
IIC2523

Sistemas Distribuidos

— Hernán F. Valdivieso López —
(2025 - 2 / Clase 12)

Replicación de Datos I

¿Cómo aseguramos consistencia en las réplicas?

Temas de la clase

1. Introducción
 - a. ¿Por qué replicar los datos?
 - b. Modelos de Consistencia
2. Modelos de consistencia fuerte (Linealización) y Secuencial
3. Modelo de consistencia Causal
4. Modelo de consistencia eventual

Introducción

¿Por qué replicar los datos?

Modelos de Consistencia

¿Por qué replicar la información?

- ◆ En un sistema distribuido, la replicación de información permite ofrecer una serie de propiedades.
- ◆ **Alta Disponibilidad (*High Availability*):**
 - ◆ La replicación garantiza que si un nodo falla, el servicio general permanece disponible para los usuarios al tener copias de los datos en otros lugares.
 - ◆ Esto también ayuda a ser tolerante a fallos.
 - ◆ Si hay n servidores y la probabilidad de fallo de uno es p , la disponibilidad con replicación es $1 - \text{Prob}(\text{todos fallan}) = 1 - p^n$

¿Por qué replicar la información?

- ◆ En un sistema distribuido, la replicación de información permite ofrecer una serie de propiedades.
- ◆ **Rendimiento y Escalabilidad (*Performance and Scalability*):**
 - ◆ **Reducción de Latencia:** Al tener copias de los datos más cerca de los clientes (físicamente o en la red), se reduce el tiempo de respuesta para las operaciones de lectura.
 - ◆ **Balanceo de Carga:** La replicación permite distribuir las solicitudes de lectura y escritura entre múltiples servidores, evitando cuellos de botella en un único punto.

¿Por qué replicar la información?

- ◆ En un sistema distribuido, la replicación de información permite ofrecer una serie de propiedades.
- ◆ **Operación Desconectada (*Disconnected Operation*):**
 - ◆ Permite a los clientes (por ejemplo, en dispositivos móviles) seguir trabajando con sus copias locales de datos incluso cuando no hay conectividad de red, y luego sincronizar los cambios al reconectarse.

¿Por qué replicar la información? Desafíos

- ◆ La replicación introduce un problema... cada vez que se actualiza una réplica, esta se diferencia de las demás.
 - ◆ Sin reloj global precisó es difícil identificar la "última" operación de escritura.

¿Por qué replicar la información? Desafíos

- ◆ La replicación introduce un problema... cada vez que se actualiza una réplica, esta se diferencia de las demás.
 - ◆ Sin reloj global precisó es difícil identificar la "última" operación de escritura.
- ◆ Es necesario propagar las actualizaciones de tal manera que las inconsistencias temporales no se noten.
 - ◆ Pero... esta propagación puede degradar severamente el rendimiento, especialmente en sistemas distribuidos a gran escala.

¿Por qué replicar la información? Desafíos

- ◆ La replicación introduce un problema... cada vez que se actualiza una réplica, esta se diferencia de las demás.
 - ◆ Sin reloj global precisó es difícil identificar la "última" operación de escritura.
- ◆ Es necesario propagar las actualizaciones de tal manera que las inconsistencias temporales no se noten.
 - ◆ Pero... esta propagación puede degradar severamente el rendimiento, especialmente en sistemas distribuidos a gran escala.
- ◆ La solución a este problema es relajar la consistencia en cierta medida.
 - ◆ Esto produce la necesidad de diferentes **modelos de consistencia**

Modelos de Consistencia

- ◆ Un modelo de consistencia es esencialmente un contrato entre los procesos/nodos y la base de datos.
 - ◆ Si los procesos/nodos aceptan obedecer ciertas reglas, la base de datos promete funcionar consistentemente.
- ◆ Los modelos se basan principalmente en operaciones que modifican algún dato (*write*) y operaciones de lecturas (*read*).
- ◆ **La replicación de un nodo "A" a otro nodo "B" sólo se confirma cuando el nodo "B" hace un *read* sobre el dato consultado.**

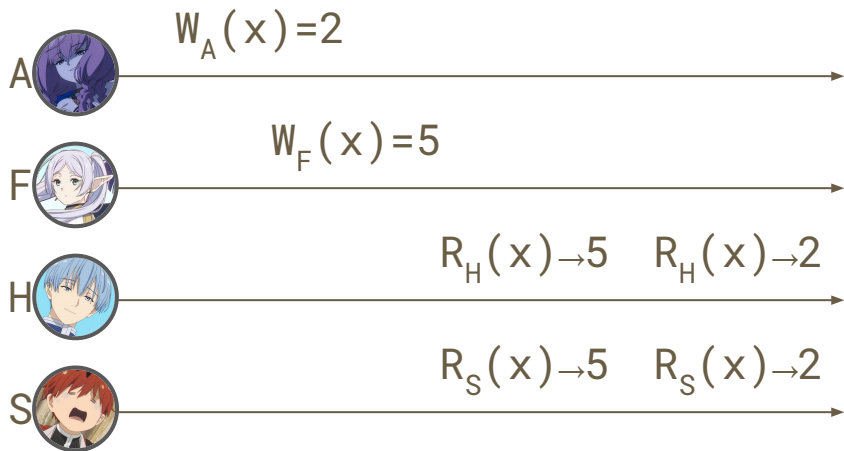
Modelo de Consistencia Secuencial y Fuerte

Consistencia Secuencial [Lamport, 1979]

- ◆ El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos se ejecutarán **en algún orden secuencial** ... y cada proceso individual respeta dicho orden.
 - ◆ Cualquier intercalación válida de las operaciones entre nodos es aceptable... pero todos los nodos ven la misma intercalación.

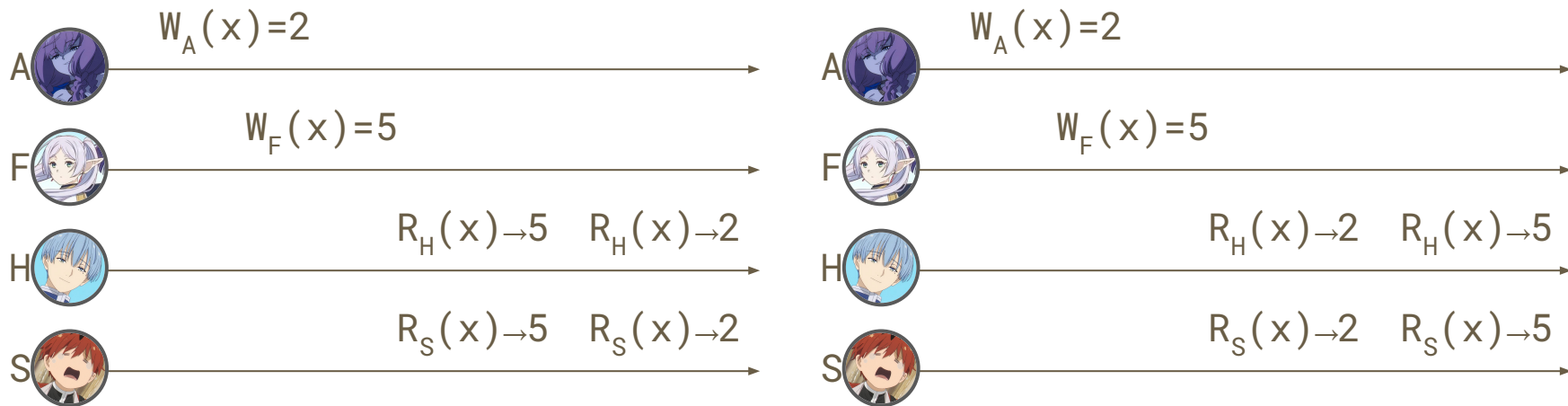
Consistencia Secuencial [Lamport, 1979]

- ◆ El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos se ejecutarán **en algún orden secuencial** ... y cada proceso individual respeta dicho orden.
 - ◆ Cualquier intercalación válida de las operaciones entre nodos es aceptable... pero todos los nodos ven la misma intercalación.



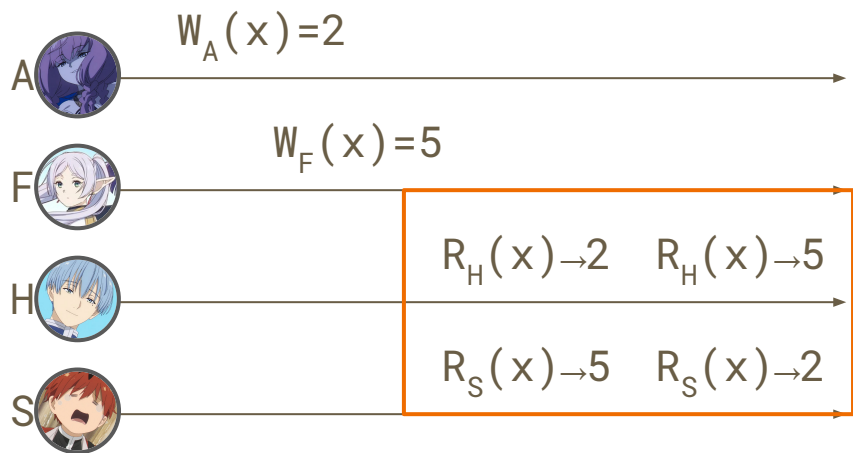
Consistencia Secuencial [Lamport, 1979]

- ◆ El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos se ejecutarán **en algún orden secuencial** ... y cada proceso individual respeta dicho orden.
 - ◆ Cualquier intercalación válida de las operaciones entre nodos es aceptable... pero todos los nodos ven la misma intercalación.



Consistencia Secuencial [Lamport, 1979]

- ◆ El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos se ejecutarán **en algún orden secuencial** ... y cada proceso individual respeta dicho orden.
 - ◆ Cualquier intercalación válida de las operaciones entre nodos es aceptable... pero todos los nodos ven la misma intercalación.

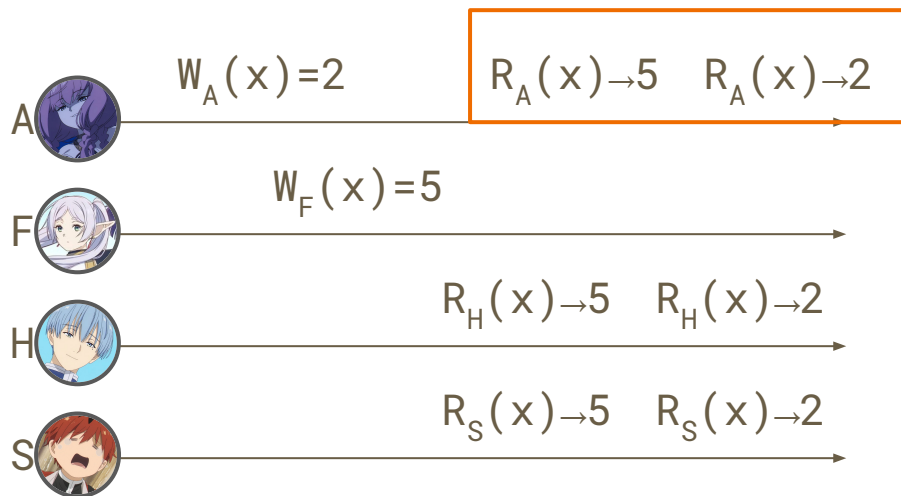


No es consistente secuencialmente.

¿Cuál fue primero, $W_A(x) = 2$ o $W_F(x) = 5$?

Consistencia Secuencial [Lamport, 1979]

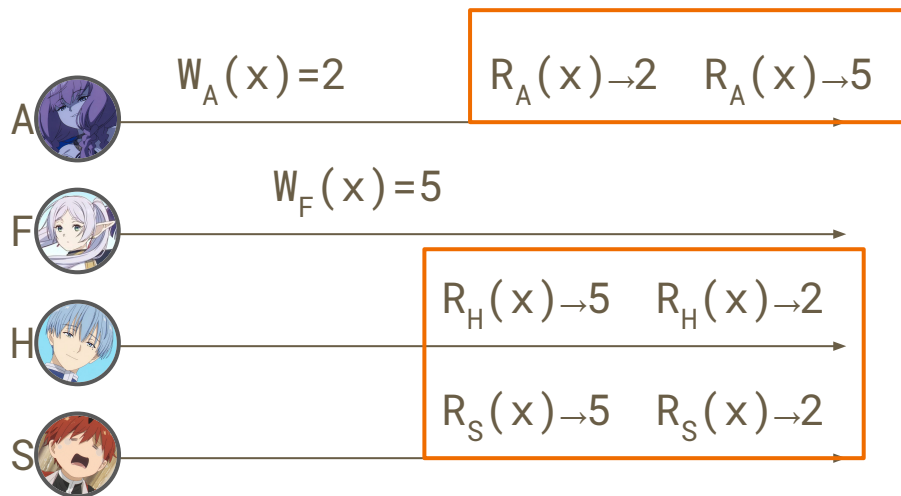
- ◆ El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos se ejecutarán **en algún orden secuencial** ... y cada proceso individual respeta dicho orden.
 - ◆ Cualquier intercalación **válida** de las operaciones entre nodos es aceptable... pero todos los nodos ven la misma intercalación.



No es válido que el nodo A no puede leer 2 después del 5

Consistencia Secuencial [Lamport, 1979]

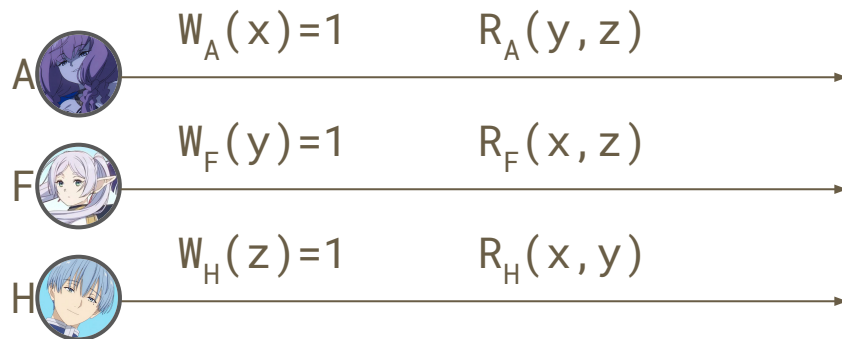
- ◆ El resultado de cualquier ejecución es el mismo que si las operaciones de todos los procesos se ejecutarán **en algún orden secuencial** ... y cada proceso individual respeta dicho orden.
 - ◆ Cualquier intercalación **válida** de las operaciones entre nodos es aceptable... pero todos los nodos ven la misma intercalación.



Nodo A indica que 5 se escribe después de 2, pero nodo H y S dan a entender lo opuesto.

No hay una secuencia de operaciones que todos los nodos respeten

Consistencia Secuencial [Lamport, 1979]



Modifican una variable y leen las otras.

¿es posible ver un
read 00 00 00 o
00 10 01? 🤔

$W_A(x)=1$
 $W_F(y)=1$
 $W_H(z)=1$
 $R_A(y, z)$ [11]
 $R_F(x, z)$ [11]
 $R_H(x, y)$ [11]

Read: 11 11 11

$W_A(x)=1$
 $R_A(y, z)$ [00]
 $W_F(y)=1$
 $R_F(x, z)$ [10]
 $W_H(z)=1$
 $R_H(x, y)$ [11]

Read: 00 10 11

$W_A(x)=1$
 $W_F(y)=1$
 $R_F(x, z)$ [10]
 $R_A(y, z)$ [10]
 $W_H(z)=1$
 $R_H(x, y)$ [11]

Read: 10 10 11

$W_F(y)=1$ [00]
 $W_H(z)=1$
 $R_F(x, z)$ [01]
 $R_H(x, y)$ [01]
 $W_A(x)=1$
 $R_A(y, z)$ [11]

Read: 01 01 11

Consistencia Secuencial [Lamport, 1979]

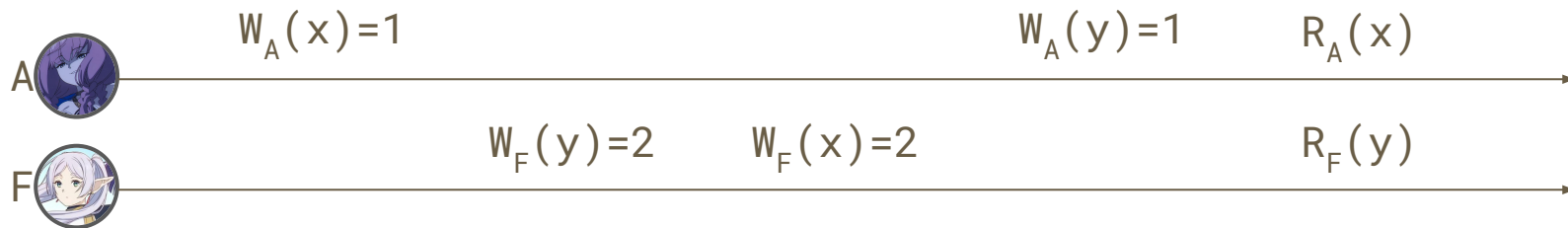
Resumen

El resultado final es como si las operaciones de todos los nodos se hubieran entrelazado de alguna manera serial/secuencial, preservando el orden individual de cada nodo.

Consistencia Secuencial [Lamport, 1979]

Problema

Muchas opciones válidas que son secuencialmente consistente.



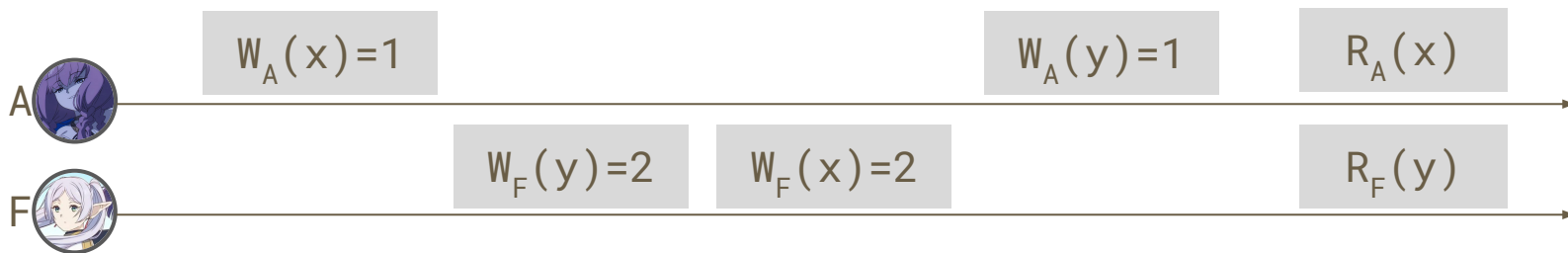
Existen 6 formas de ordenar $W_A(x)=1$ $W_A(y)=1$ $W_F(y)=2$ $W_F(x)=2$ que serán secuencialmente consistentes.

Consistencia Fuerte [Herlihy & Wing, 1986]

- ◆ Consistencia más estricta que la secuencia. Llamada también linealizable.
- ◆ Cada operación debe aparecer como que tiene efecto instantáneo en algún momento entre su inicio y su finalización.
 - ◆ La consistencia fuerte está **totalmente ligado al tiempo real de las operaciones**.
- ◆ Al momento que se completa un *write*, esta operación se propaga inmediato a las demás réplicas.

Consistencia Fuerte [Herlihy & Wing, 1986]

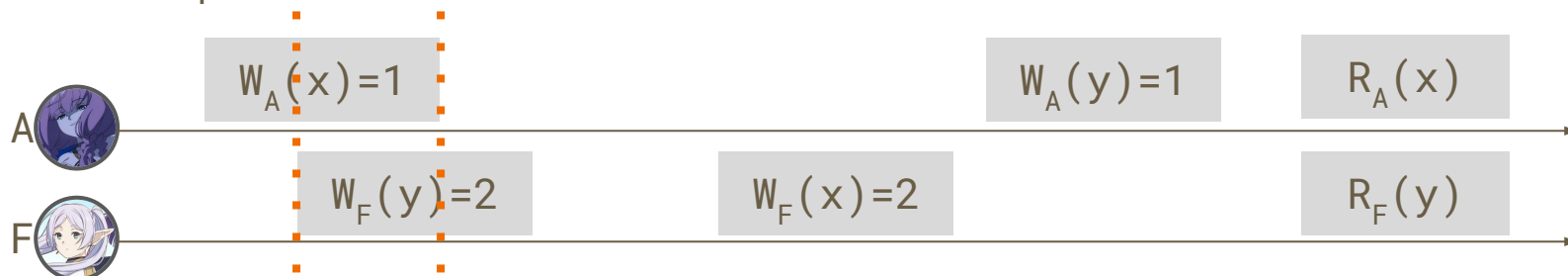
- ◆ Consistencia más estricta que la secuencia. Llamada también linealizable.
- ◆ Cada operación debe aparecer como que tiene efecto instantáneo en algún momento entre su inicio y su finalización.
 - ◆ La consistencia fuerte está **totalmente ligado al tiempo real de las operaciones**.
- ◆ Al momento que se completa un *write*, esta operación se propaga inmediato a las demás réplicas.



Solo 1 secuencia válida de *writes*

Consistencia Fuerte [Herlihy & Wing, 1986]

- ◆ Consistencia más estricta que la secuencia. Llamada también linealizable.
- ◆ Cada operación debe aparecer como que tiene efecto instantáneo en **algún momento entre su inicio y su finalización**.
 - ◆ La consistencia fuerte está **totalmente ligado al tiempo real de las operaciones**.
- ◆ Al momento que se completa un *write*, esta operación se propaga inmediato a las demás réplicas.



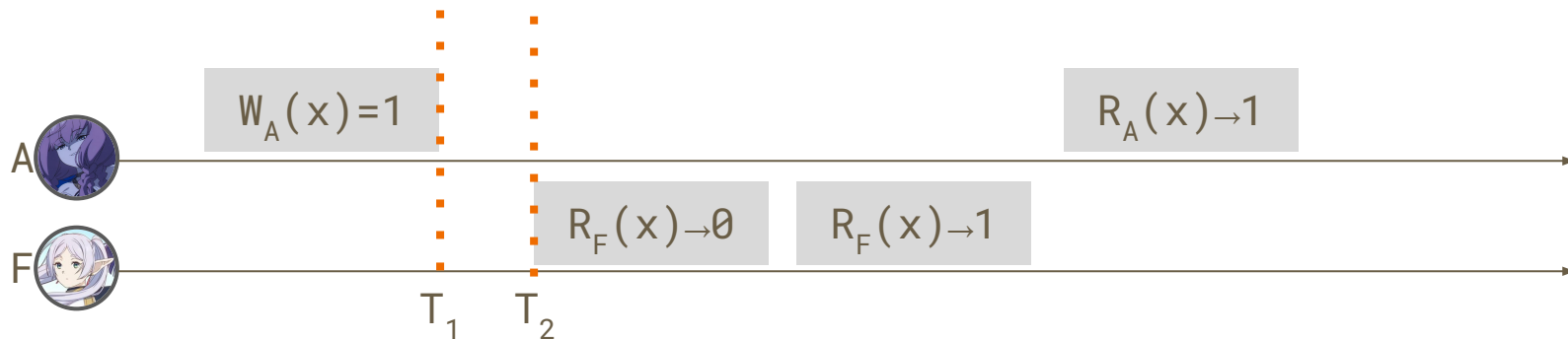
Solo 2 secuencia válida de *writes*

Partir con $W_a(x)=1$ o partir con $W_f(y)=2$

Diferencia entre Consistencia Fuerte y Secuencial

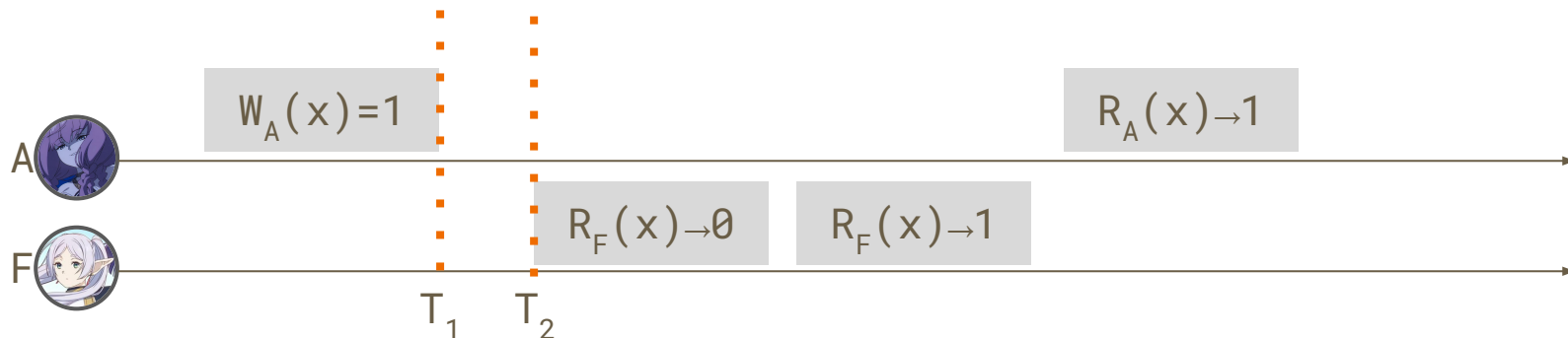
- ◆ Toda linealización (consistencia fuerte) es secuencialmente consistente, pero no toda consistencia secuencial es linealizable.
 - ◆ Linealización implica consistencia secuencial
- ◆ La linealización es más estricta porque impone una relación de orden total basada **en el tiempo real**, mientras que la consistencia secuencial solo requiere que se preserve el orden de programa de cada cliente.
- ◆ En un sistema distribuido, es muy difícil garantizar consistencia fuerte dada la latencia de los mensajes.

Diferencia entre Consistencia Fuerte y Secuencial



◆ Asumiendo $T_1 < T_2$ ¿Se cumple algún modelo de consistencia?

Diferencia entre Consistencia Fuerte y Secuencial



- ◆ Asumiendo $T_1 < T_2$ ¿Se cumple algún modelo de consistencia?
- ◆ Es consistente secuencialmente, pero no es linealizable.
- ◆ Puede ser que nodo F leyó la variable X antes que le llegara la propagación del cambio de X por el nodo A.
 - ◆ Existe una secuencia **válida secuencial** ($R_F \rightarrow W_A \rightarrow R_F \rightarrow R_A$)
 - ◆ No respeta el requisito de **tiempo real** de la consistencia fuerte.

Modelo de Consistencia Causal

Consistencia Causal [Huto & Ahamad, 1990]

- ◆ El modelo de consistencia causal es un debilitamiento de la consistencia secuencial.
- ◆ Hace una distinción entre los eventos que están potencialmente relacionados causalmente y aquellos que no lo están.

Consistencia Causal [Huto & Ahamad, 1990]

- ◆ El modelo de consistencia causal es un debilitamiento de la consistencia secuencial.
- ◆ Hace una distinción entre los eventos que están potencialmente relacionados causalmente y aquellos que no lo están.
 - ◆ Si un evento B es causado o influenciado por un evento anterior A, la causalidad requiere que todos los demás vean primero A y luego B.
 - ◆ Si un evento C, ejecutado por otro nodo de forma concurrente a B, no tiene ninguna causa, puede estar posicionado en diferentes partes para diversas réplicas. No está "obligado" a una única secuencia.

Consistencia Causal [Huto & Ahamad, 1990]

Ejemplo foro de discusión

- ◆ Persona 1 publica un *post* sobre Memes.
- ◆ Persona 2 lee el post de meme y hace un comentario
- ◆ Persona 3 publica un *post* sobre Anime

Consistencia Causal [Huto & Ahamad, 1990]

Ejemplo foro de discusión

- ◆ Persona 1 publica un *post* sobre Memes.
- ◆ Persona 2 lee el post de meme y hace un comentario
- ◆ Persona 3 publica un *post* sobre Anime

En un sistema con consistencia causal ocurre que:

- ◆ Todos los usuarios deben ver primero el post de meme y luego el comentario.
- ◆ No se impone un orden global entre el *post* de "Memes" y "Anime", ya que no hay una relación causal directa entre ellos.

Consistencia Causal [Huto & Ahamad, 1990]

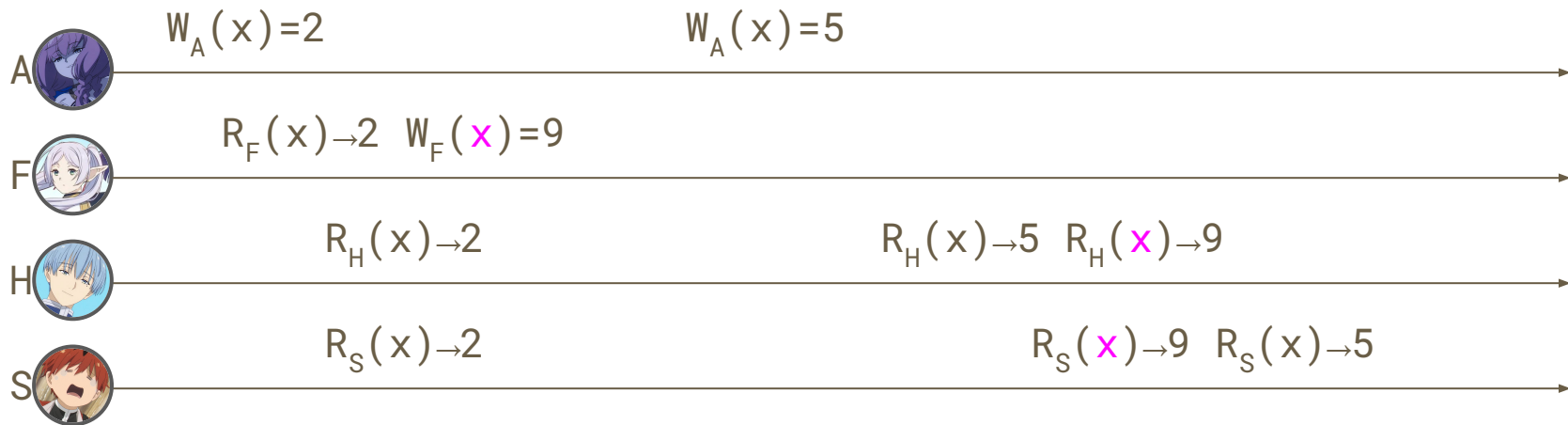
Ejemplo foro de discusión

- ◆ Persona 1 publica un *post* sobre Memes.
- ◆ Persona 2 lee el post de meme y hace un comentario
- ◆ Persona 3 publica un *post* sobre Anime

En un sistema con consistencia causal ocurre que:

- ◆ Todos los usuarios deben ver primero el post de meme y luego el comentario.
- ◆ No se impone un orden global entre el *post* de "Memes" y "Anime", ya que no hay una relación causal directa entre ellos.
 - ◆ Un usuario le puede aparecer primero el post de "Memes", mientras que otro ve primero el post de "Anime". Lo único que se exige es que el comentario que hizo persona 2 esté posterior a la publicación del *post* de Memes.

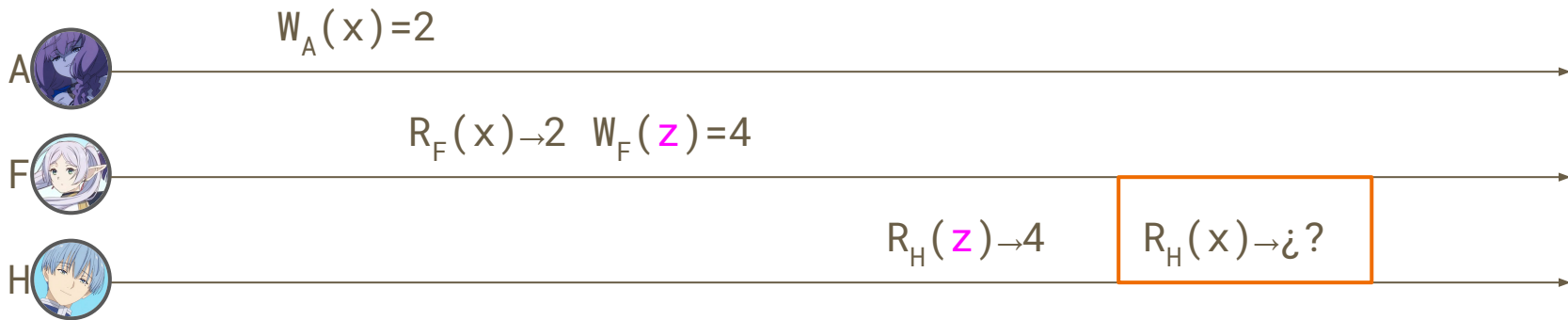
Consistencia Causal [Huto & Ahamad, 1990]



- ◆ Nodos H y S ven los *write* en órdenes distintos \rightarrow No es consistente secuencial
- ◆ Nodo F leyó la escritura del nodo A ($x=2$) antes de escribir él ($x=9$)
 - ◆ Solo se exige que se respete que 2 viene antes de 9. Si algún nodo lee el 2 posterior al 9, se rompe la consistencia causal.

Consistencia Causal [Huto & Ahamad, 1990]

- ◆ Solo se exige causalidad.



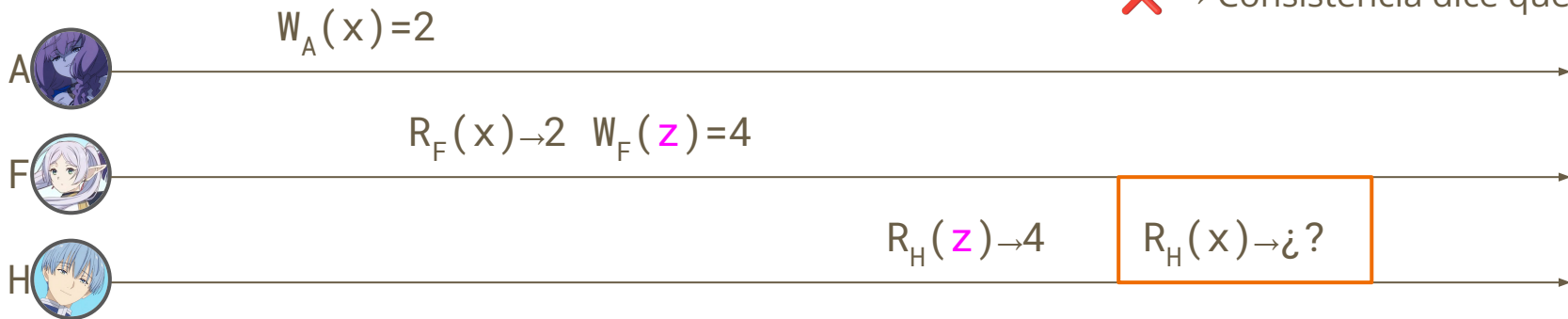
- ◆ ¿Qué podemos decir del valor de **X** al final si aseguramos consistencia causal?

Consistencia Causal [Huto & Ahamad, 1990]

- ◆ Solo se exige causalidad.

✓ → Consistencia dice que **Si**

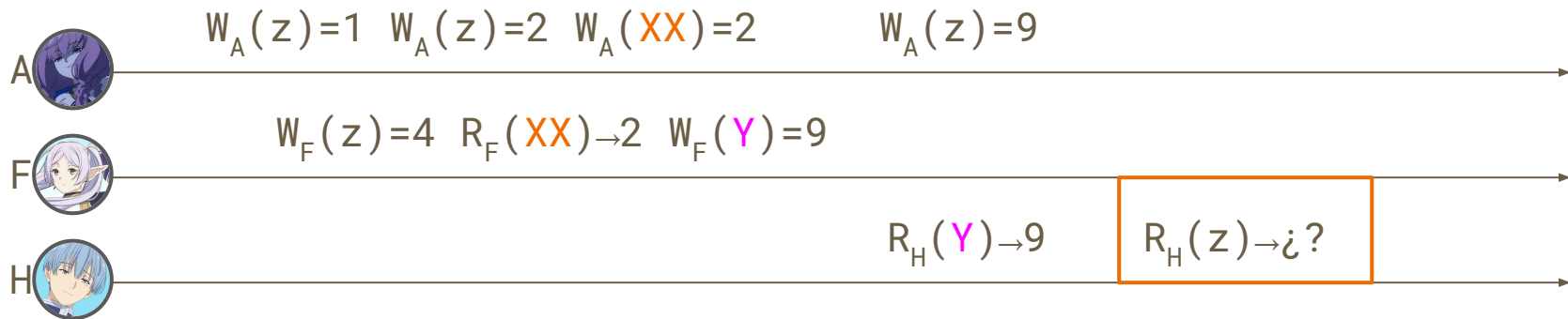
✗ → Consistencia dice que **No**



- ◆ ¿Qué podemos decir del valor de **X** al final si aseguramos consistencia causal?
 - ◆ ¿Puede tomar el valor 2? ✓
 - ◆ ¿Puede tomar el valor NULL (sin definir la variable todavía)? ✗

Consistencia Causal [Huto & Ahamad, 1990]

- ◆ Solo se exige causalidad.



- ◆ ¿Qué podemos decir del valor de **Z** al final si aseguramos consistencia causal?

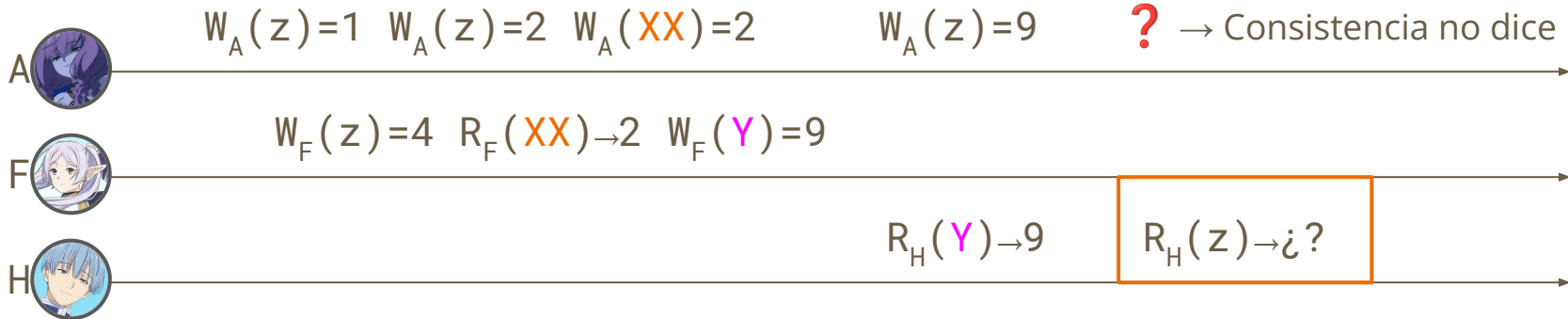
Consistencia Causal [Huto & Ahamad, 1990]

◆ Solo se exige causalidad.

✓ → Consistencia dice que **Si**

✗ → Consistencia dice que **No**

? → Consistencia no dice nada.



◆ ¿Qué podemos decir del valor de **Z** al final si aseguramos consistencia causal?

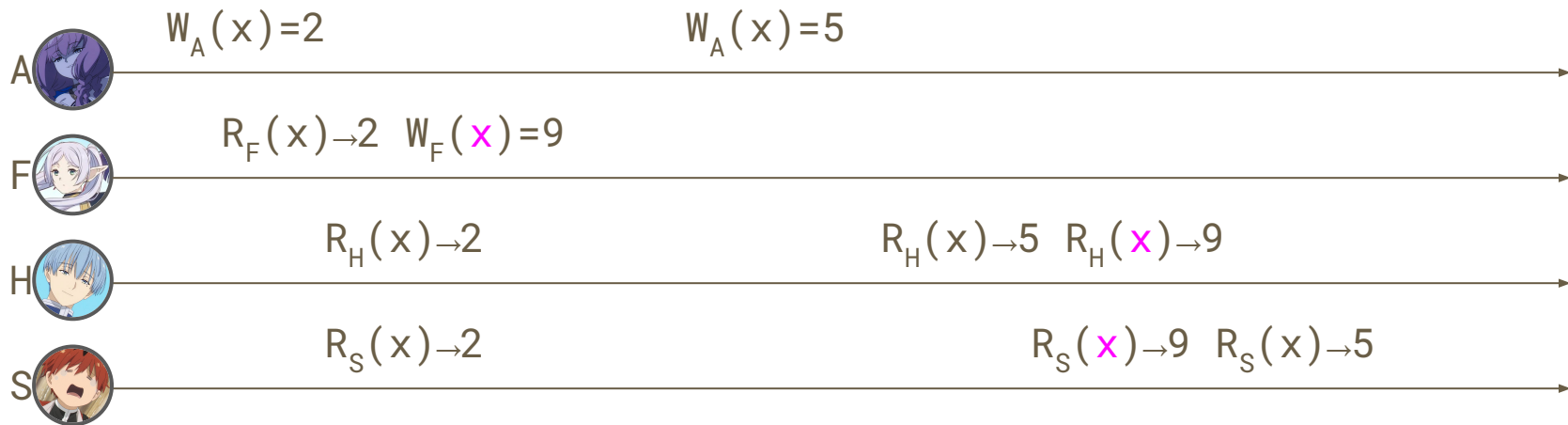
- ◆ ¿Puede tomar el valor 1? ✗
- ◆ ¿Puede tomar el valor 2? ✓
- ◆ ¿Puede tomar el valor 4? ✓
- ¿Puede tomar el valor 9? ?
- ¿Puede tomar el valor NULL? ✗

Modelo de Consistencia Eventual

Consistencia Eventual [Vogels, 2009]

- ◆ Modelo que prioriza la alta disponibilidad y el rendimiento sobre la consistencia.
- ◆ Si 2 o más nodos no están escribiendo exactamente al mismo tiempo (no hay conflicto de escritura), las réplicas convergerá hacia copias idénticas entre sí.
- ◆ Se garantiza que las actualizaciones se propagan eventualmente a las réplicas.
 - ◆ Esto permite una relajación en la consistencia entre réplicas. Dos clientes pueden observar réplicas diferentes, e incluso un cliente puede observar datos obsoletos.
- ◆ En caso que exista un conflicto de escritura, necesitamos alguna estrategia... por ejemplo, se escoge uno que sobrescriba al otro.

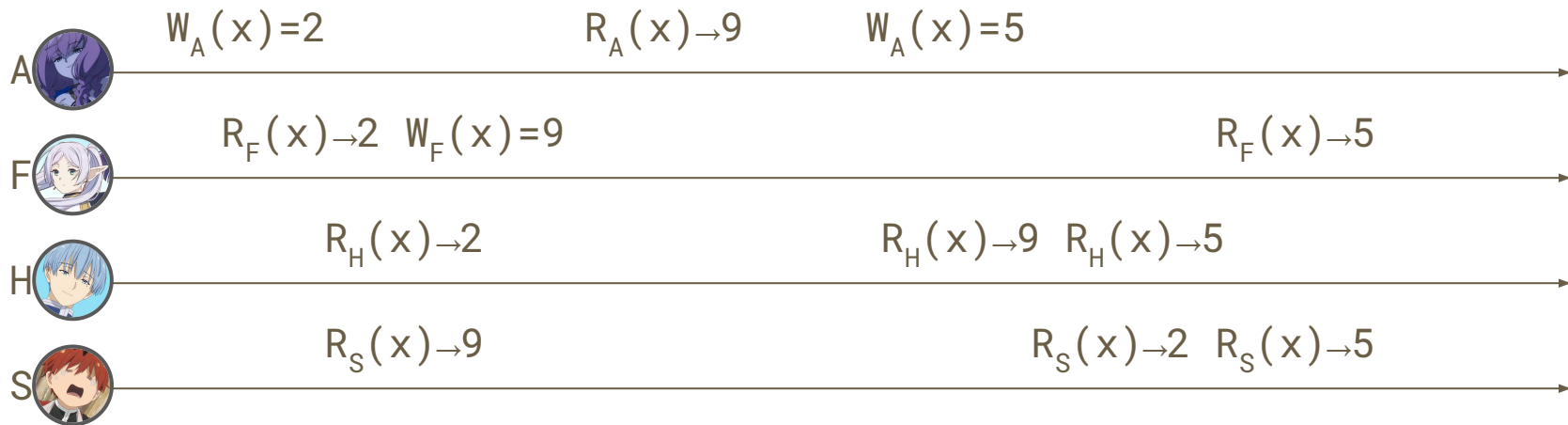
Consistencia Eventual [Vogels, 2009]



◆ Vimos anteriormente que este caso es consistente causal, pero no es consistente eventual.

- ◆ H primero recibió $W_A(x)=5$ y luego recibió $W_F(x)=9$
- ◆ S primero recibió $W_F(x)=9$ y luego recibió $W_A(x)=5$
- ◆ No van a llegar a un resultado igual.

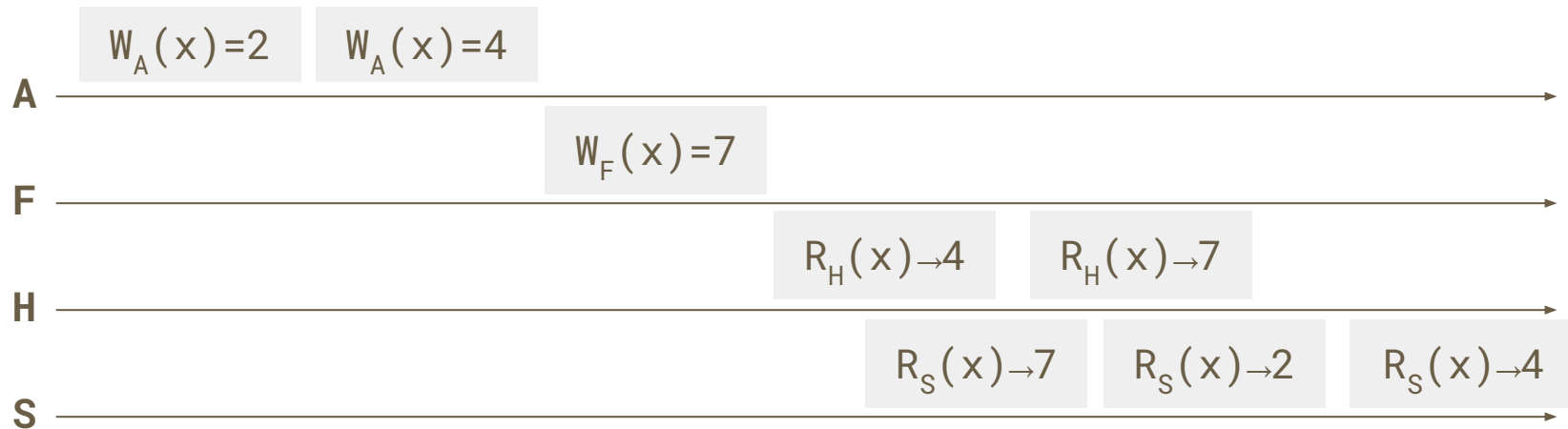
Consistencia Eventual [Vogels, 2009]



- ◆ Nodos H y S ven los *write* en distinto orden → No es consistente secuencial.
- ◆ Nodo S ve primero que x es 9 y luego que es 2 → No es consistente causal
 - ◆ Ojo que este caso sigue siendo factible... Si la comunicación de $A \rightarrow S$ es más lenta, puede pasar que $W_A(x)=2$ llegue después que nodo F propagó su cambio.
- ◆ Todos llegan finalmente a ver $x=5$ → Consistente eventual

Poniendo a prueba lo que hemos aprendido

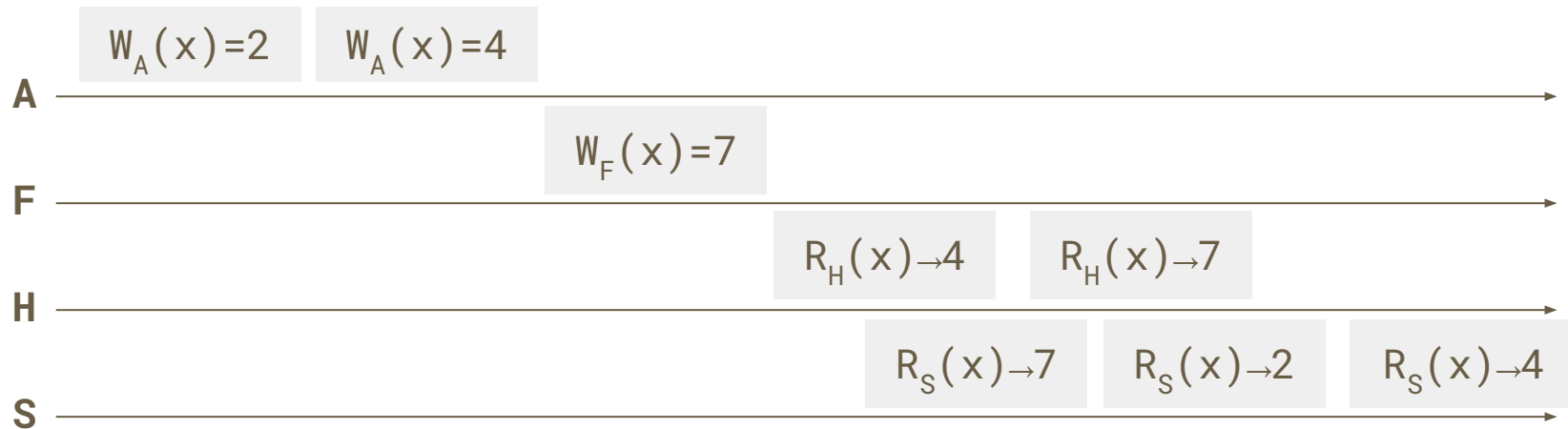
Para cada modelo indique si se cumple o no



- Consistencia Fuerte (linealización)
- Consistencia Secuencial
- Consistencia Causal
- Consistencia Eventual

Poniendo a prueba lo que hemos aprendido

Para cada modelo indique si se cumple o no



- a. Consistencia Fuerte (linealización) ✗
- b. Consistencia Secuencial ✗
- c. Consistencia Causal ✓
- d. Consistencia Eventual ✗

Próximos eventos

Próxima clase

- ◆ Replicación de Datos II - Protocolos de Replicación
 - ◆ ¿Cuales existen?
 - ◆ ¿Qué tipo de consistencia intenta abordar los Protocolos?

Evaluación

- ◆ El otro lunes su publica control 4, evalúa hasta la próxima clase.

IIC2523

Sistemas Distribuidos

— Hernán F. Valdivieso López —
(2025 - 2 / Clase 12)

Créditos (animes utilizados)

Sōsō no Frieren

