# Causal Message Delivery, Cooked Three Ways 

Lindsey Kuper<br>UC Santa Cruz<br>IFP WG 2.16 (Language Design), January 2023

2
I








what meeting with Lindsey?

















































































some design considerations for the "can you keep a secret?" protocol

## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!
- When a process wants to send a message as a result of an eager send it previously received, can the new message go into the output buffer while waiting for the now-you-can-tell message, or should it not even be buffered yet?


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!
- When a process wants to send a message as a result of an eager send it previously received, can the new message go into the output buffer while waiting for the now-you-can-tell message, or should it not even be buffered yet?
- I don't know!


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!
- When a process wants to send a message as a result of an eager send it previously received, can the new message go into the output buffer while waiting for the now-you-can-tell message, or should it not even be buffered yet?
- I don't know!
- (my excuse for giving a distributed systems talk at WGLD) How about some kind of information flow analysis?


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!
- When a process wants to send a message as a result of an eager send it previously received, can the new message go into the output buffer while waiting for the now-you-can-tell message, or should it not even be buffered yet?
- I don't know!
- (my excuse for giving a distributed systems talk at WGLD) How about some kind of information flow analysis?
- if an analysis shows $m_{2}$ is in fact not related to $m_{1}$, I can send $m_{2}$ without regard to whether I've heard back about $m_{1}$


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!
- When a process wants to send a message as a result of an eager send it previously received, can the new message go into the output buffer while waiting for the now-you-can-tell message, or should it not even be buffered yet?
- I don't know!
- (my excuse for giving a distributed systems talk at WGLD) How about some kind of information flow analysis?
- if an analysis shows $m_{2}$ is in fact not related to $m_{1}$, I can send $m_{2}$ without regard to whether I've heard back about $m_{1}$
- happens-before is too coarse; we use it because it's an overapproximation of actual causality that's easy to compute


## some design considerations for the "can you keep a secret?" protocol

- Should a recipient acknowledge eager sends right away, or wait until it gets the now-you-can-tell message?
- Right away. In fact, for FIFO delivery, the sender shouldn't send now-you-can-tell until the eager send is ack'd!
- Can you do an eager send as a result of an eager send?
- No. E.g., if Ron can eagerly send $m_{3}$, then it can overtake $m_{1}$ on the way to Sam, which was the original problem!
- When a process wants to send a message as a result of an eager send it previously received, can the new message go into the output buffer while waiting for the now-you-can-tell message, or should it not even be buffered yet?
- I don't know!
- (my excuse for giving a distributed systems talk at WGLD) How about some kind of information flow analysis?
- if an analysis shows $m_{2}$ is in fact not related to $m_{1}$, I can send $m_{2}$ without regard to whether I've heard back about $m_{1}$
- happens-before is too coarse; we use it because it's an overapproximation of actual causality that's easy to compute
- it's 2023 and we're PL people, dang it; let's use language-level techniques that weren't available when these protocols were originally designed in the '90s

