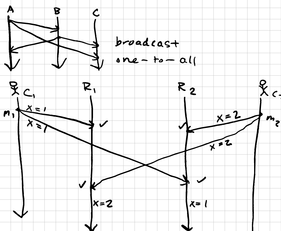
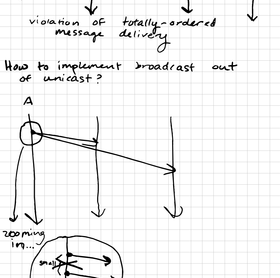
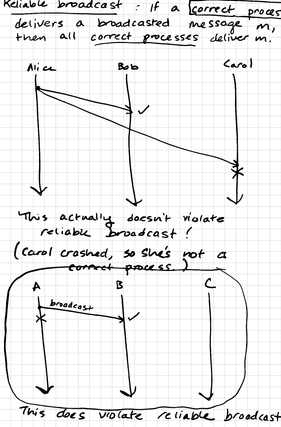
CSE138 Lecture 9

✓- reliable broadcast ← define & implement
✓- intro to replication
- review for the midterm





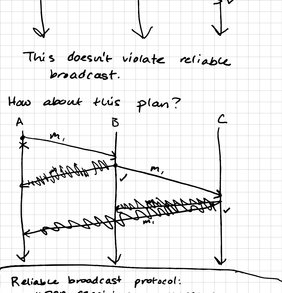
How to implement broadcast out of unicast?



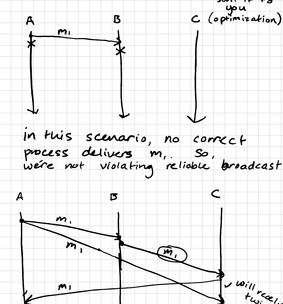
Problem: we could crash while trying to broadcast.

in the crash model, this means a process that stops crashed

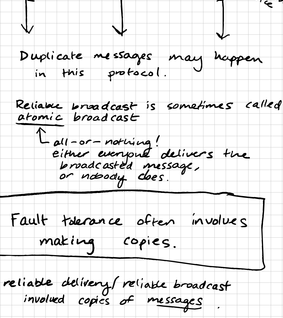
Reliable broadcast: If a correct process delivers a broadcasted message m, then all correct processes deliver m.



This actually doesn't violate reliable broadcast!
(Carol crashed, so she's not a correct process)



This does violate reliable broadcast.



This doesn't violate reliable broadcast.

How about this plan?

Reliable broadcast protocol:
- upon receiving a message, don't deliver it immediately, instead, broadcast it to everyone else*! then deliver it.

* except whoever sent it to you (optimization)

in this scenario, no correct process delivers m_1. So, we're not violating reliable broadcast.

Duplicate messages may happen in this protocol.

Reliable broadcast is sometimes called atomic broadcast
↳ all-or-nothing! either everyone delivers the broadcasted message, or nobody does.

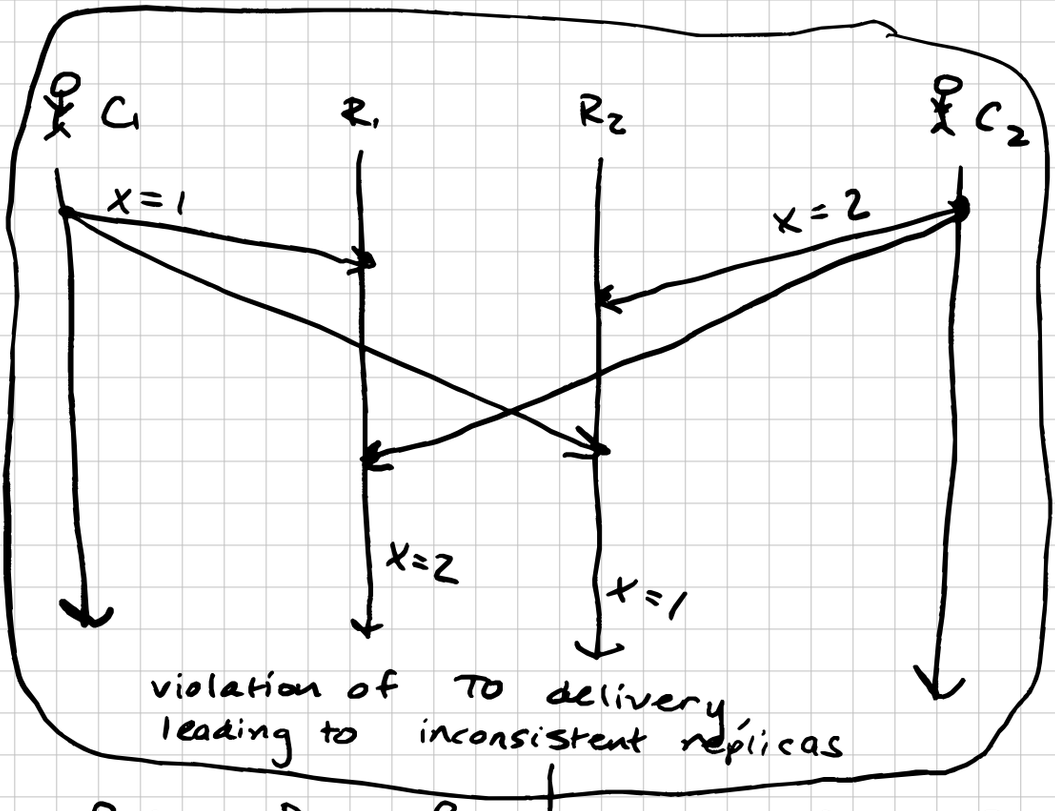
Fault tolerance often involves making copies.

reliable delivery/reliable broadcast involved copies of messages.

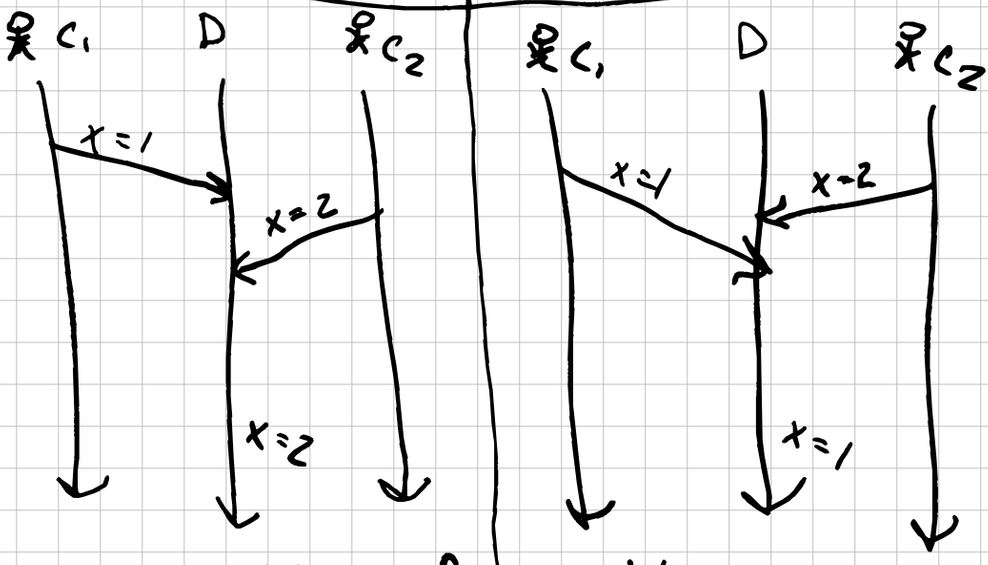
REPLICATION (making copies of data)

Replication: reasons to do it
- guard against data loss
- data locality (keep data close to clients that need it)
- dividing up the work (scalability); handling more requests at once.

availability



violation of TO delivery leading to inconsistent replicas



Not a problem of consistency, rather, a problem of determinism.

determinism is a property that relates <u>different runs</u> of a system.

↑ i.e. a hyperproperty

OK, back to consistency.

informally: a replicated storage system is <u>strongly consistent</u> if clients can't tell that the data is replicated.

somehow we have to make everyone deliver messages in the same order.

if you want a <u>formal definition</u>

<u>linearizability</u> ← Herlihy & Wing