

Distributed Computing

Using multiple ***sites*** to solve a problem

Distributed Computing

Using multiple ***sites*** to solve a problem

Network filesystem

Distributed Computing

Using multiple ***sites*** to solve a problem

Network filesystem

Cloud computing

Distributed Computing

Using multiple ***sites*** to solve a problem

Network filesystem

Cloud computing

Multi-player game

Distributed Computing

Using multiple **sites** to solve a problem

Network filesystem

Cloud computing

Multi-player game

Collaborative spreadsheet

Distributed Computing

Using multiple **sites** to solve a problem

Network filesystem

Cloud computing

Multi-player game

Collaborative spreadsheet

E-commerce site

Distributed Computing

Using multiple **sites** to solve a problem

Network filesystem

Cloud computing

Multi-player game

Collaborative spreadsheet

E-commerce site

Scientific simulation

Distributed Computing

Using multiple **sites** to solve a problem

Network filesystem

Cloud computing

Multi-player game

Collaborative spreadsheet

E-commerce site

Scientific simulation

Monitoring a volcano

Distributed Computing

Using multiple *sites* to solve a problem

Network filesystem

Cloud computing

Multi-player game

Collaborative spreadsheet

E-commerce site

Scientific simulation

Monitoring a volcano

Searching for aliens

Distributed Computing

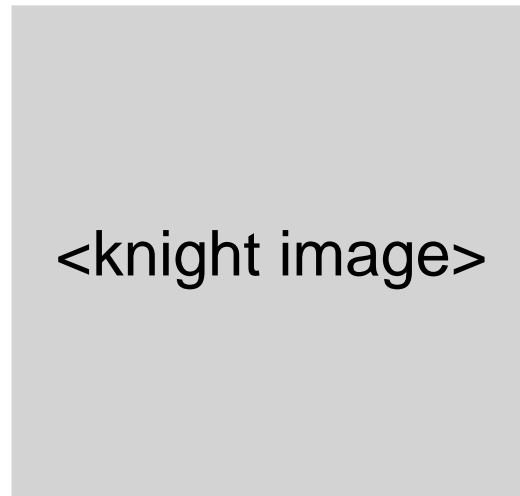
Different applications \Rightarrow different levels of transparency:

- Hardware as a service: use of multiple computer is automatic for each application
- Volcano-monitoring sensors: distribution and communication explicit in the application

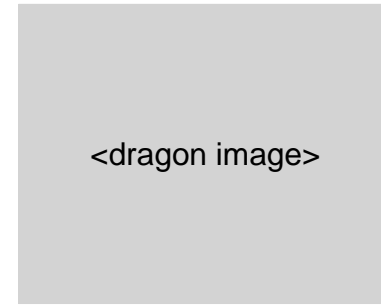
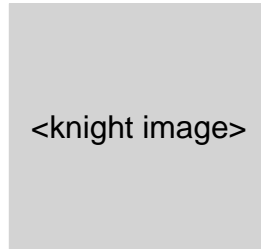
Different applications \Rightarrow same core problems:

- Synchronization
- Possibility of failure (site or communication)

Happens Before



Virtual Clocks



mumbling

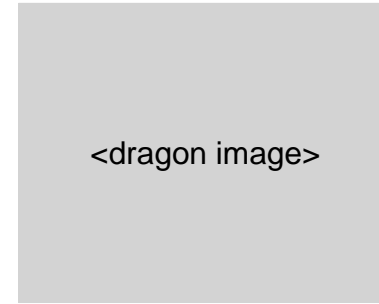
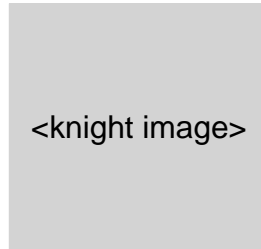
muttering

sheilding

sleeping

breathing fire

Virtual Clocks



mumbling

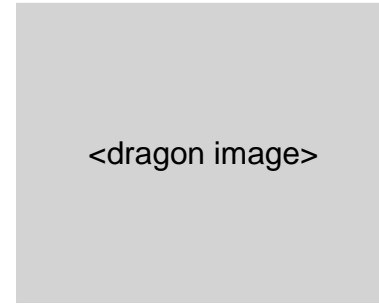
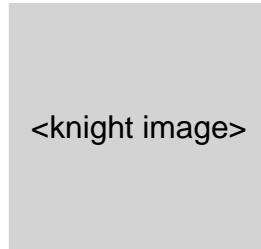
muttering

sheilding

sleeping

breathing fire

Virtual Clocks



mumbling

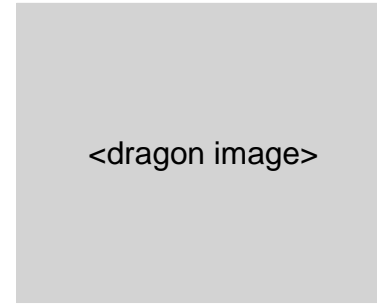
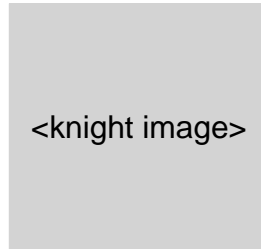
sleeping

muttering

sheilding

breathing fire

Virtual Clocks



mumbling @ 0

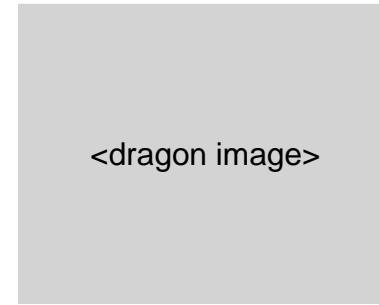
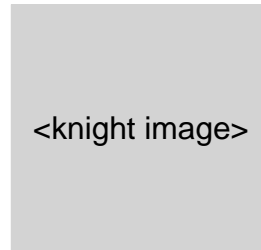
sleeping @ 0

muttering @ 1

sheilding @ 2

breathing fire @ 1

Virtual Clocks



mumbling @ 0

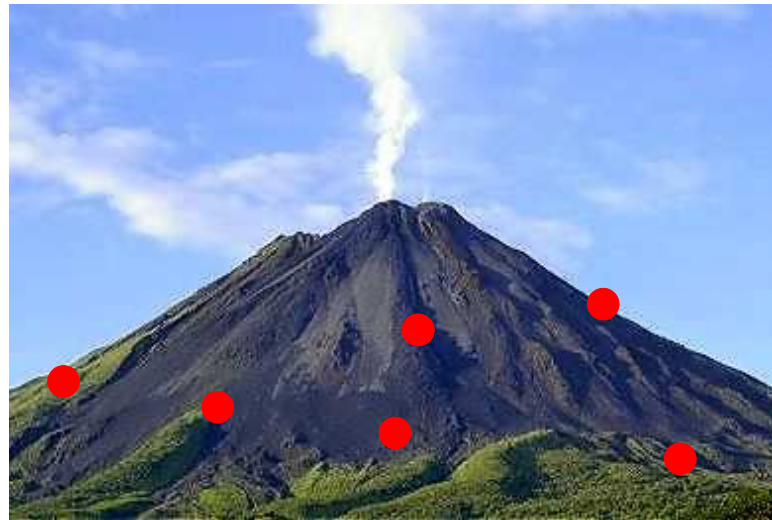
muttering @ 1

sheilding @ 2

sleeping @ 0

breathing fire @ 3

Mutual Exclusion and Locks



Centralized Lock

One site manages a given lock

Advantages:

- Simple to implement
- Easy to make fair

Disadvantages:

- Lock manager might fail

Distributed Lock

Ask everyone else's permission for lock

Advantages:

Disadvantages:

- Lots of communication
- One failure prevents use of lock

Majority Lock

Ask permission from more than 50% for lock

Advantages:

- Handles failure of sites well
- Less communication than asking everyone

Disadvantages:

- Still lots of communication

Token Ring

Wait for permission, then pass it on afterward

Advantages:

- Relatively little communication
- Clearly fair
- Makes sense when permission is needed often

Disadvantages:

- Have to set up ring
- Single failure breaks the ring
- Extra work when permission is needed rarely

Elections

In case sites need to pick a new ***coordinator*** (e.g., for a lock)

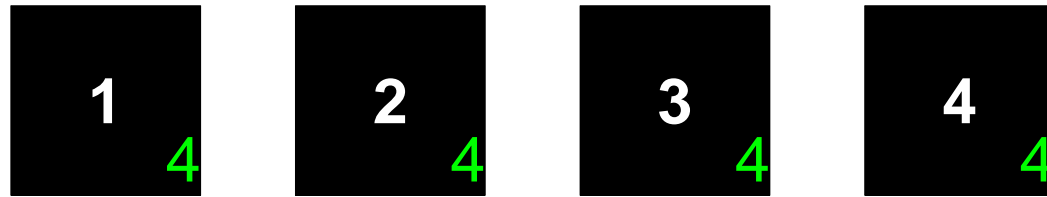
- Bully algorithm
- Ring algorithm

Bully Algorithm

Rank all sites

- If a process i doesn't hear back from a coordinator:
 - Send an `elect` message to every $j > i$
 - No responses? Then i considers itself elected and notifies everyone
 - Got response? Wait for notify from new coordinator...
- If a process i receives an `elect` message, treat it like not hearing from the coordinator

Bully Algorithm



Bully Algorithm



Bully Algorithm



Bully Algorithm



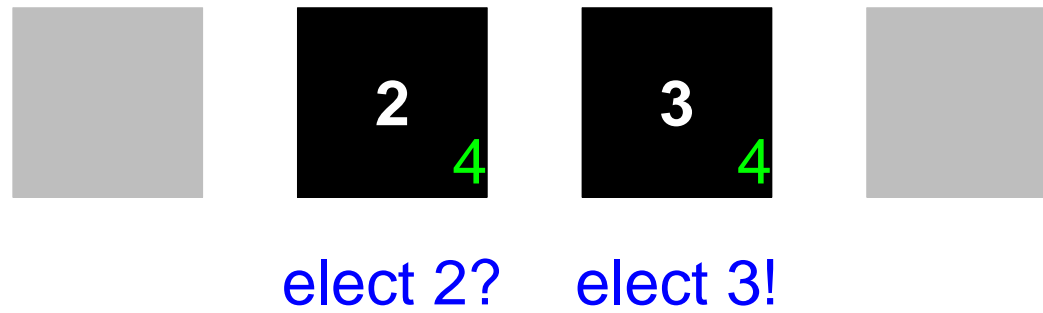
wait on 4

Bully Algorithm

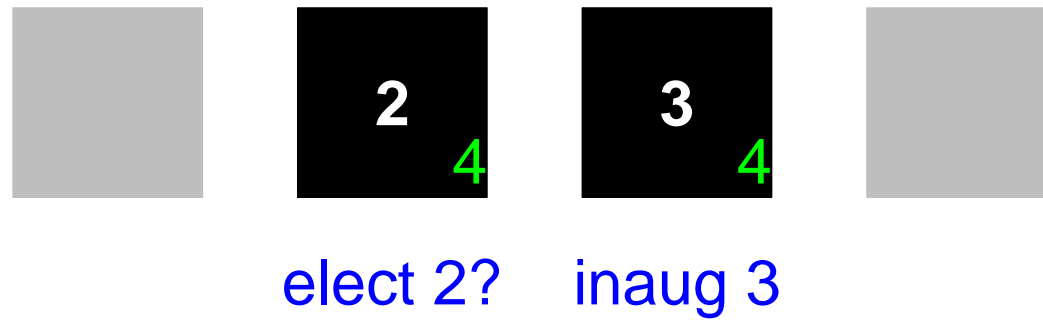


elect 2!

Bully Algorithm



Bully Algorithm



Bully Algorithm



Bully Algorithm

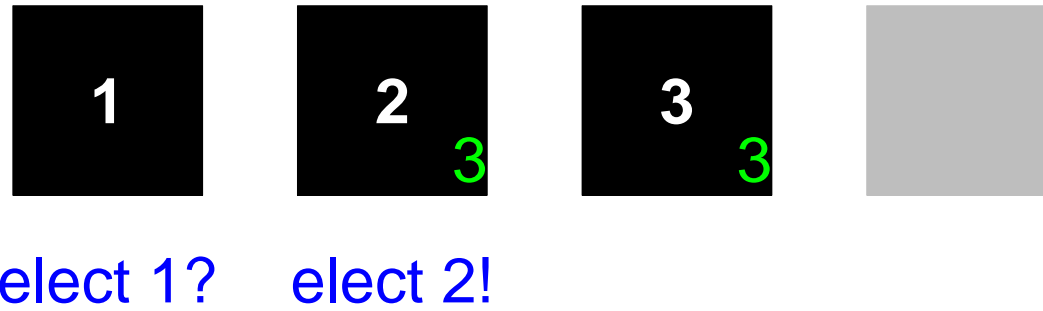


Bully Algorithm



elect 1!

Bully Algorithm



Bully Algorithm



elect 1?



elect 2?



elect 3!



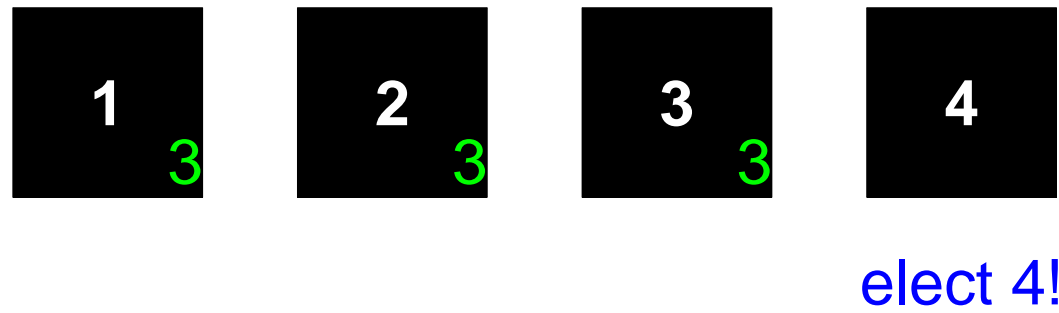
Bully Algorithm



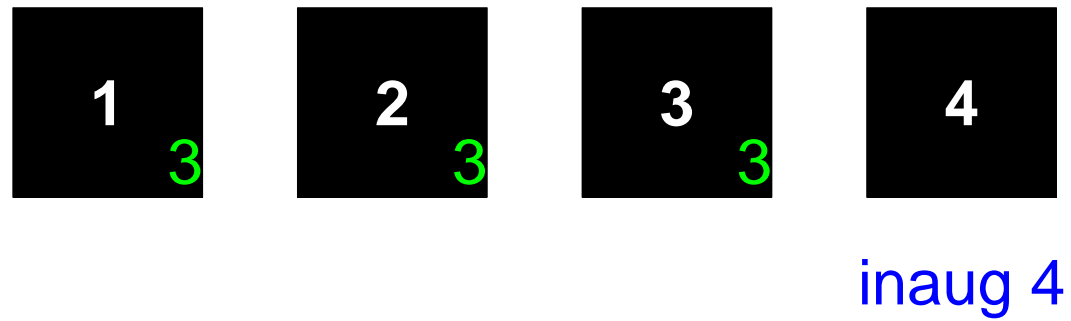
Bully Algorithm



Bully Algorithm



Bully Algorithm



Bully Algorithm



Ring Algorithm

When communication is always to the next live site

- If a process i doesn't hear back from a coordinator:
 - Send an `elect` message to neighbor
 - Accumulate `elect` messages to learn about live sites
 - Get own `elect` \Rightarrow all sites known

Ring Algorithm



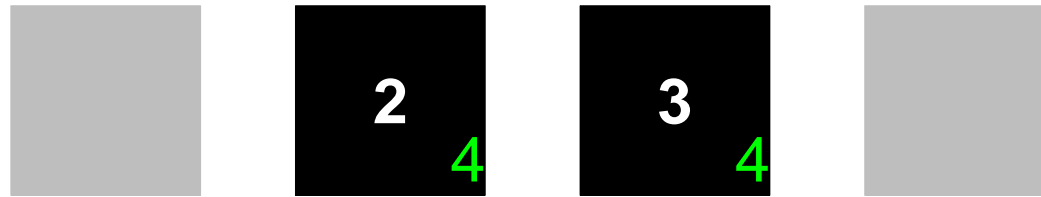
Ring Algorithm



Ring Algorithm

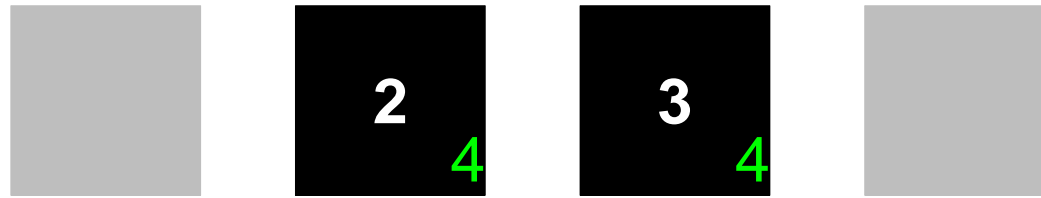


Ring Algorithm



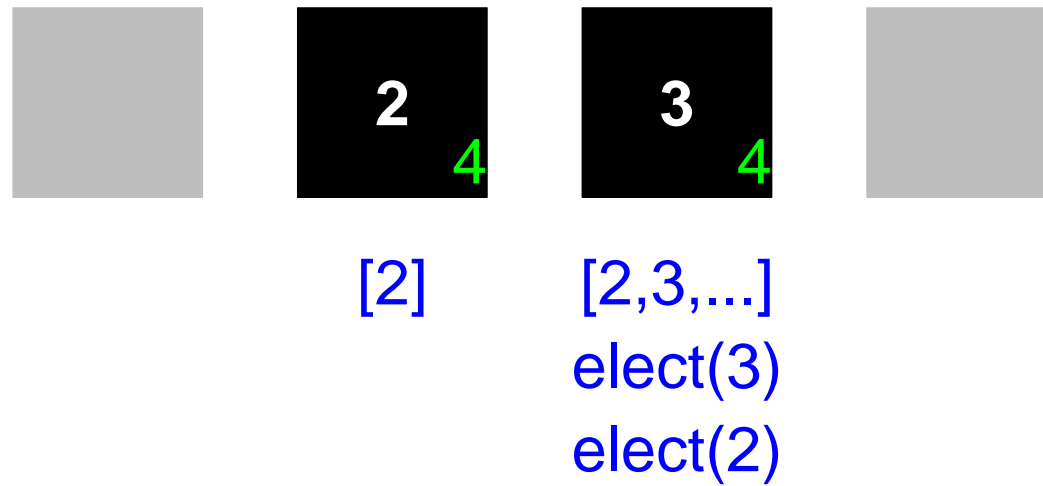
wait on 4

Ring Algorithm

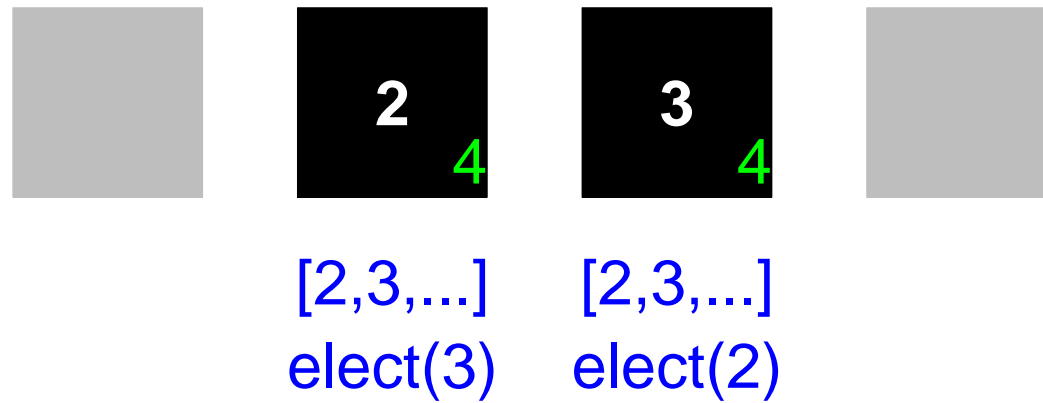


[2,...]
elect(2)

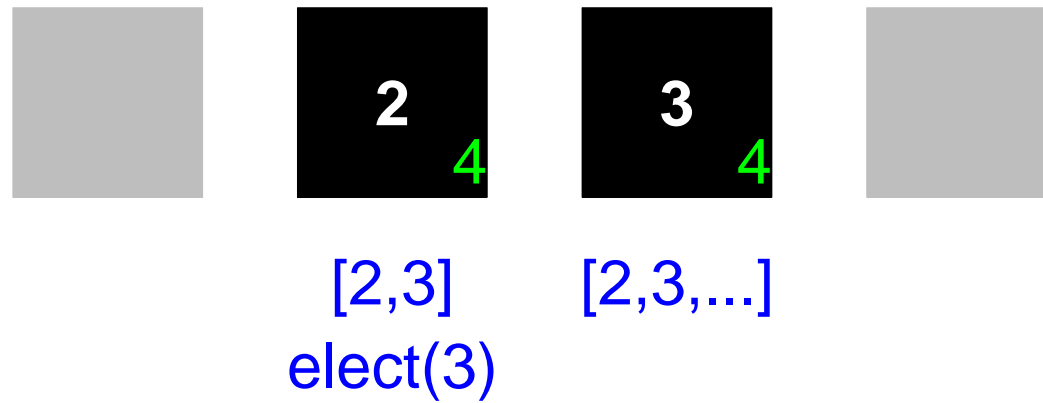
Ring Algorithm



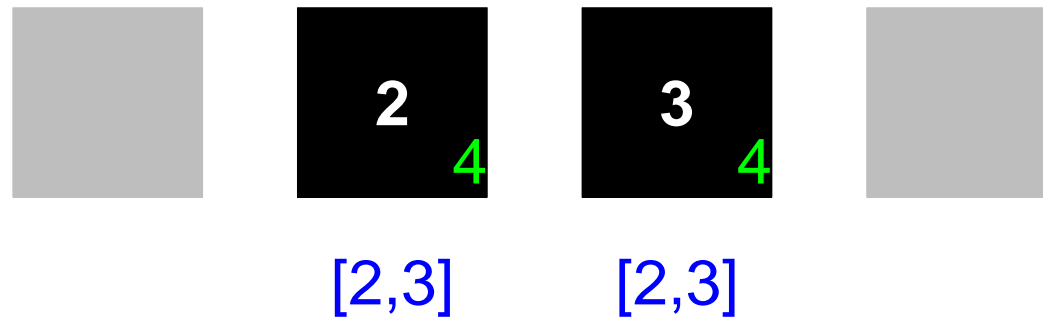
Ring Algorithm



Ring Algorithm



Ring Algorithm




Ring Algorithm



Multi-Site Atomicity

Multi-Site Atomicity



<buyer image>

<seller image>

<bank image>

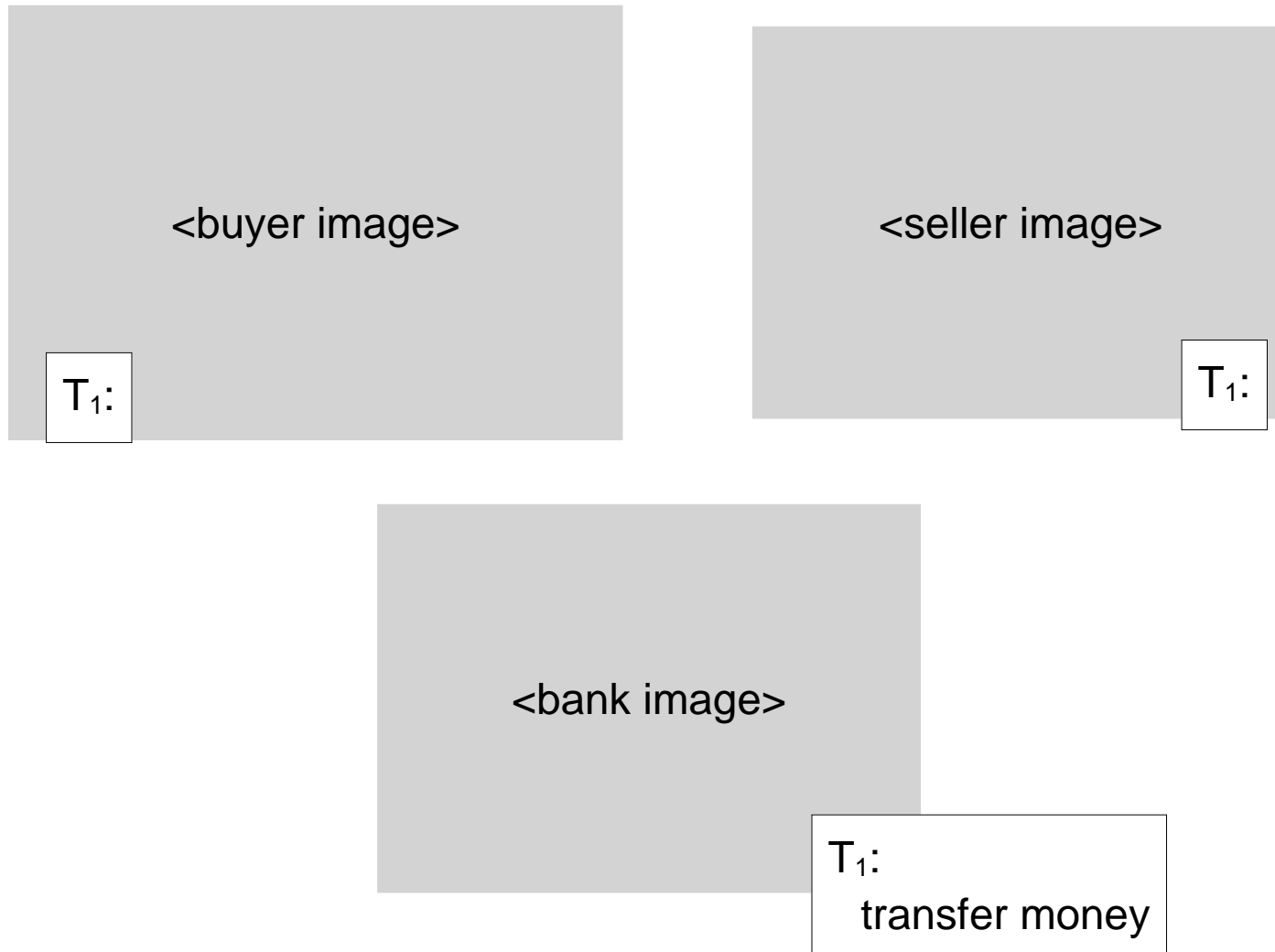
Two-Phase Commit



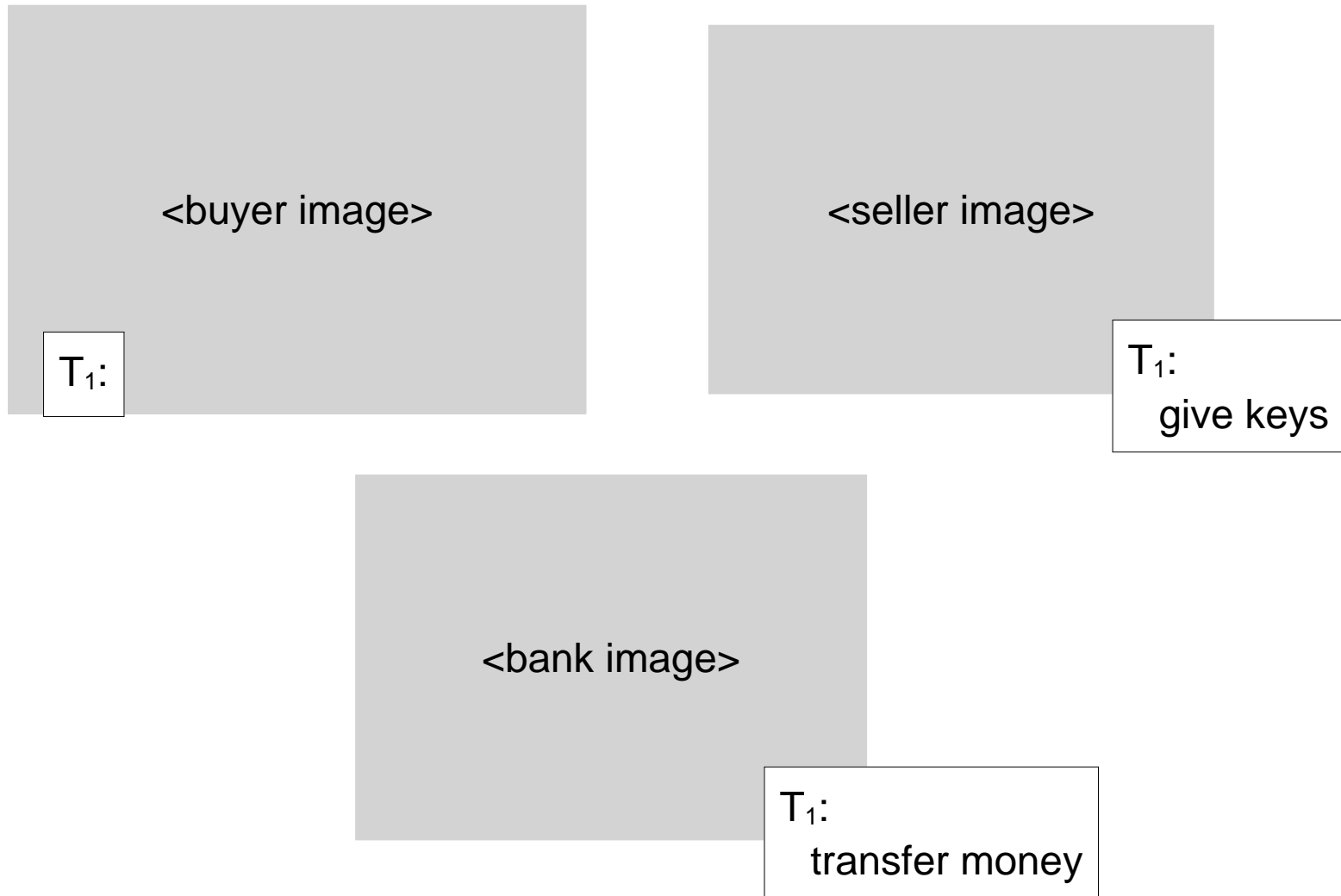
Two-Phase Commit



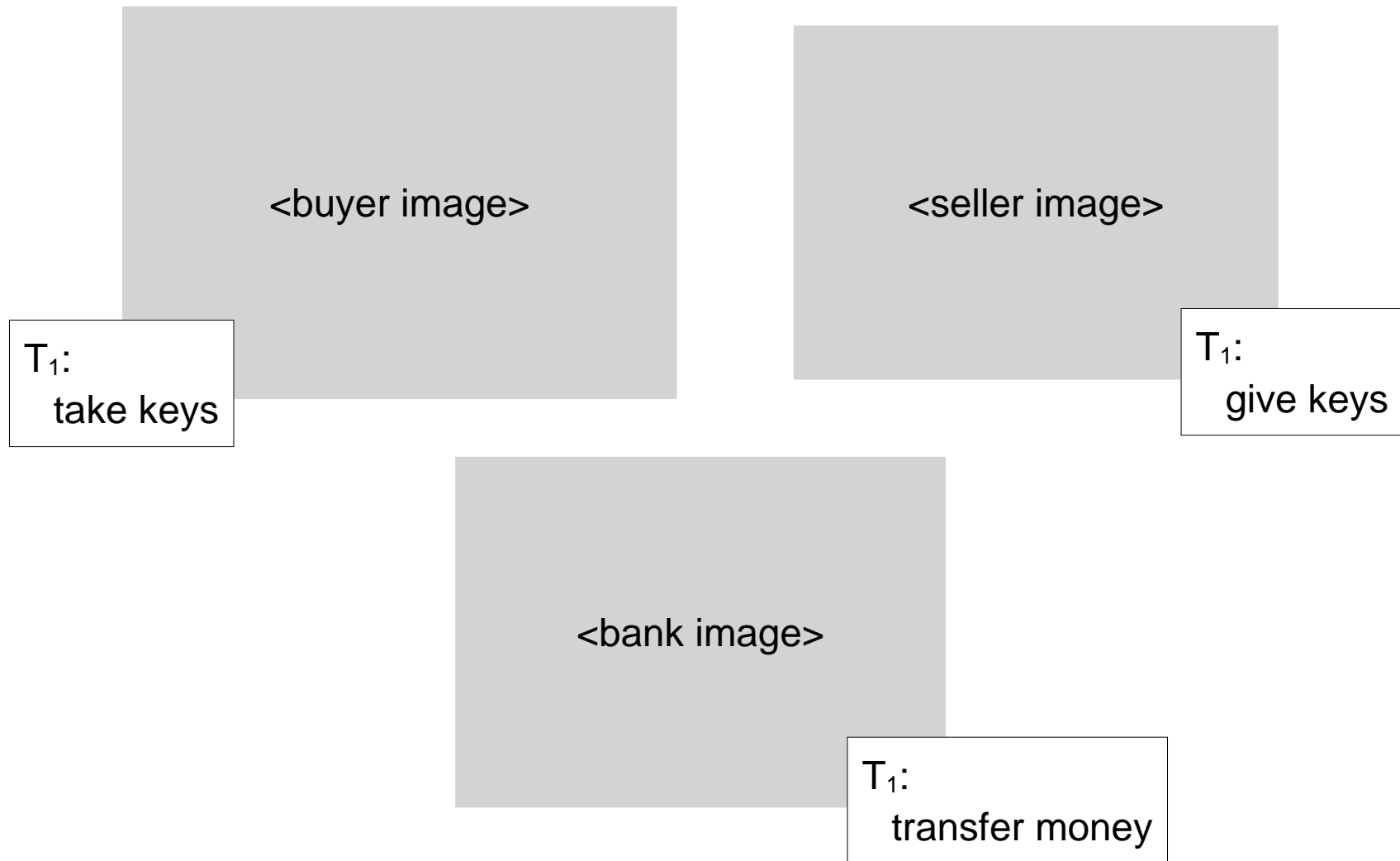
Two-Phase Commit



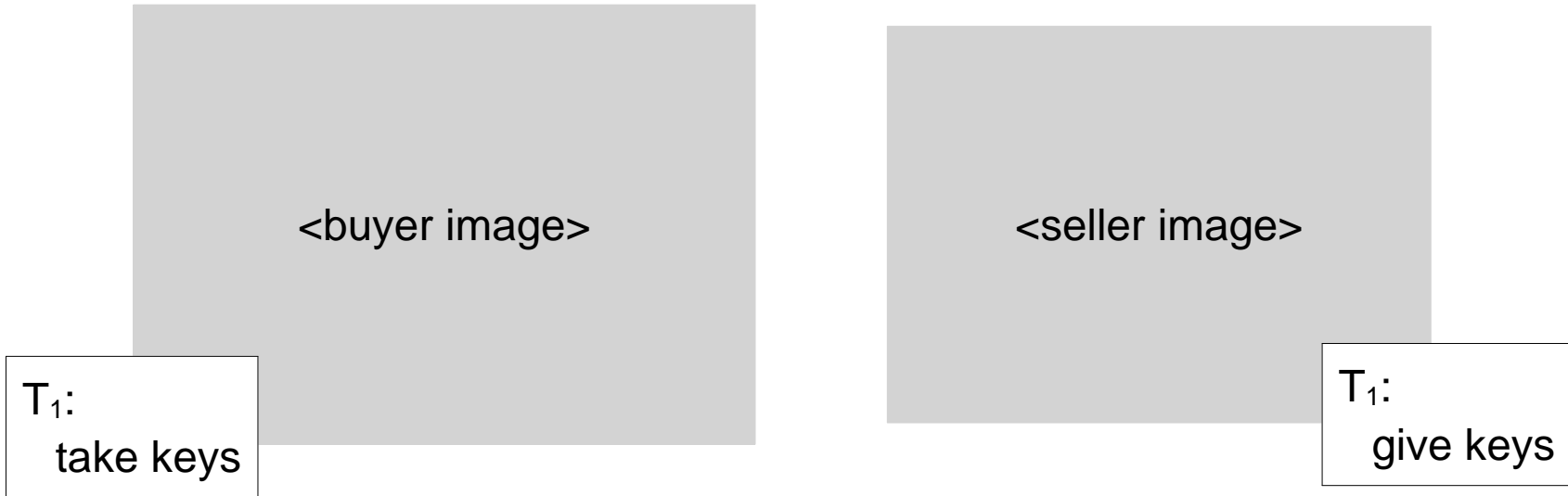
Two-Phase Commit



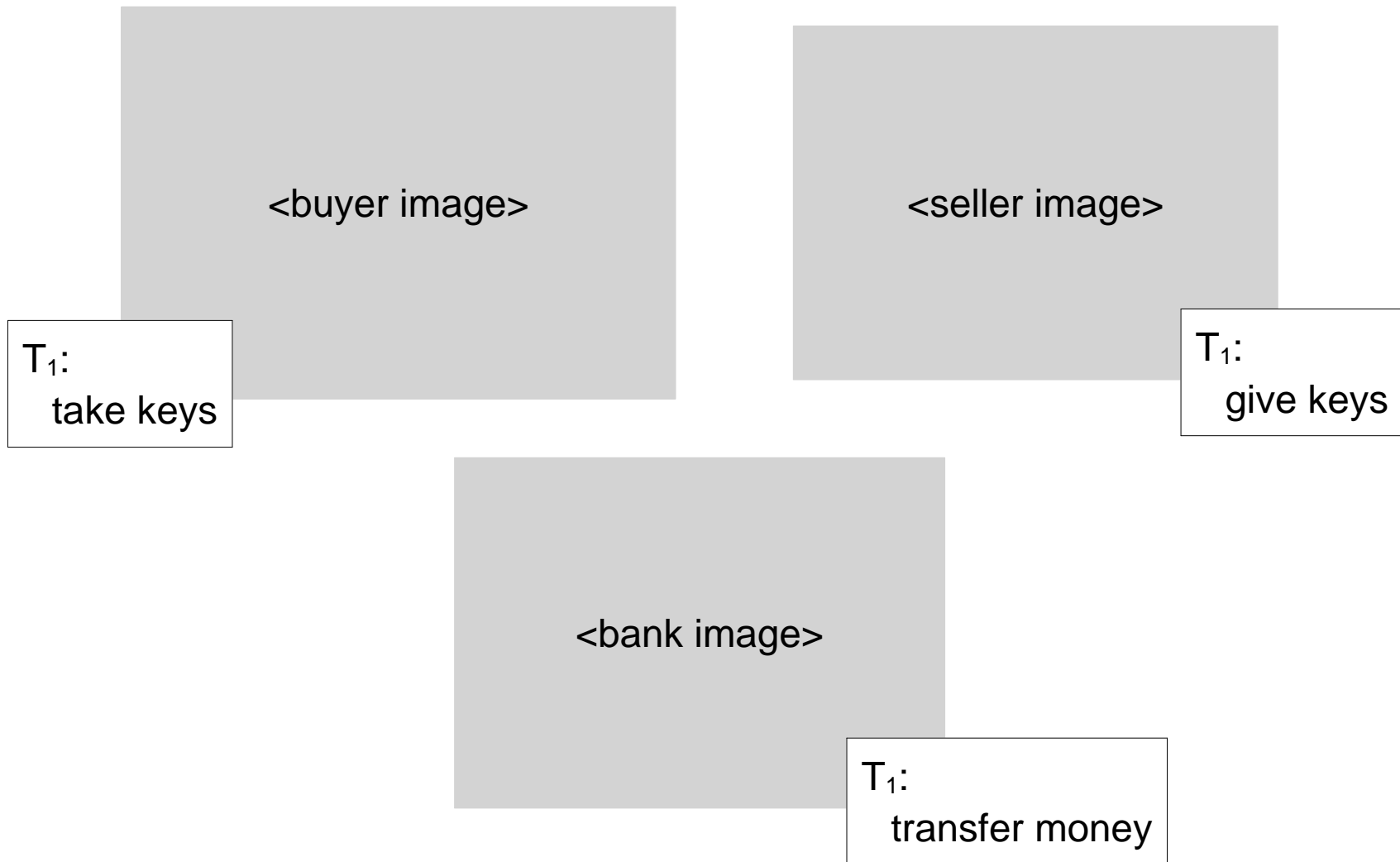
Two-Phase Commit



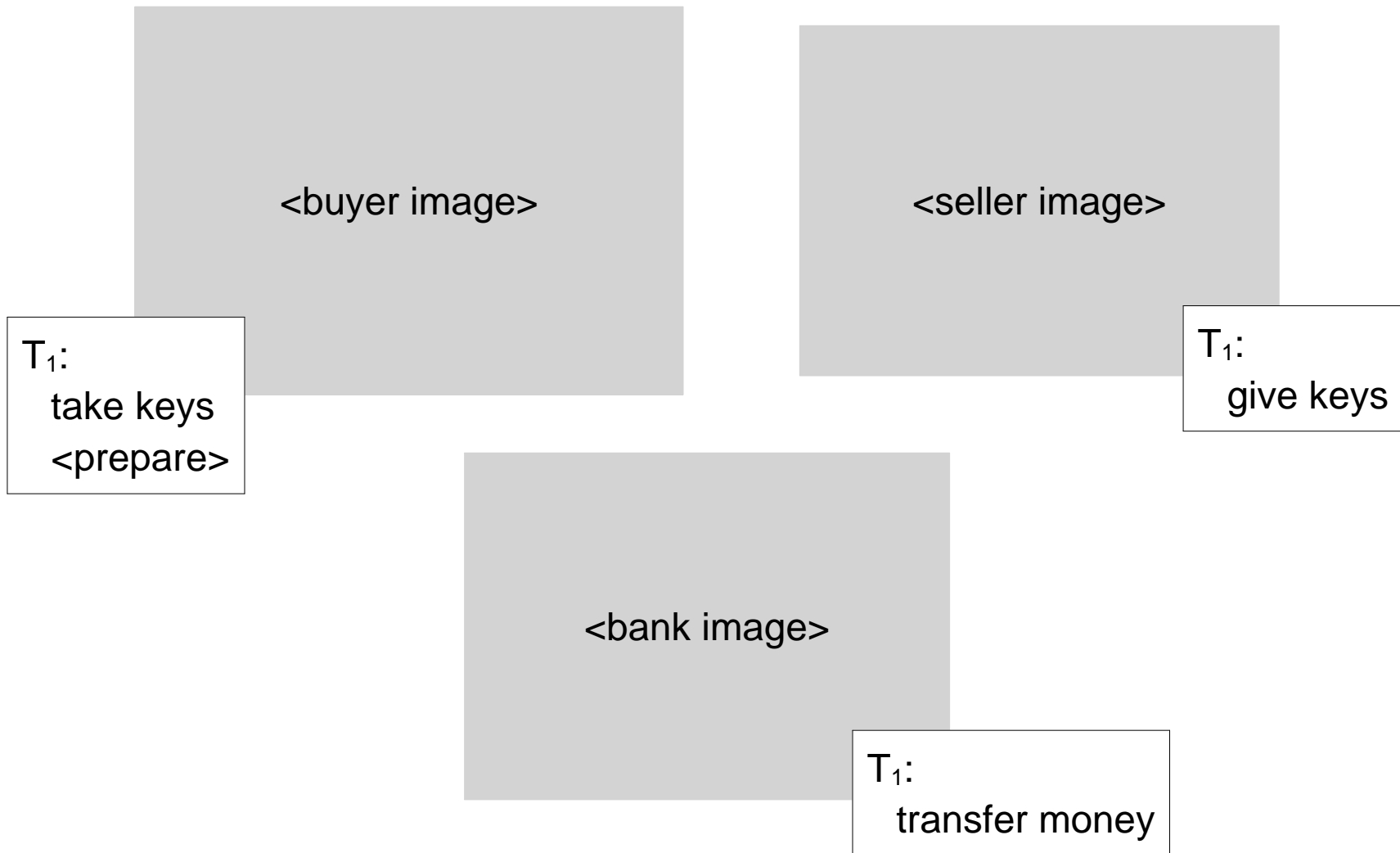
Two-Phase Commit



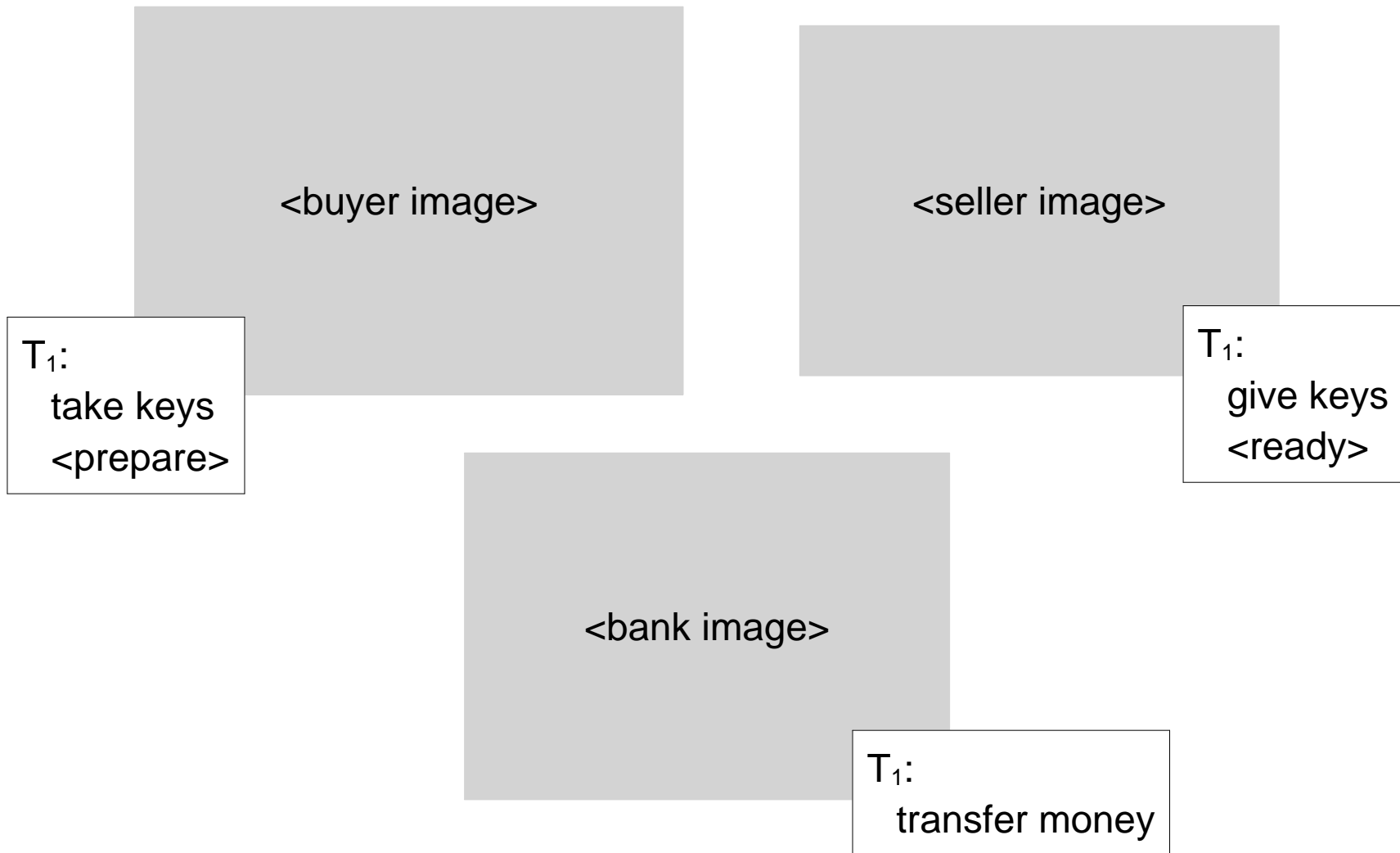
Two-Phase Commit



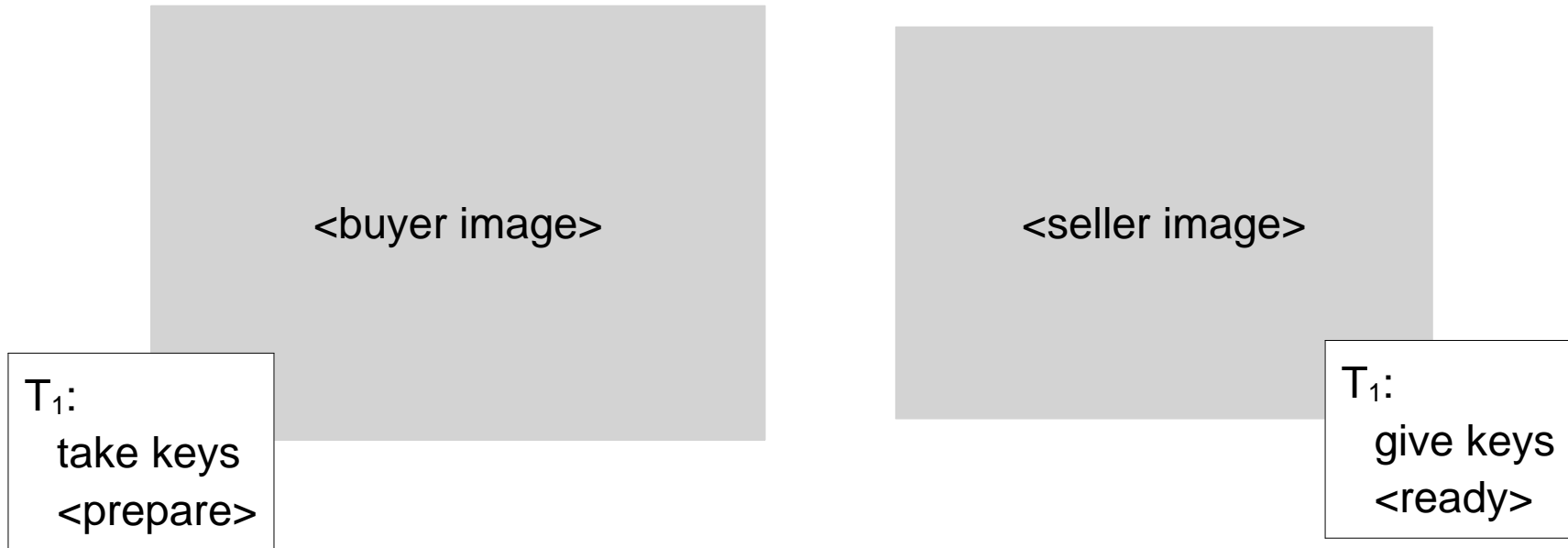
Two-Phase Commit



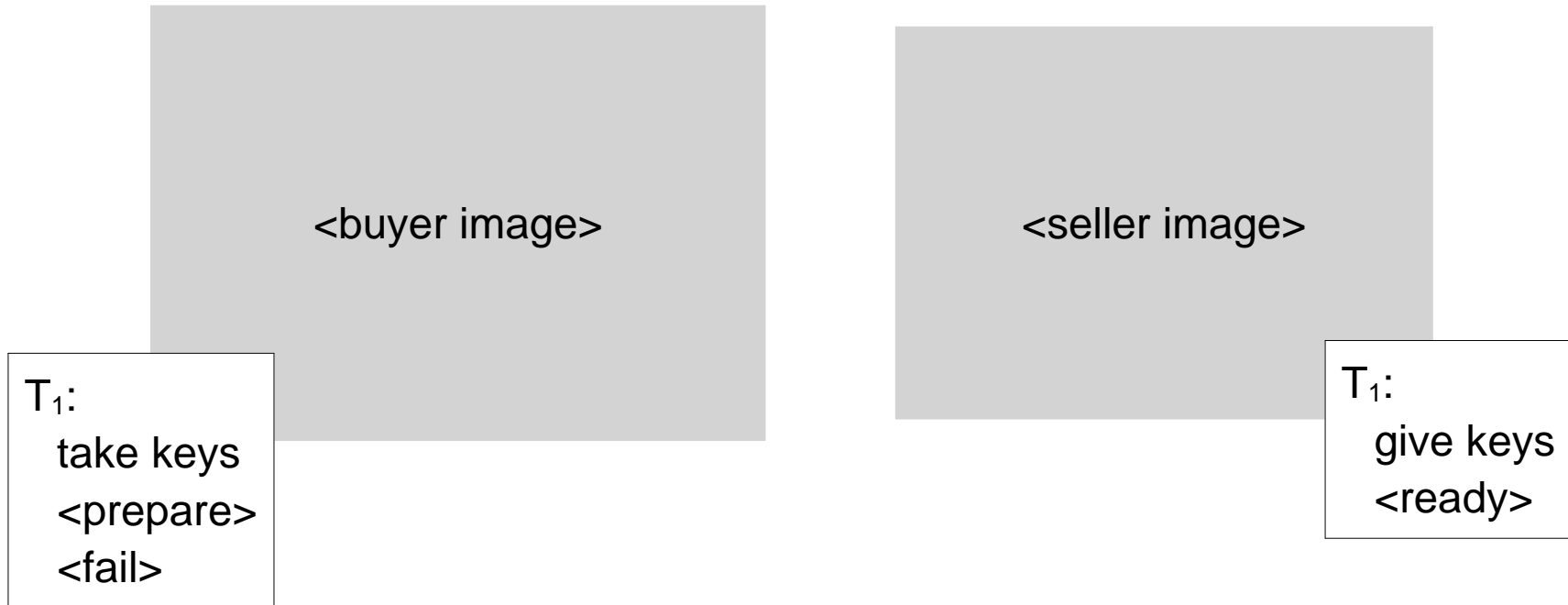
Two-Phase Commit



Two-Phase Commit



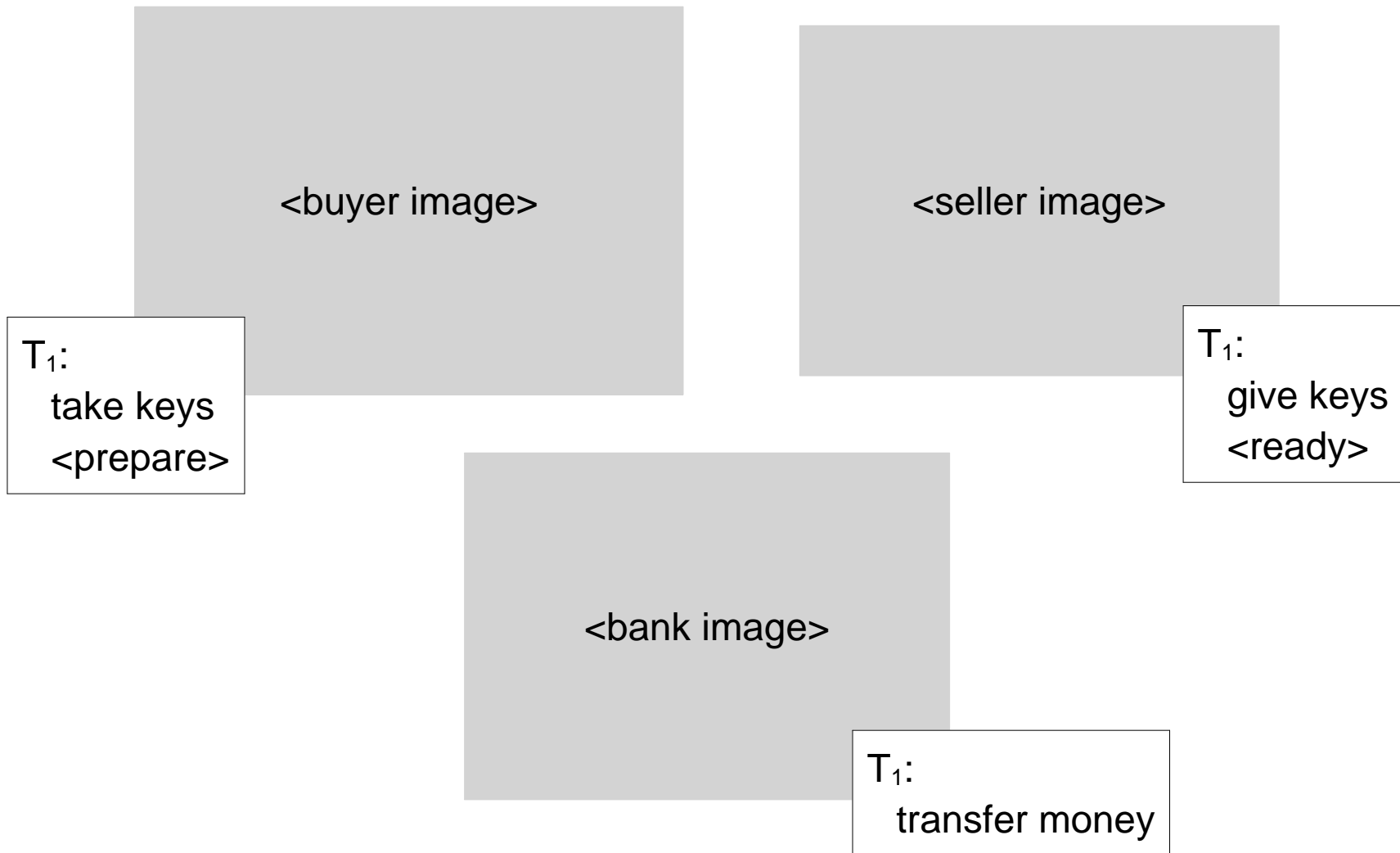
Two-Phase Commit



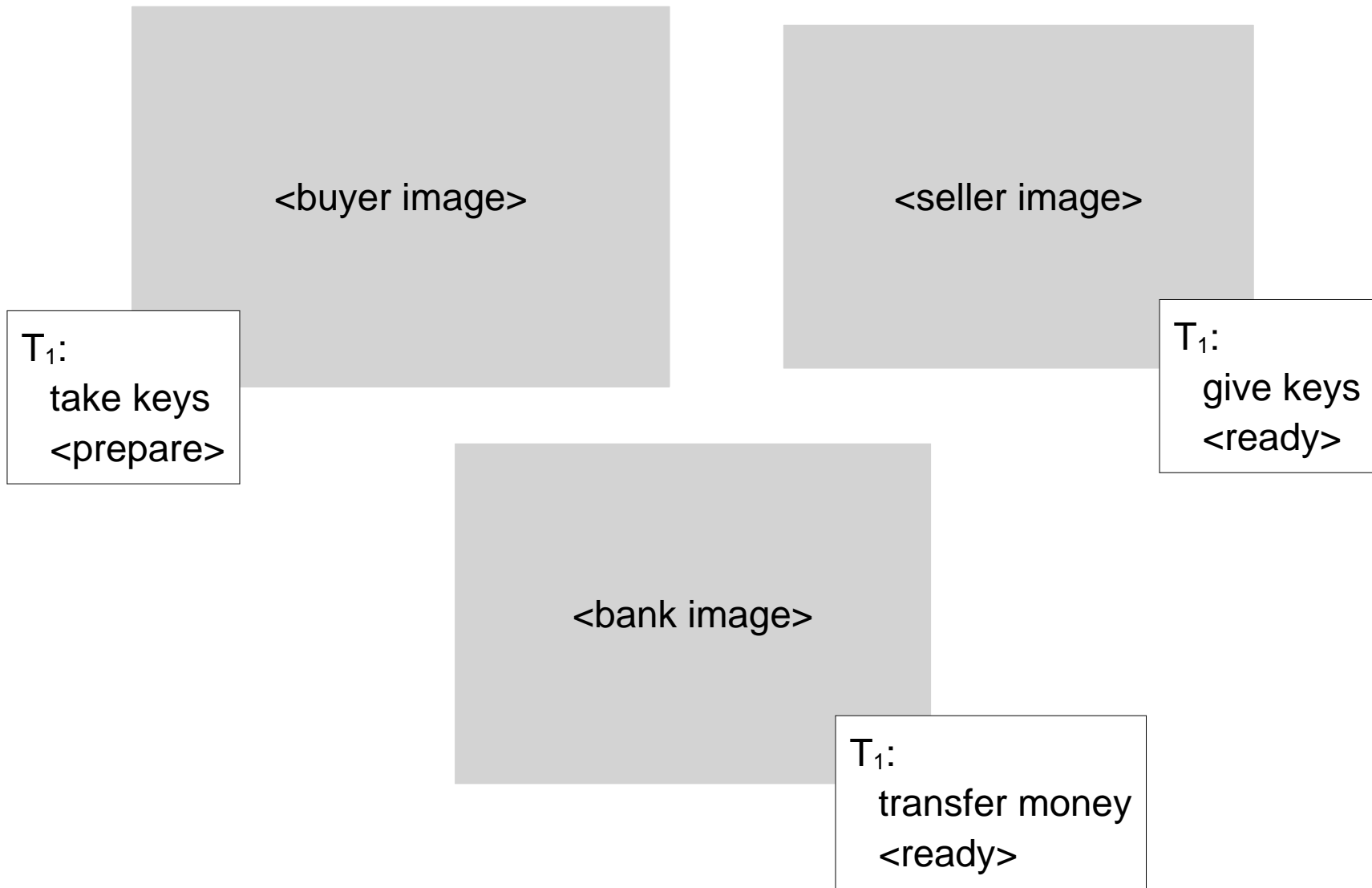
Two-Phase Commit



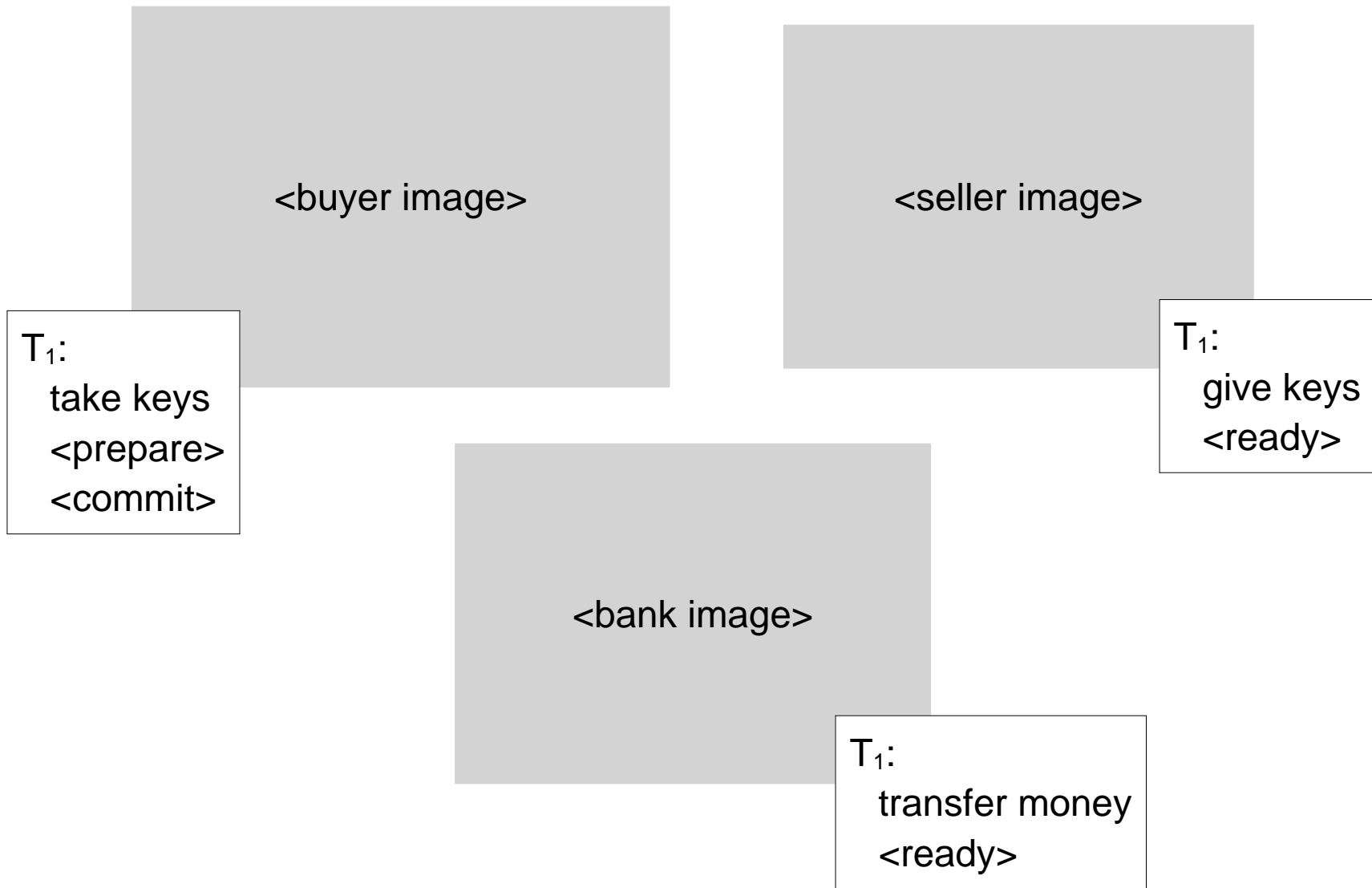
Two-Phase Commit



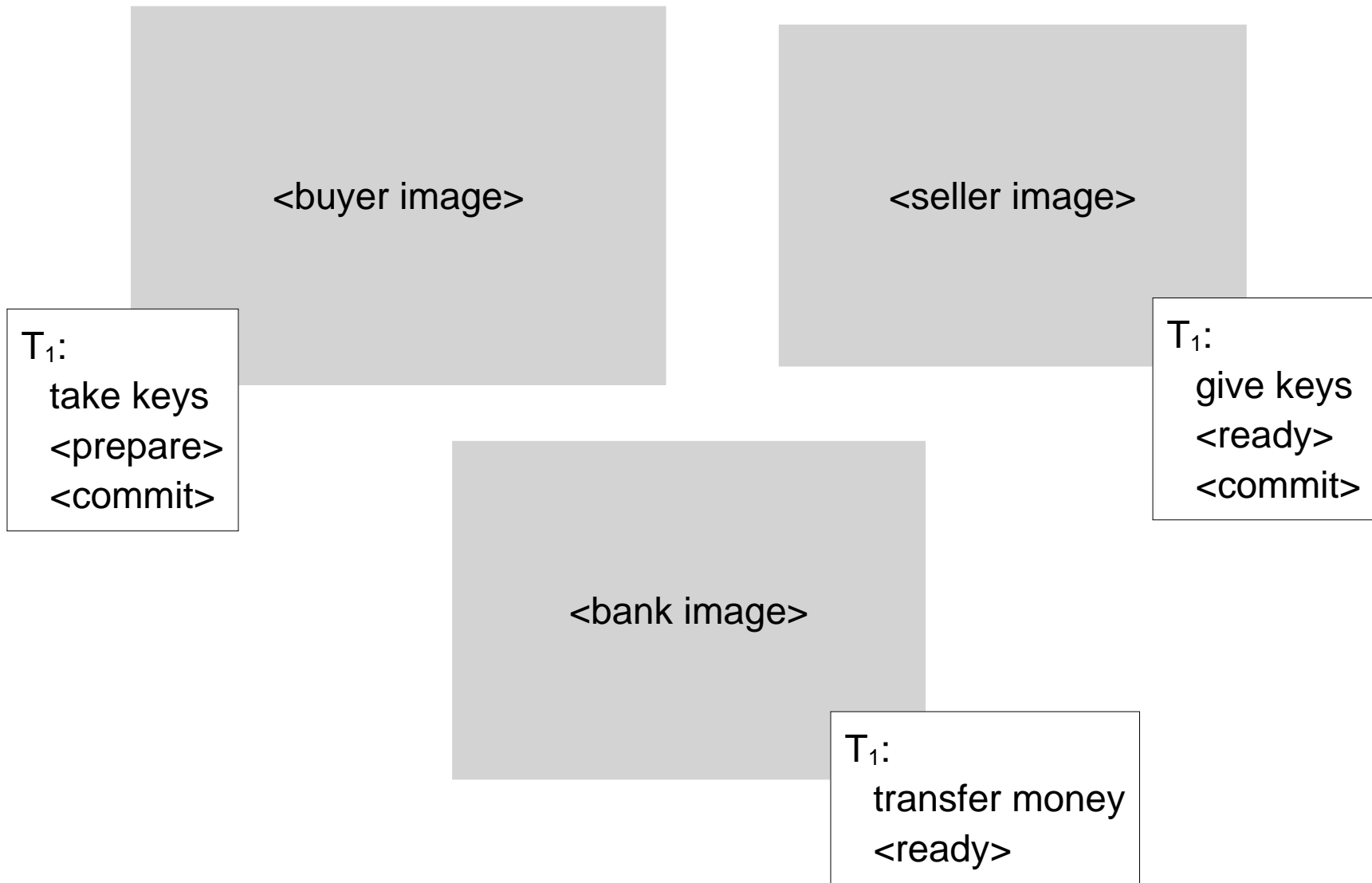
Two-Phase Commit



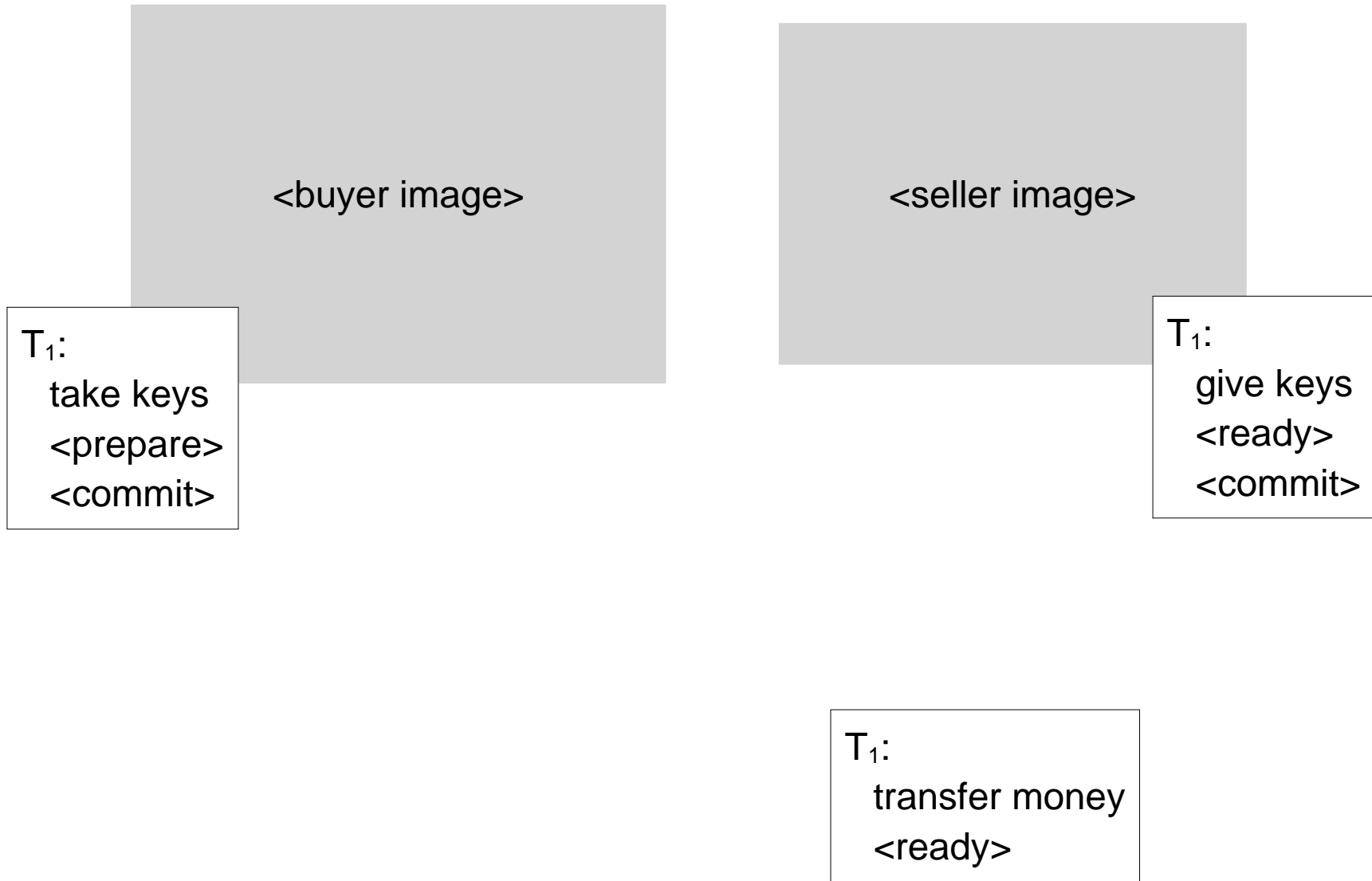
Two-Phase Commit



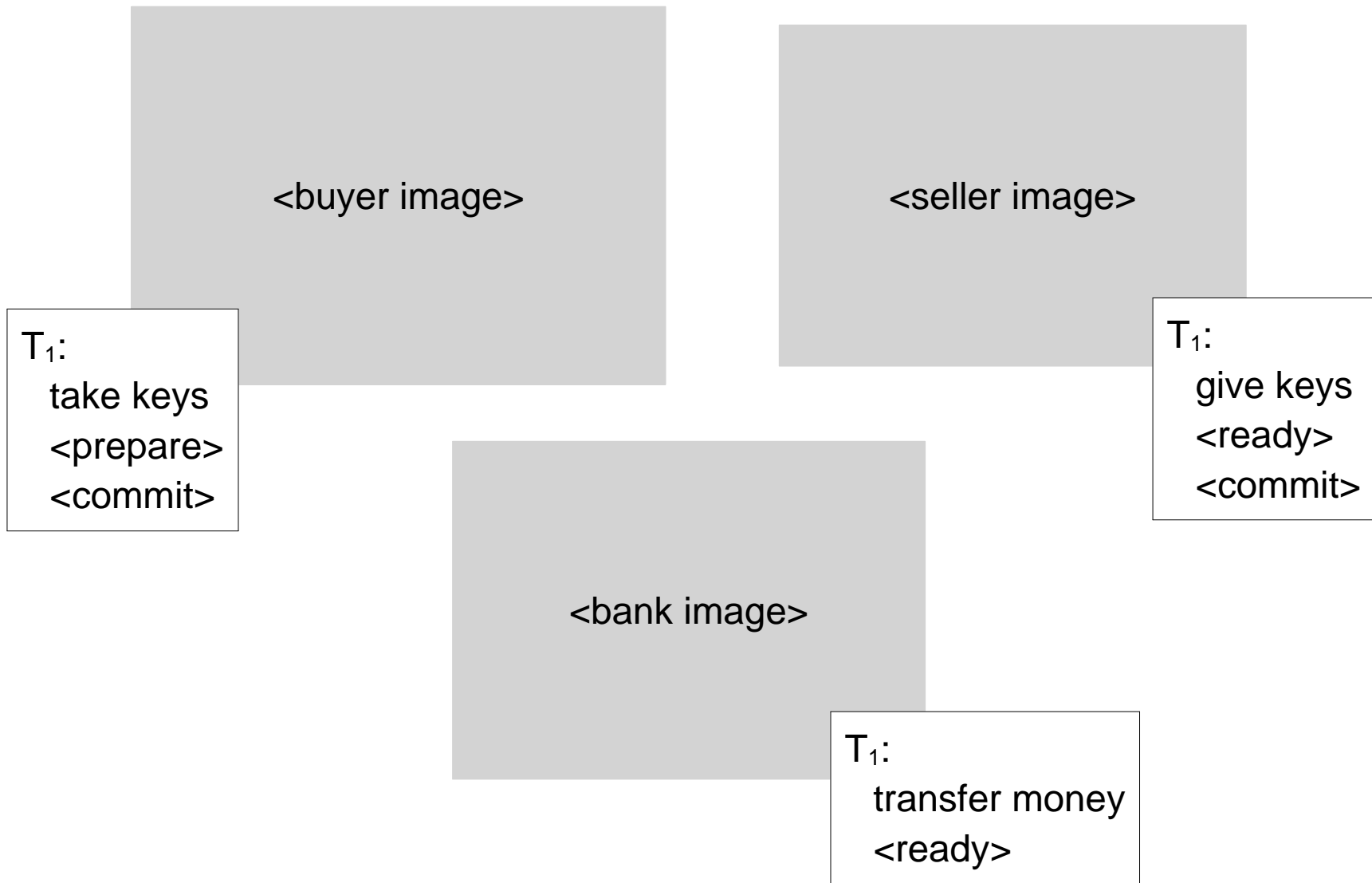
Two-Phase Commit



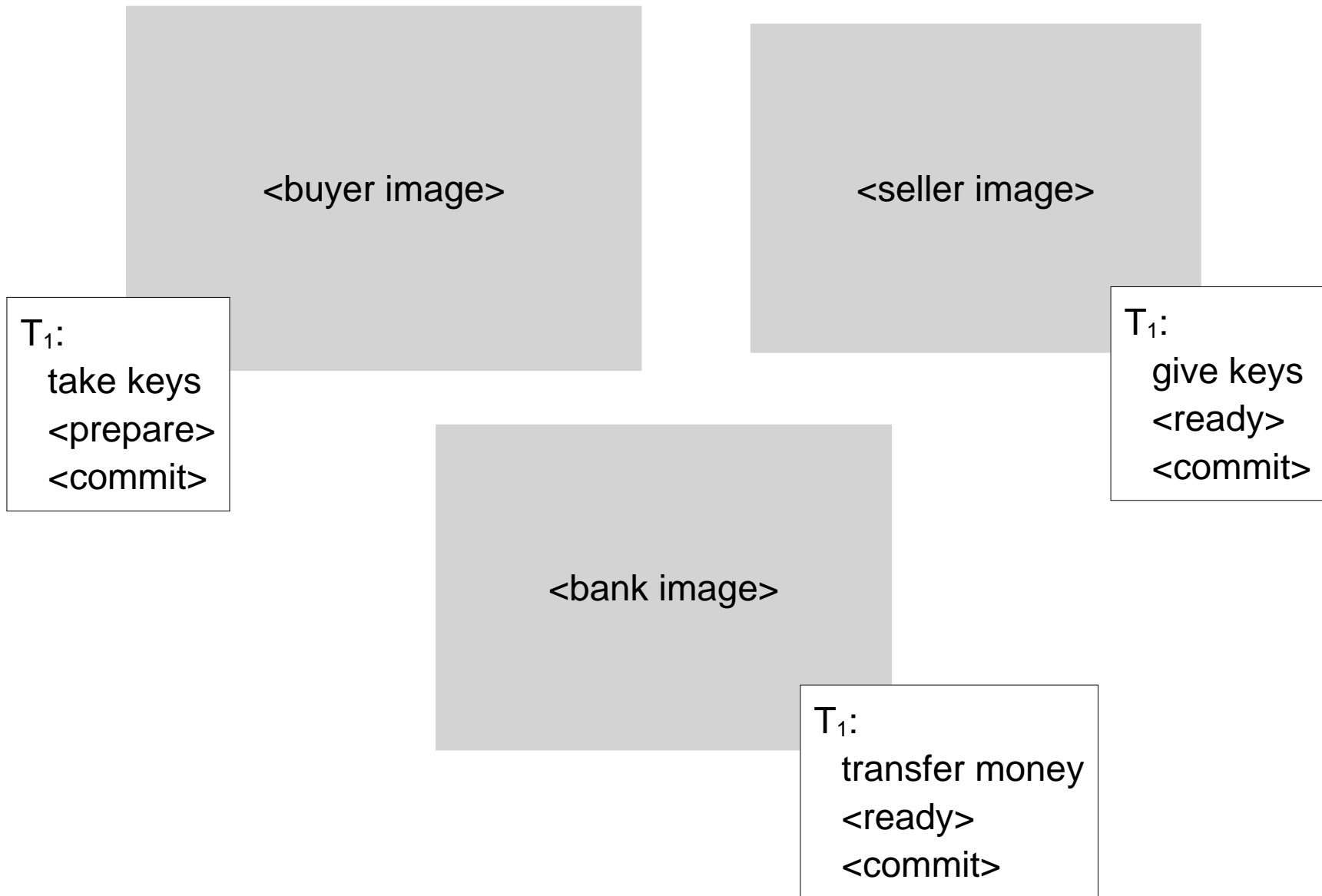
Two-Phase Commit



Two-Phase Commit



Two-Phase Commit



Things You Can't Do

- Two-way common knowledge

e.g., buyers commit only if they know that the bank received the message to finalize the transaction

- Handle too many ***byzantine*** failures

Can use voting to detect when up to $1/3$ of the sites are faulty, but not when more are faulty