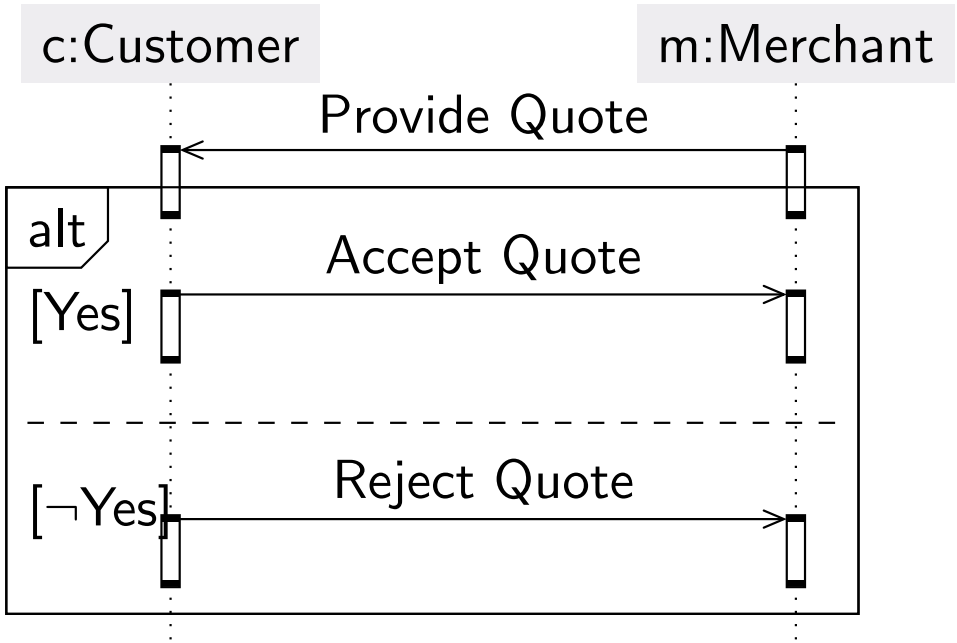
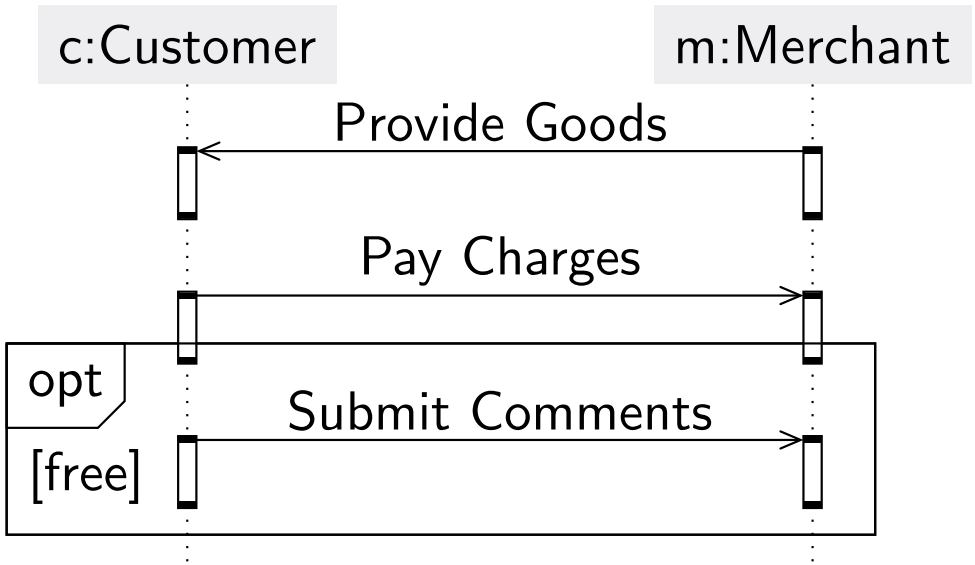
# The Alternative Block

Nondeterministially choose and execute any fragment whose guard is true



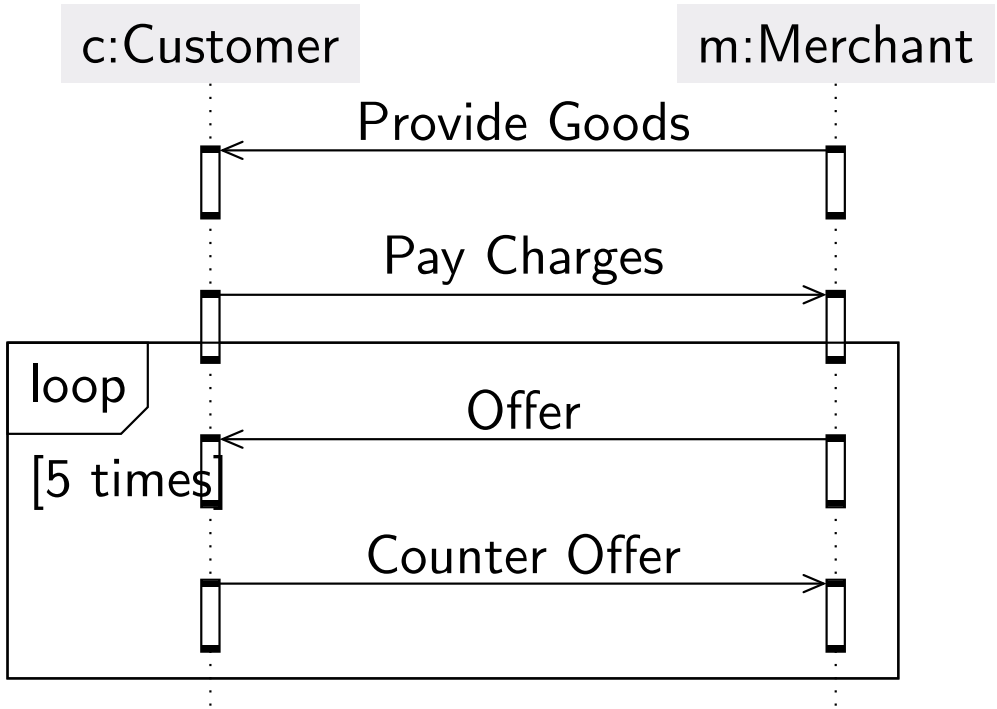
# The Optional Block

Modeling error here: Showing internal detail (free (spare time)) in a protocol



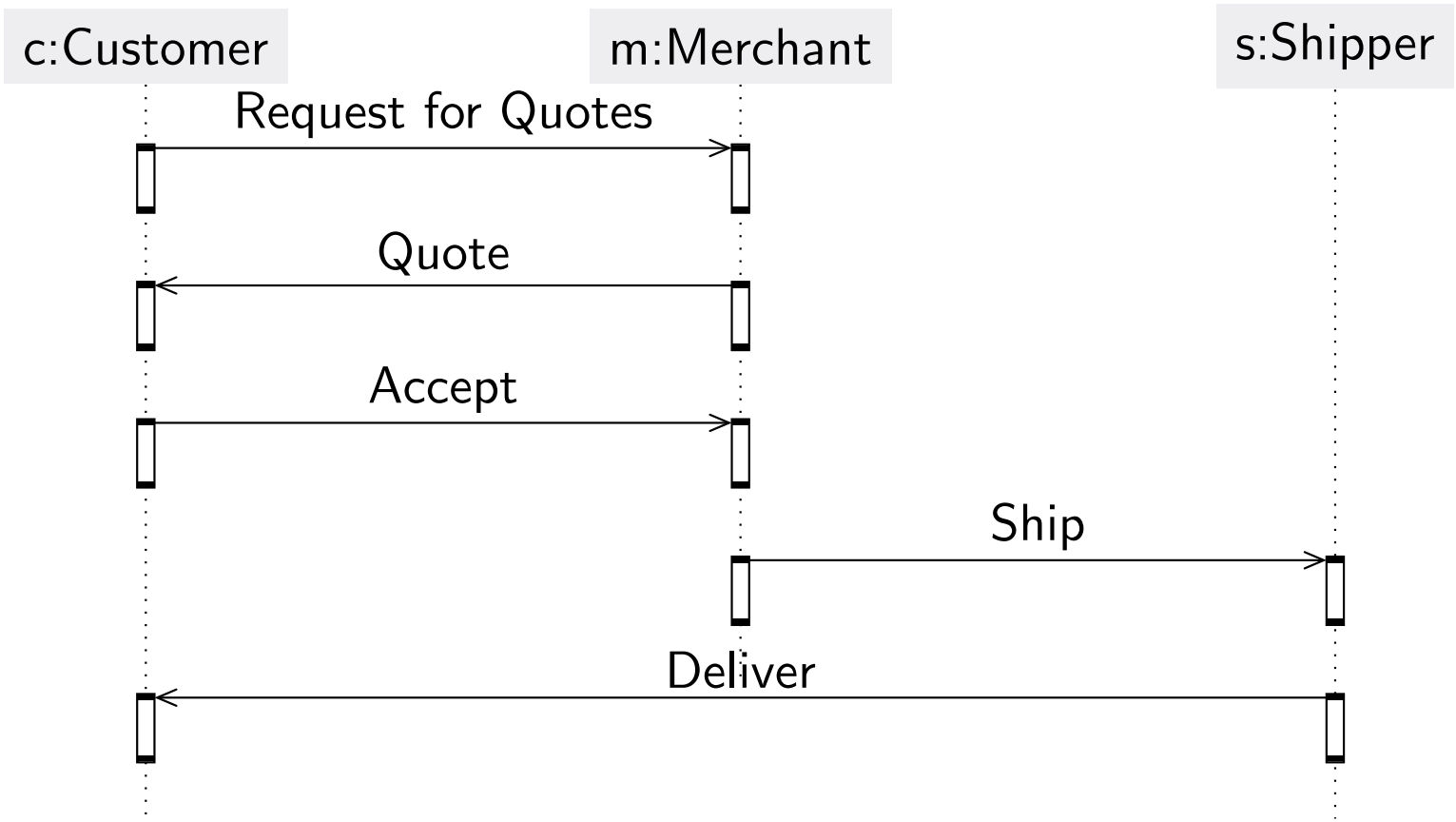
# The Loop Block

## Usually bounded in our examples

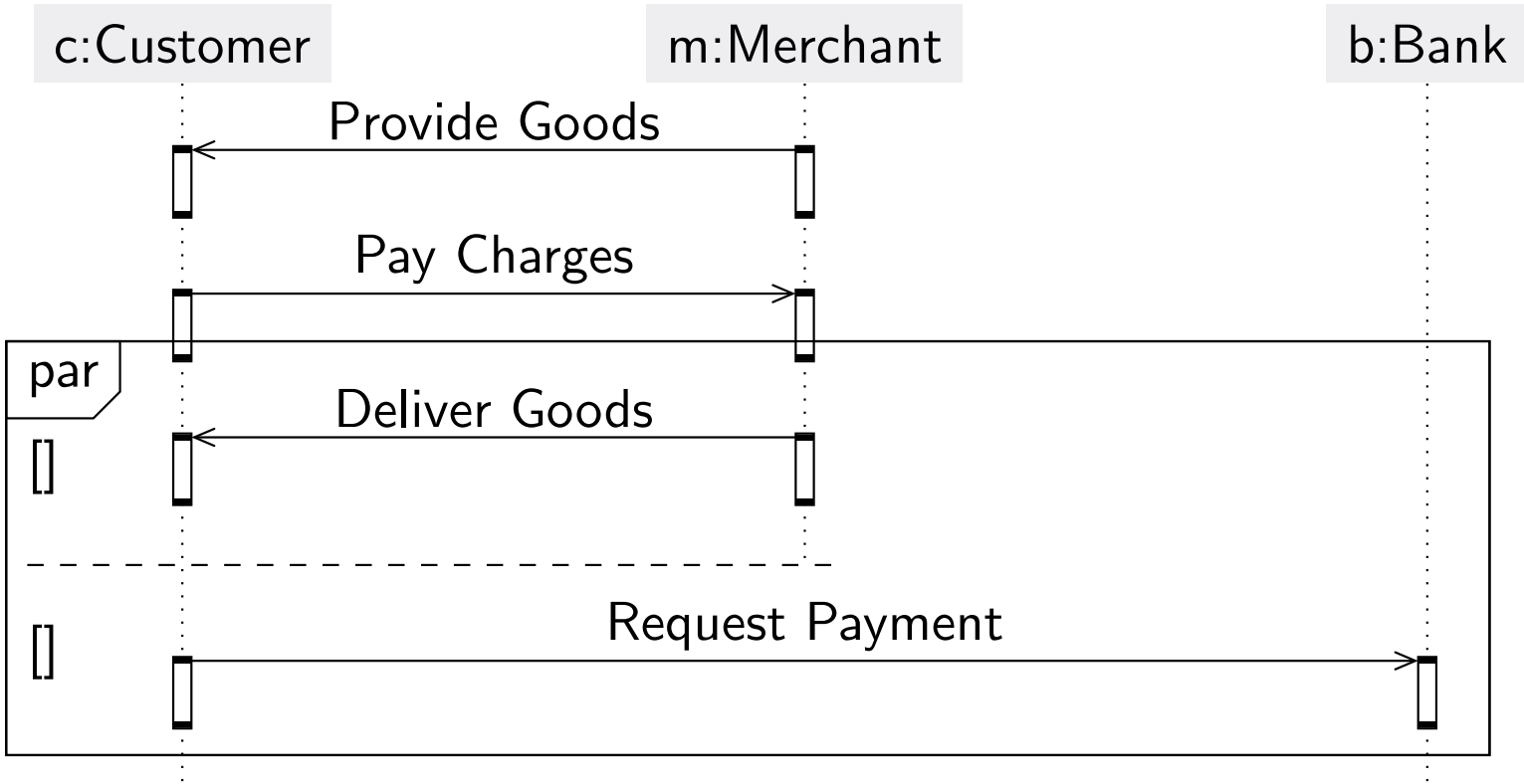


# Purchase (Just the Happy Path)

## Notice the hand off pattern, indicative of delegation



# The Parallel Block

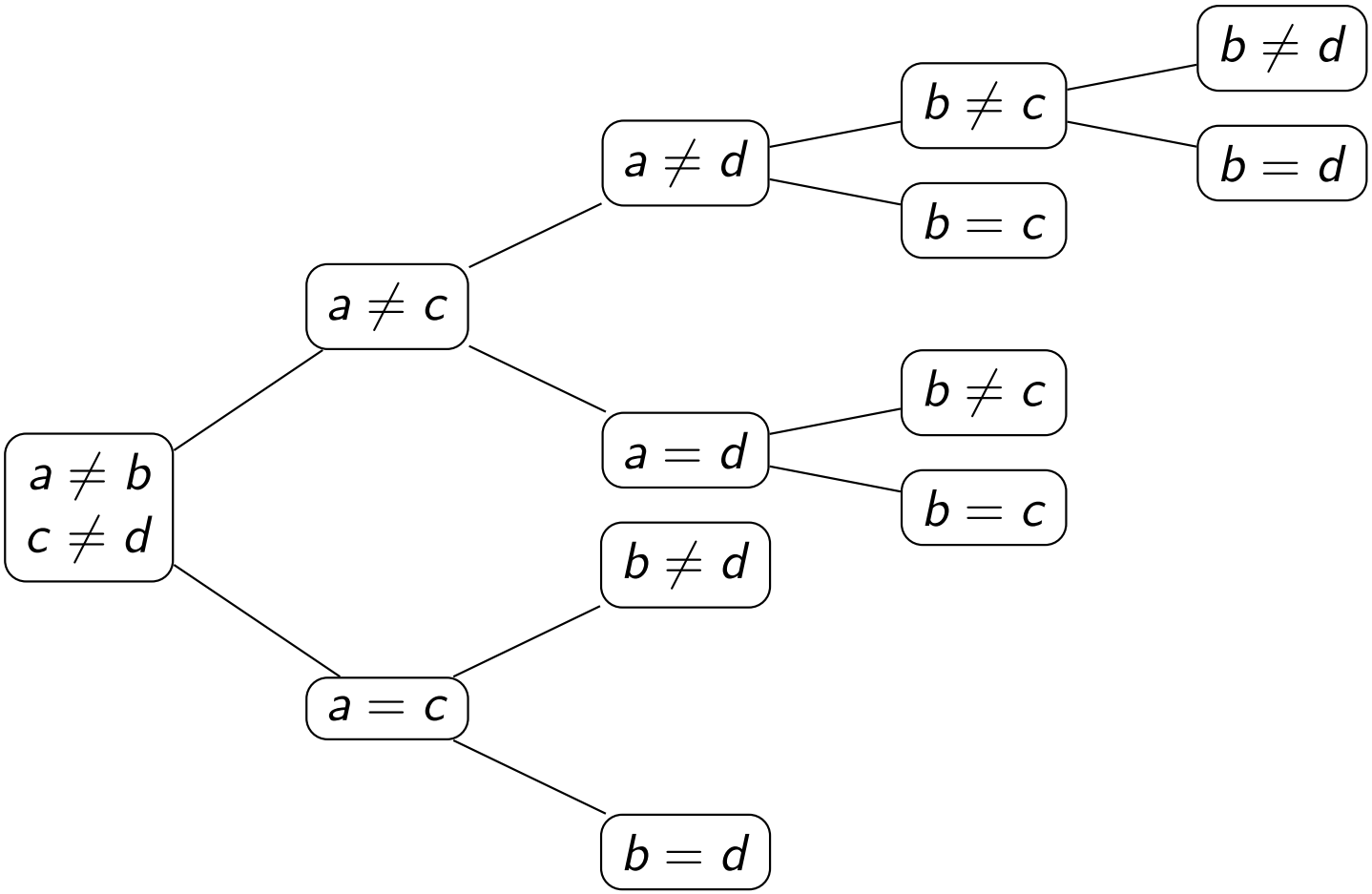


# Exercise: Diagramming Precedence

- Four roles: $A$, $B$, $C$, $D$ (could map to the same parties)
- Two messages: $m_{AB}$ and $m_{CD}$ (sender to receiver: distinct parties)
- We would like to assert that $m_{AB}$ precedes $m_{CD}$

# All Possible Sequence Diagram Structures

Given messages from $a$ to $b$ and from $c$ to $d$, seven are possible



# Exercise: Which of the Sequence Diagrams for Precedence are Compatible with Asynchrony?

Invariant outcomes regardless of relative execution speed, communication delays, and no global clock

# Exercise: Diagramming Occurrence and Exclusion

Use guards that refer to message occurrence

If $[m_{AB}]$ occurs then so does $[m_{CD}]$

- Four roles: $A$, $B$, $C$, $D$ (could map to the same parties)
- Two messages: $m_{AB}$ and $m_{CD}$ (sender to receiver)
- We would like to assert that
  - $m_{AB}$ excludes $m_{CD}$
  - $m_{AB}$ and $m_{CD}$ mutually exclude each other
  - $m_{AB}$ requires $m_{CD}$

# Properties of a (Point-to-Point) Message Channel

Can we take a system snapshot that violates any of these properties?
How can we achieve each property?

**Noncreative:** Must a message that is received have been sent by someone?

- Will a channel create messages?

**Reliable:** Must a message that is sent be received?

- Will a channel drop messages?

**Ordered:** Must the messages received from the same sender be received in the order in which they were sent?

- In which direction does the information flow?

**Global:** Must the messages received from different senders be received in the order in which they were sent?

- Called "causal" ordering in the literature but that term refers to potential causality

# Challenges to Correctness of Protocols

## Not specific to sequence diagrams

**Distribution:** different parties observe different messages, i.e., each lacks remote knowledge

**Asynchrony:** different parties observe messages in inconsistent orders

- Despite FIFO channels

- Intuitions about correctness
  - If each party interacts correctly, is the overall behavior correct?
  - If not, our sequence diagram is not *realizable* or *enactable*
  - Is the design of each party obvious?
  - Does the design of the parties preclude some legal enactments?

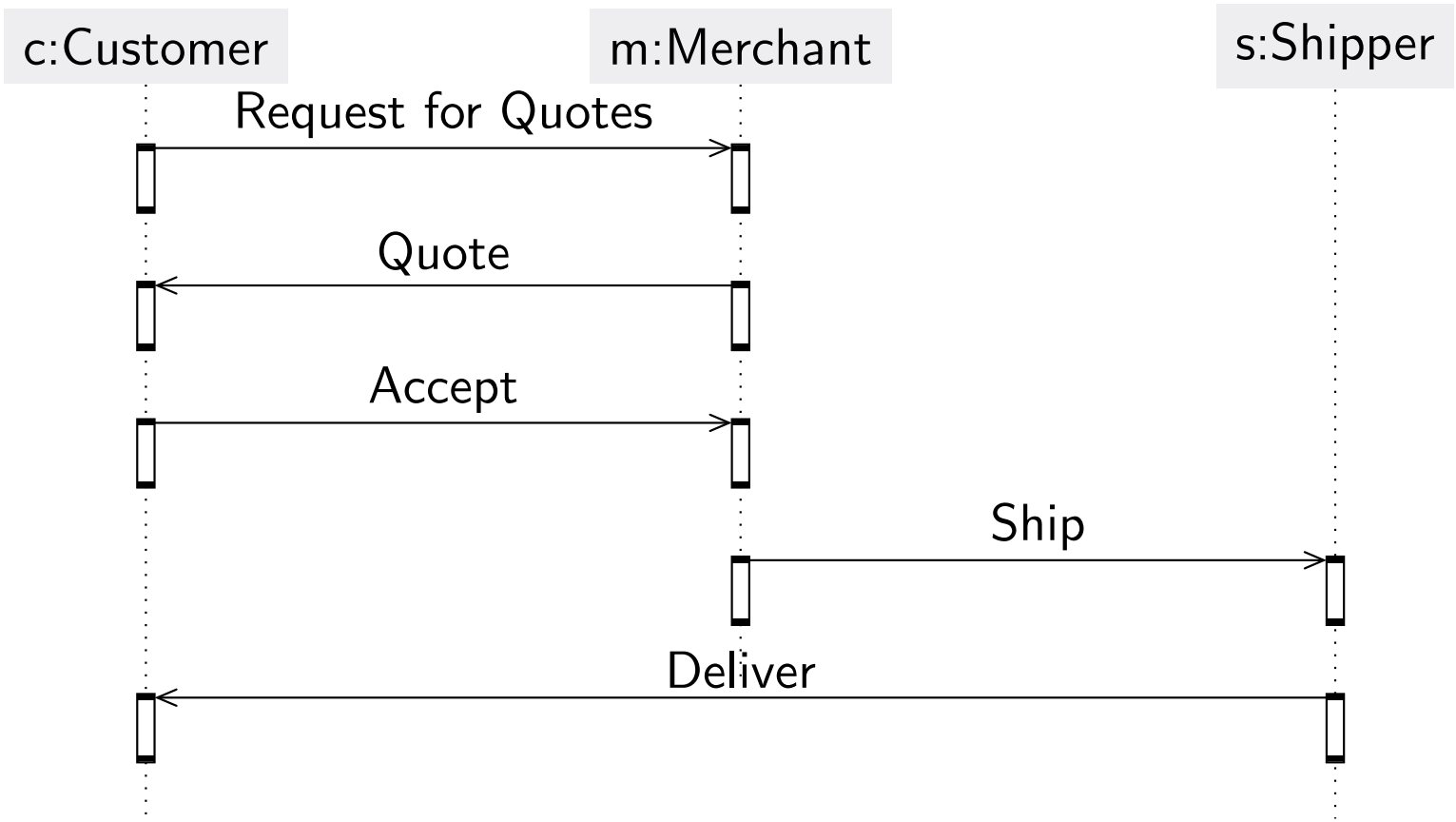
# Business Protocols

Interactions among autonomous parties, understood at the business level

- *Conversation*: An instance of a protocol
- Operational representations: steps taken
  - Procedural
    - Sequence diagrams
    - State diagrams
    - Activity diagrams
    - Petri Nets
  - Declarative
    - Temporal logic
    - Dynamic logic
    - Information-based specifications
- Meaning-based representations: underlying business transaction
  - Declarative, if captured formally at all
    - Commitment machines
    - Constitutive specifications

# Exercise: Identify the Public and Private Components

## Process = Protocol + Policies



# Exercise: How Might we Modularize Protocols?

## Consider Purchase

# Modular Business Protocols

- Identify small, well-defined interactions with clear business meanings
- Improve flexibility and concurrency
- Possibly lead to invalid executions
- How can we ensure good properties despite modularity?
  - Begin from a constraint language
  - Standardize modular fragments as patterns, e.g., RosettaNet

# Sequence Diagrams for Business Modeling

## No!

- No internal reasoning
  - No private predicates in guards
- No method calls
  - No self calls
- No synchronous messages
  - No business puts itself on indefinite hold waiting for its partner to proceed
- No causally invalid expectations
  - No *nonlocal* choice
    - No nonlocal choice that matters
  - No control of incoming message occurrence or ordering
  - No dependence on occurrence or ordering of remote message emission or reception
  - No reliance on ordering across channels
    - No reliance on ordering within a channel unless warranted