Multiagent Middleware for Application Semantics

The Holy Grail of Distributed Systems

Amit K. Chopra Samuel H. Christie V

Lancaster University

Usenix OSDI '23

Application as System of Autonomous Components

Decentralized; Communication via asynchronous messaging

Microservices; Any application involving multiple real-world principals

How to accommodate autonomy?

How can components make decisions flexibly?

Multiagent Systems: Communication semantics

Model message meaning in terms of the *social objects* computed

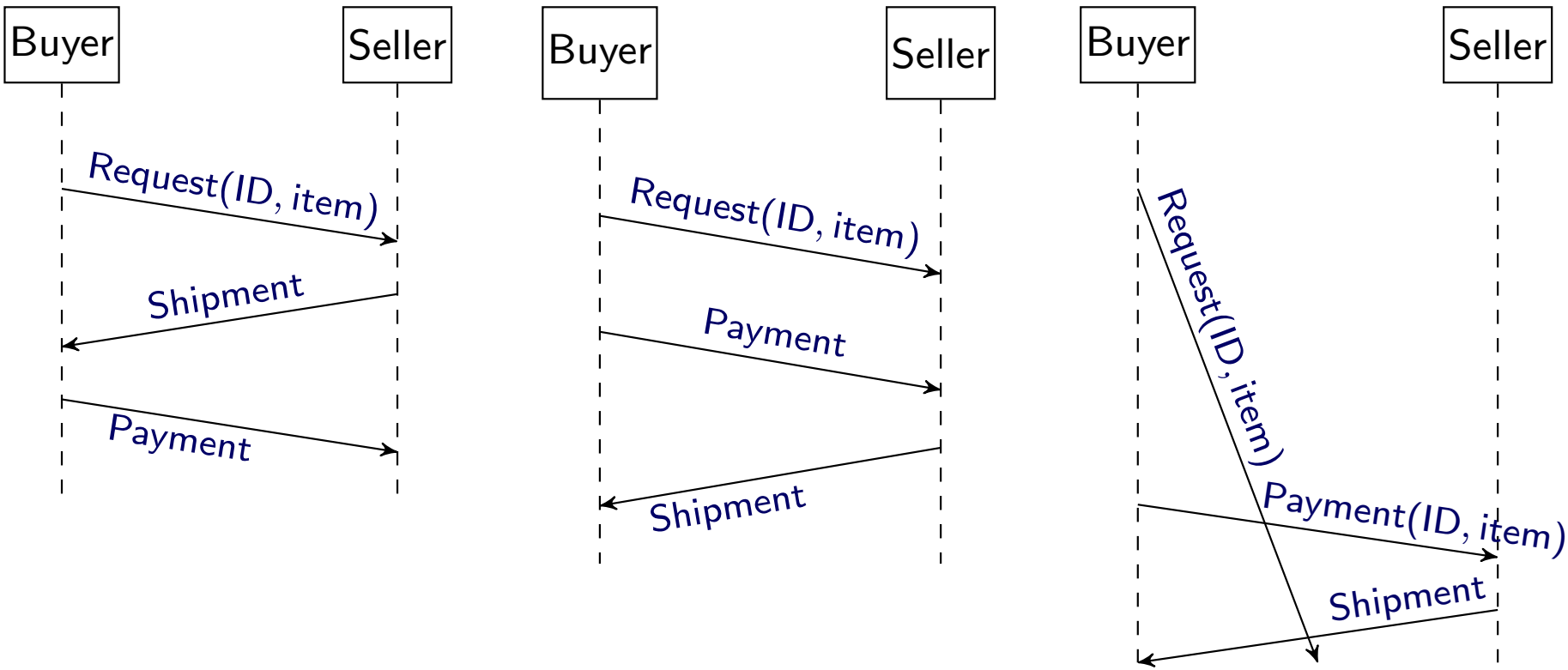
Distributed Systems: Antiautonomy

- Mostly ignores communication semantics
- Mostly ignores the end-to-end principle
- Stuck in the client-server paradigm
- Excessive hidden synchronization

Applications as Information Protocols (Munindar Singh)

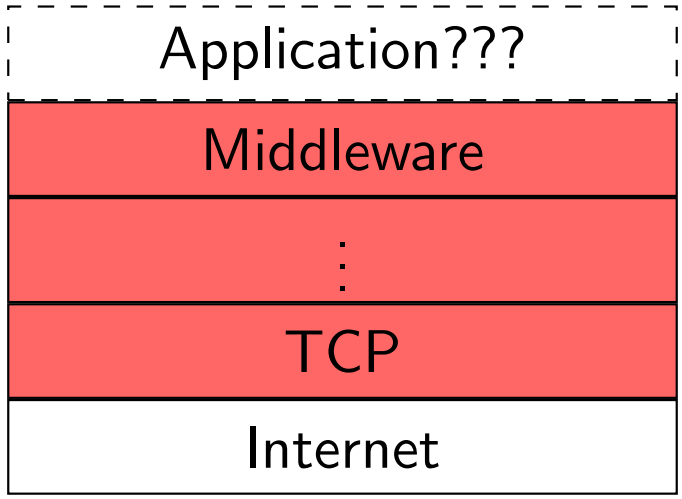
Specify true causality; forget potential causality

```
Purchase {
 role B, S
 parameter out ID key, out item, out status, out token
 B ↦ S: Request[out ID, out item]
 S ↦ B: Shipment[in ID, in item, out status]
 B ↦ S: Payment[in ID, in item, out token]
}
```



Current Systems Wisdom: Application MIA

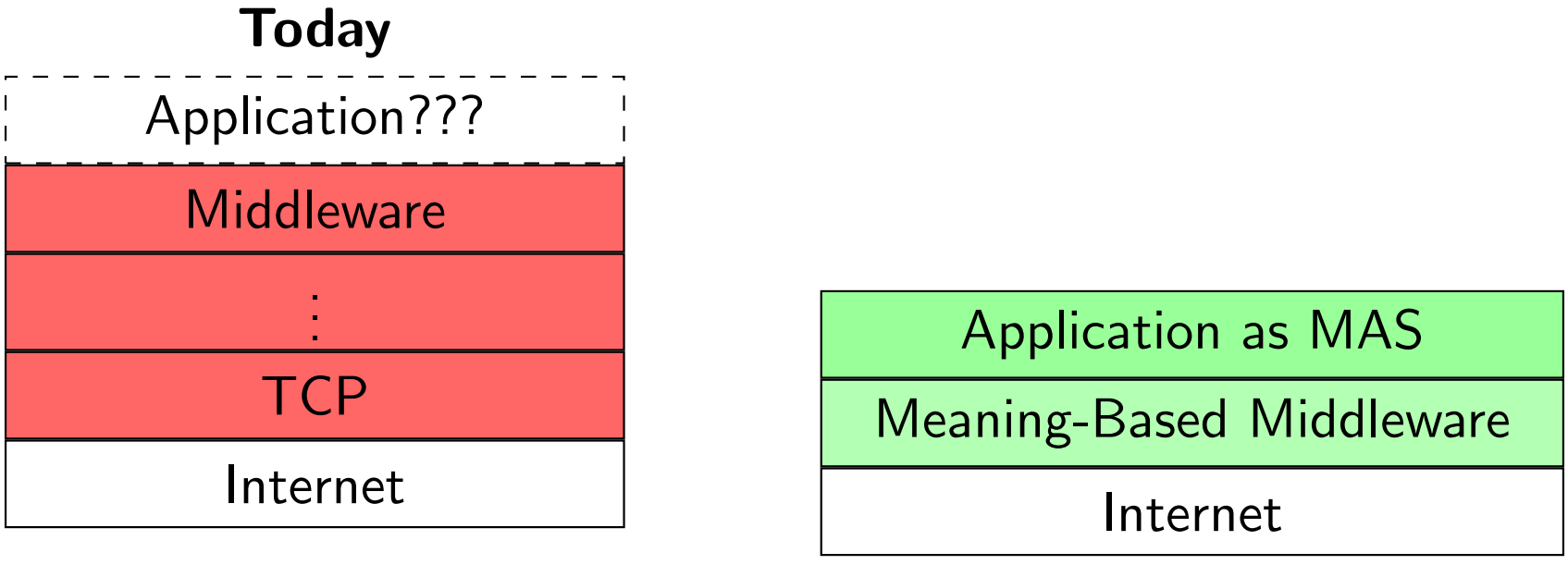
Application chooses from a palette of ordering and reliability guarantees in communication services



Ordering hits flexibility	FIFO blocks receipt of Payment until Request is received
Ordering inadequate	FIFO works only between two endpoints
Reliability superfluous	Once Buyer sends Payment, it may not care if Request is delivered
Reliability inadequate	Even if delivery is guaranteed, Buyer may retransmit Payment
Costly!	

Transformative: Multiagent Advances in Meaning

Don't need the complex stuff built into the stack over the last 50 years!



Research Program

Show that the meaning-based middleware enables conveniently building correct, flexible, loosely-coupled, fault tolerant, and high-performance distributed applications

Kiko: Middleware-Supported Programming Model

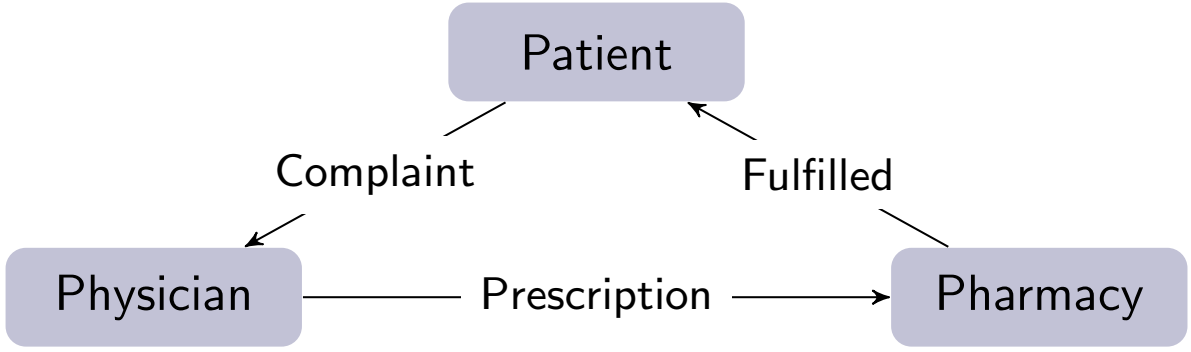
Outshines CNCF's Dapr

```
TrafficControl {
    ...
    EntryCam ↦ Collector: Entered[out entryID key,
        out regID, out entryTS]
    ExitCam ↦ Collector: Exited[out exitID key, out
        regID, out exitTS]
    Collector ↦ Manager: Fine[in entryID key, in
        exitID key, in regID, in entryTS, in exitTS,
        out avgSpeed]
}
```

```
@decision [every 2 hours] //Alice plays Collector
def decision_fines(forms):
    potentialFines = forms.messages(Fine)

    for f in potentialFines
        averageSpeed = DISTANCE / (f.entryTS -
            f.exitTS)
        if averageSpeed > SPEED_LIMIT:
            f.bind(avgSpeed=averageSpeed)
```

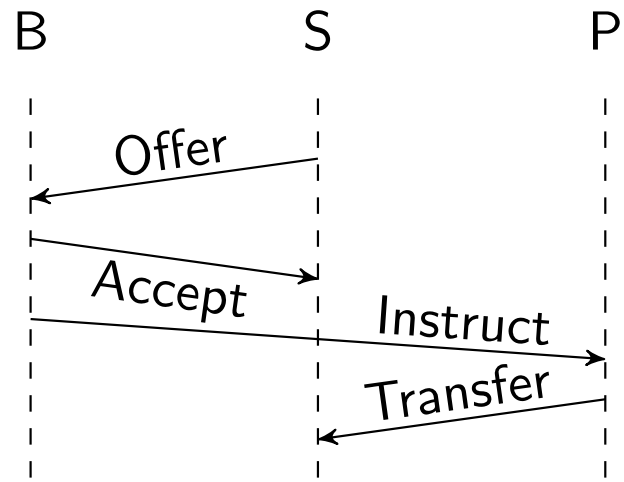
Mandrake: Declarative Recovery Policies



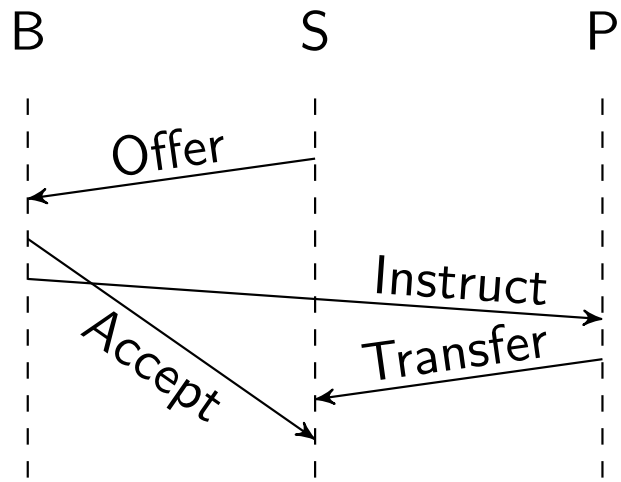
- Despite delivery guarantees, Patient times out waiting for Fulfilled from Pharmacy. Why?

```
// Alice plays Patient
- action: remind Physician of Complaint until
    Reassurance or Prescription or Filled
  when: 0 0 * * * // daily
  max tries: 5
- action: remind Pharmacist of Prescription after 2
    days until Filled
  when: 0 0 * * * // daily
  max tries: 5
```

No Go (or Erlang)



(a) Protocol



(b) Violated despite FIFO channels.
Enter selective reception!

FIFO channels topped up with selective reception

- Absurd, limits flexibility

The Fix

- Amend with a protocol-based programming model
- Use protocols to coordinate asynchronous computations!

Not So QUIC

The Rabbithole of Ordering

Recently standardized by the IETF, widely adopted

- Multiple FIFO streams within a connection to address TCP's head of line blocking problem

But why bother? No two messages ever have to be delivered in order!

Also, don't need MPTCP, SCTP, MQTT,...

How does the field of networking and middleware change when ordering is obviated?

- How to do congestion control without TCP?

Protocol: The Fundamental Distributed Abstraction

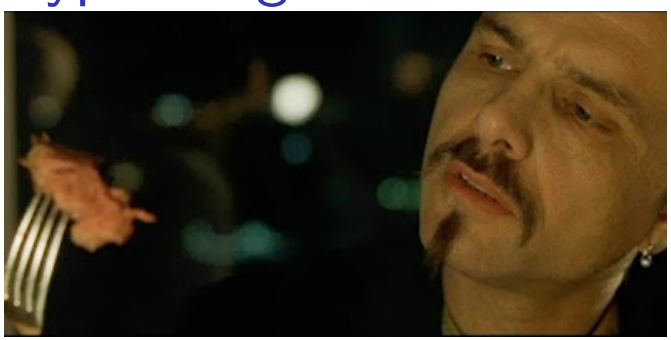
David Clark: We'd use UDP, not TCP, if we knew how to do application-level protocols

Well, we know now!

Scott Shenker: Teaching networks as a bag of protocols is dull, need fundamental principles and abstractions

Teach how to model applications as protocols and the implications for communication services

Cypher: Ignorance is bliss



Knowledge empowers. Try our software:
`https://gitlab.com/masr/`