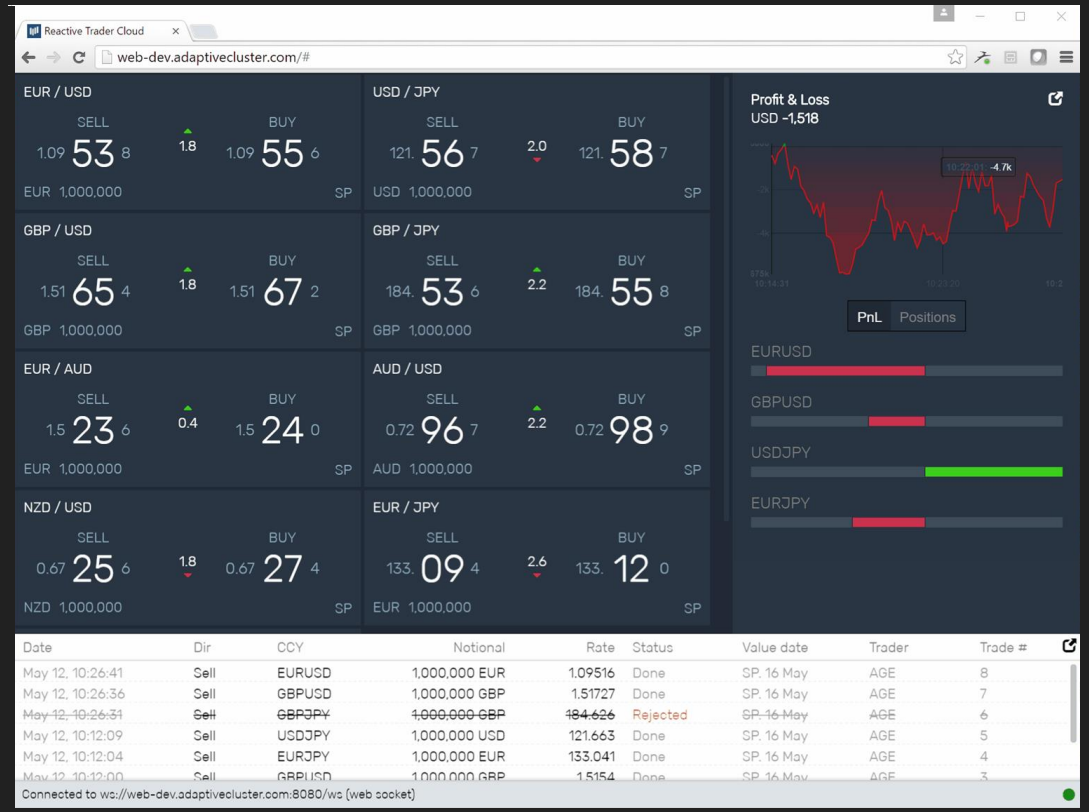
Clustered Event-Driven Services

Simple model in a complex event-driven world

YOW! 2017 - Sydney

Adaptive

Yow 2014 - Reactive user interfaces

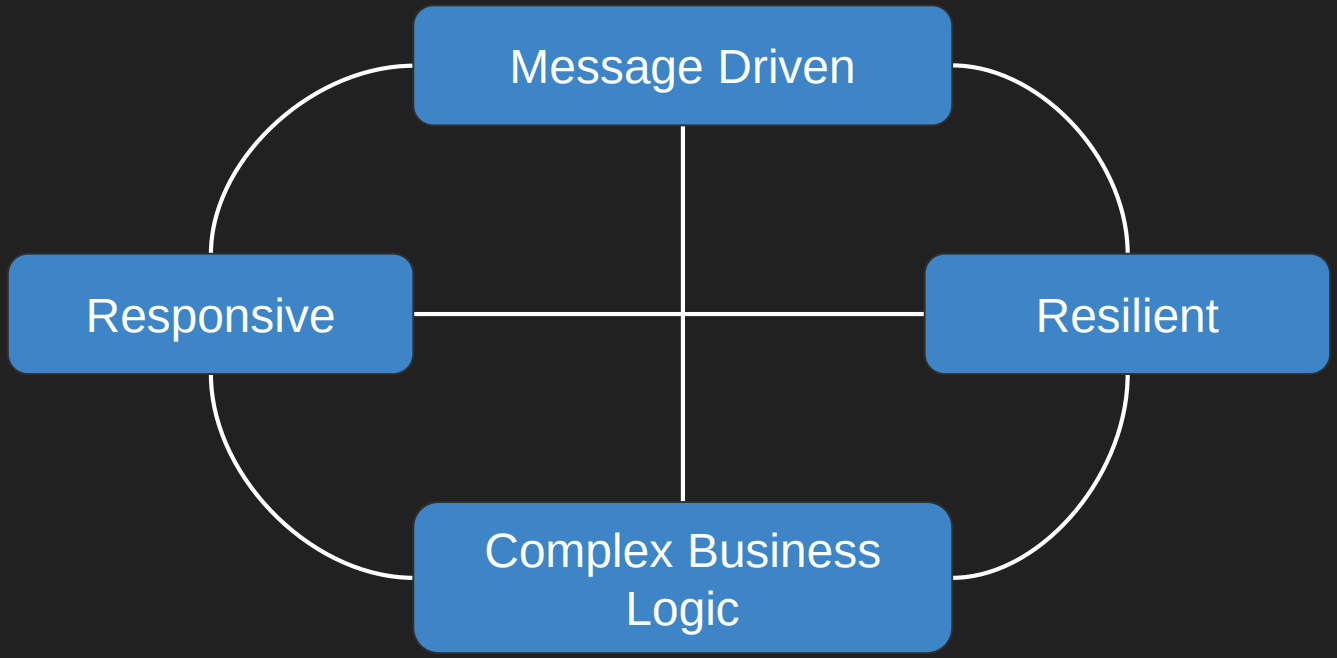


Lee Campbell

Matt Barrett

Adaptive

Attributes of a trading system



Adaptive

The problem

Complex multi-threaded programs

⇒ Hard to reason about

⇒ Hard to debug

The Solution - LMAX Architecture

Blueprint for reactive services

Clean separation of concerns
- technical requirements
- business logic

Keep the business logic as simple as possible

Very easy to debug

Dave Farley

Martin Thompson

Adaptive

Our journey to the LMAX Architecture

Simple Model
Easy To Debug

Fault Tolerance
State Replication

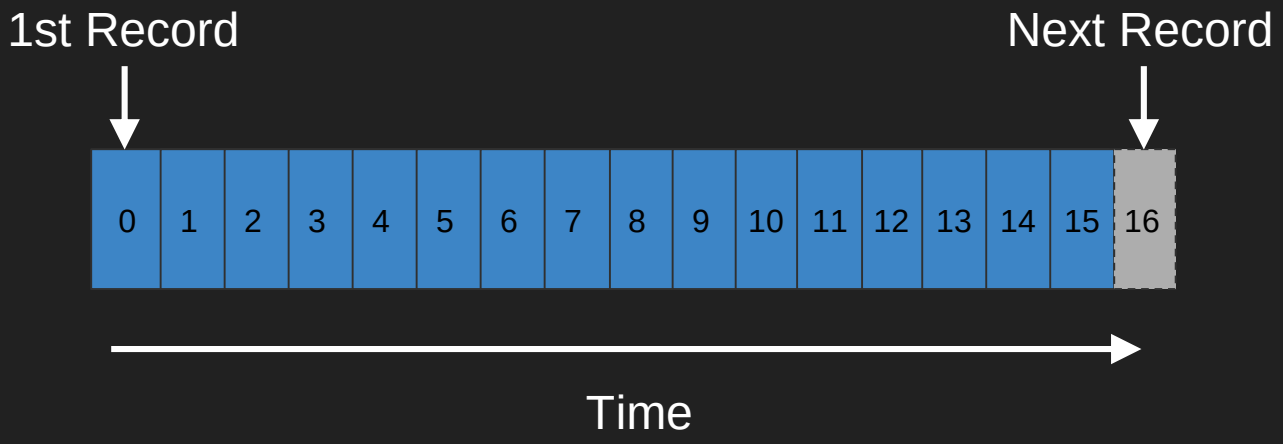
Durability without a
Database

Deterministic execution in a distributed event system

Determinism

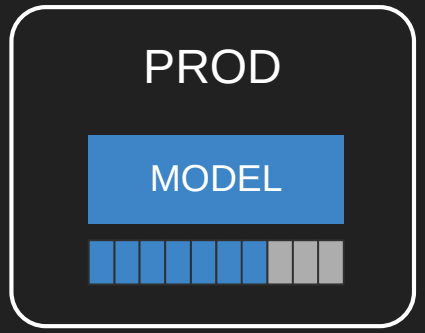
Given an **initial state** and a **command**, a deterministic model will always produce the same **output state** and **side-effect(s)**

Log as Sequencing Mechanism

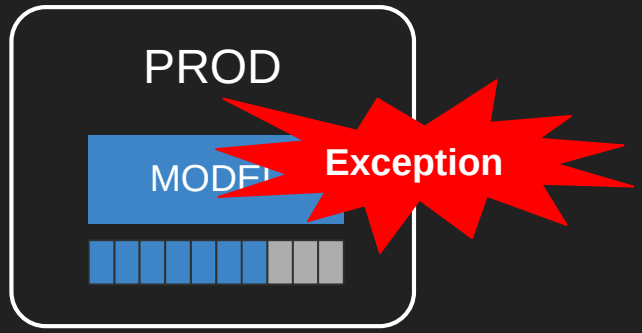


Single Threaded Log Consumer

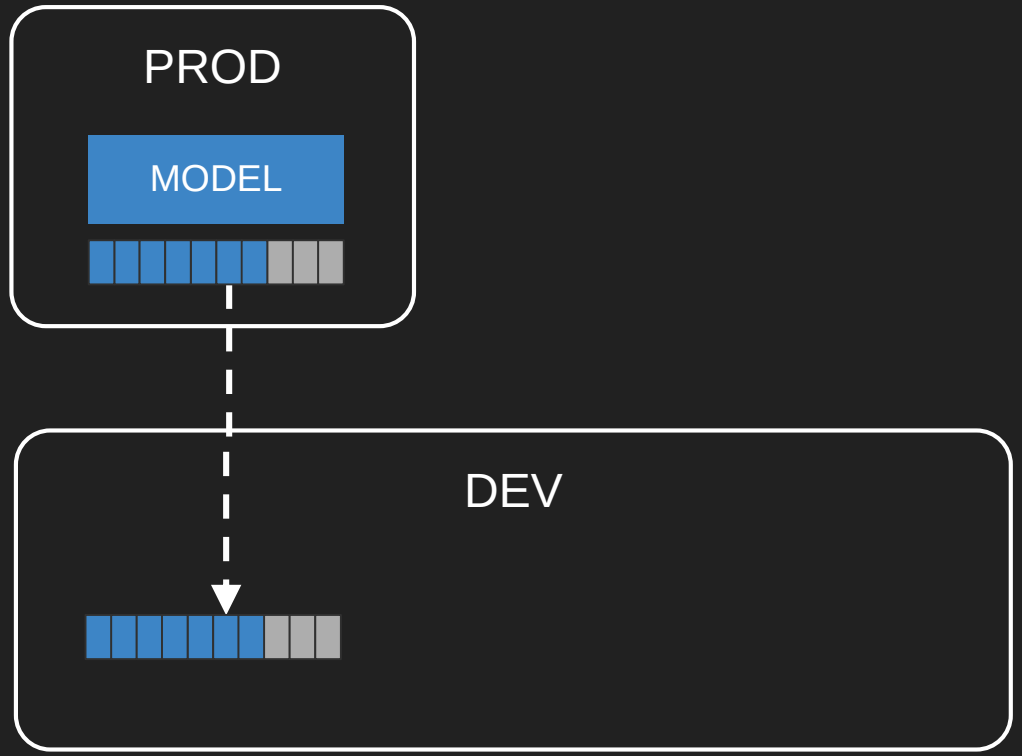
Easy Debugging by Replicating Exact State



Production issue

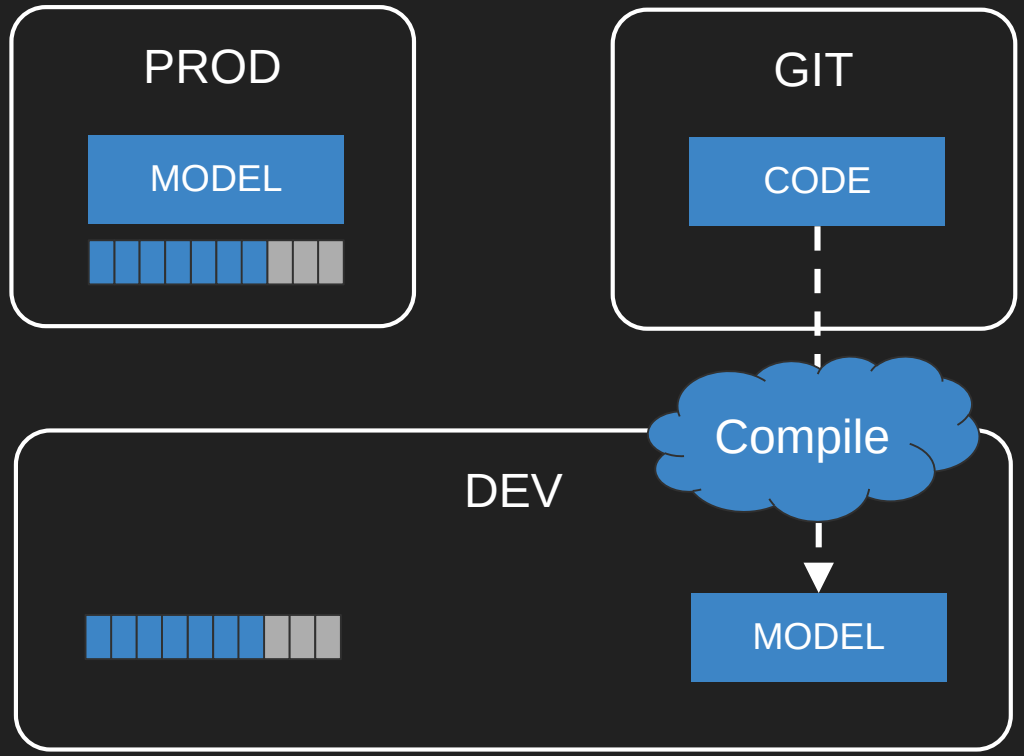


Get the log



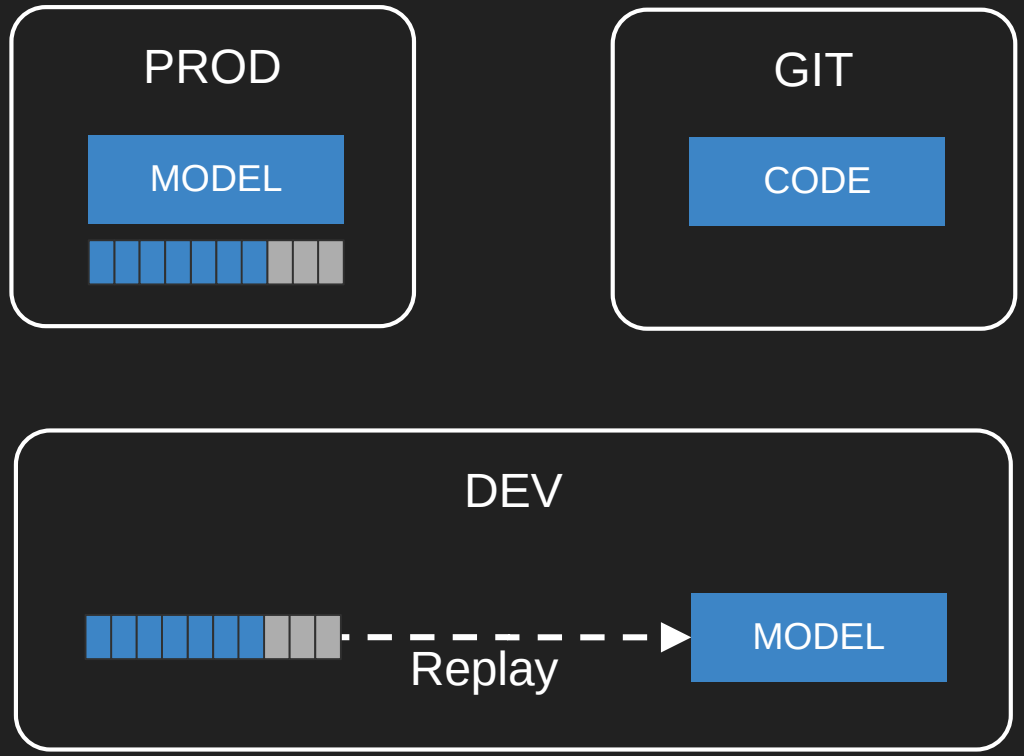
Adaptive

Get the same version of the code, compile



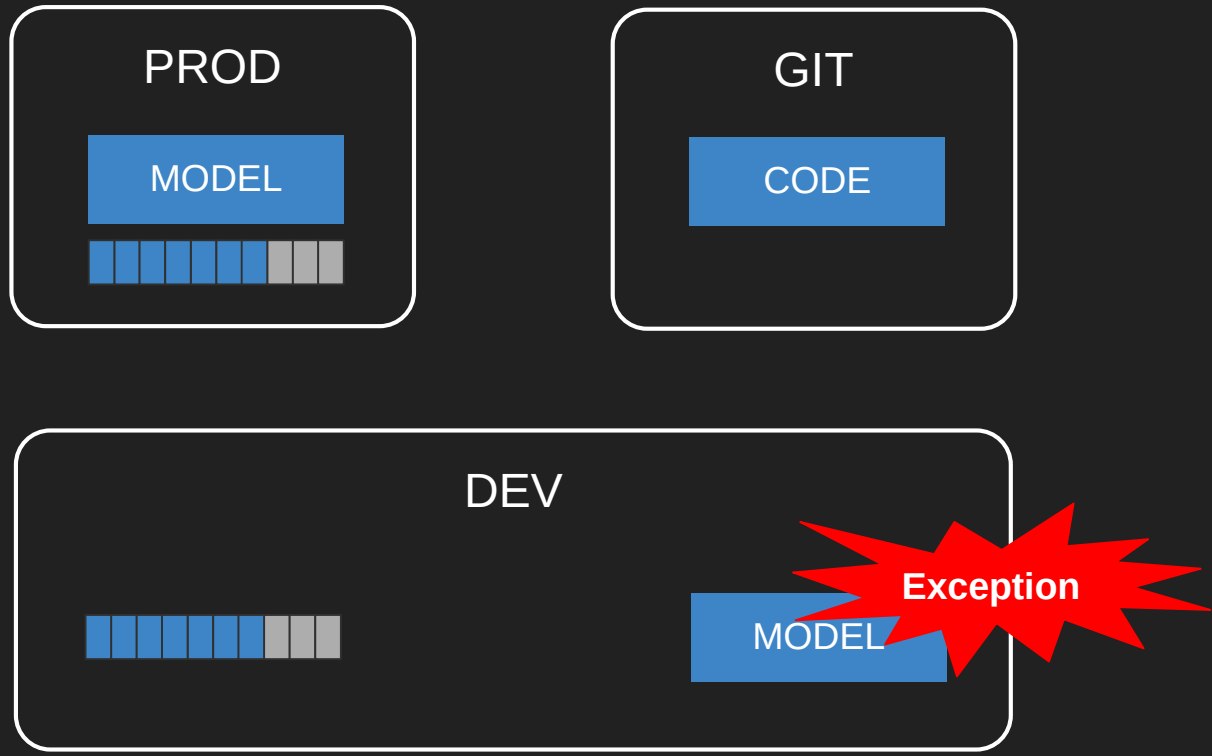
Adaptive

Replay the log



Adaptive

Reproduce the same behaviour, debugger attached



Adaptive

Performance

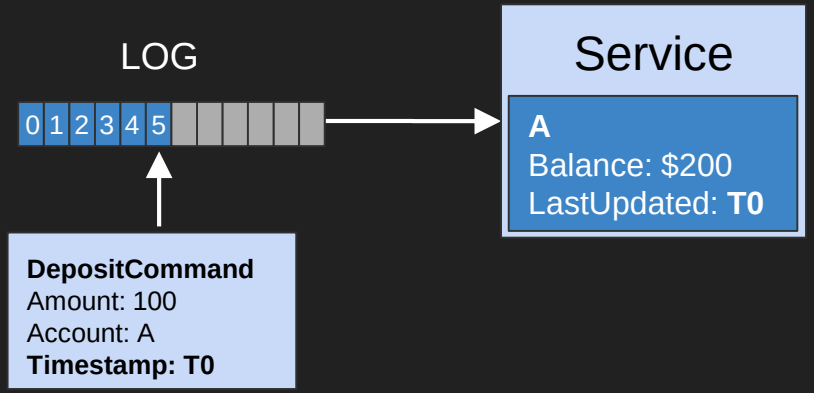
Locality of state and behaviour

10K ops/sec - quite easy to achieve with sensible code
100K ops/sec - profiling required
1M+ ops/sec

- Optimised data structures
- Low complexity algorithms - O(1), O(log(N))
- Minimise allocation

Handling Time Deterministically

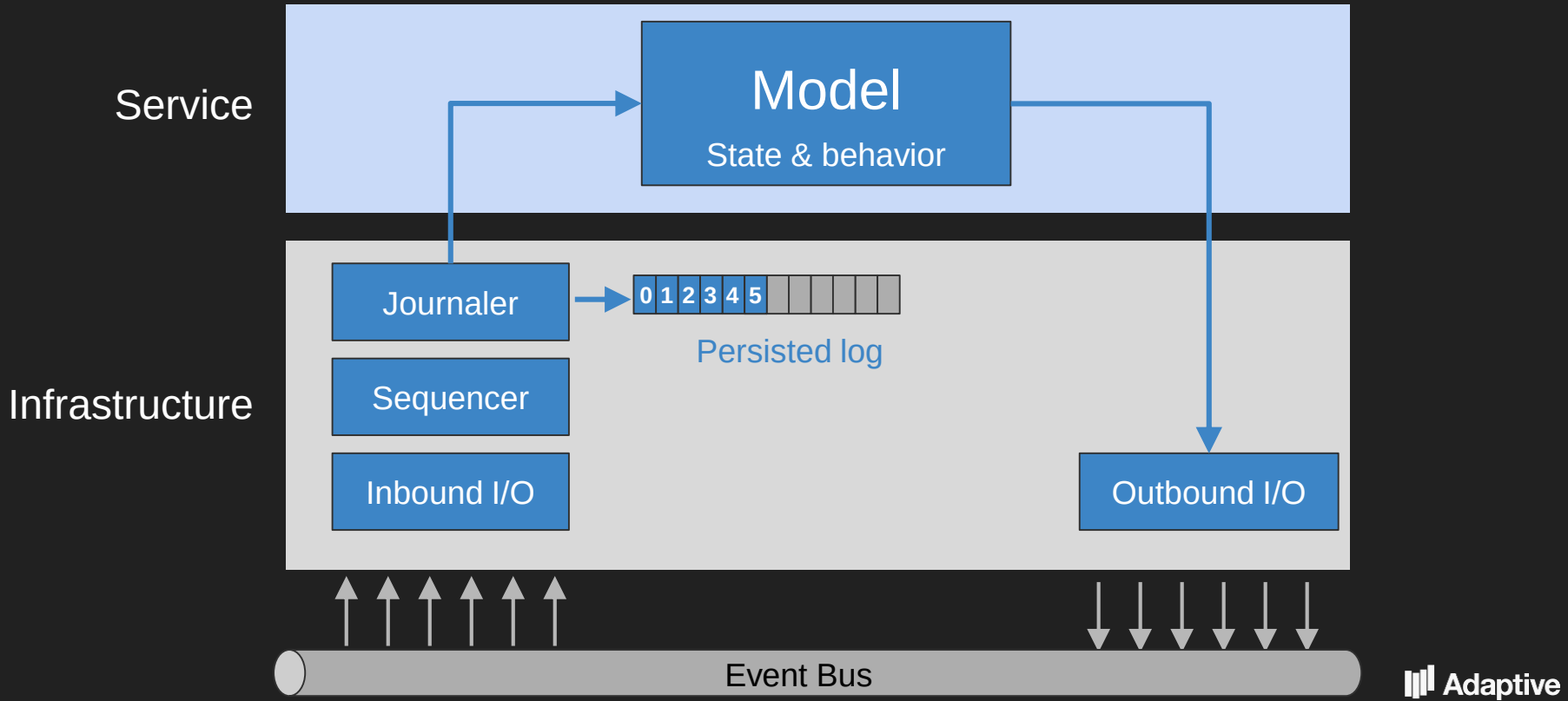
- System time is not deterministic
- Inject time through the log
- Great for testability



Adaptive

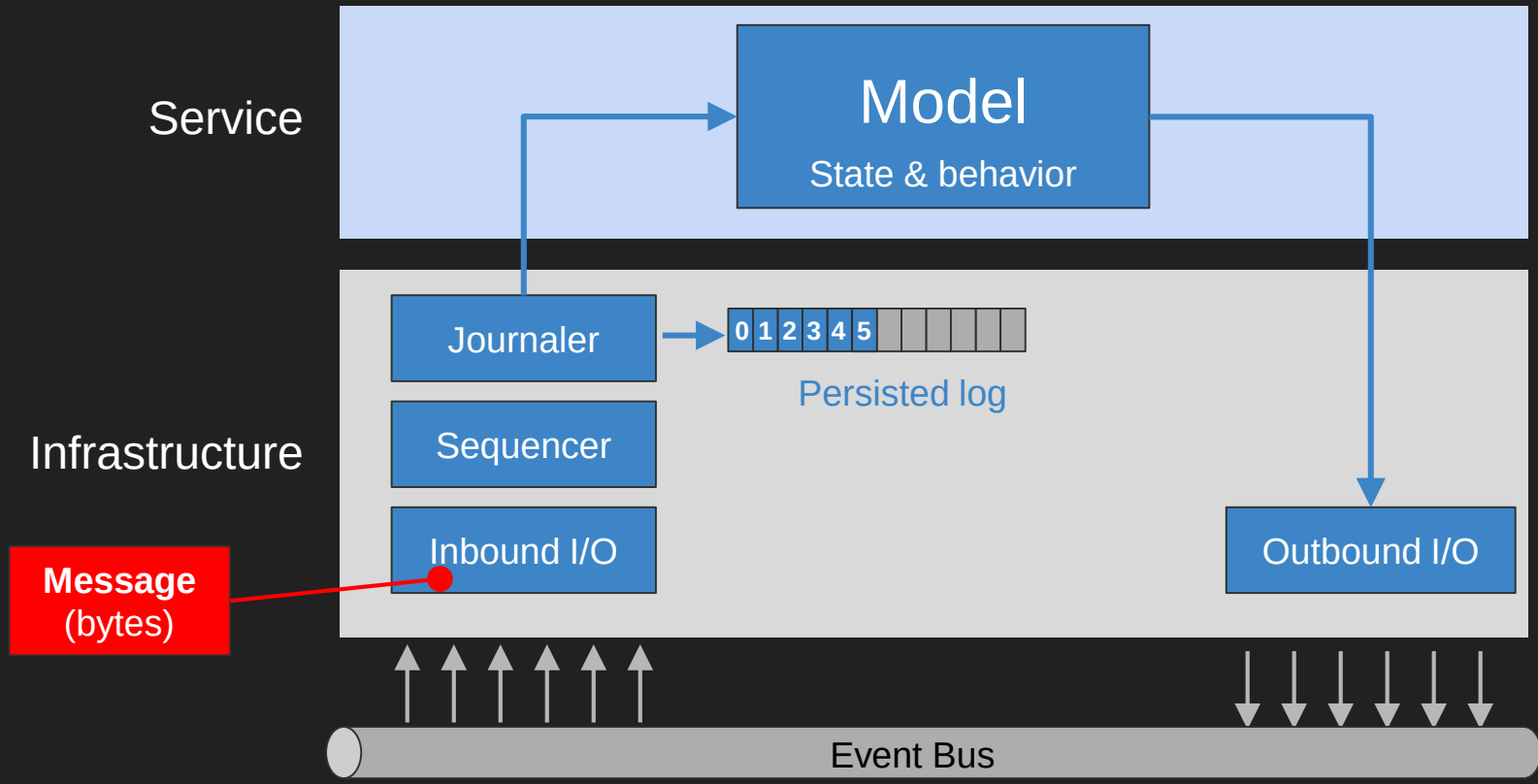
Service Anatomy

Service Anatomy: Separate Infrastructure from Model



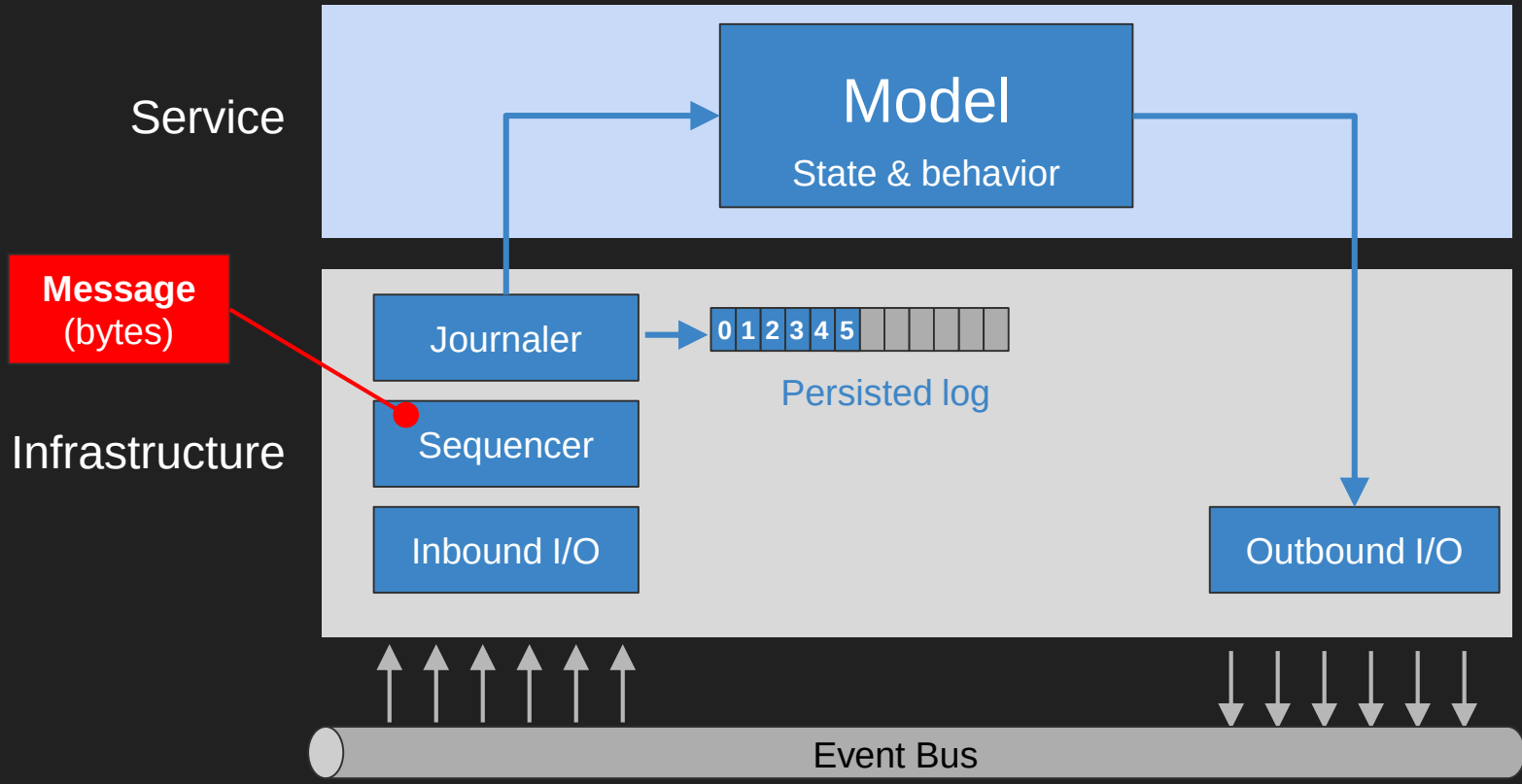
Adaptive

Incoming message



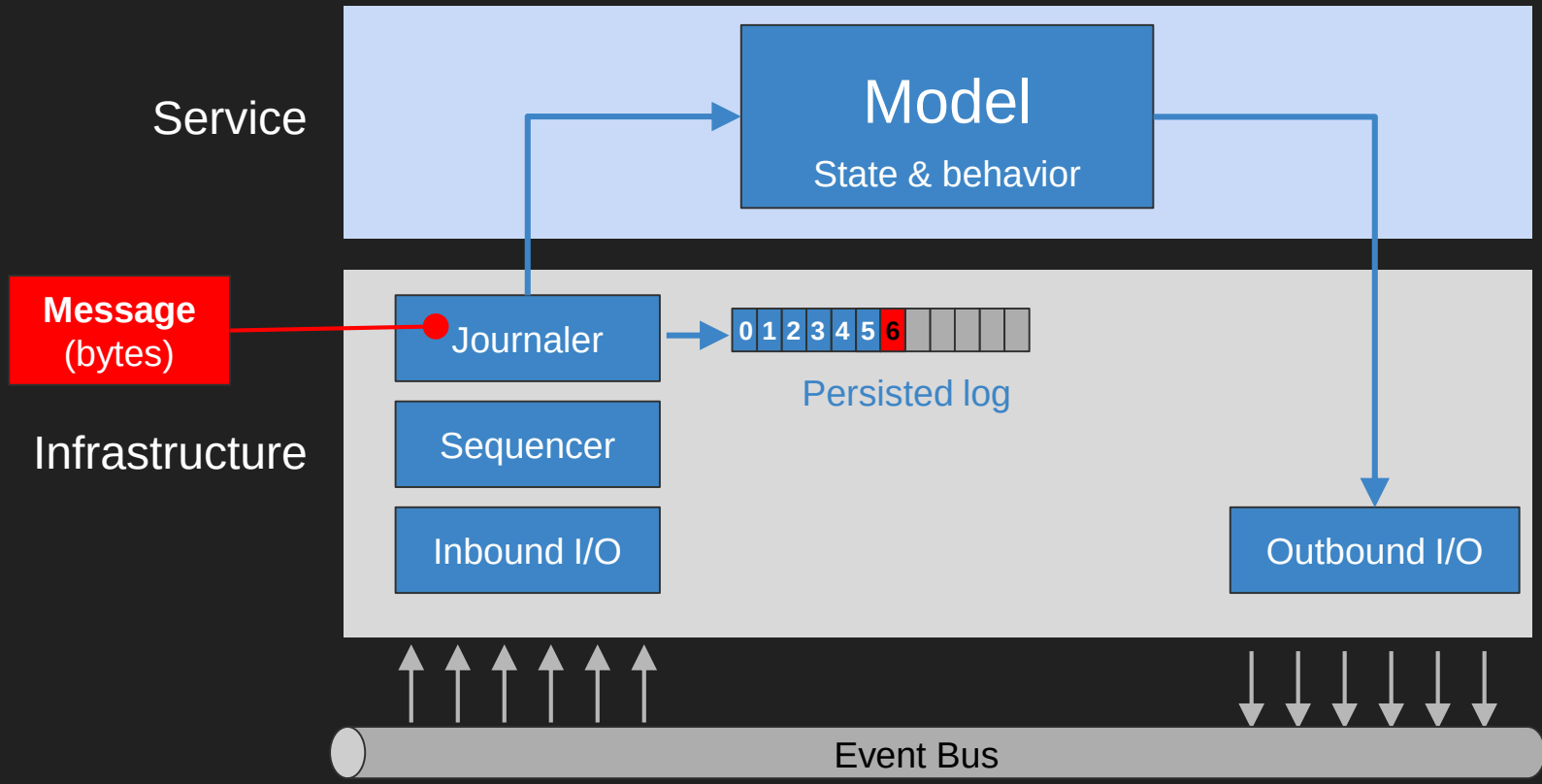
Adaptive

Message is sequenced



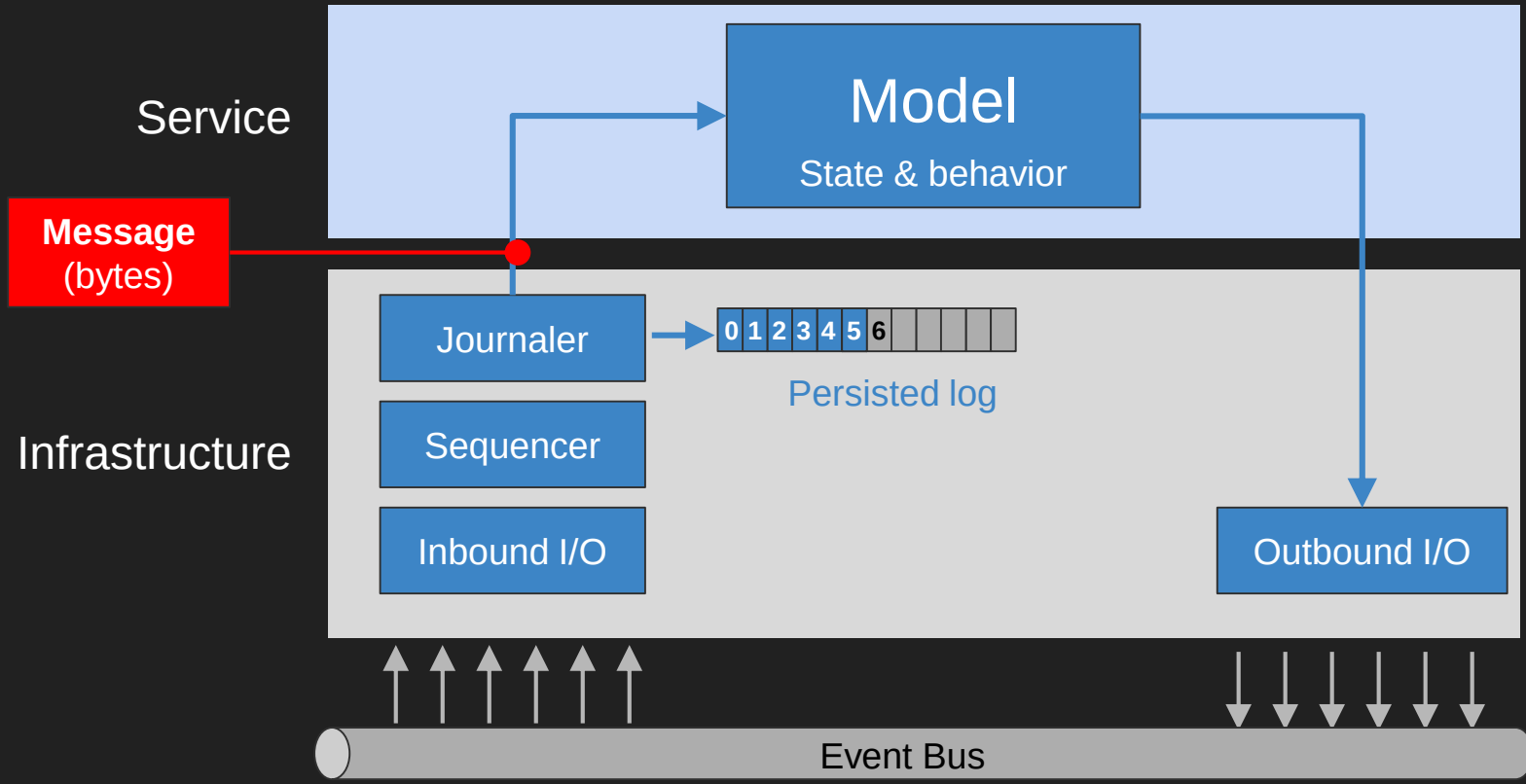
Adaptive

Message is written to disk



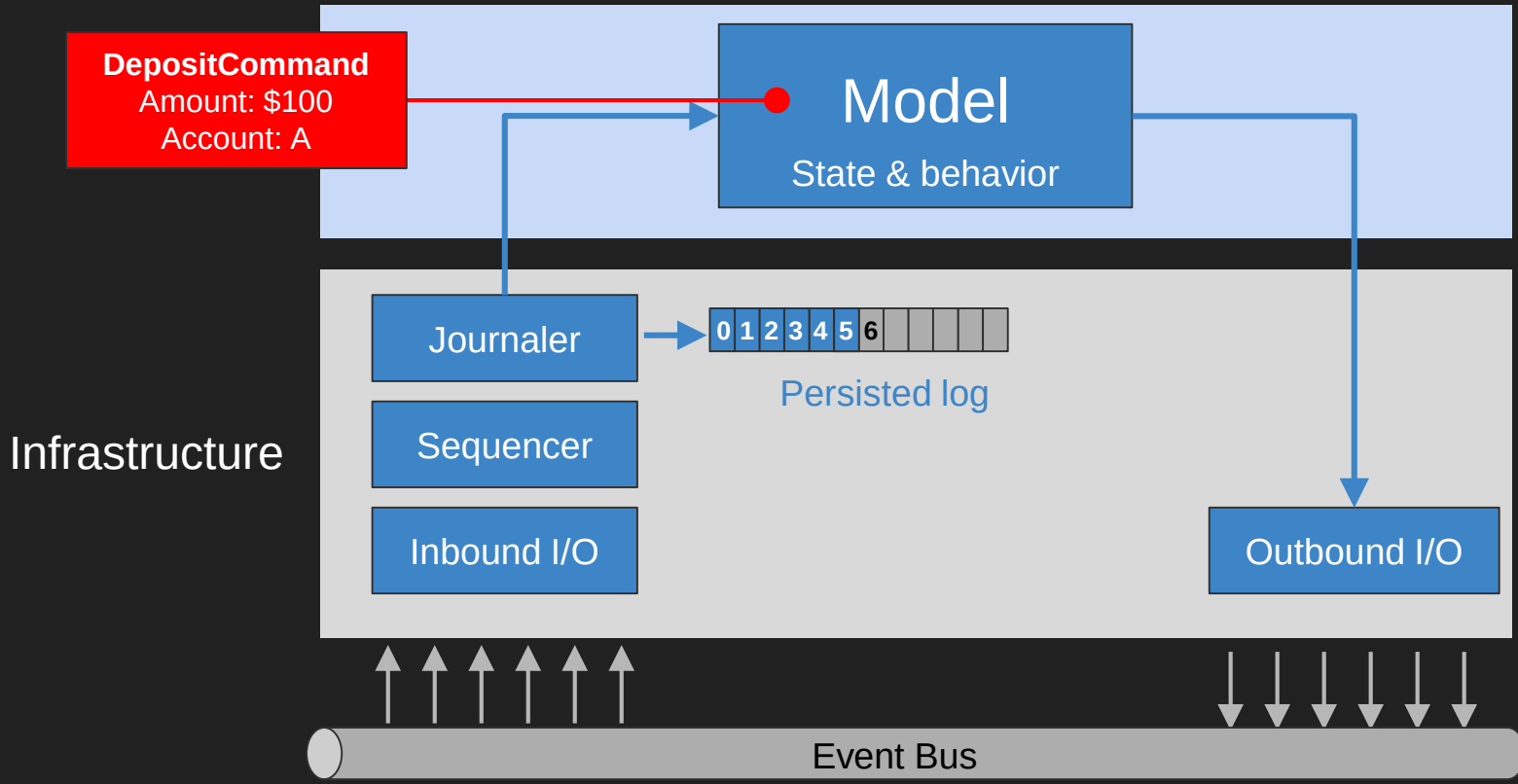
Adaptive

Message passed to the model



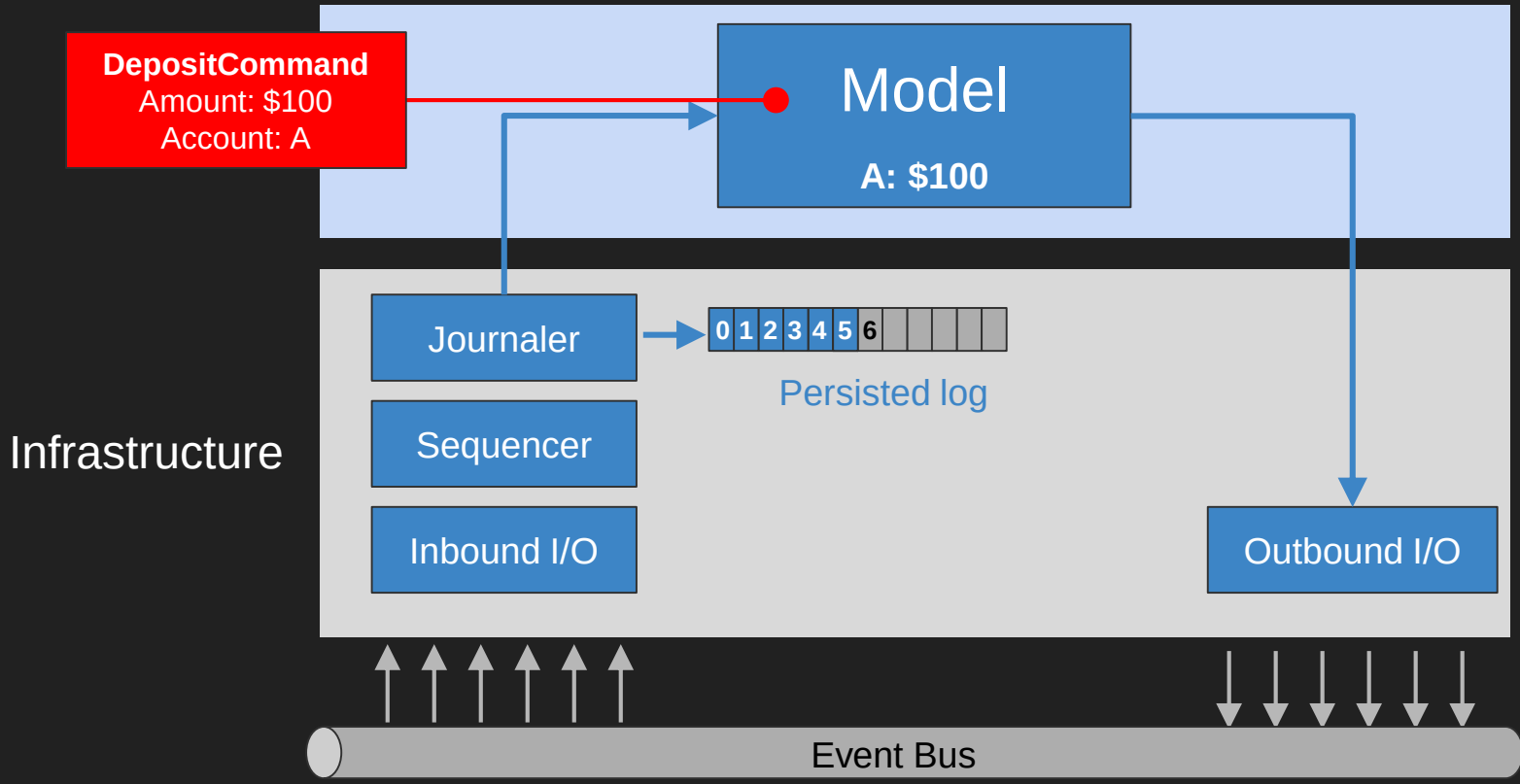
Adaptive

Message is decoded



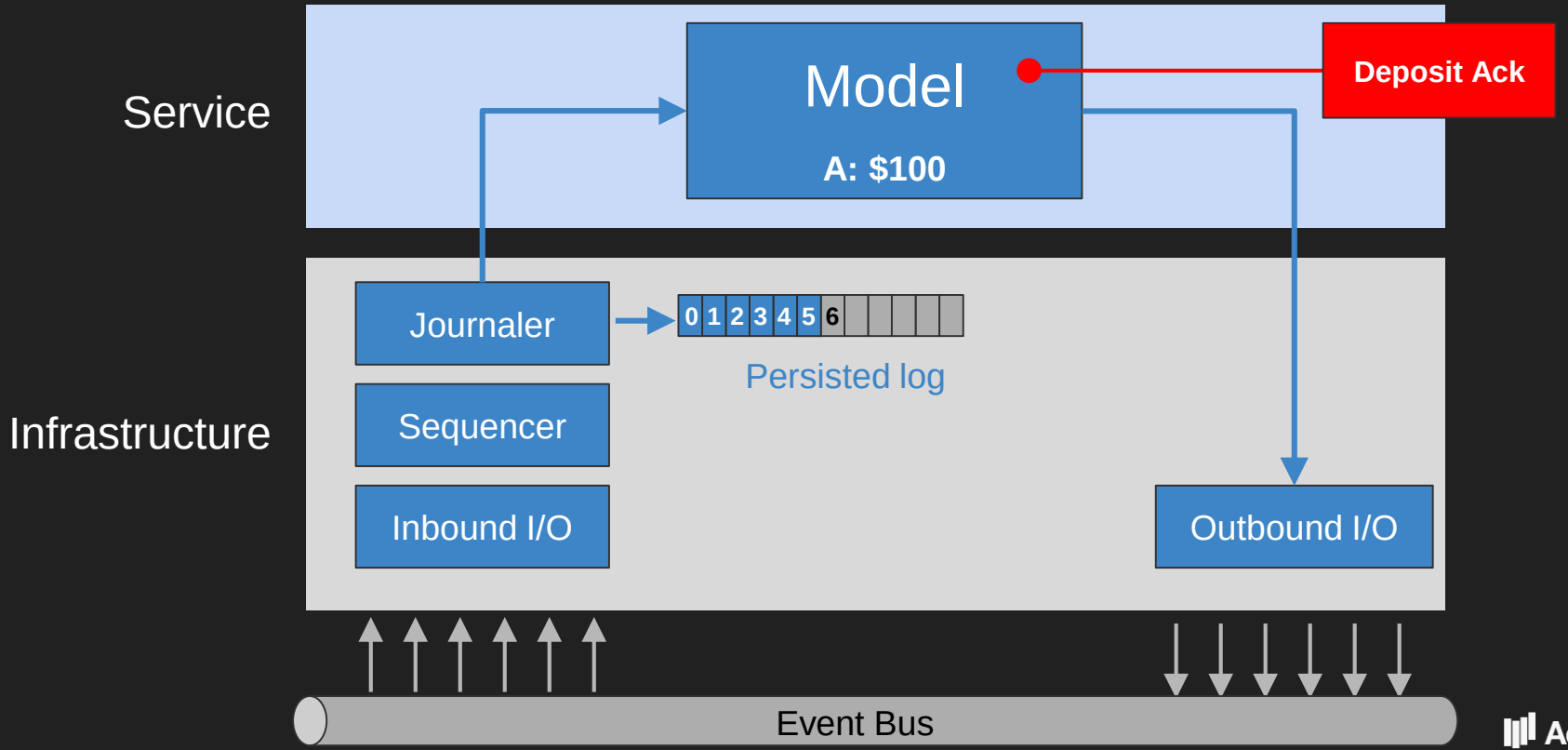
Adaptive

Command is applied to the model



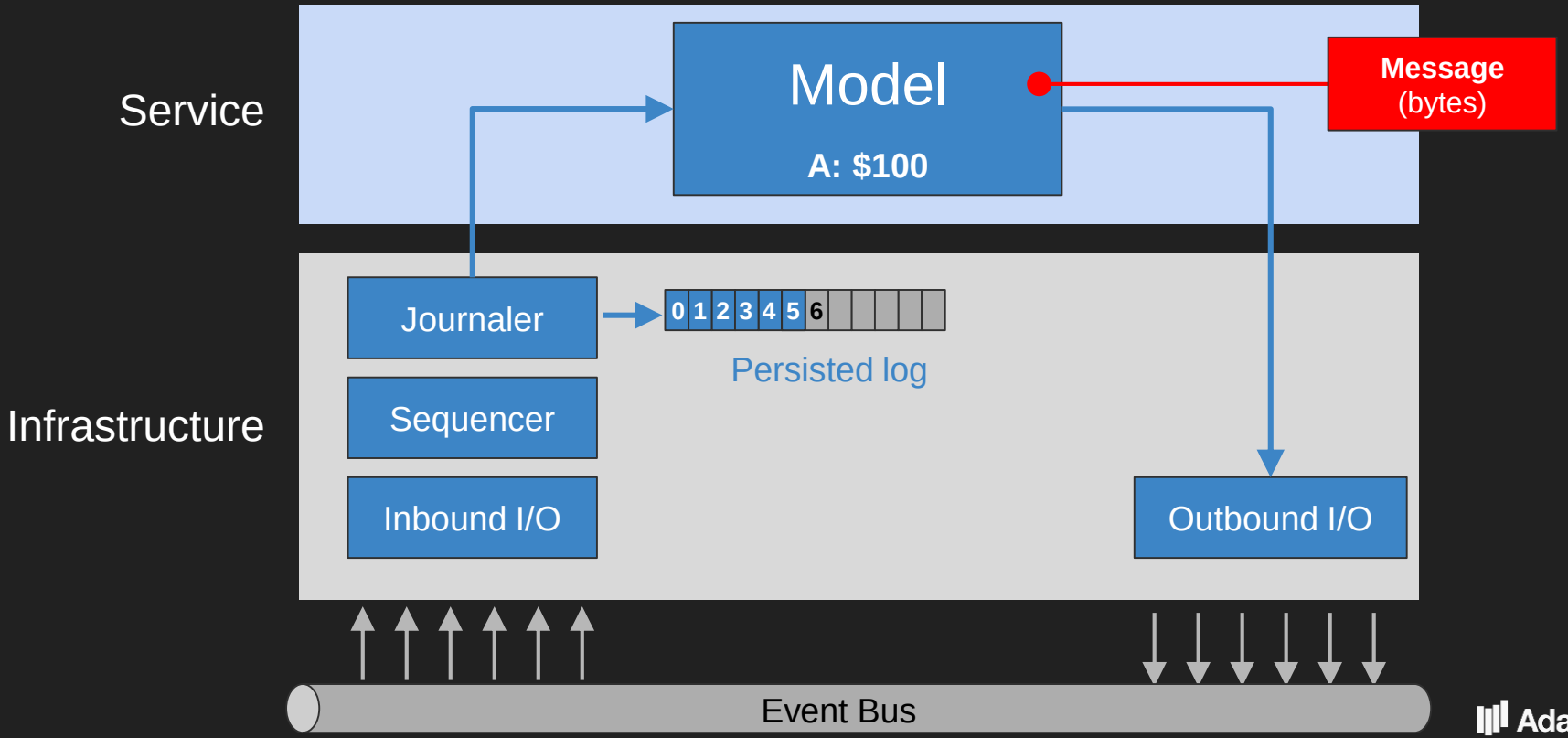
Adaptive

Response message is created



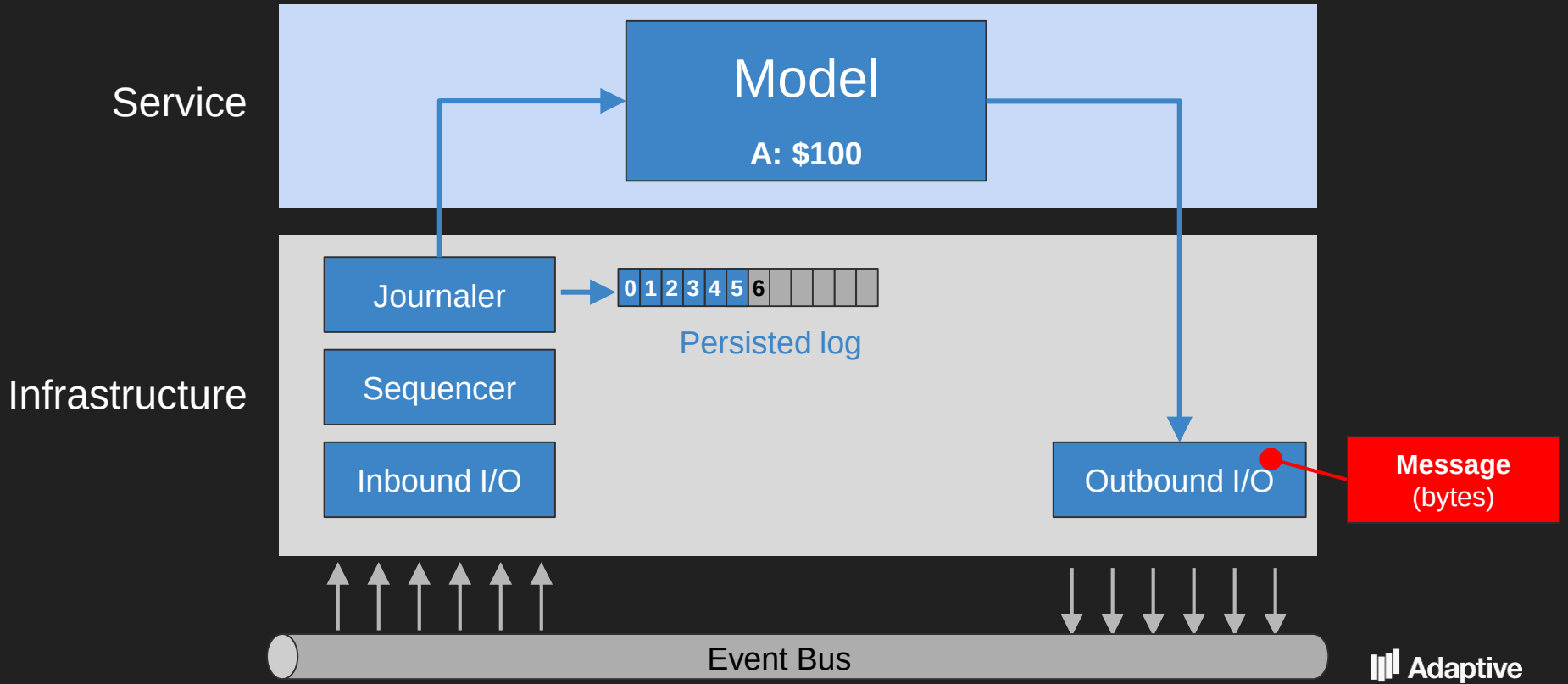
Adaptive

Response message is encoded



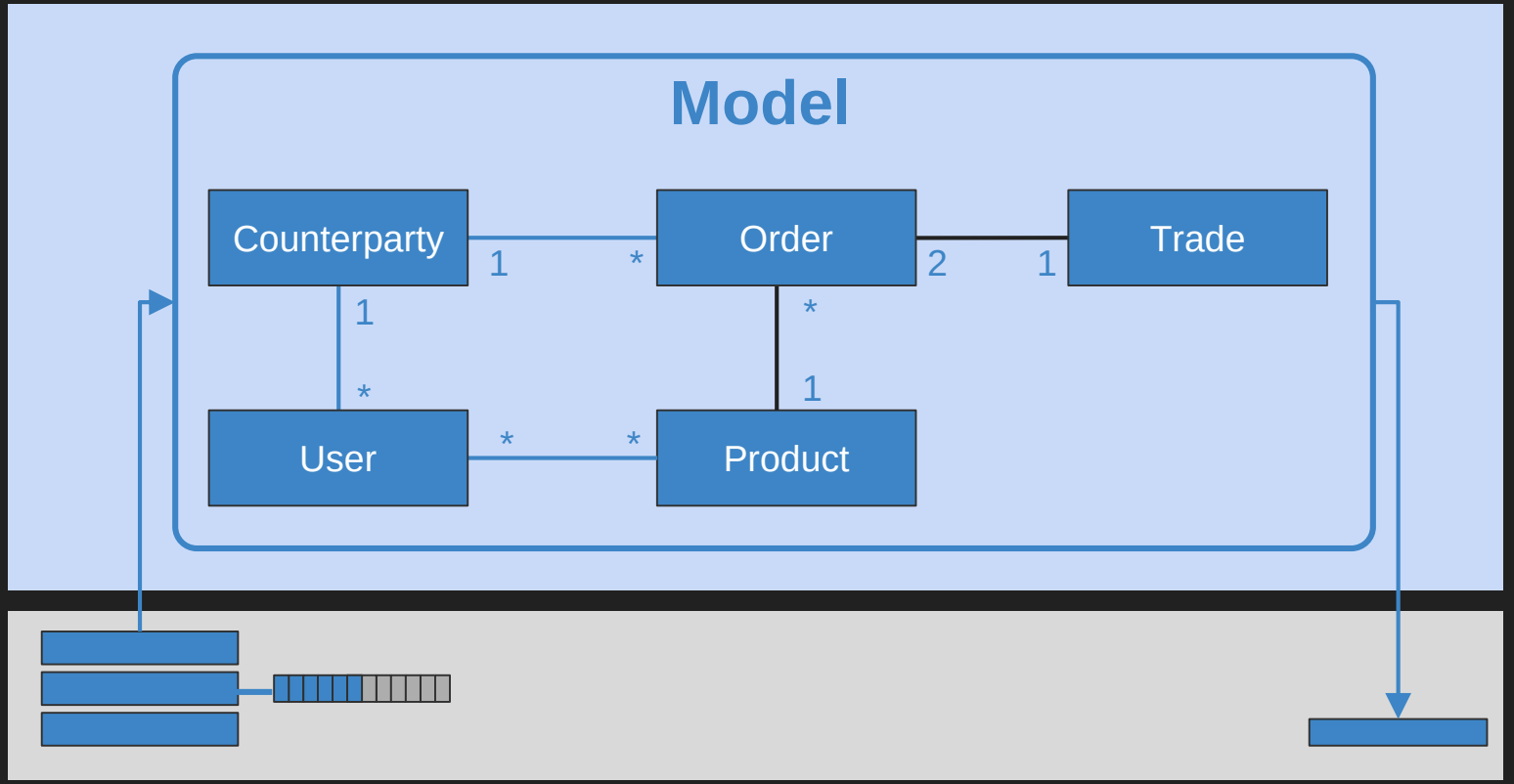
Adaptive

Message is sent over the network



Adaptive

Stateful Model

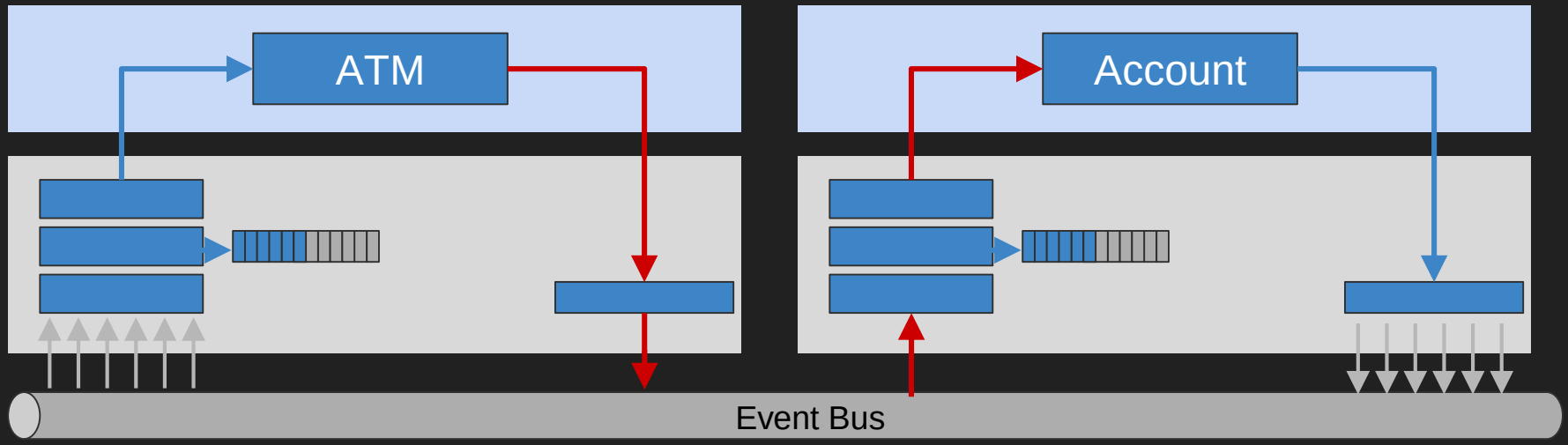


Adaptive

Service Contracts & Message Domain Model

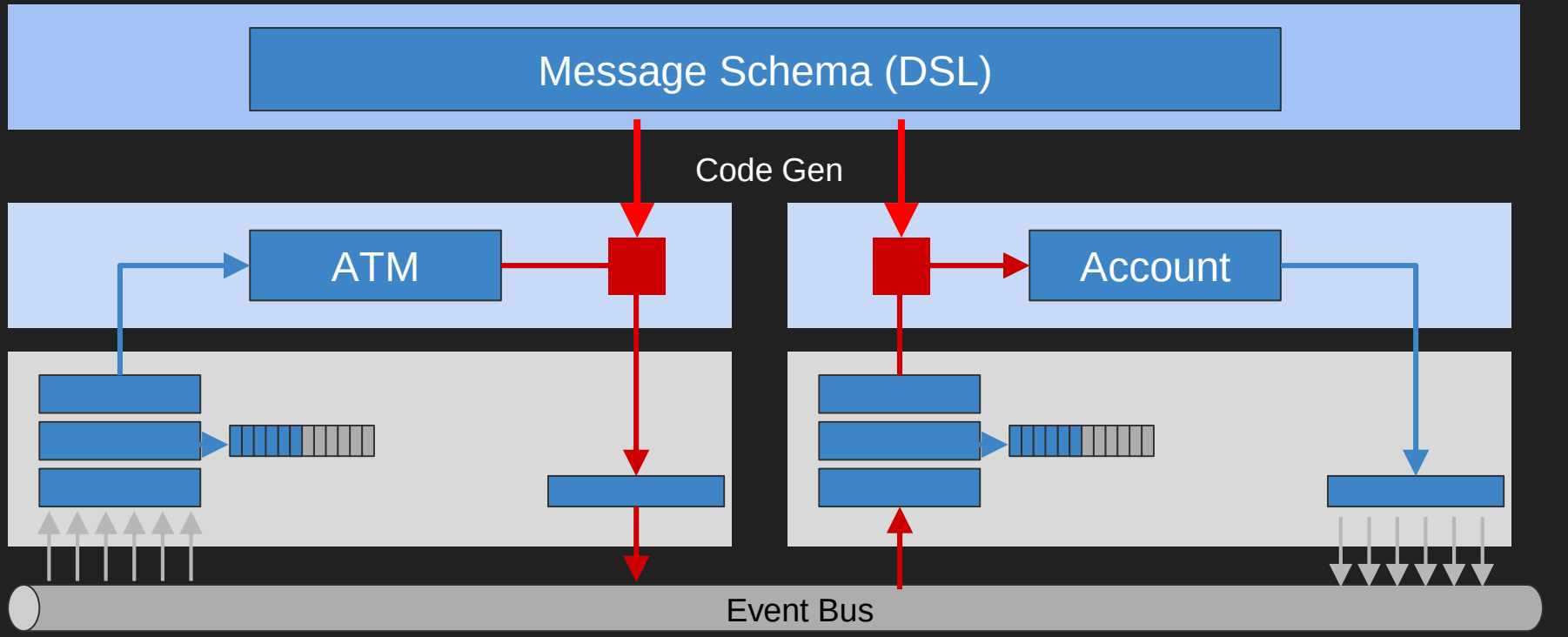
Adaptive

Service Binding

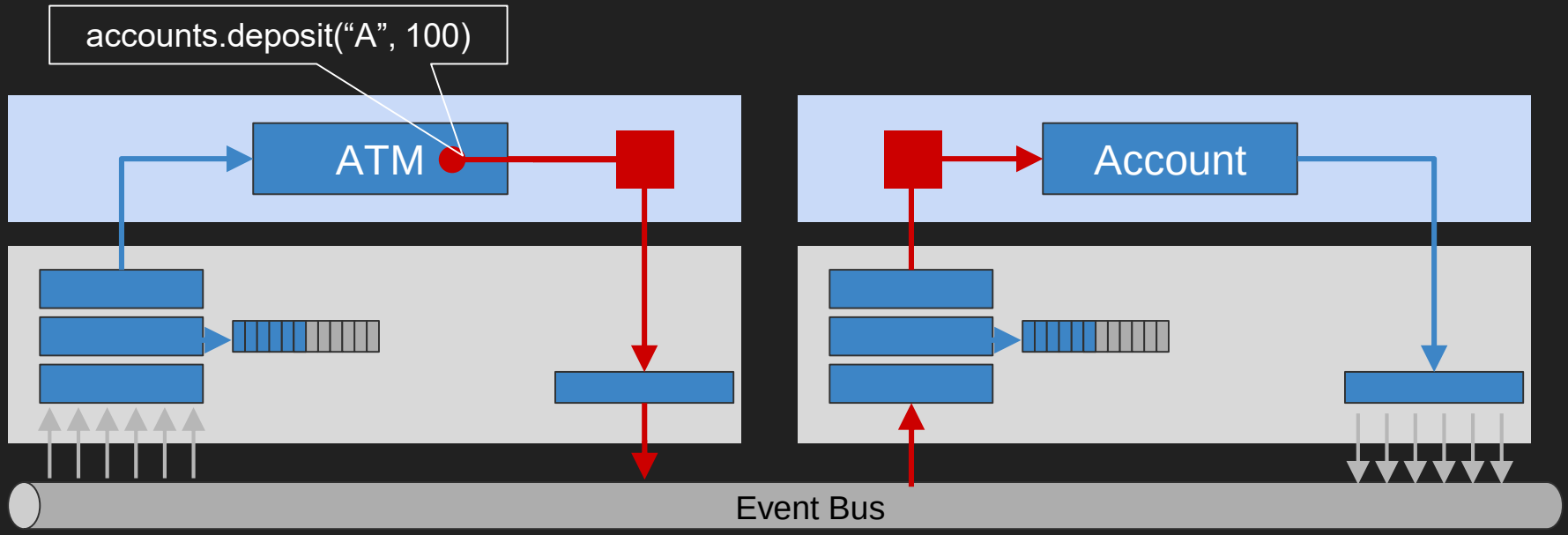


Adaptive

Code Generation of Marshallers

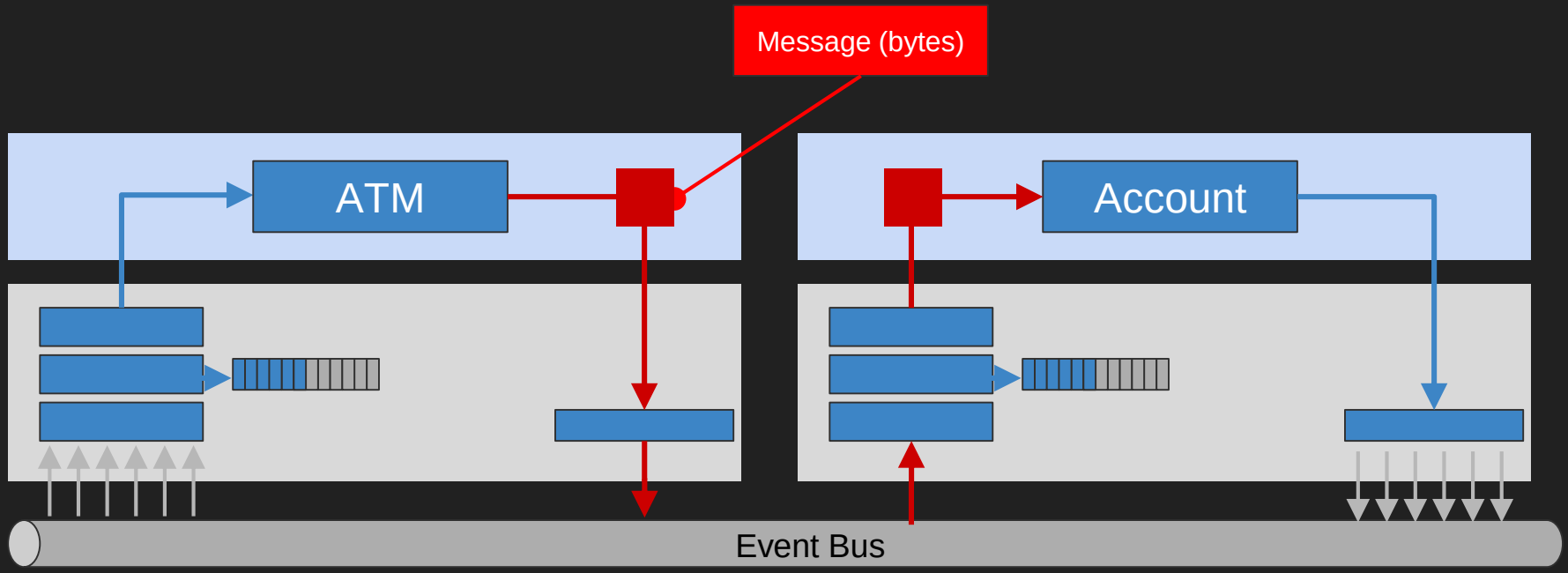


Service Interface injected in ATM Model



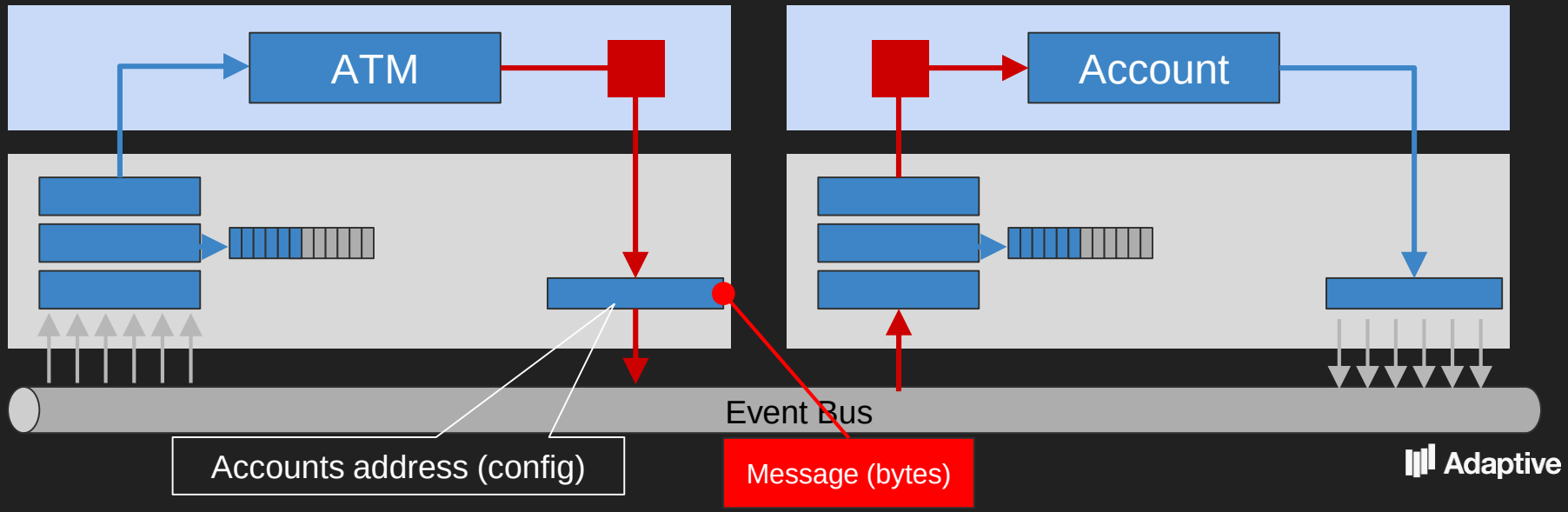
Adaptive

Message is encoded



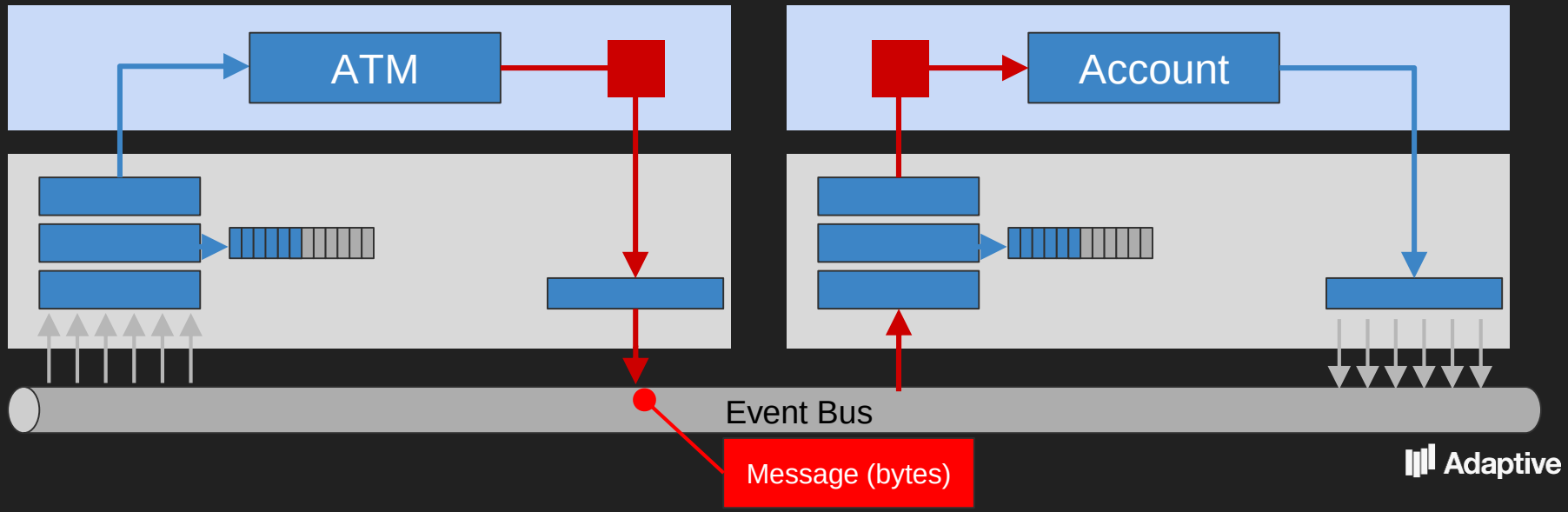
Adaptive

Message is routed



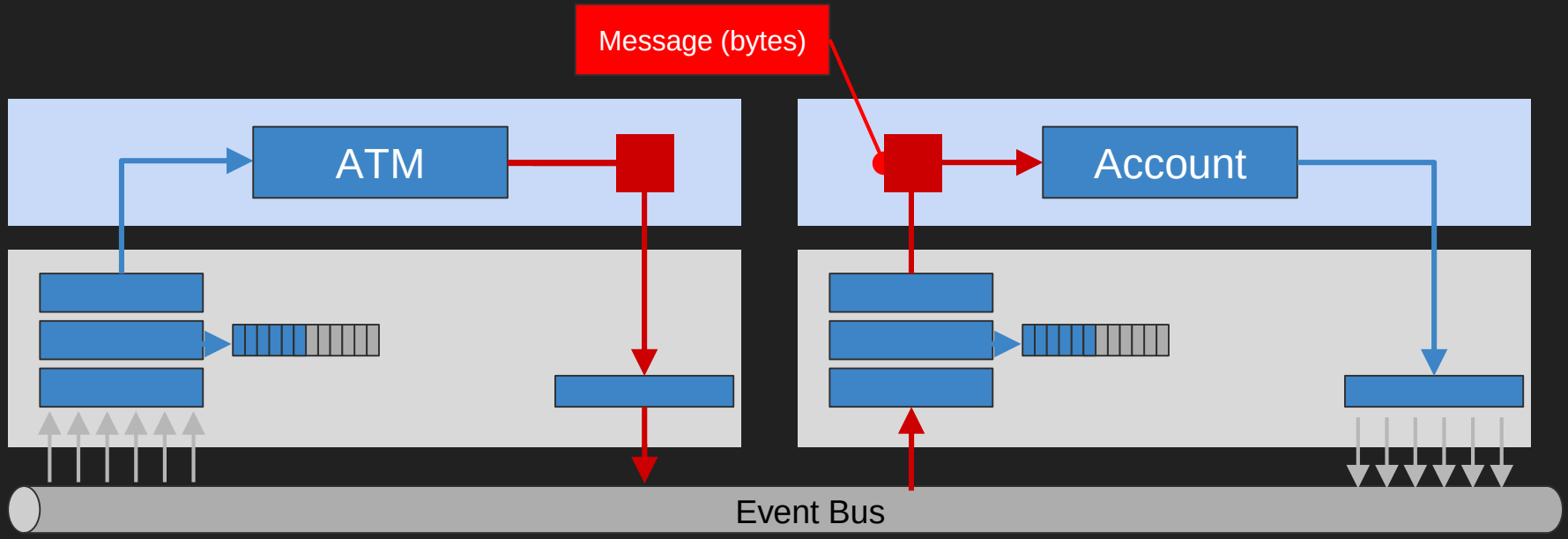
Adaptive

Message is sent on the wire



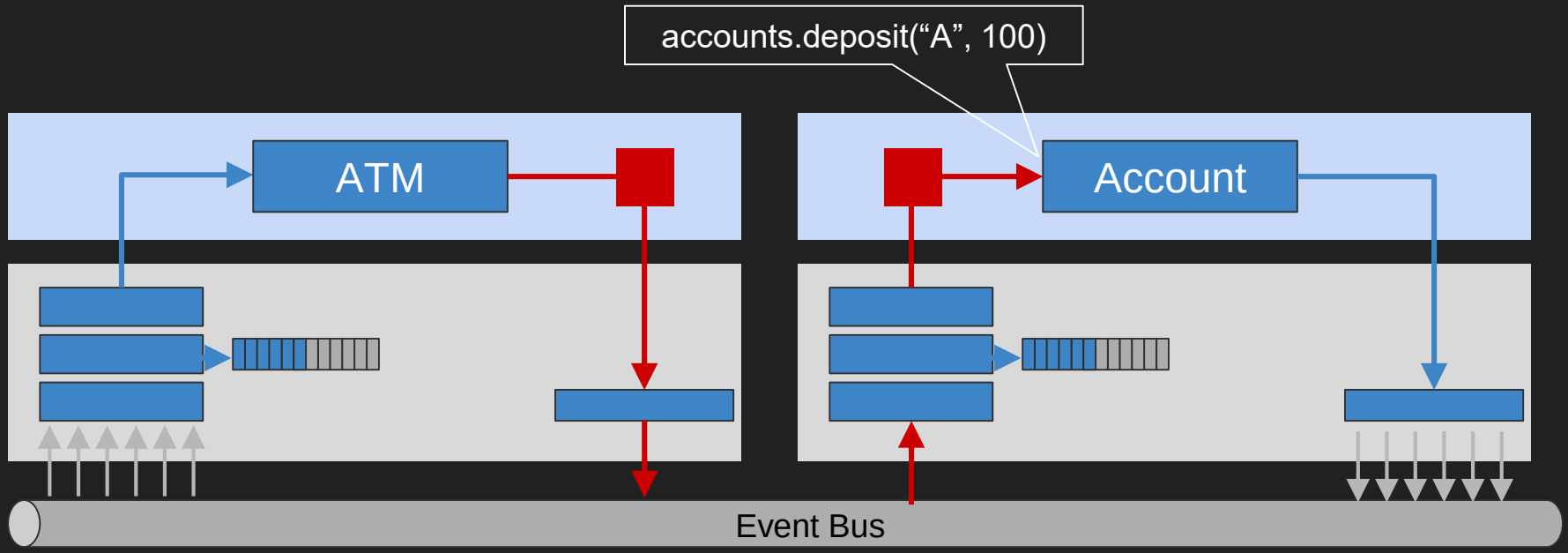
Adaptive

Message header contains message type



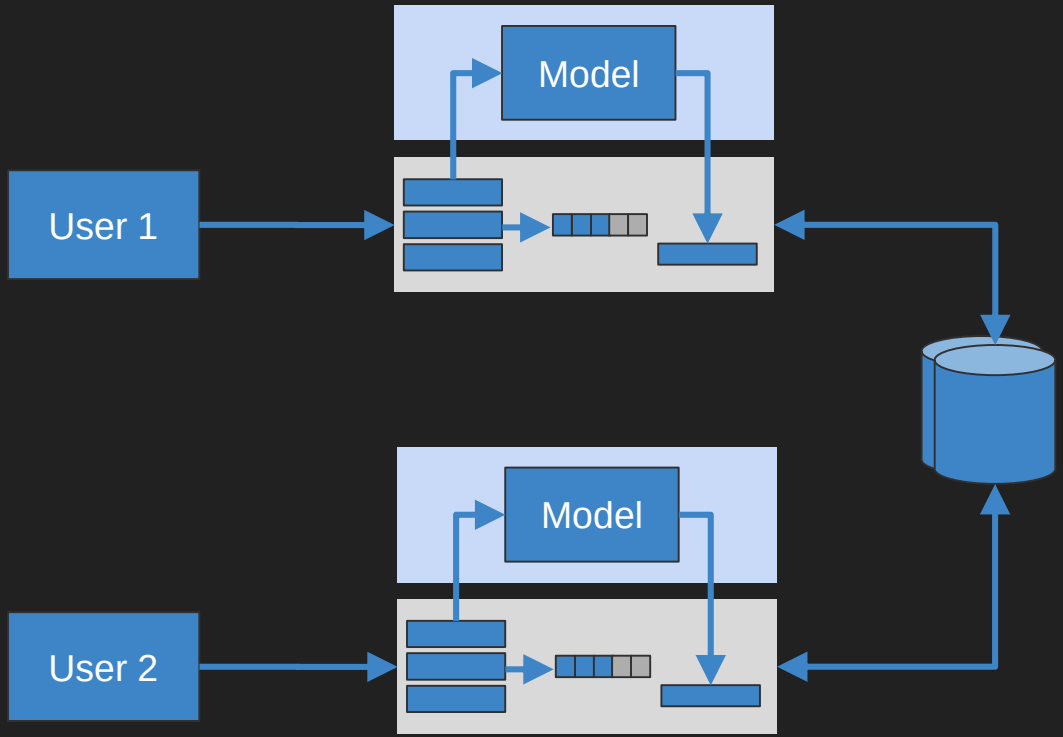
Adaptive

Marshaller decodes and a method on account



Adaptive

1st Iteration - State Synchronized via Database



Benefits

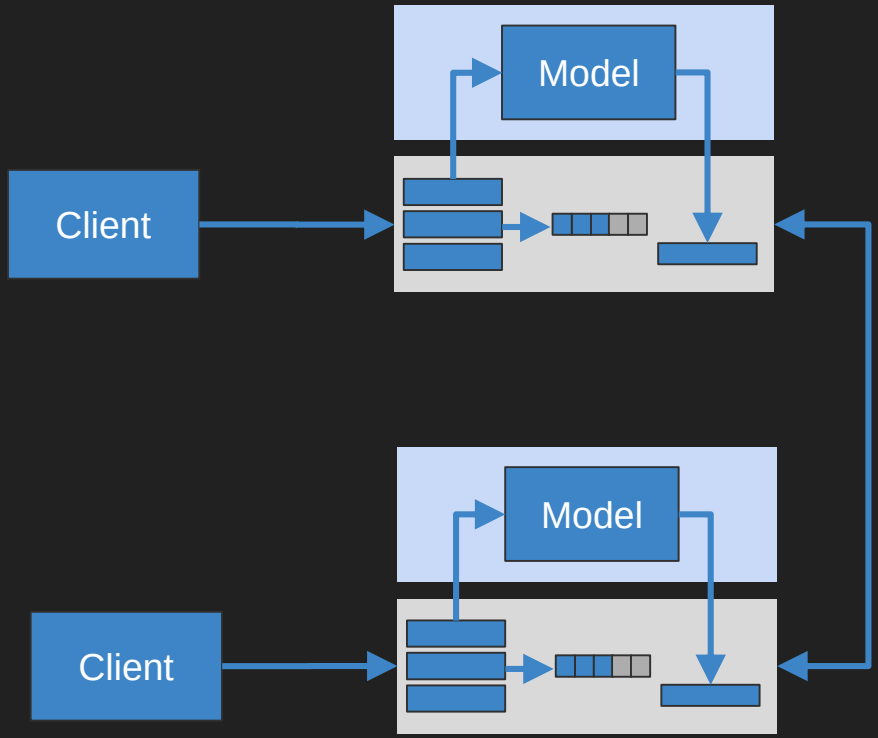
- Simple model
- Debugability

Pain points

- Synchronisation through database

Fault Tolerance & State Replication

2nd Iteration - Clustering



Adaptive

Fault tolerant data replication

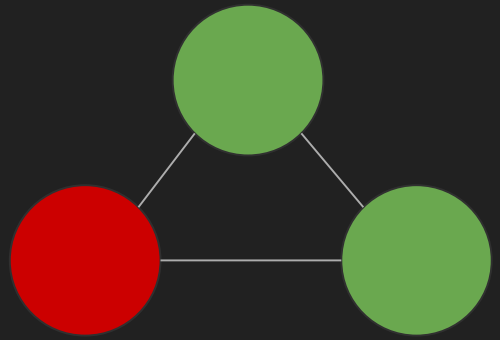
Difficult problem:

- The log must stay consistent between servers
- Even if a server fails or the network fails

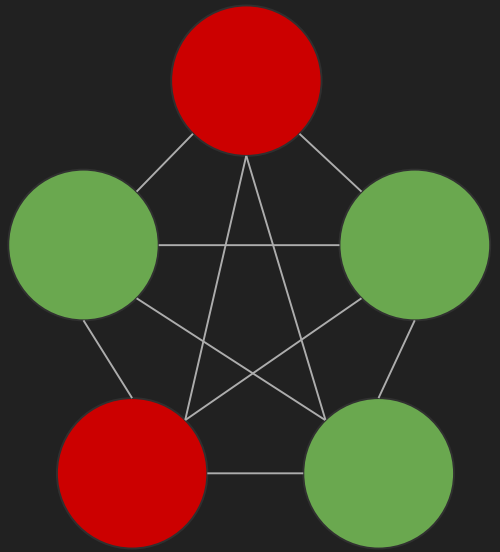
Adaptive

Distributed consensus problem

RAFT



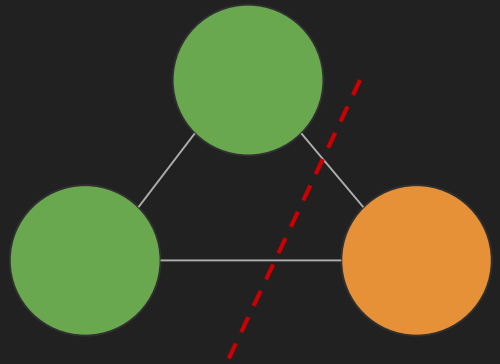
Tolerates fault of 1 node



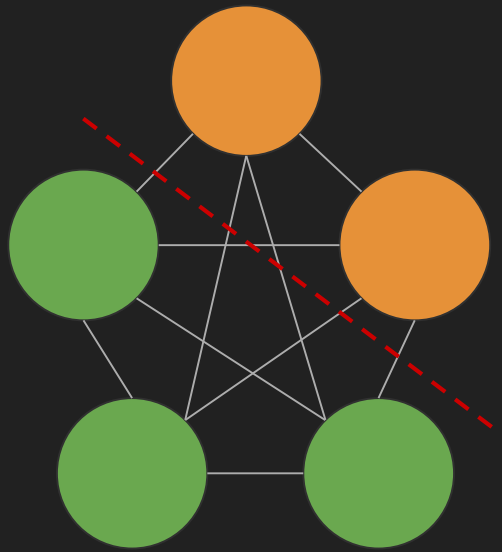
Tolerates fault of 2 nodes

Adaptive

RAFT - Network failure



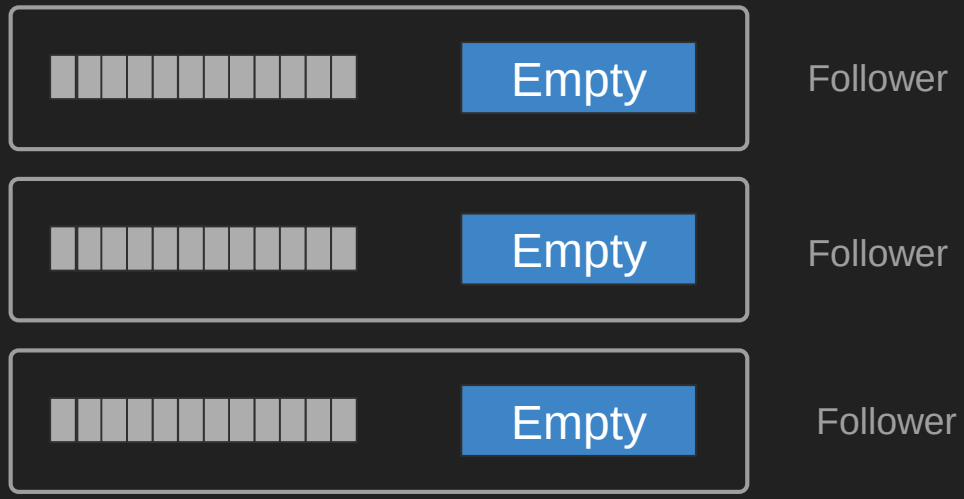
1 node partitioned from the other nodes



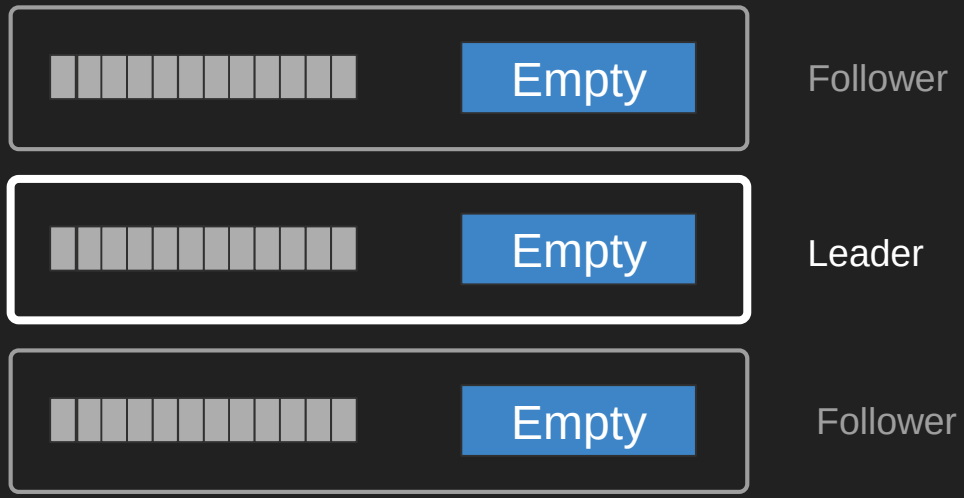
2 nodes partitioned from the other nodes

Adaptive

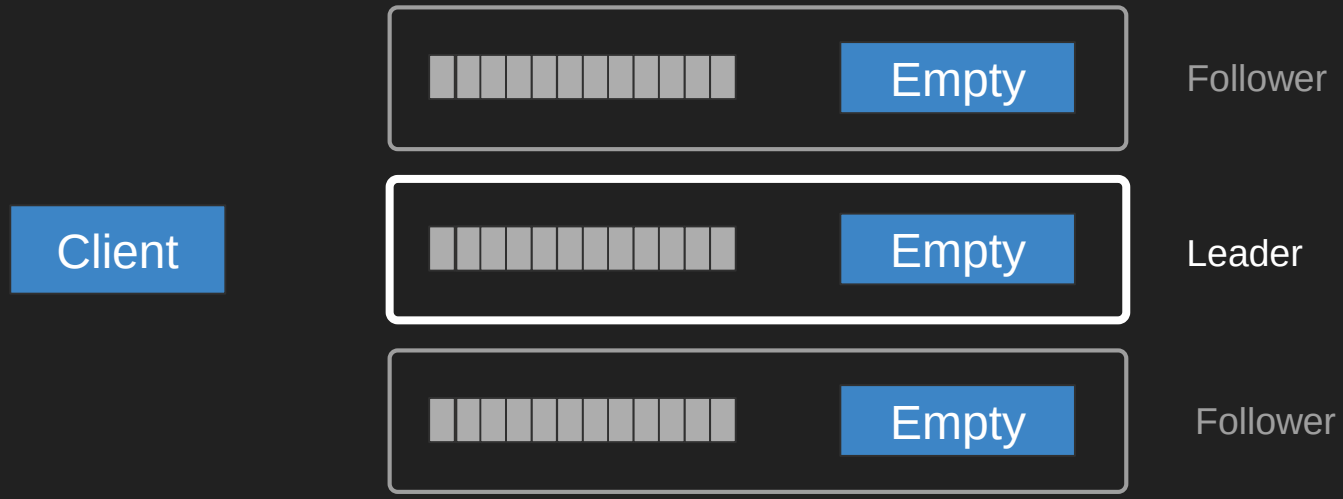
Leader election



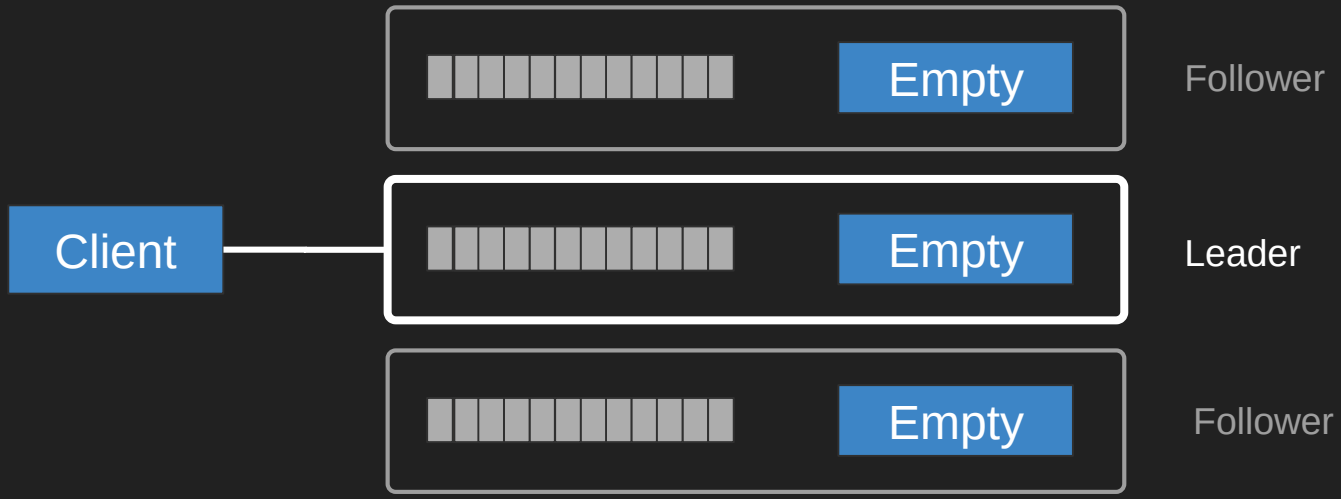
Leader election



Leader discovery

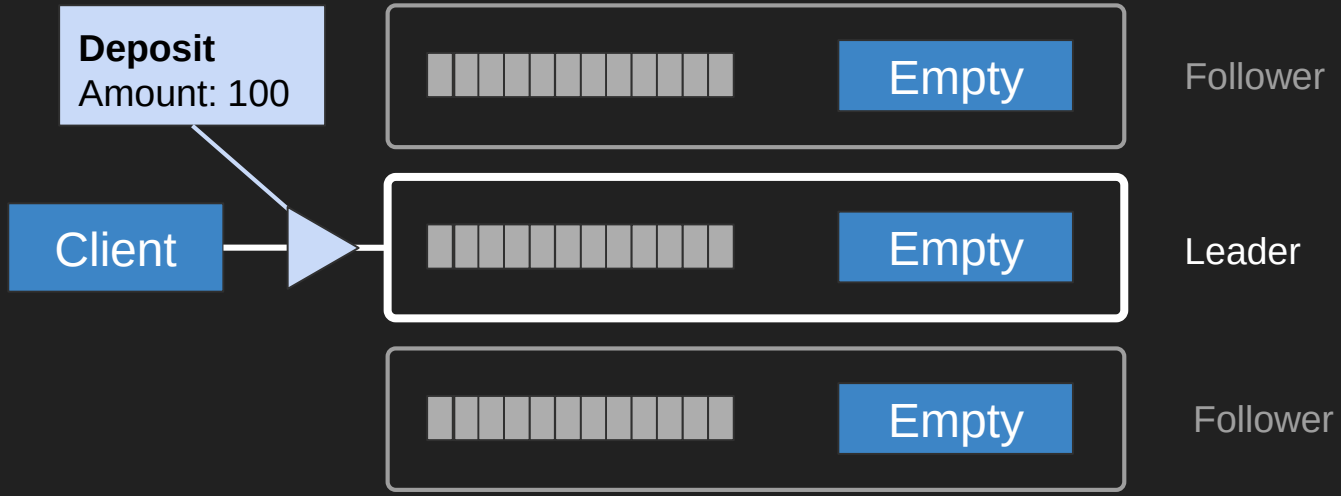


Client connects

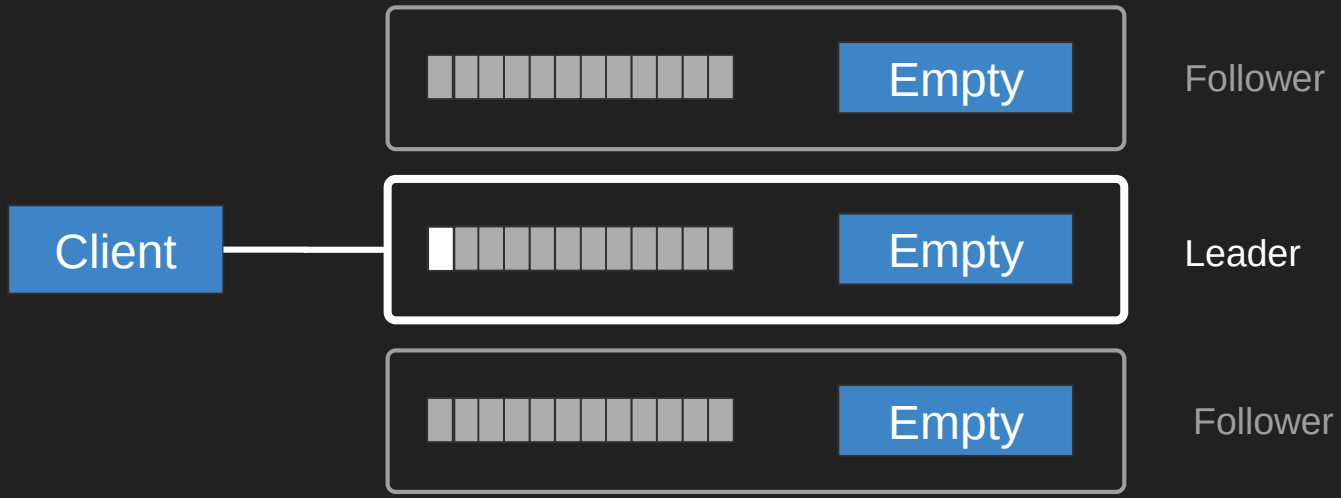


Adaptive

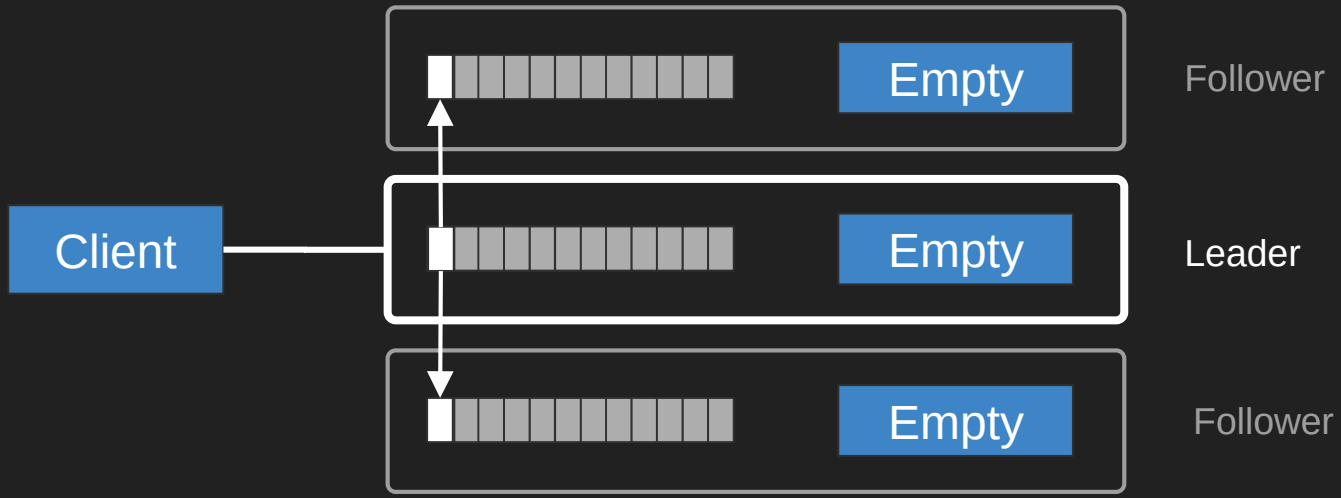
Client sends a command



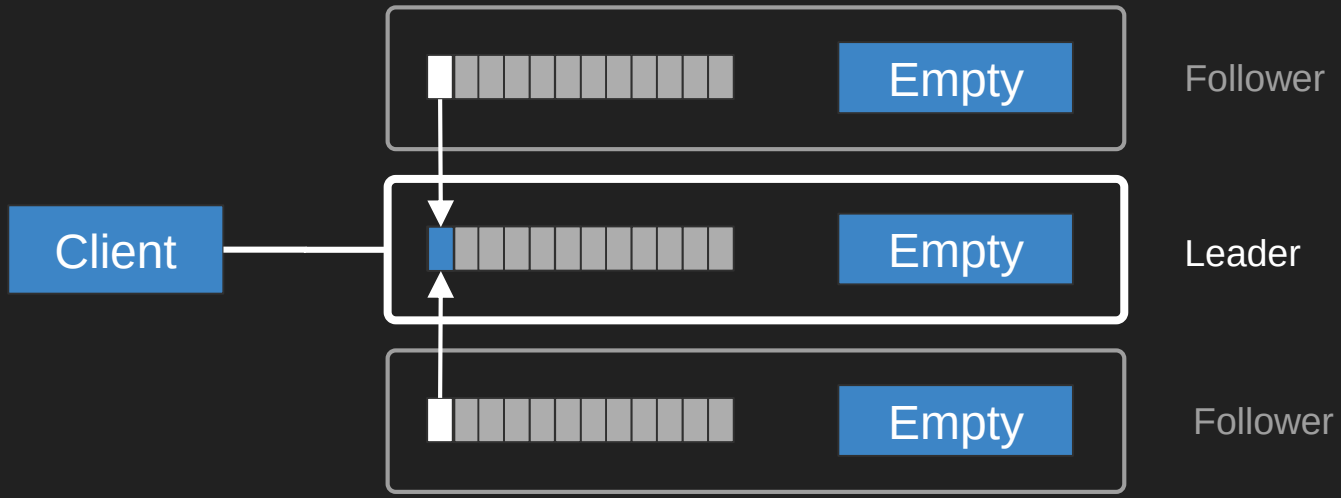
Leader writes command to its log



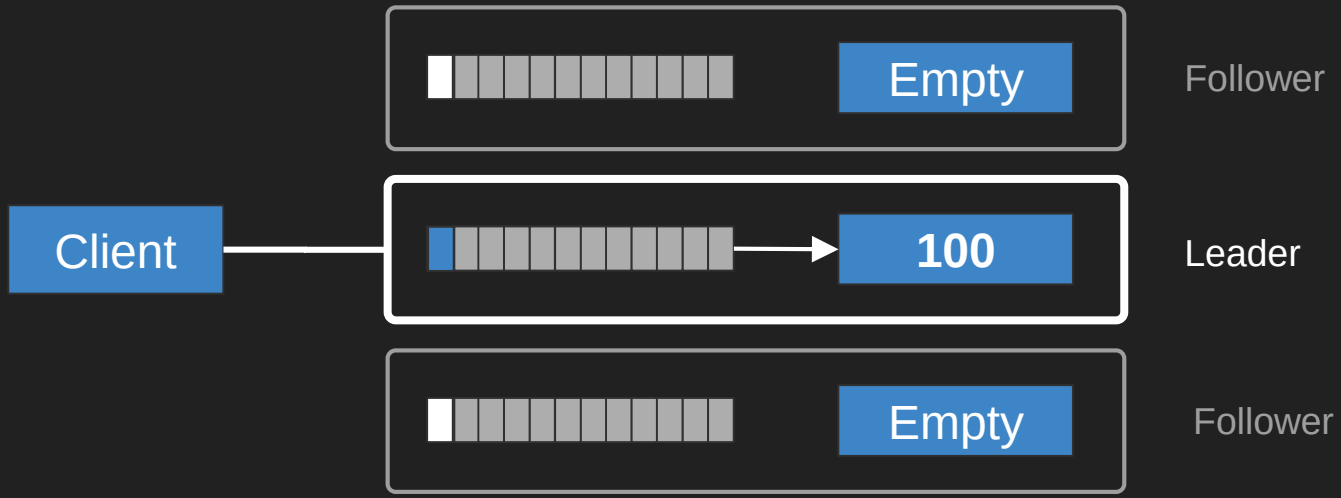
Command is replicated



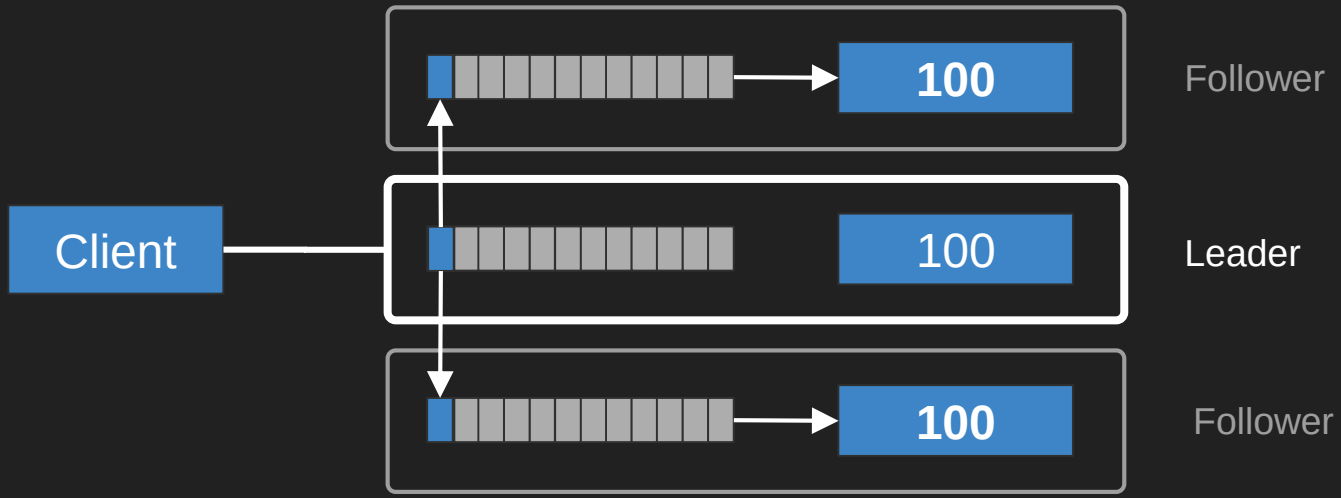
Followers Acknowledge



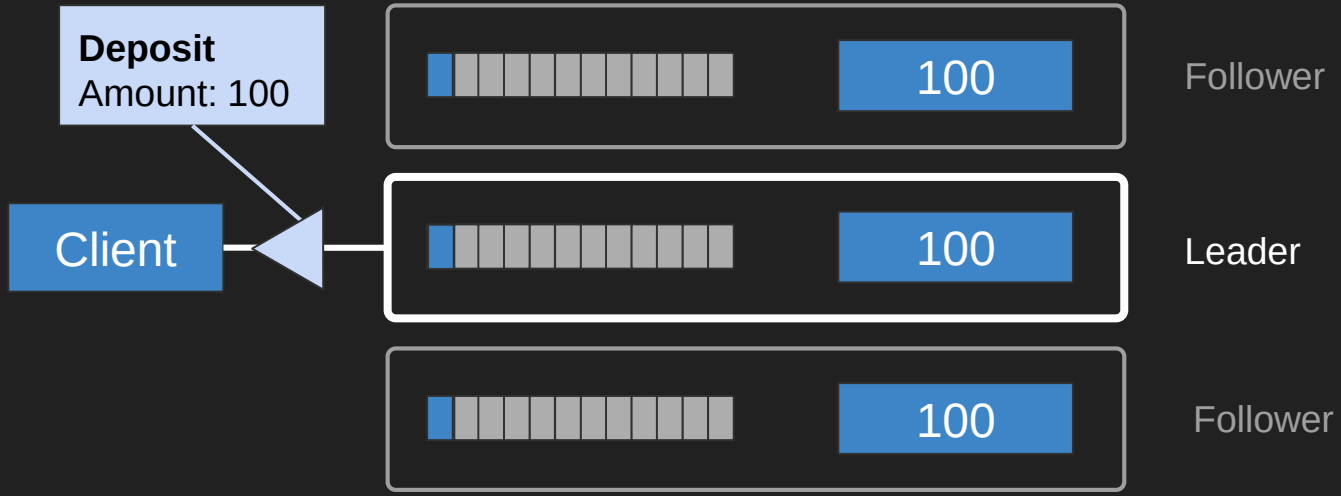
Leader commits command and applies to model



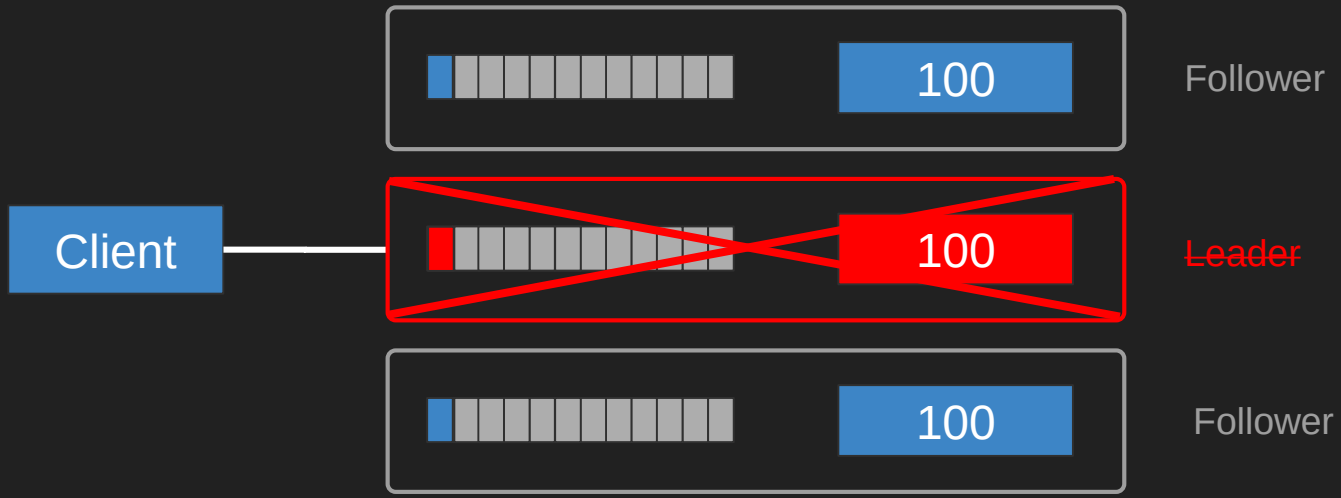
Leader notifies followers to commit



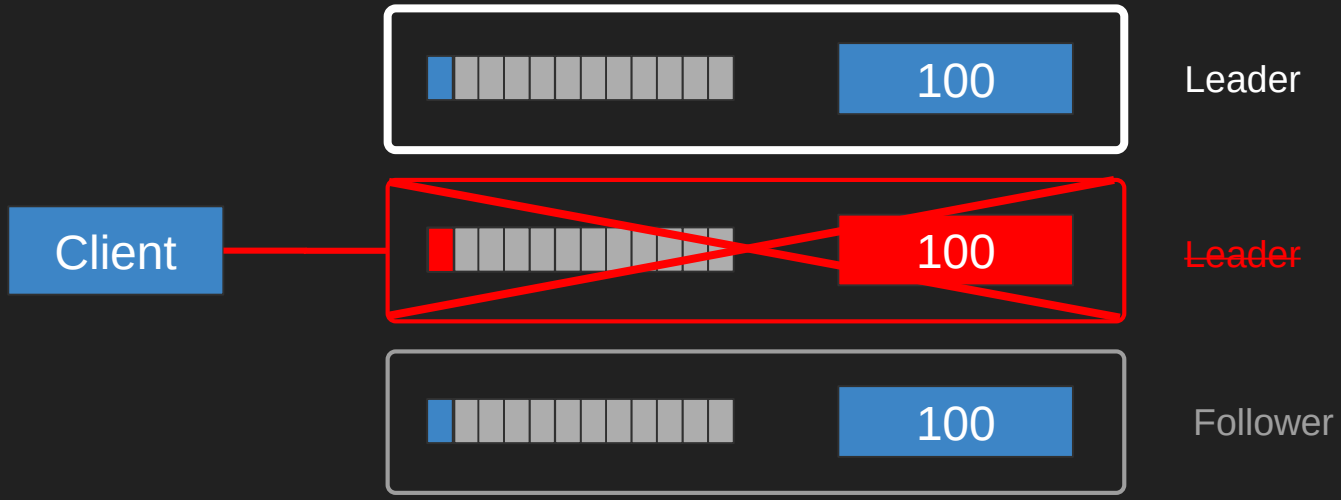
Leader notifies client



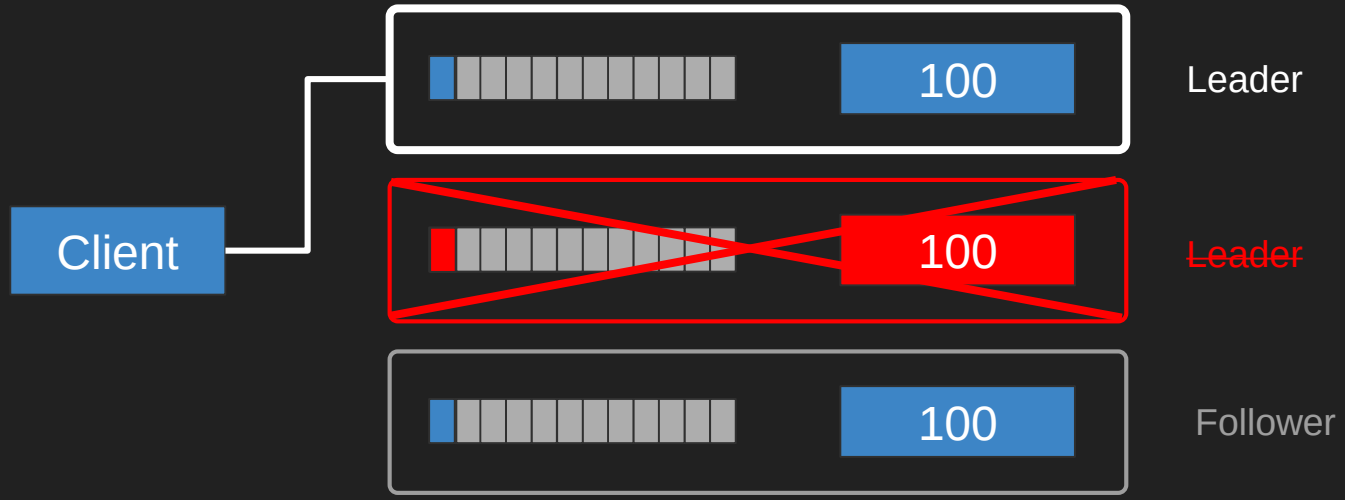
Leader dies



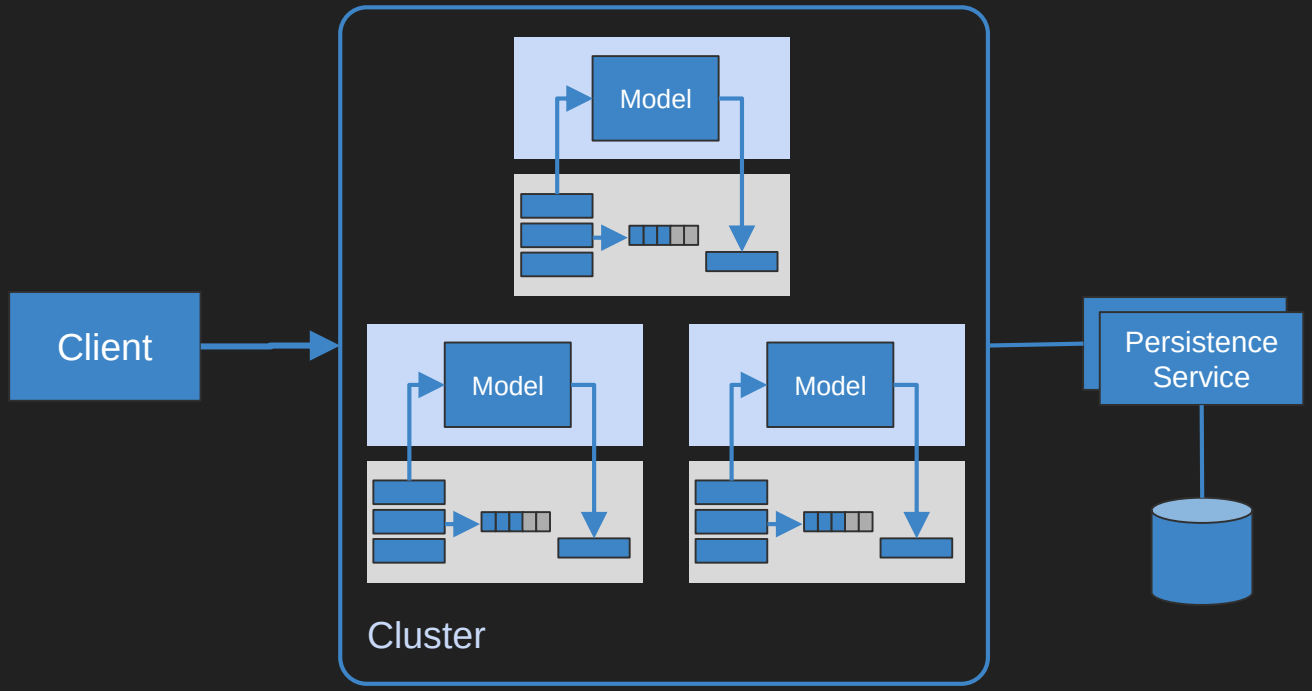
New leader election



Client times out and connects to new leader



2nd Iteration - Clustering



Benefits

- Fault tolerant
- Consistent model

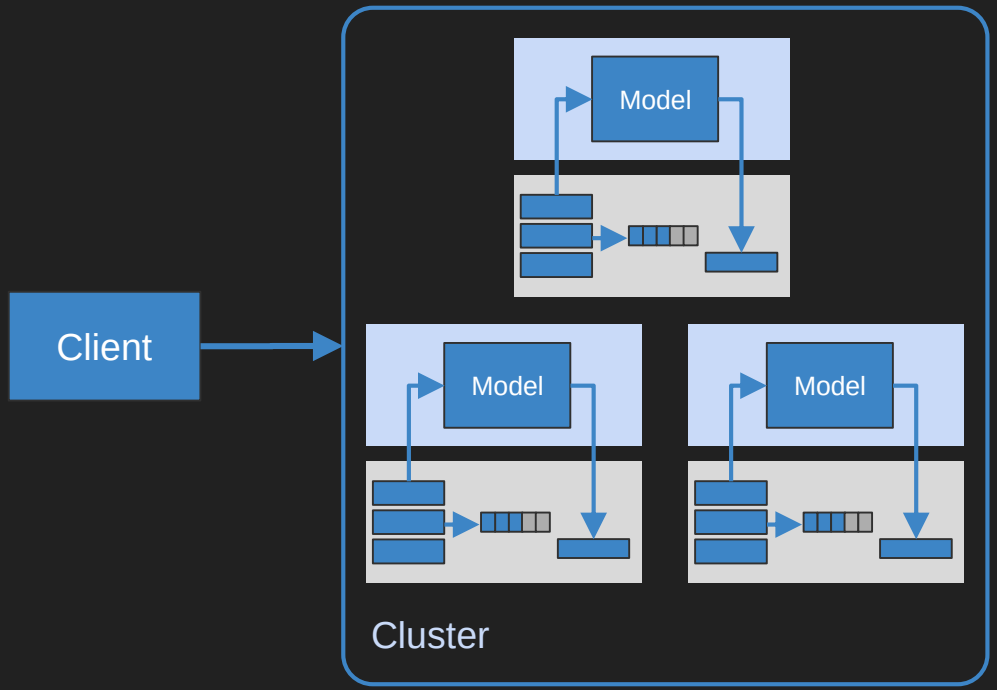
Pain points

- Database as golden source

Adaptive

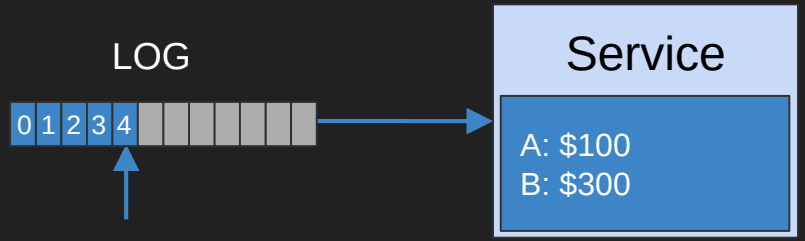
Durability

3rd Iteration - Durable Services



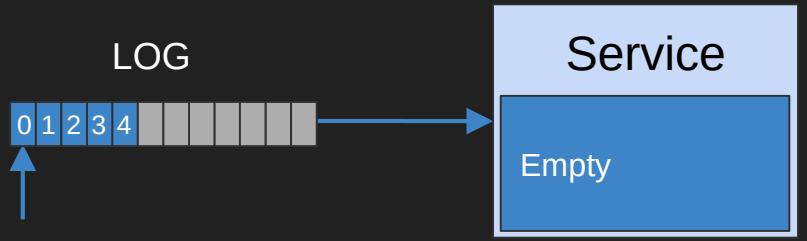
Adaptive

System restart

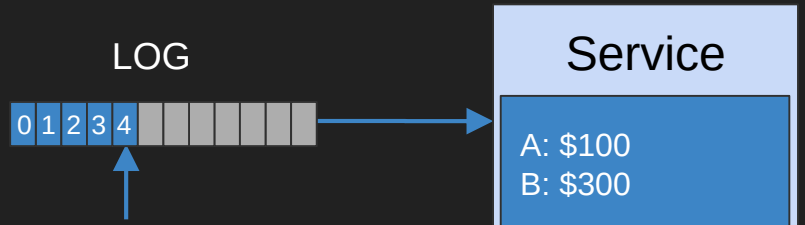


Stop the service

Start the service



Replay commands



Model is back in the same state because it's deterministic.

Managing the log

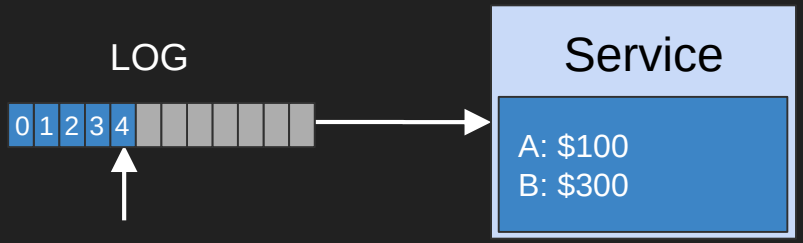
OK if

- Log grows slowly
- Your model has short lifetime

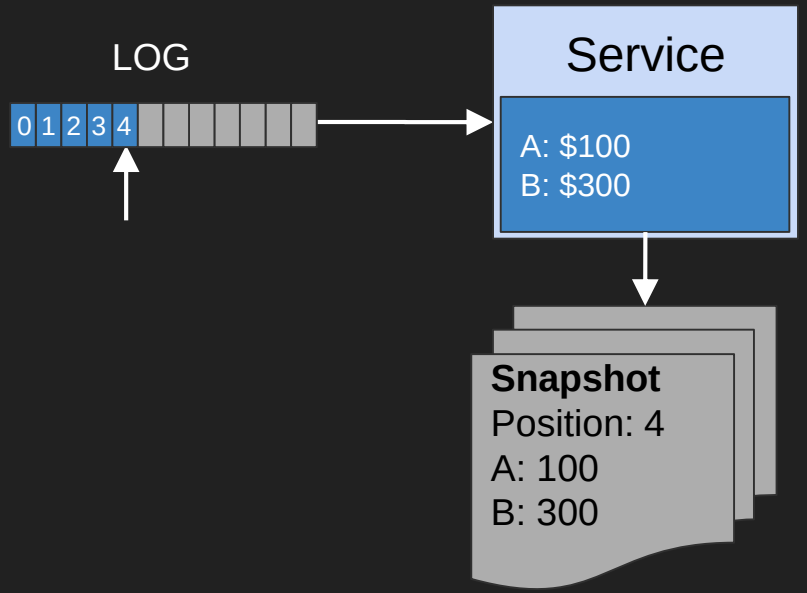
Problem: the size of the log affects recovery time

Need log compaction.

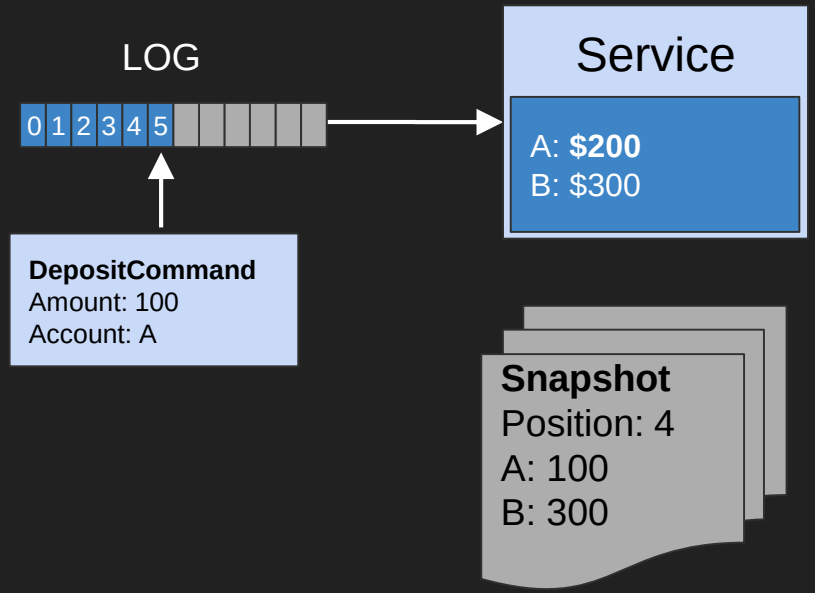
Snapshotting



Write state to disk

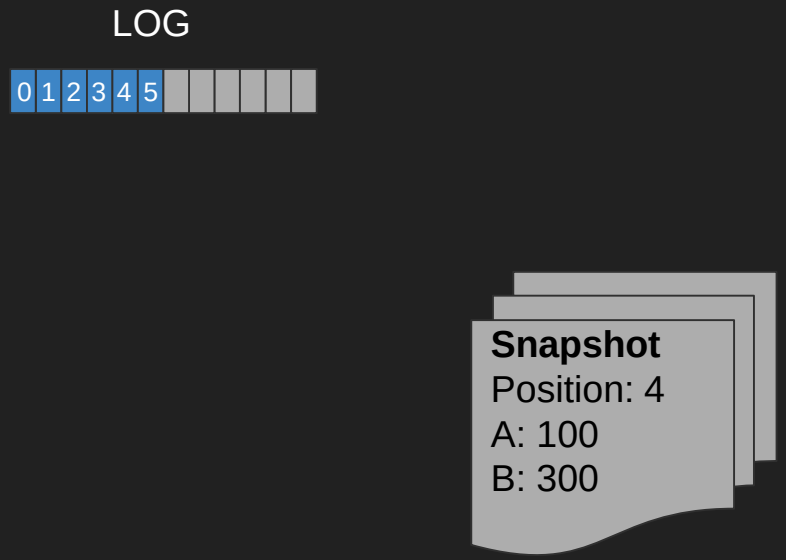


New Commands Are Processed



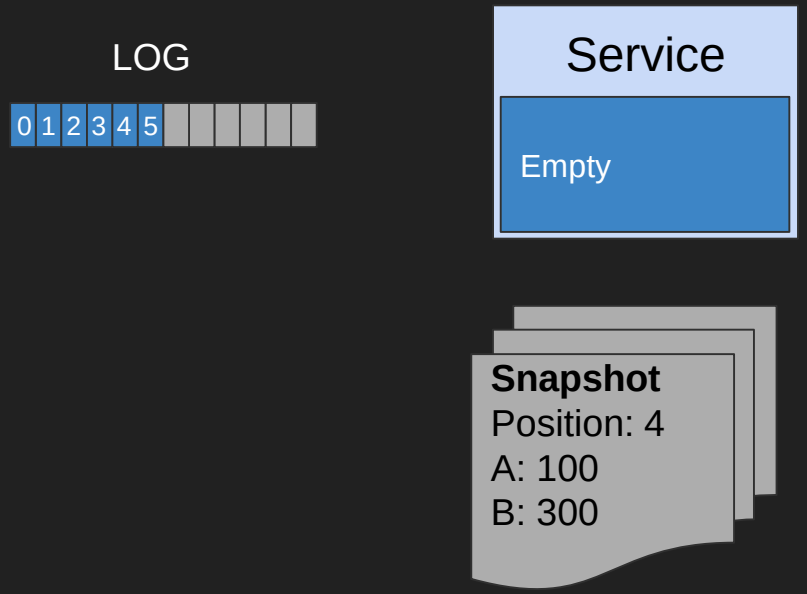
Adaptive

Stop the Service

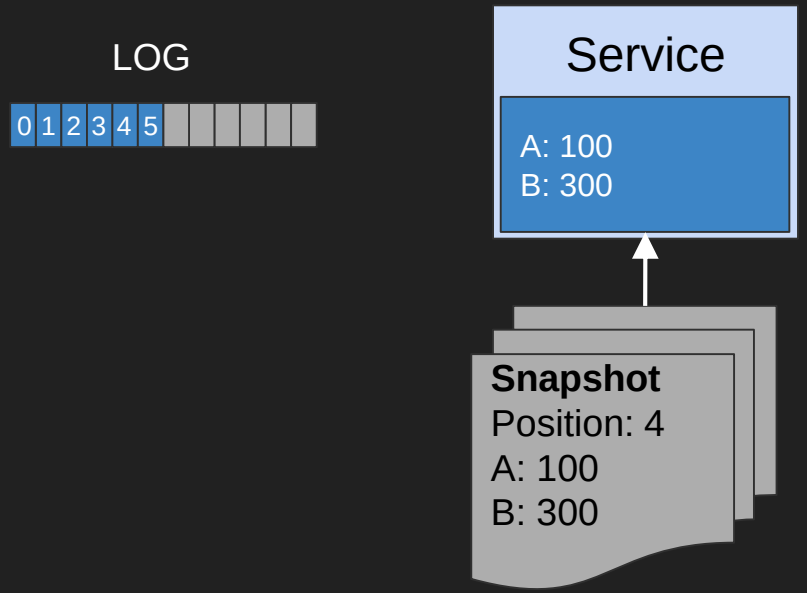


Adaptive

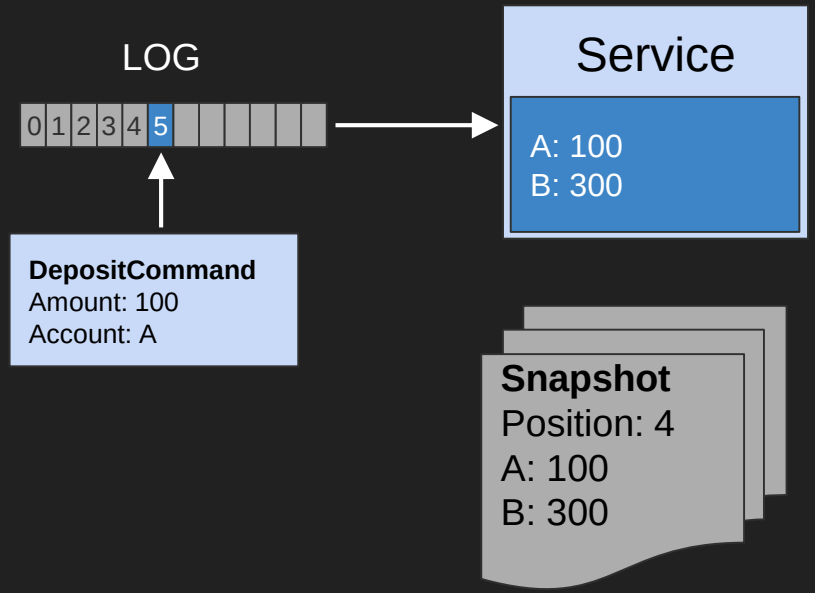
Restart the service



Restore the snapshot



Replay only commands from after the snapshot



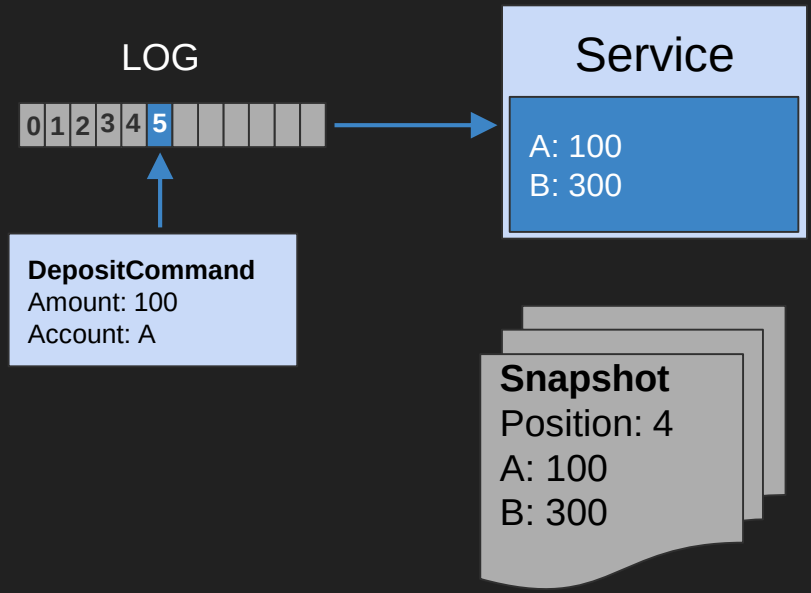
Adaptive

Snapshotting

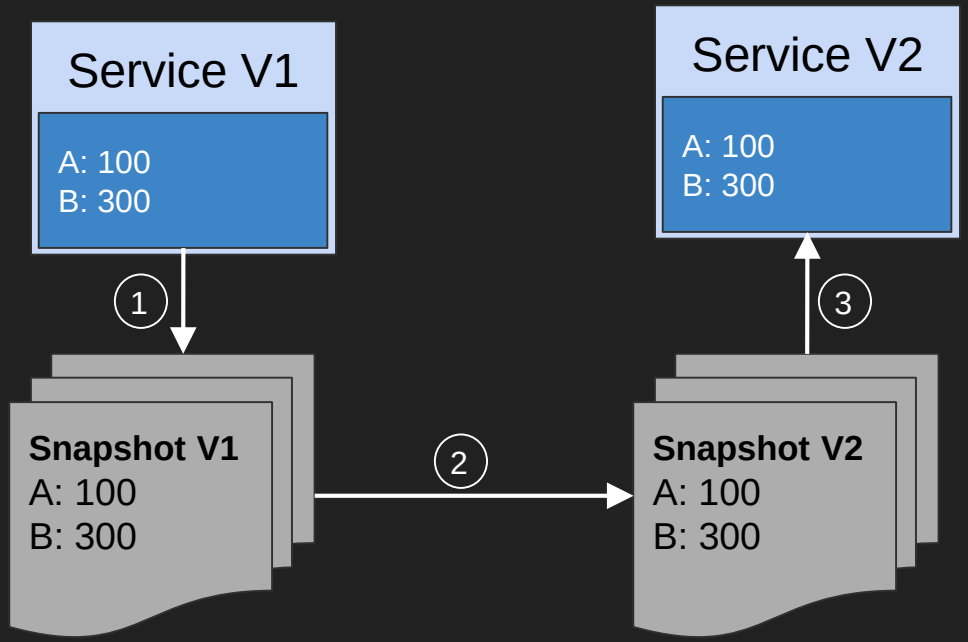
System is back in the same state.

Snapshotting makes it possible to trim the log.

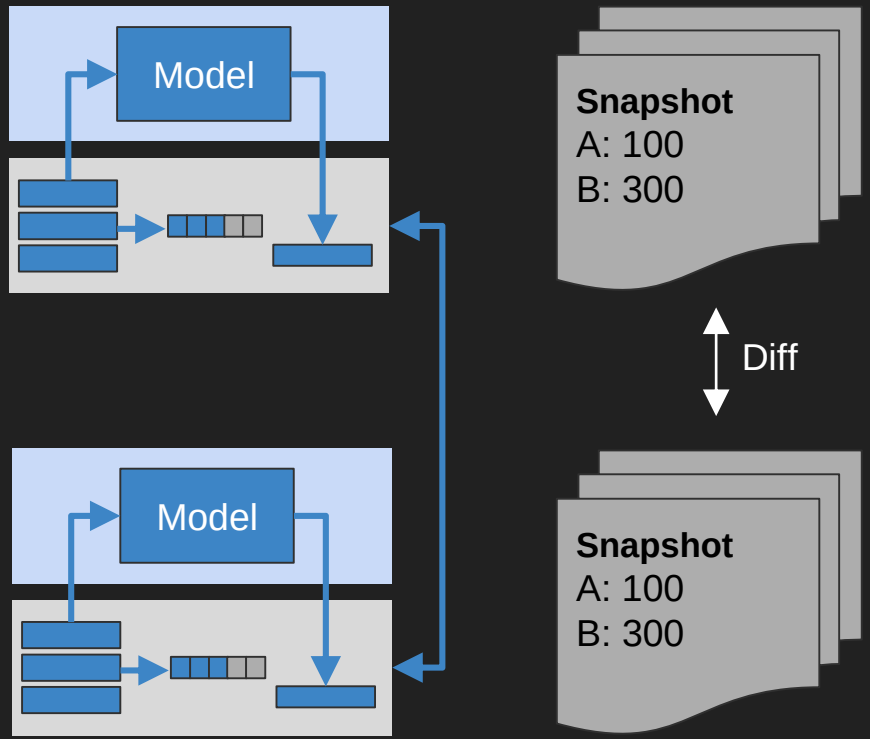
With a snapshot at position N, we can archive the log up to the same position.



Upgrades

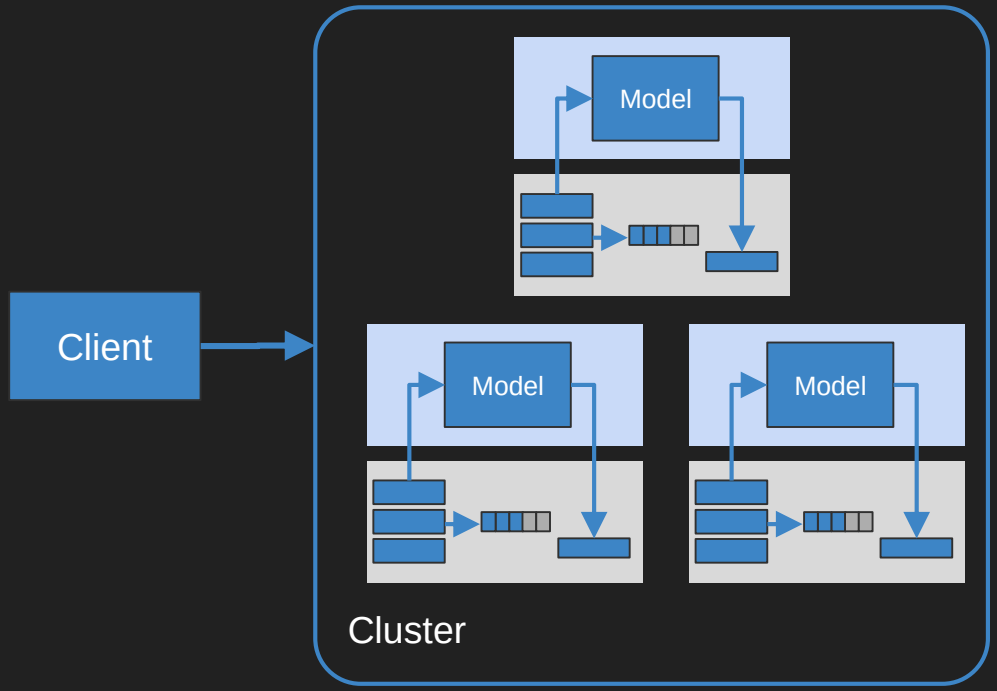

Service V1
A: 100
B: 300
1
Snapshot V1
A: 100
B: 300
2
Service V2
A: 100
B: 300
3
Snapshot V2
A: 100
B: 300

Consistency Check



Adaptive

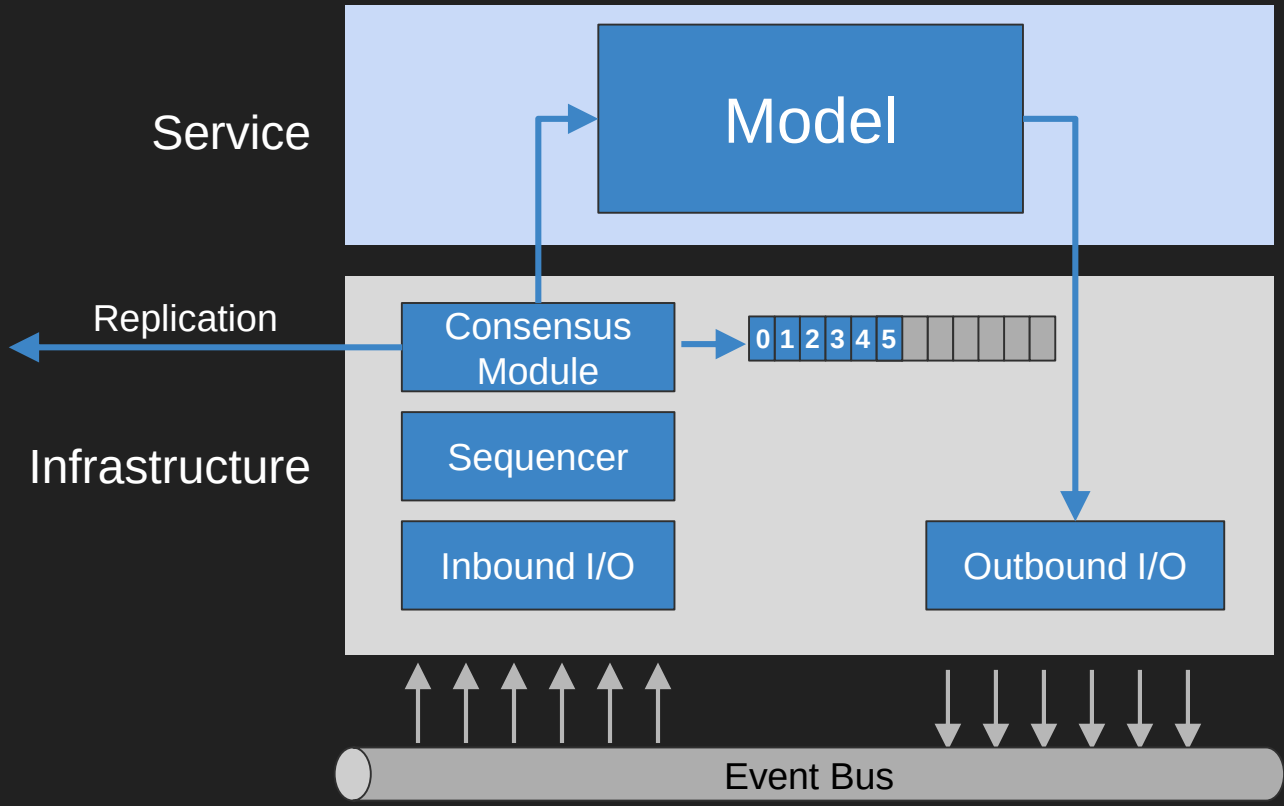
Iteration 3 - Durable Services



Benefit

State is managed in one place

Reactive Application Server



Simple Model
Easy To Debug
Fast

Fault Tolerance
State Replication
Durability
Concurrency
Messaging

Adaptive

Don’t build the infrastructure yourself!

Open source

- Copycat - http://atomix.io/
- Aeron - https://github.com/real-logic/aeron

Martin Thompson

Todd Montgomery

Adaptive

Other use-cases

- Online games
  - MMO
  - Gambling
- Ticketing system
- Consistent Databases / Caches
- Many more..

Wrapping up

- Demonstrated the simplicity of this approach
- Widely applicable
- Deserves more attention!
- Learn more
  - LMAX Architecture https://goo.gl/q1iSCB
  - Raft paper and website - https://raft.github.io/
  - The Log - https://goo.gl/m4iWqn
  - White paper - http://weareadaptive.com/blog

Thank you!

Any questions?

http://weareadaptive.com/careers/
careers@weareadaptive.com

Adaptive

Possible topics

Out of scope

- Timers & scheduling
- Testing with time
  - 23-25 hour day exemple
- Messaging
  - Code gen stuff
  - Error scenario / state machine

Adaptive