leading practices

leading practices

events should be processed in realtime
(i.e. within a few seconds)

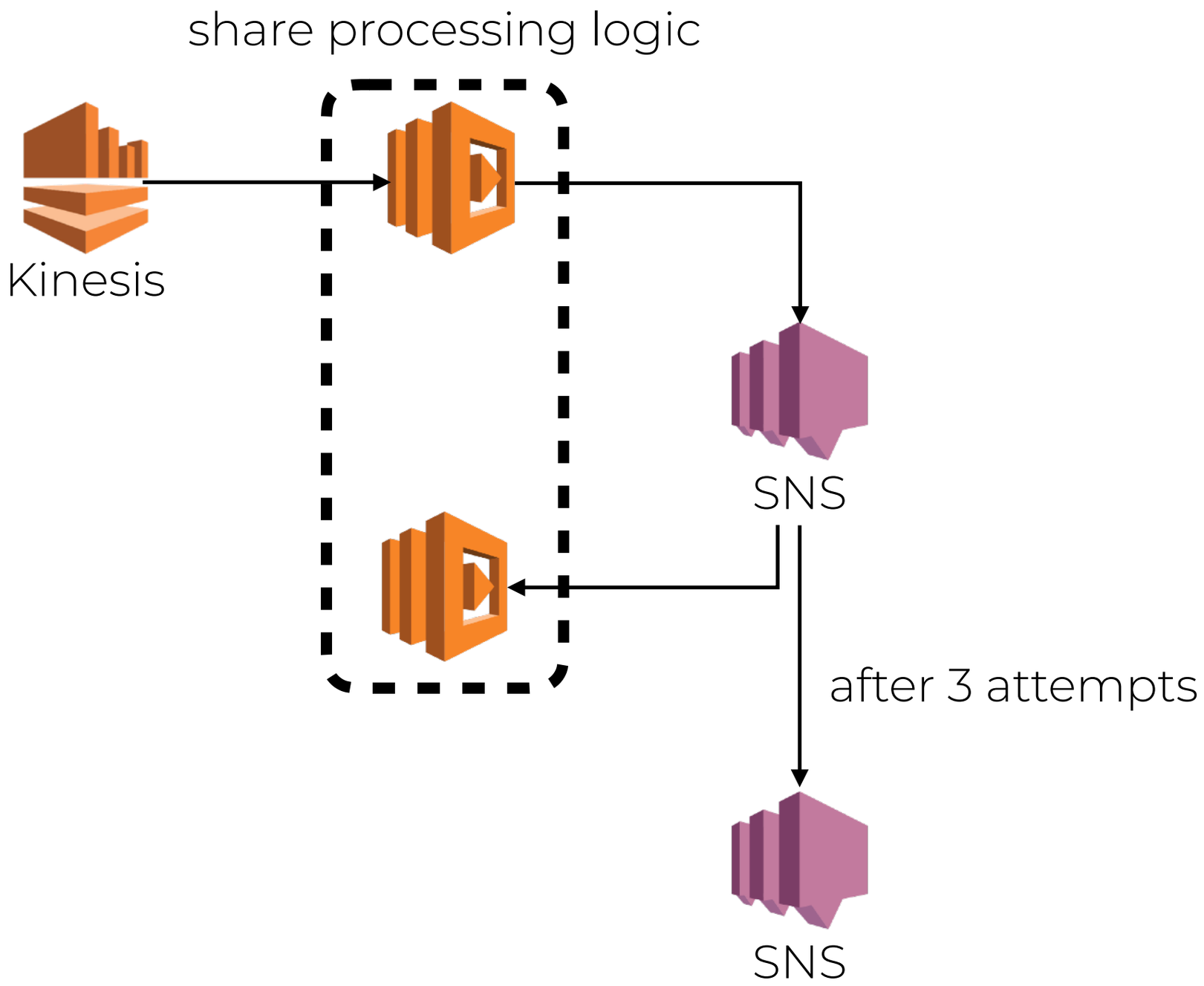
leading practices

failed events should be retried, but the retries **should not violate** the realtime constraint

leading practices

unprocessed events should be retrievable
(i.e. available for human intervention, or root cause analysis)

leading practices



leading practices

Put Requests	$0.014 per mill (at 25KB chunks)
Shard Hours	$0.015 per hour per shard 1MB/s ingress, 1000 records/s 2MB/s egress, 5 read req/s
Extended Retention	$0.020 per shard hour

http://amzn.to/2ubyaot

leading practices

optimise payload size

Put Requests	$0.014 per mill (at 25KB chunks)
Shard Hours	$0.015 per hour per shard 1MB/s ingress, 1000 records/s 2MB/s egress, 5 read req/s
Extended Retention	$0.020 per shard hour

http://amzn.to/2ubyaot

leading practices

consider using a binary format such as Protocol Buffer and Thrift

leading practices

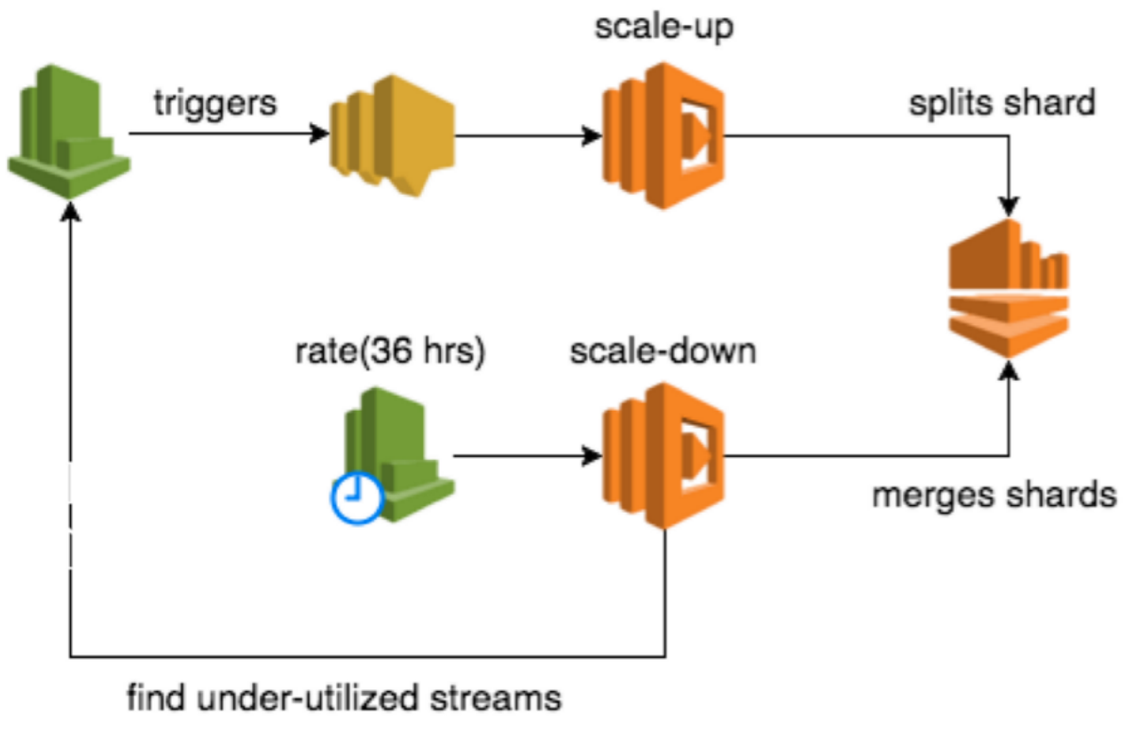
alarm on **IteratorAge** and

WriteProvisionedThroughputExceeded

Yan Cui
Senior Developer @ Space Ape Games.
May 5 · 6 min read

Auto-scaling Kinesis streams with AWS Lambda

A recipe for creating a cost-effective solution for auto-scaling Kinesis streams using Lambda functions

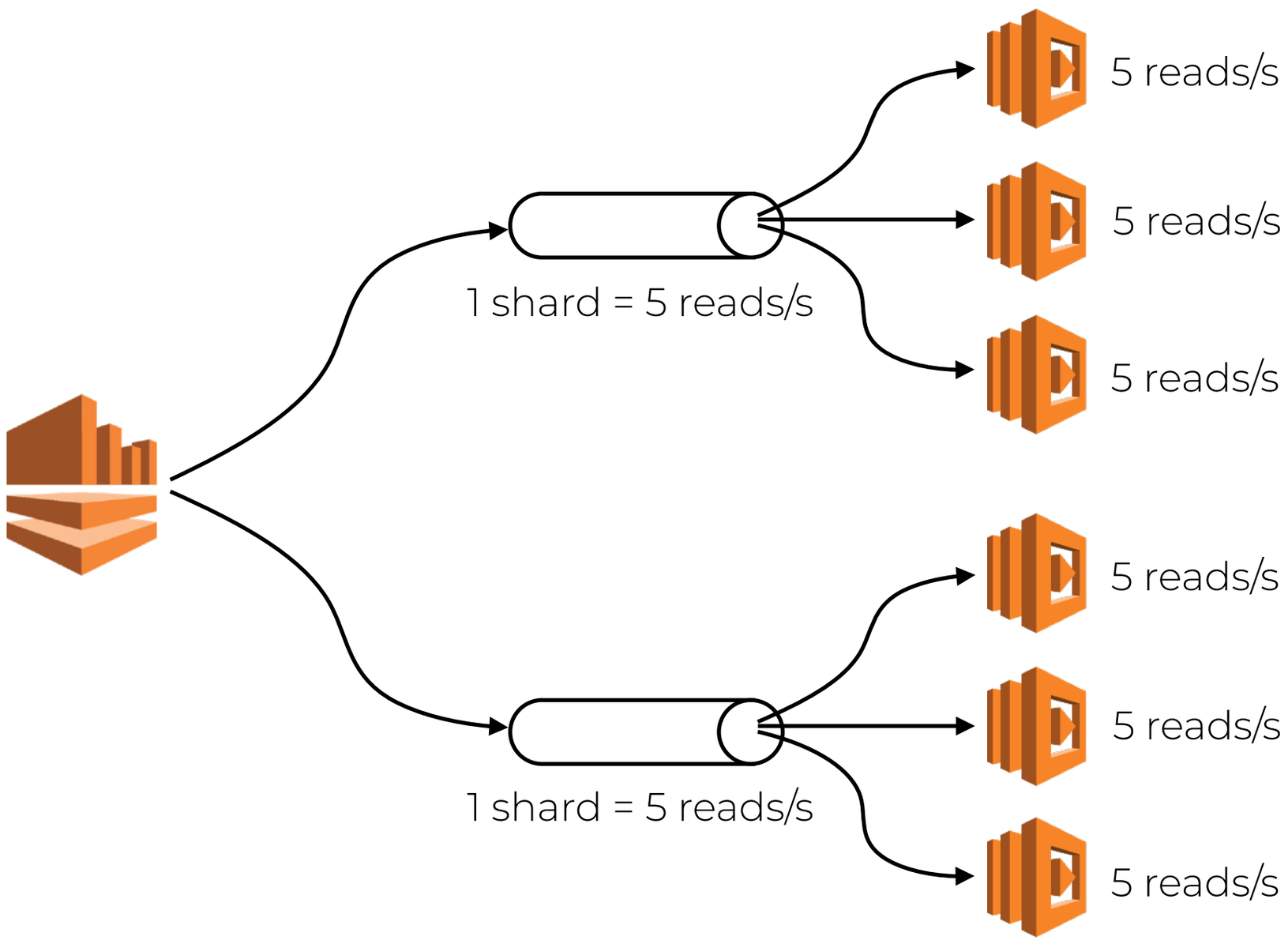


http://bit.ly/2hxZGui

leading practices

avoid “hot” streams

leading practices



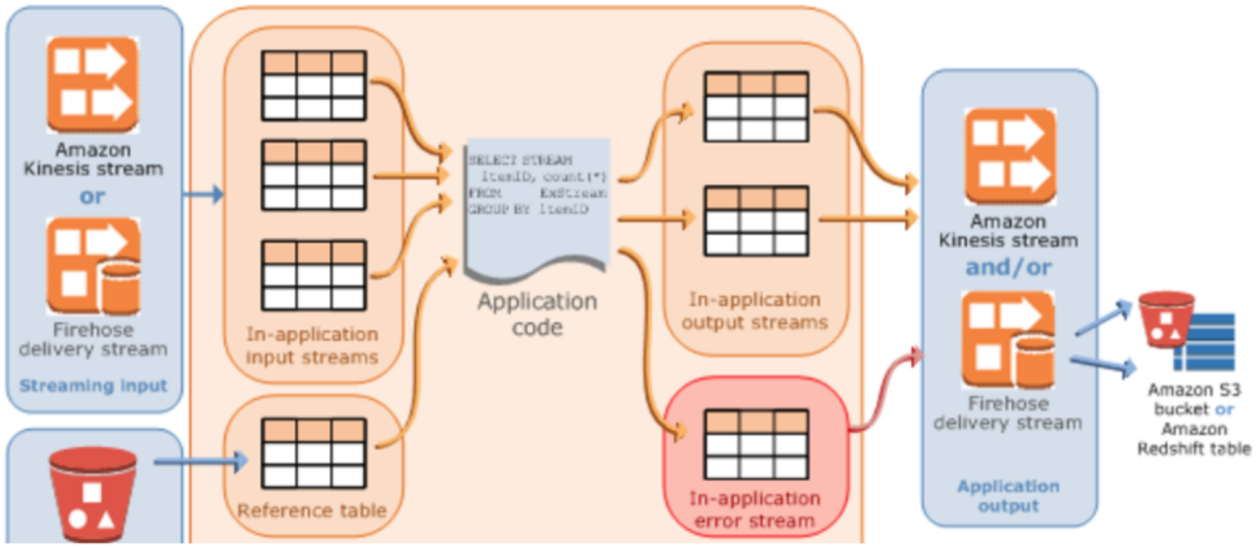
AWS Lambda docs

If your stream has 100 active shards, there will be at most 100 Lambda function invocations running concurrently. Then, each Lambda function processes events on a shard **in the order that they arrive**.

http://amzn.to/2va5GN5

leading practices

Amazon Kinesis Streams fan-out via Kinesis Analytics - made with serverless ⚡



Source: https://github.com/alexcasalboni/kinesis-streams-fan-out-kinesis-analytics

How to fan-out Amazon Kinesis Streams?

Published on June 23, 2017

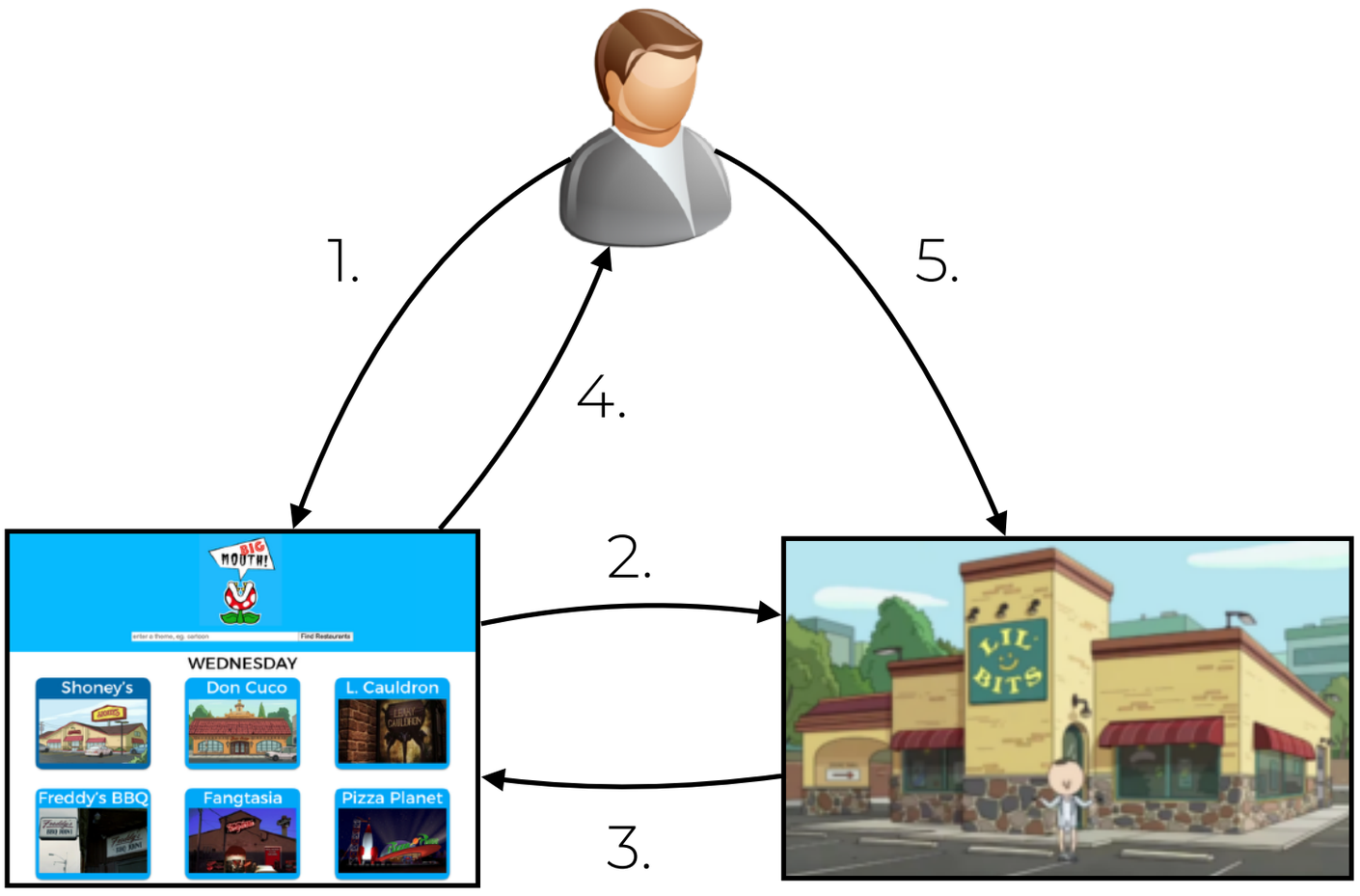
Alex Casalboni | ✓ Following
Cloud Evangelist, Serverless lover, AI enthusiast
4 articles

26 0 23

http://bit.ly/kinesis-fan-out

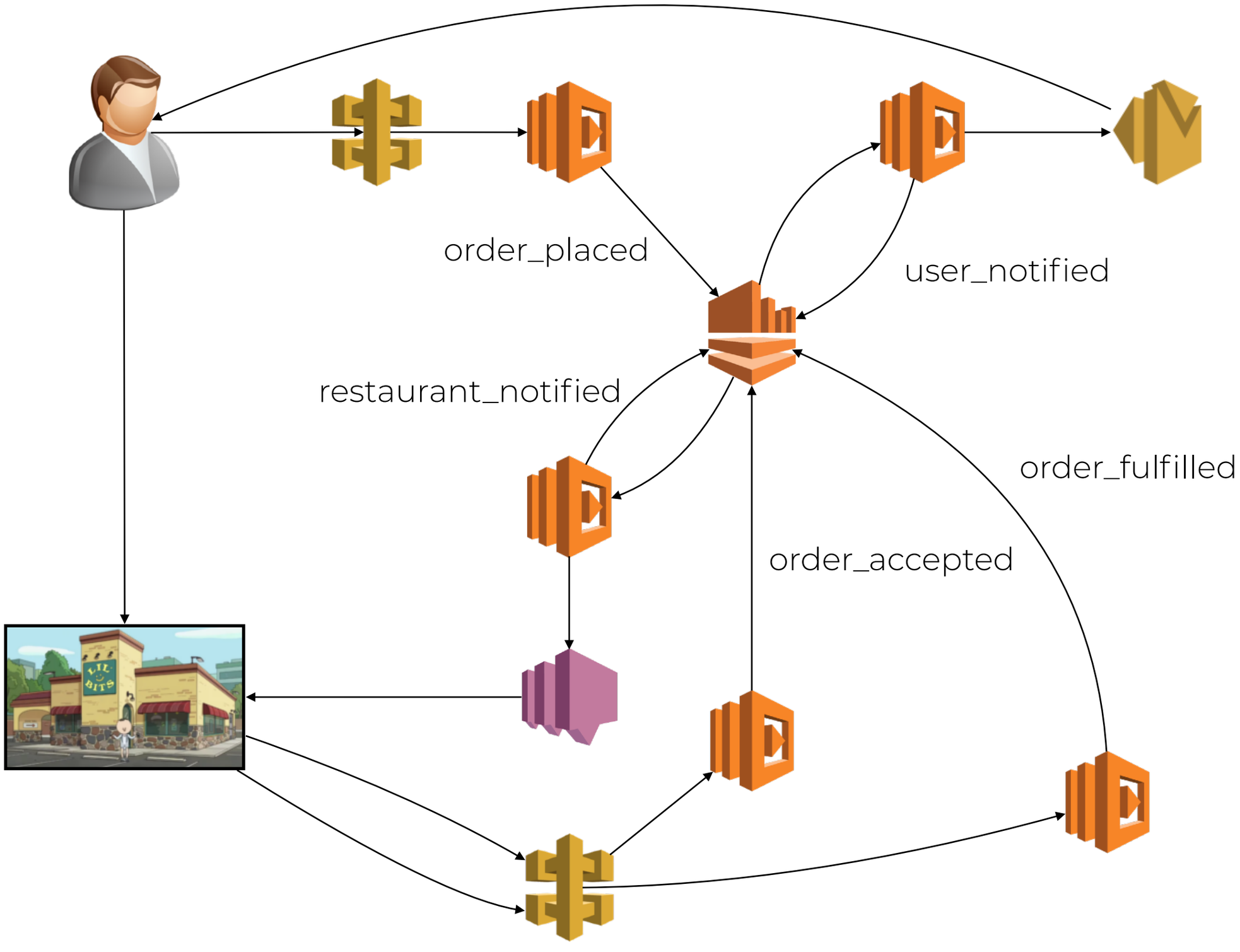
A case against events

order flow



1. customer places order
2. notifies restaurant
3. restaurant accepts order
4. customer confirmation
5. pick up

order flow


order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted
LIL BITS

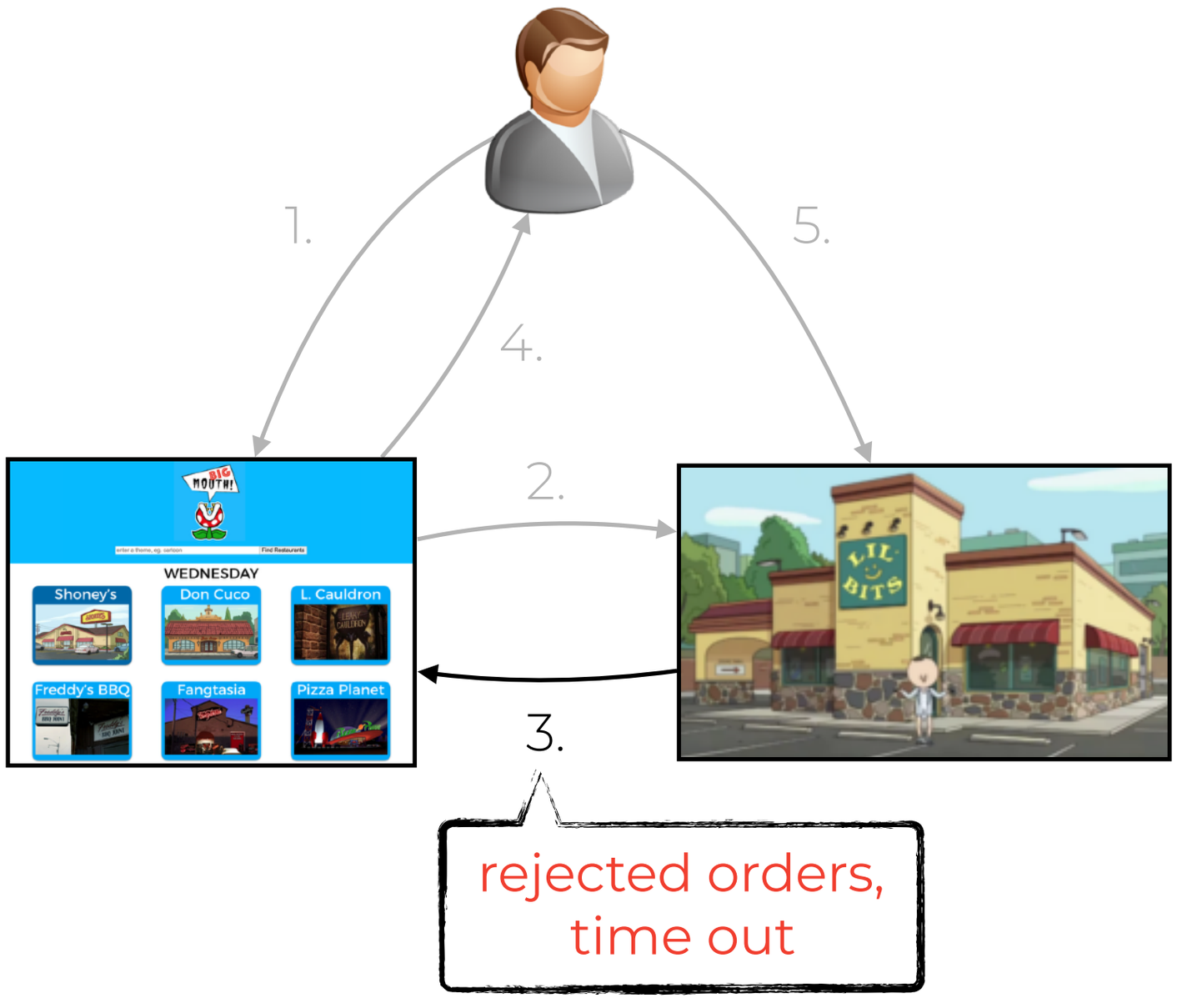
event-driven

- loose coupling
- scalability

failure modes

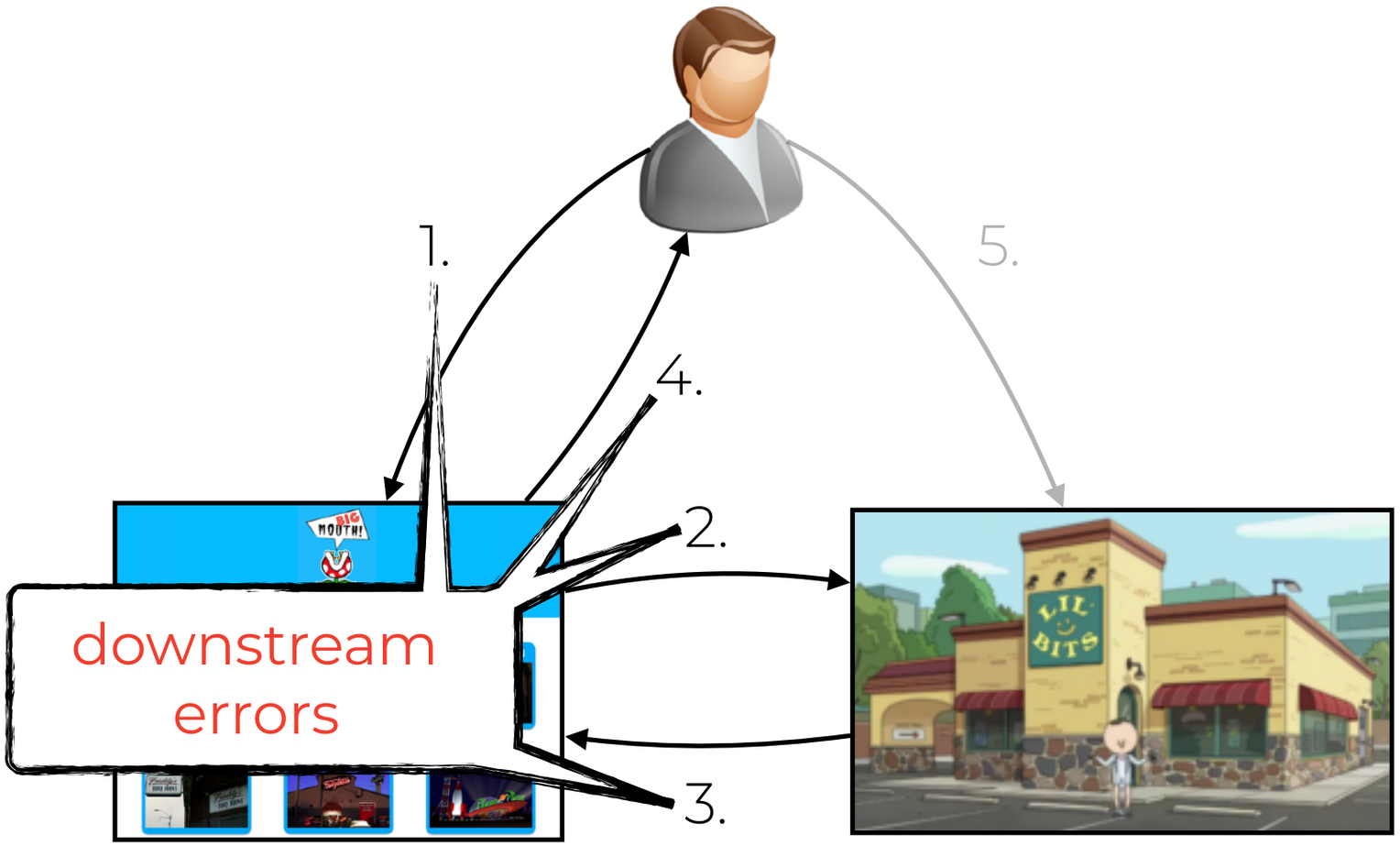
1. customer places order
2. notifies restaurant
3. ant accepts order
4. er confirmation

failure modes



1. customer places order
2. notifies restaurant
3. restaurant accepts order
4. customer confirmation
5. pick up

failure modes

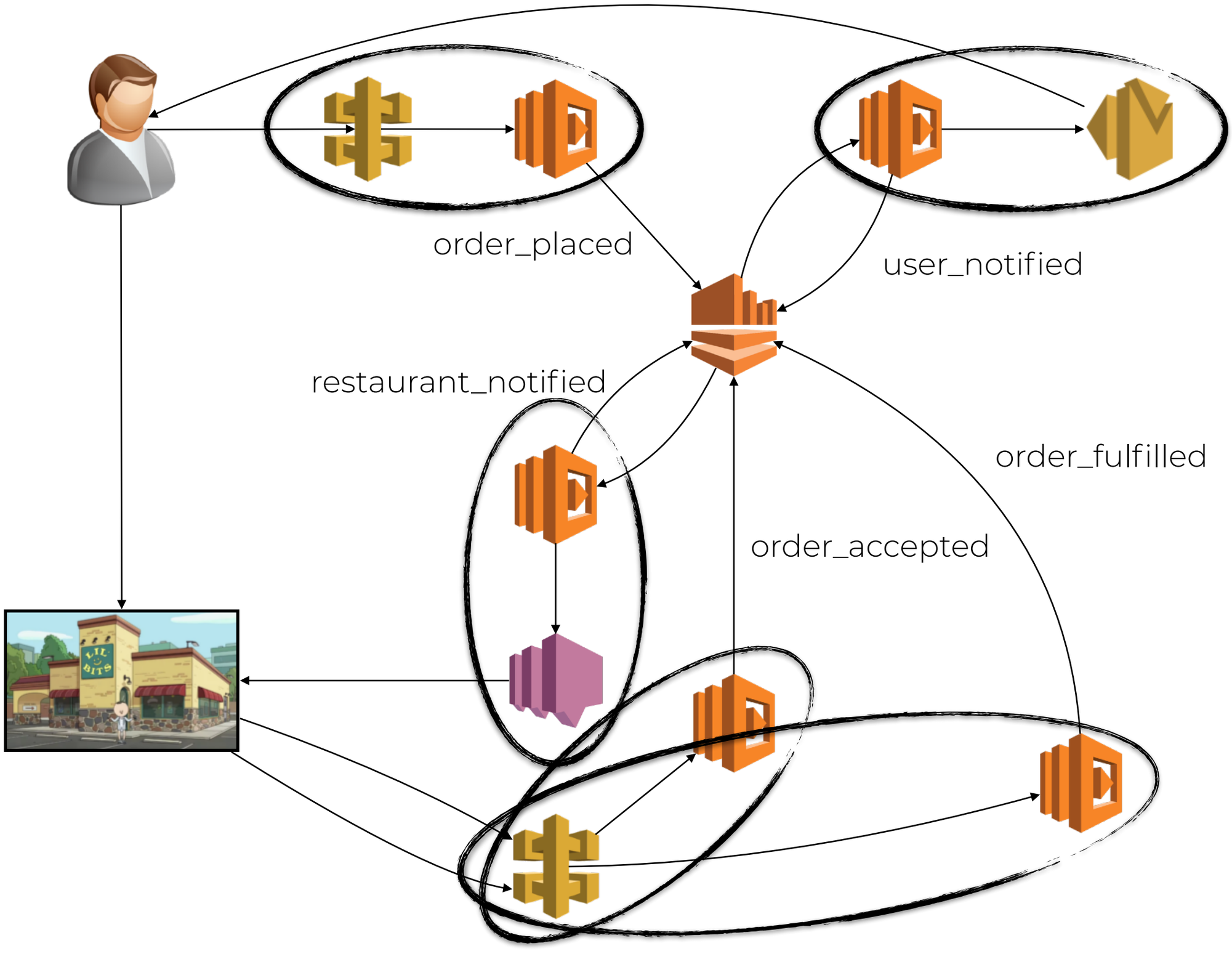


1. customer places order
2. notifies restaurant
3. restaurant accepts order
4. customer confirmation
5. pick up

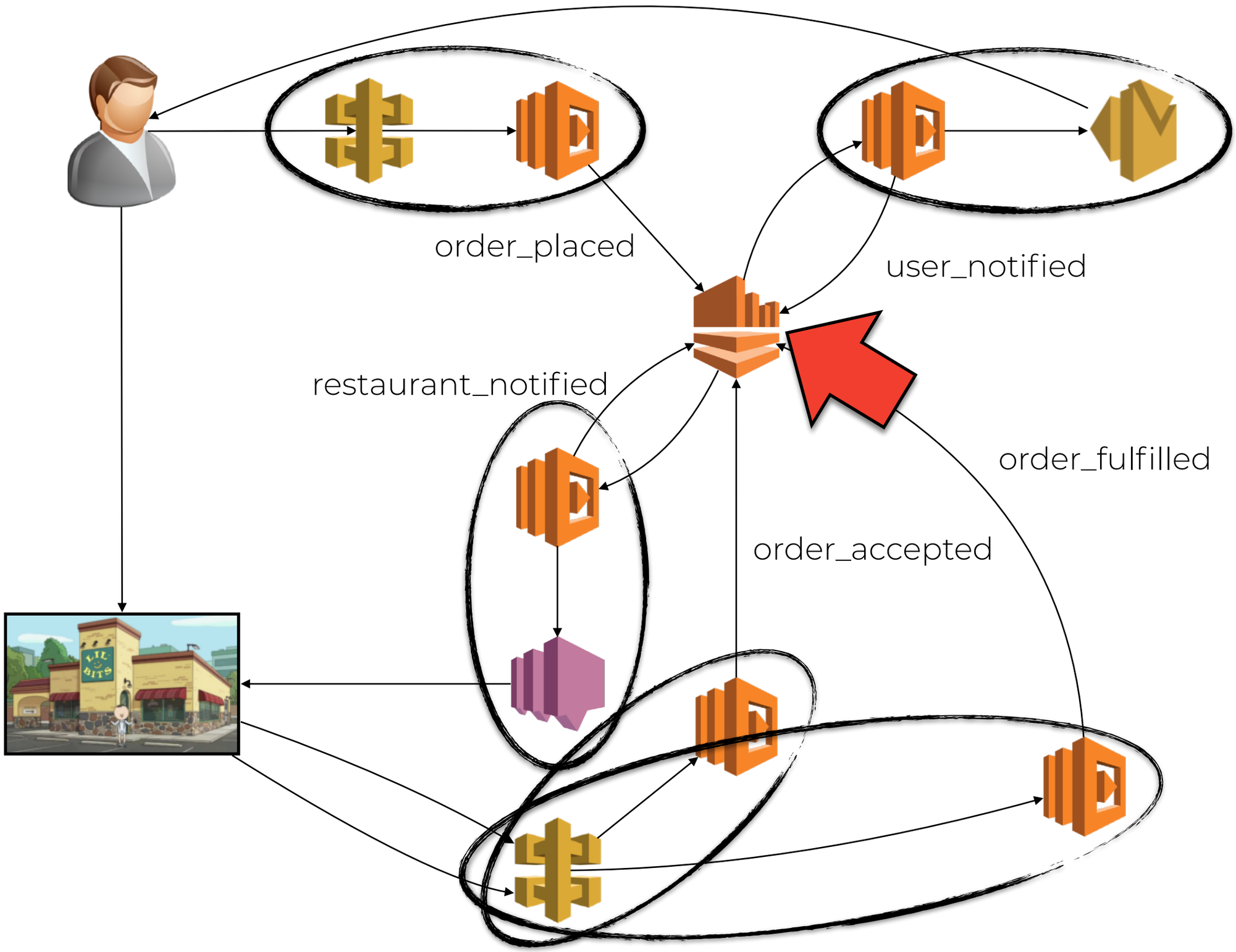
failure modes

- minimise impact on user experience

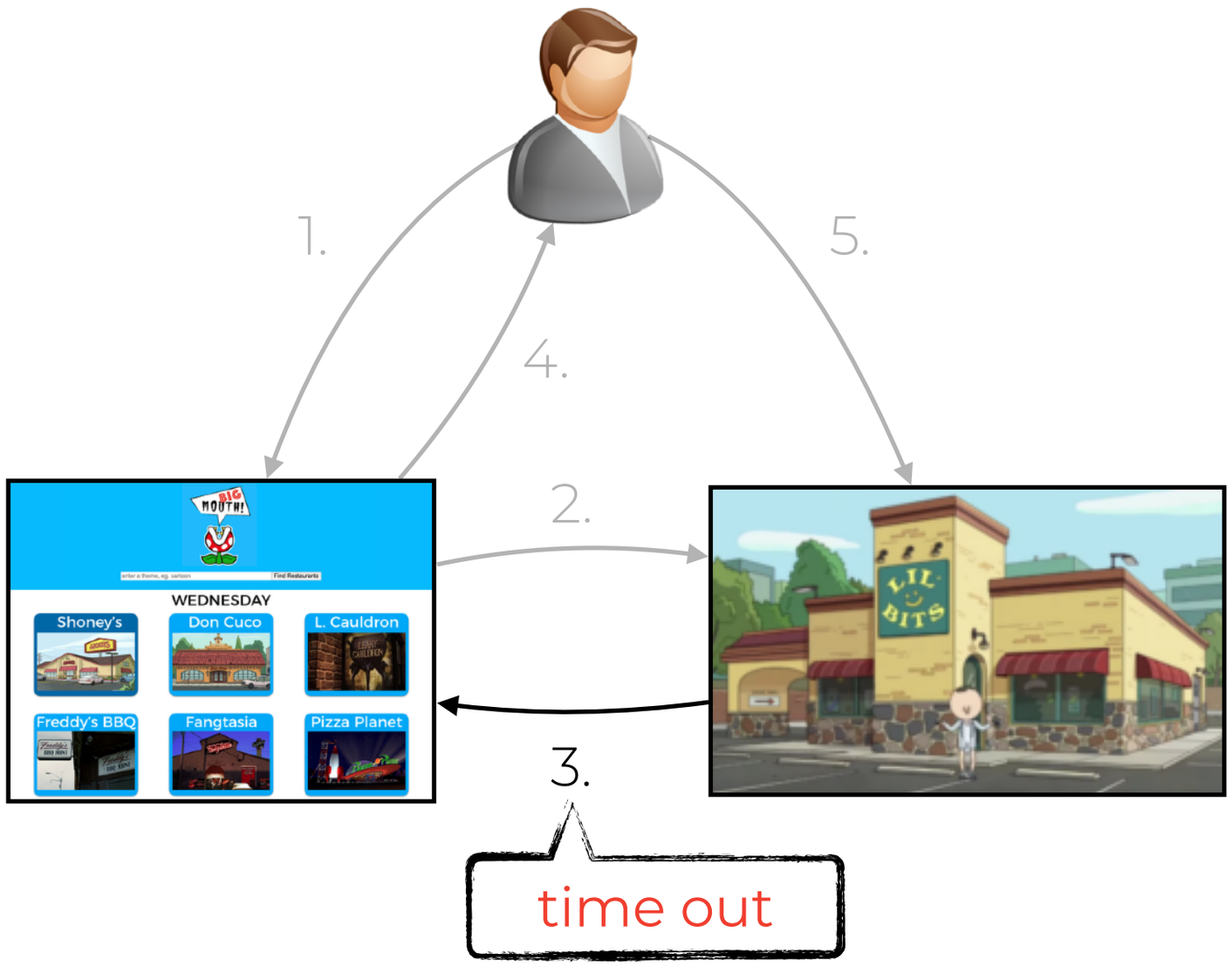
order flow


order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted

order flow

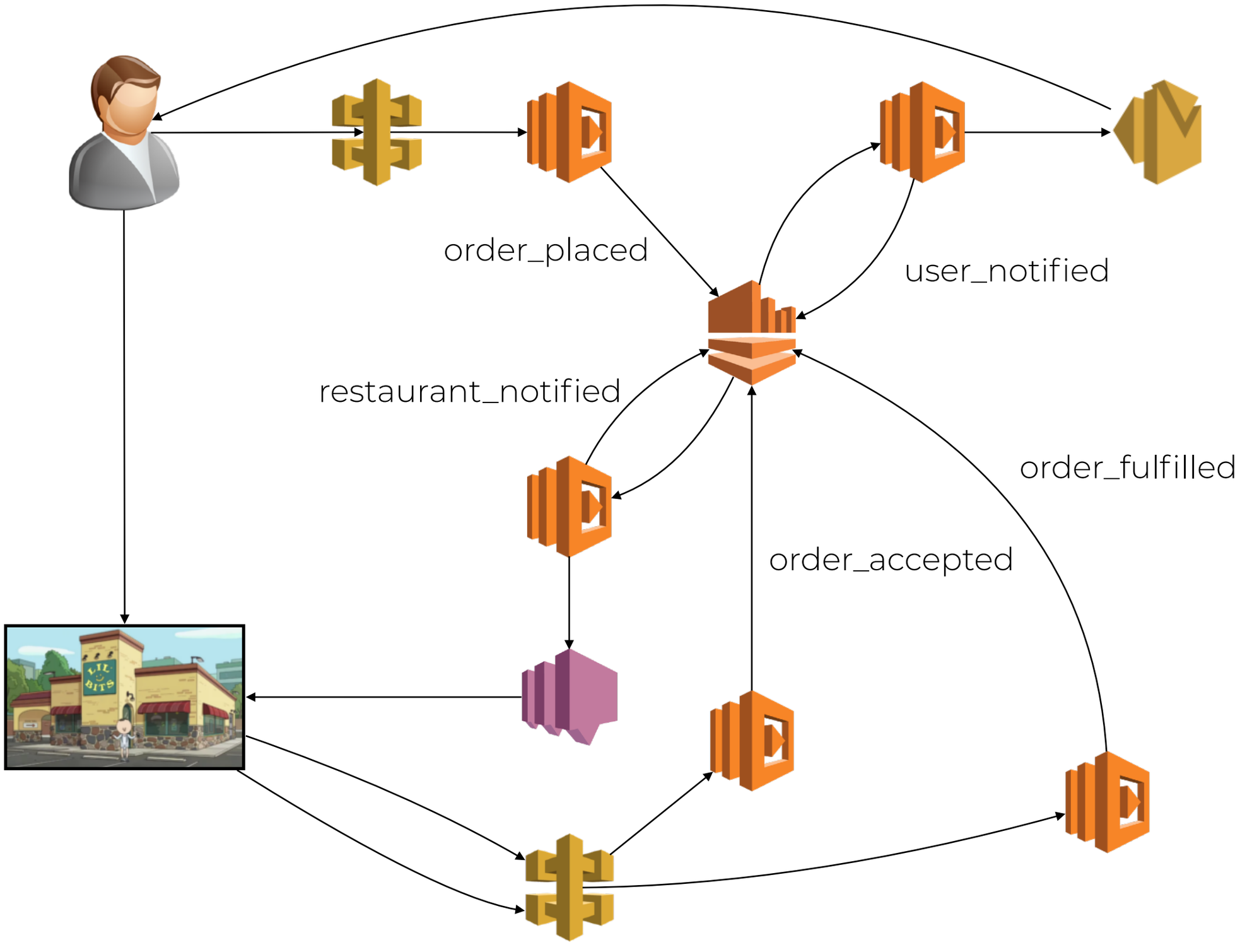

order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted
LIL BITS

failure modes

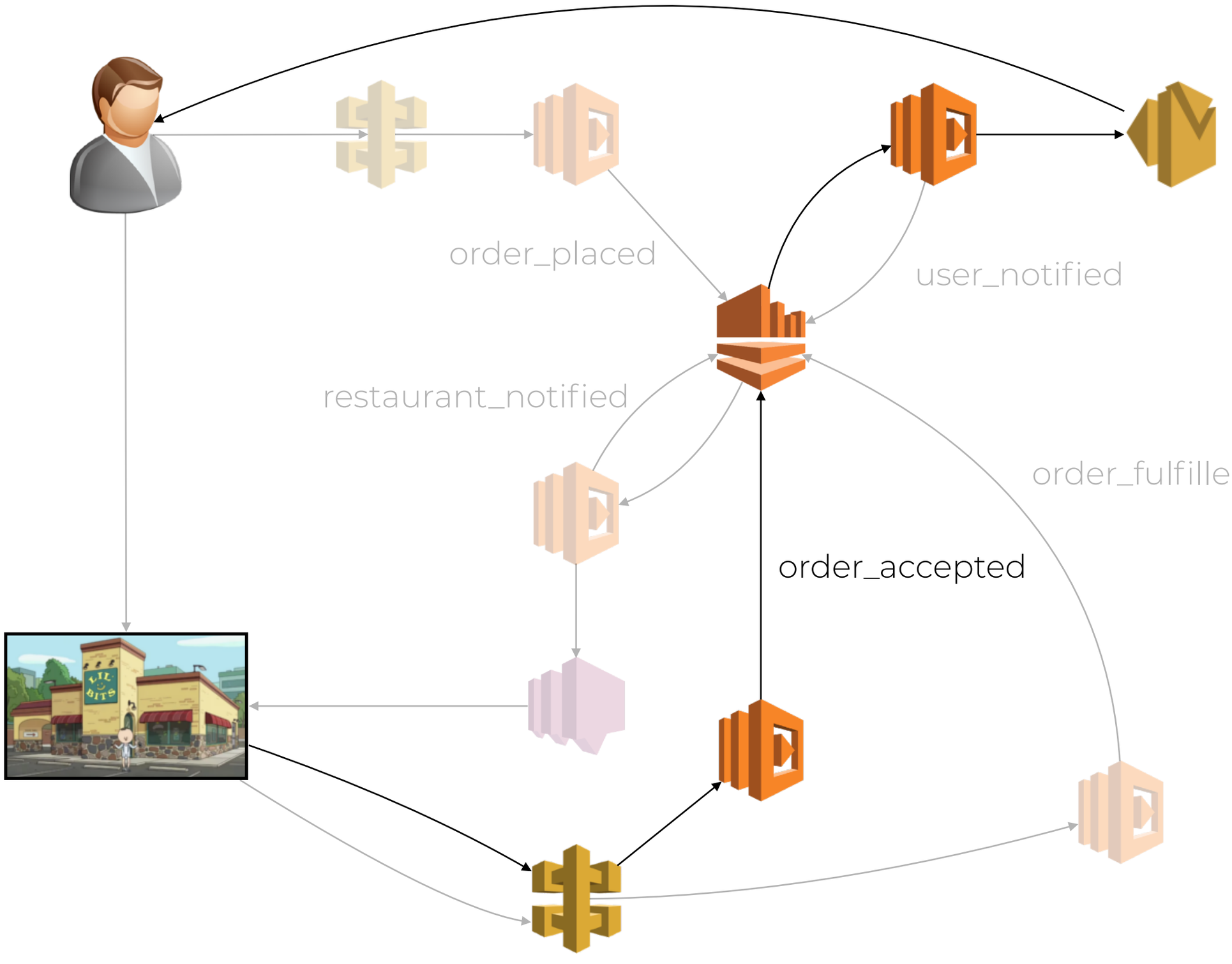


1. customer places order
2. notifies restaurant
3. restaurant accepts order
4. customer confirmation
5. pick up

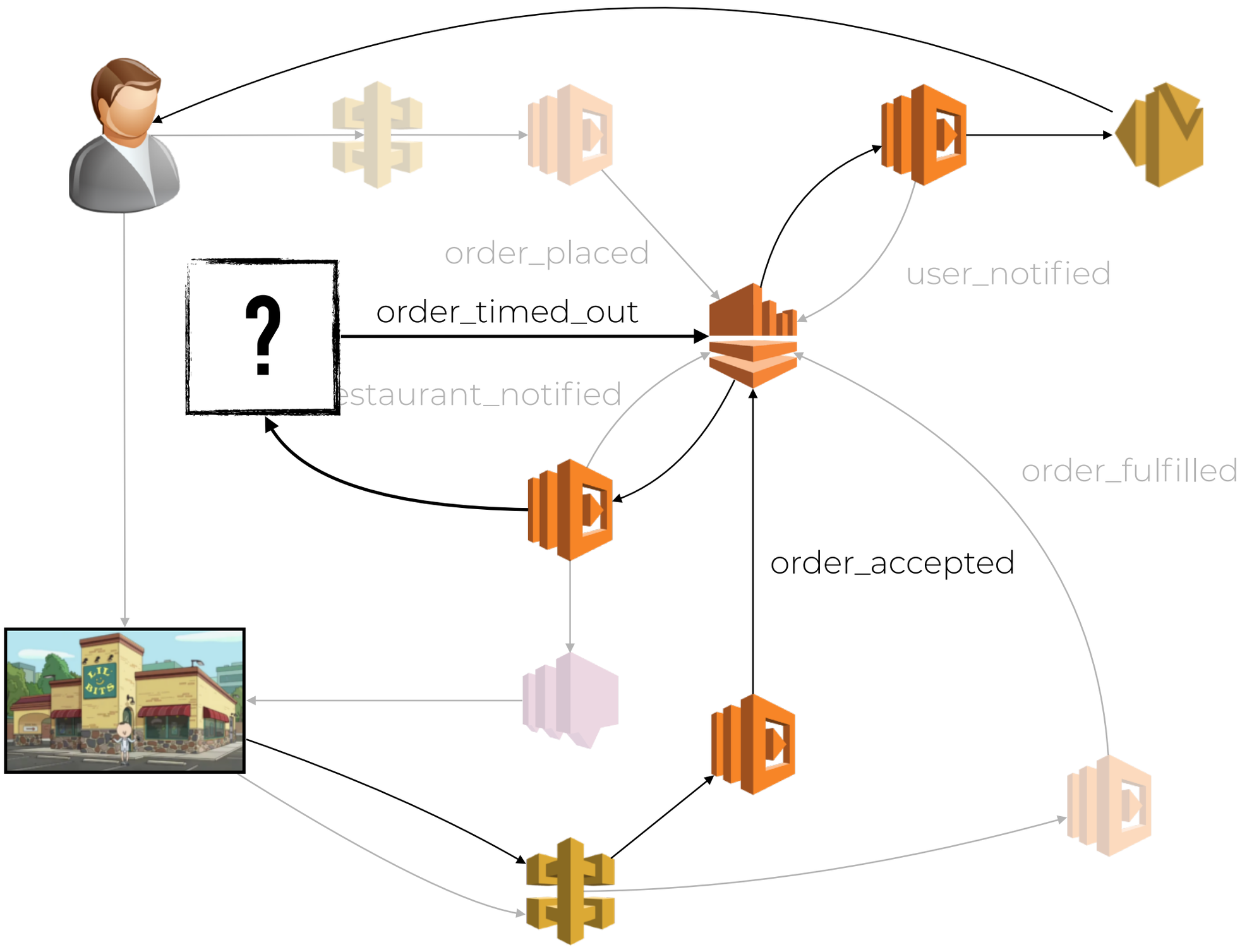
order flow


order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted
LIL BITS

order flow


order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted
LIL BITS

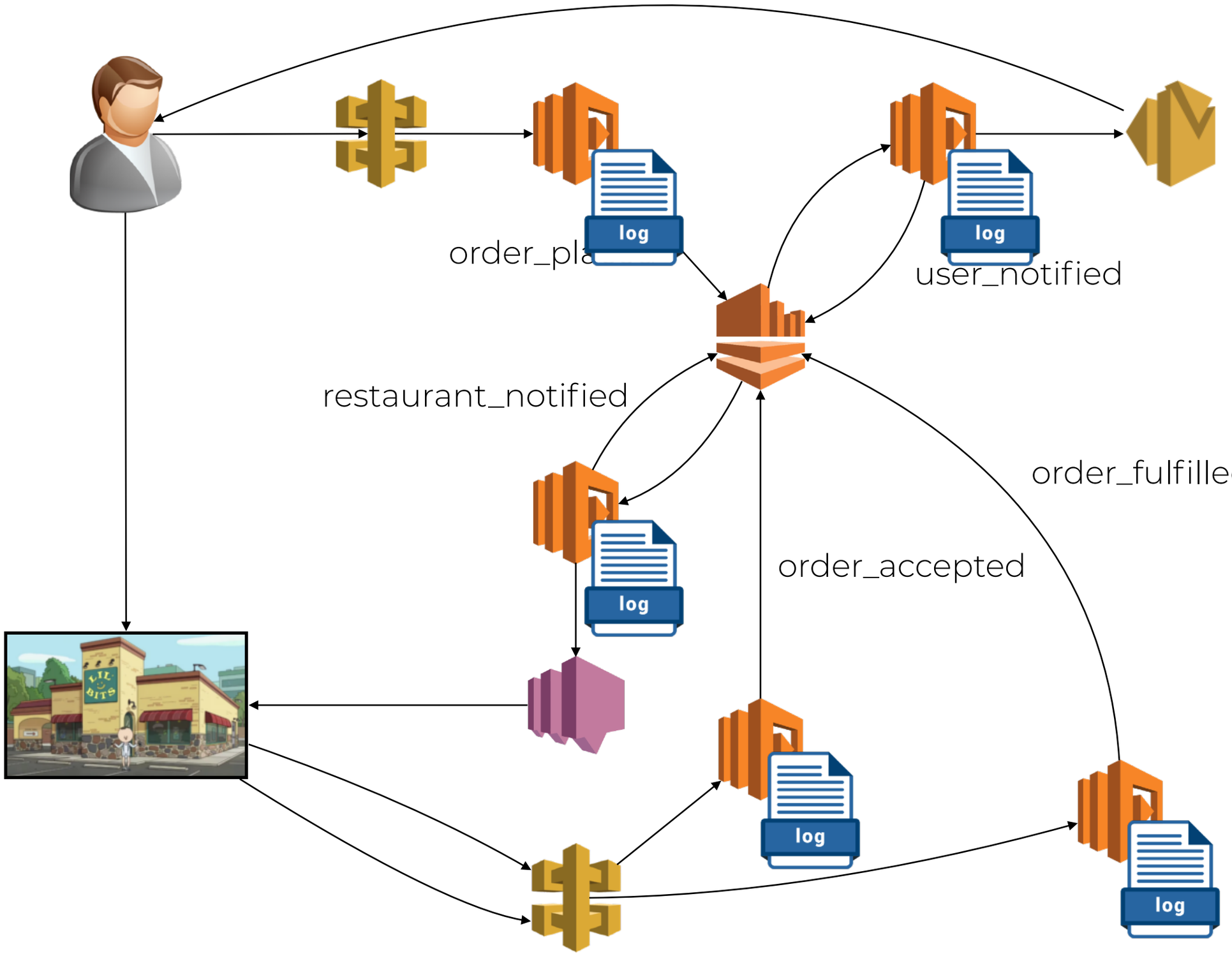
order flow


order_placed
user_notified
?
order_timed_out
estaurant_notified
order_fulfilled
order_accepted
LIL BITS

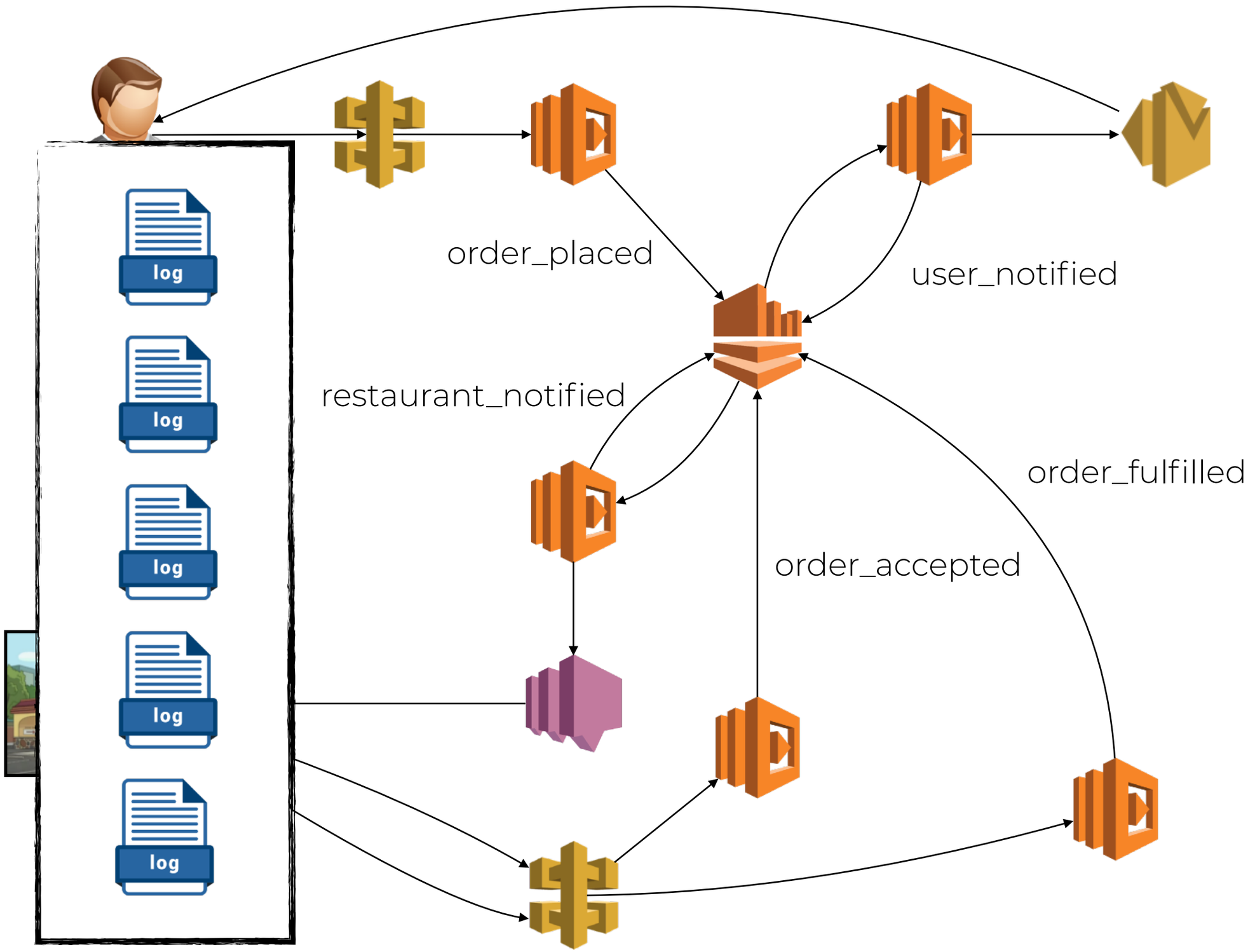
failure modes

- minimise impact on user experience
- strong traceability

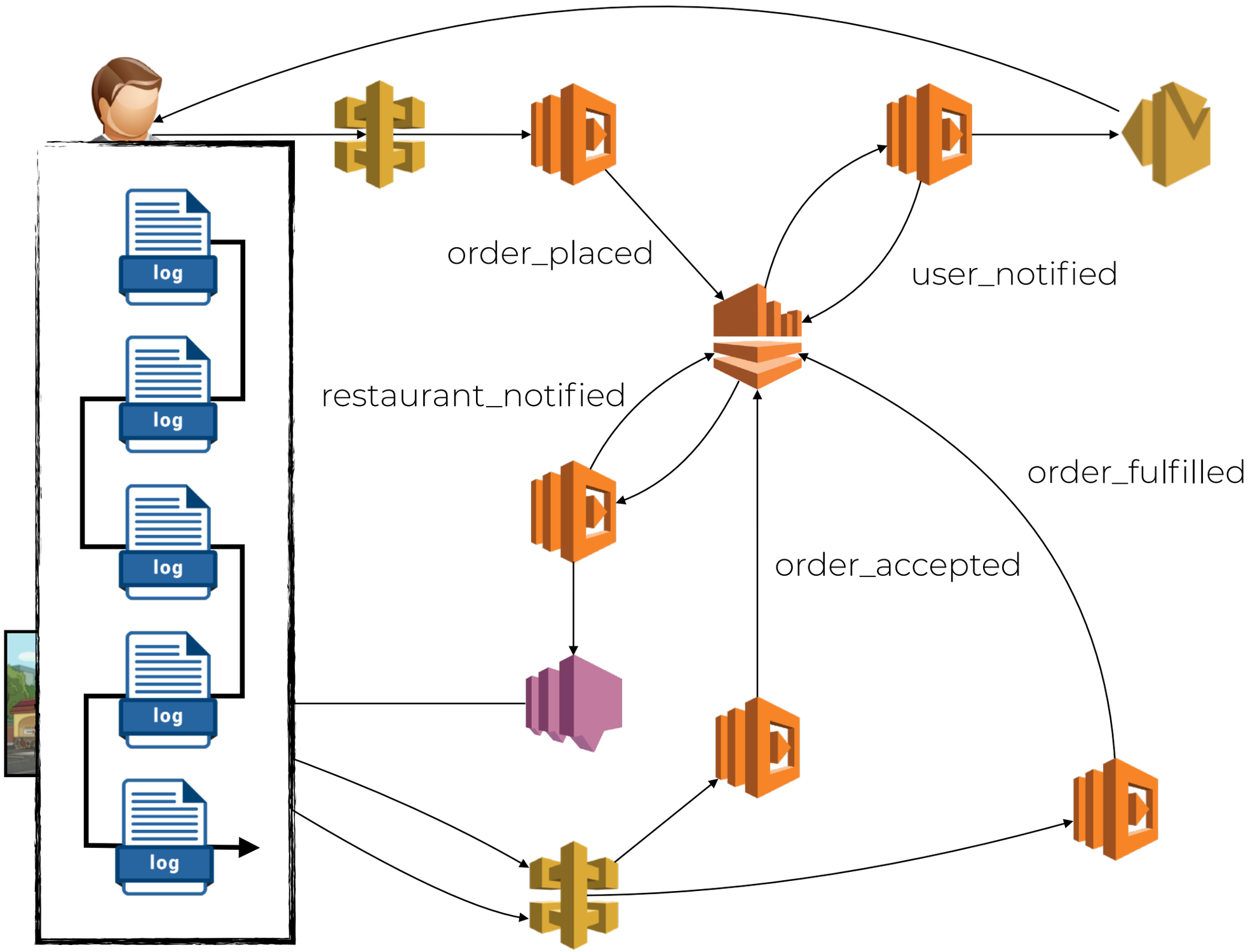
order flow


log
order_pla
log
user_notified
restaurant_notified
log
order_accepted
order_fulfilled
log
log

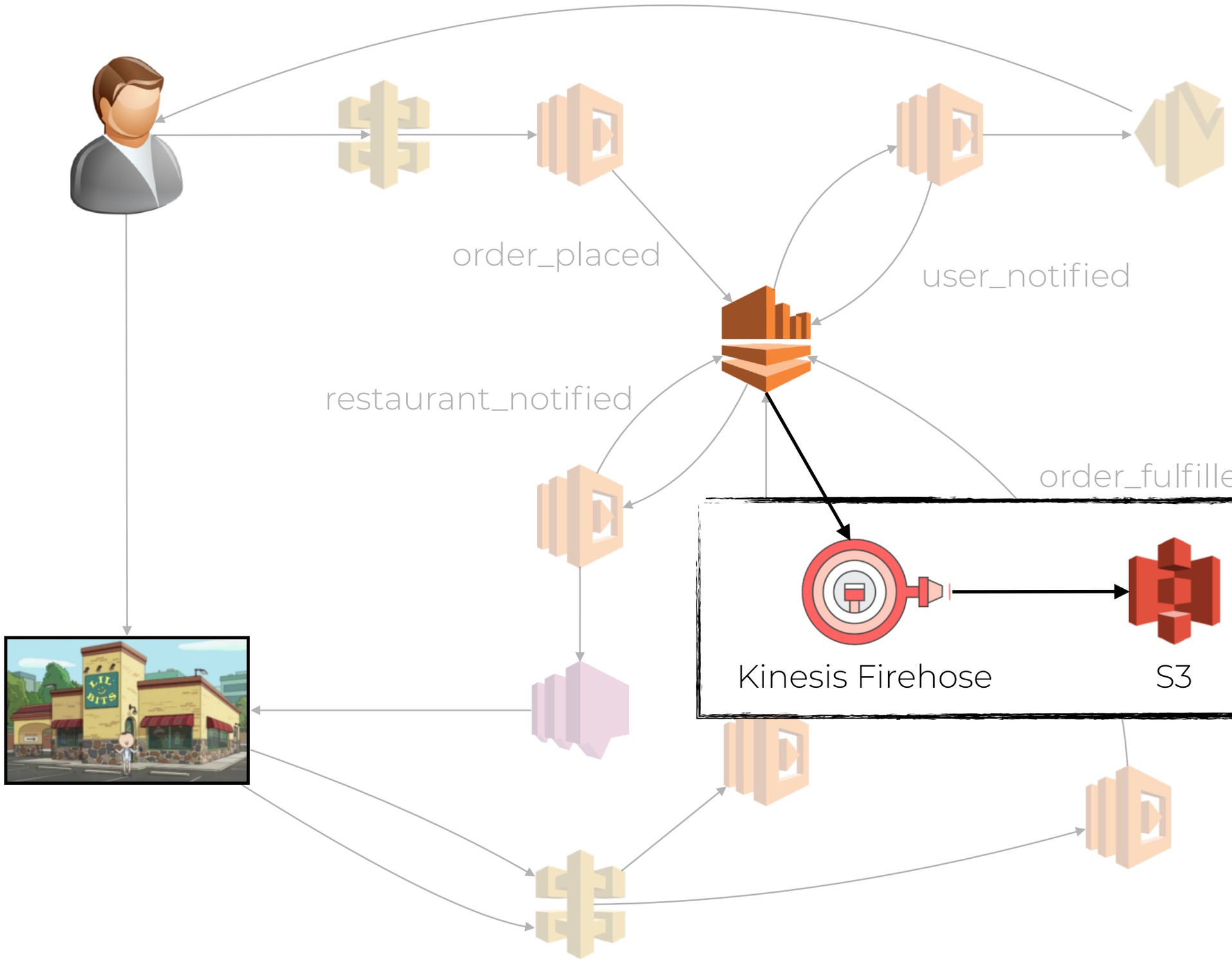
order flow


log
log
log
log
log
order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted

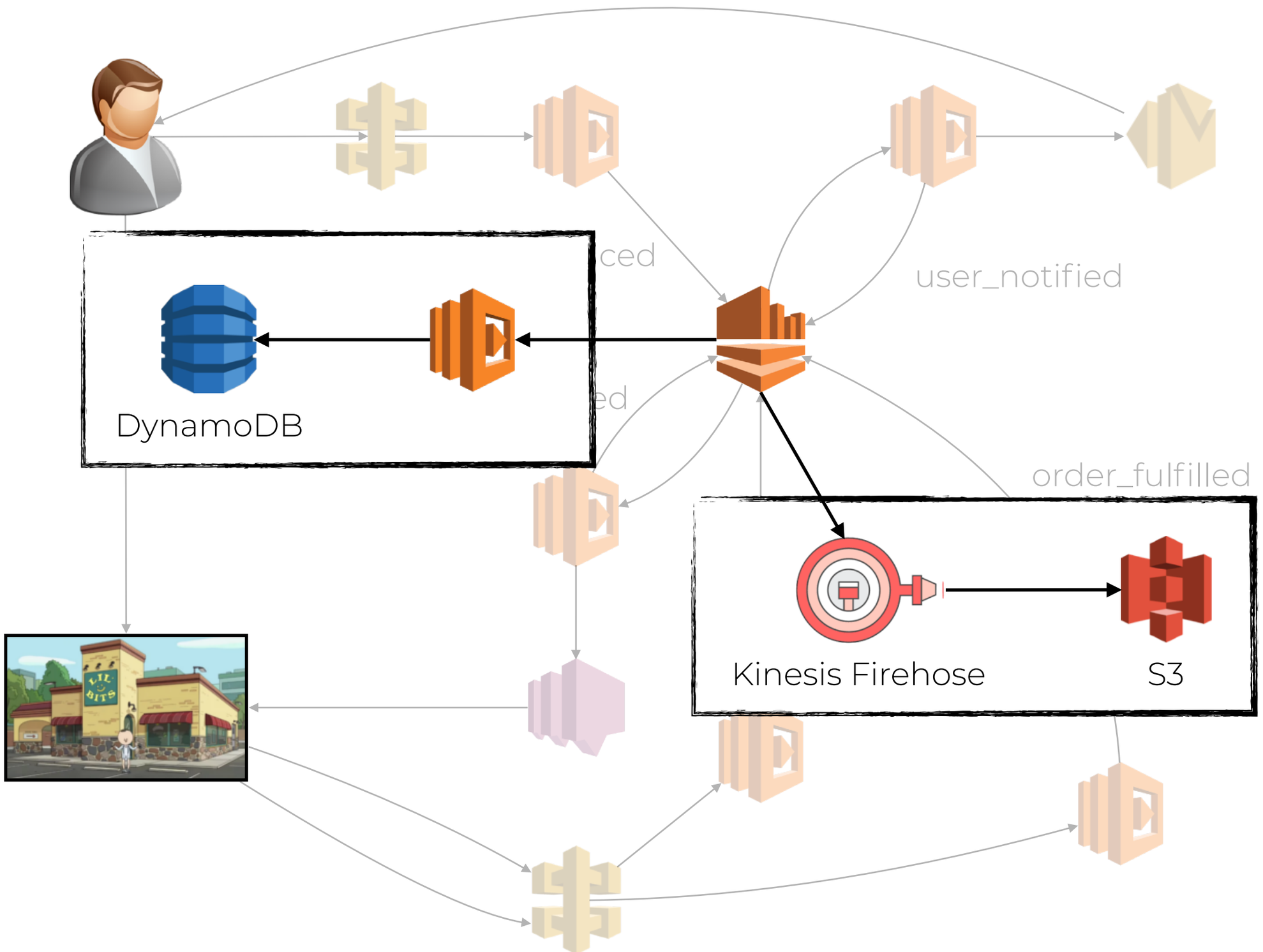
order flow


log
log
log
log
log
order_placed
user_notified
restaurant_notified
order_fulfilled
order_accepted

order flow

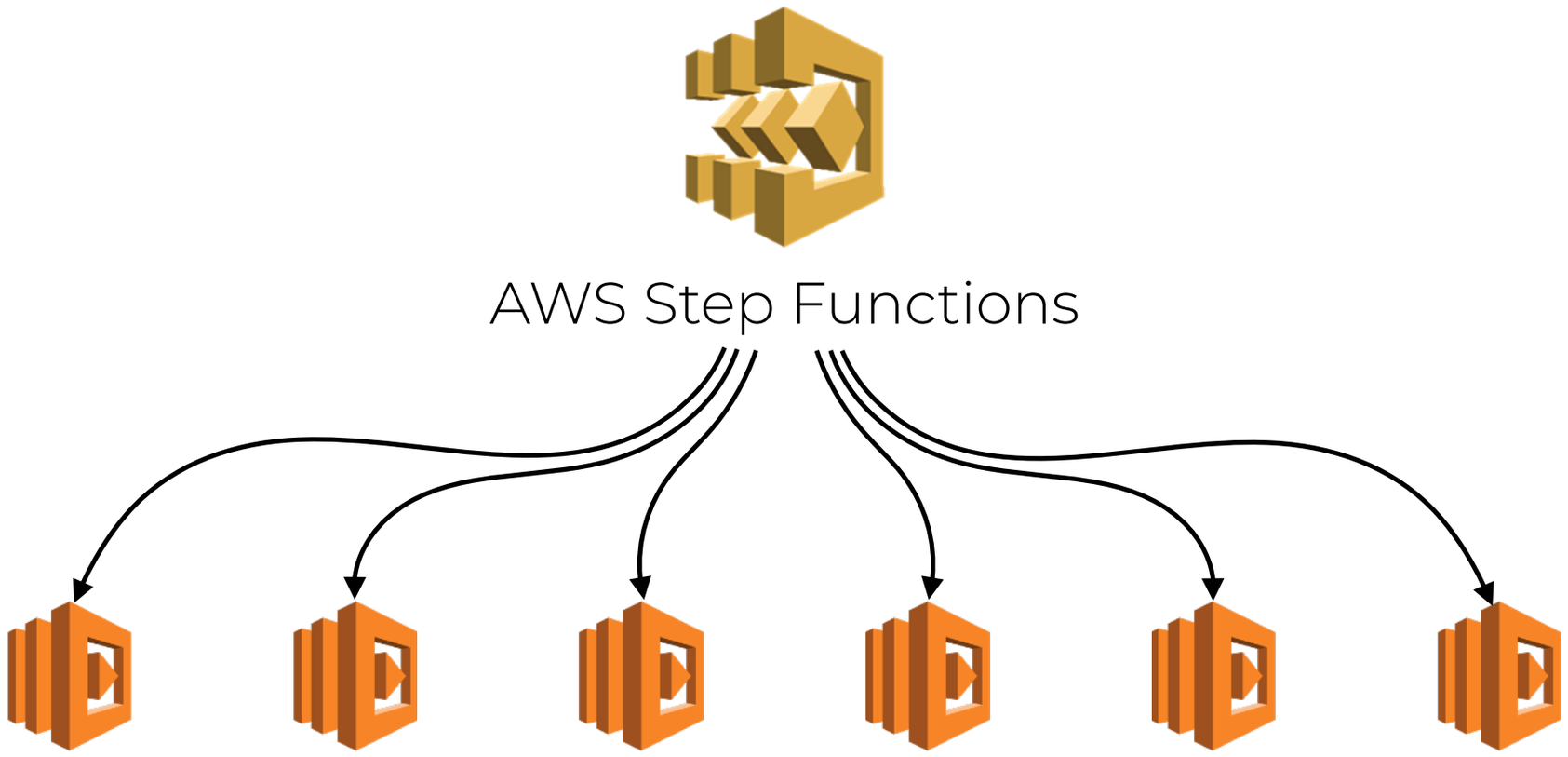

order_placed
user_notified
restaurant_notified
order_fulfilled
Kinesis Firehose
S3
LIL BITS

order flow

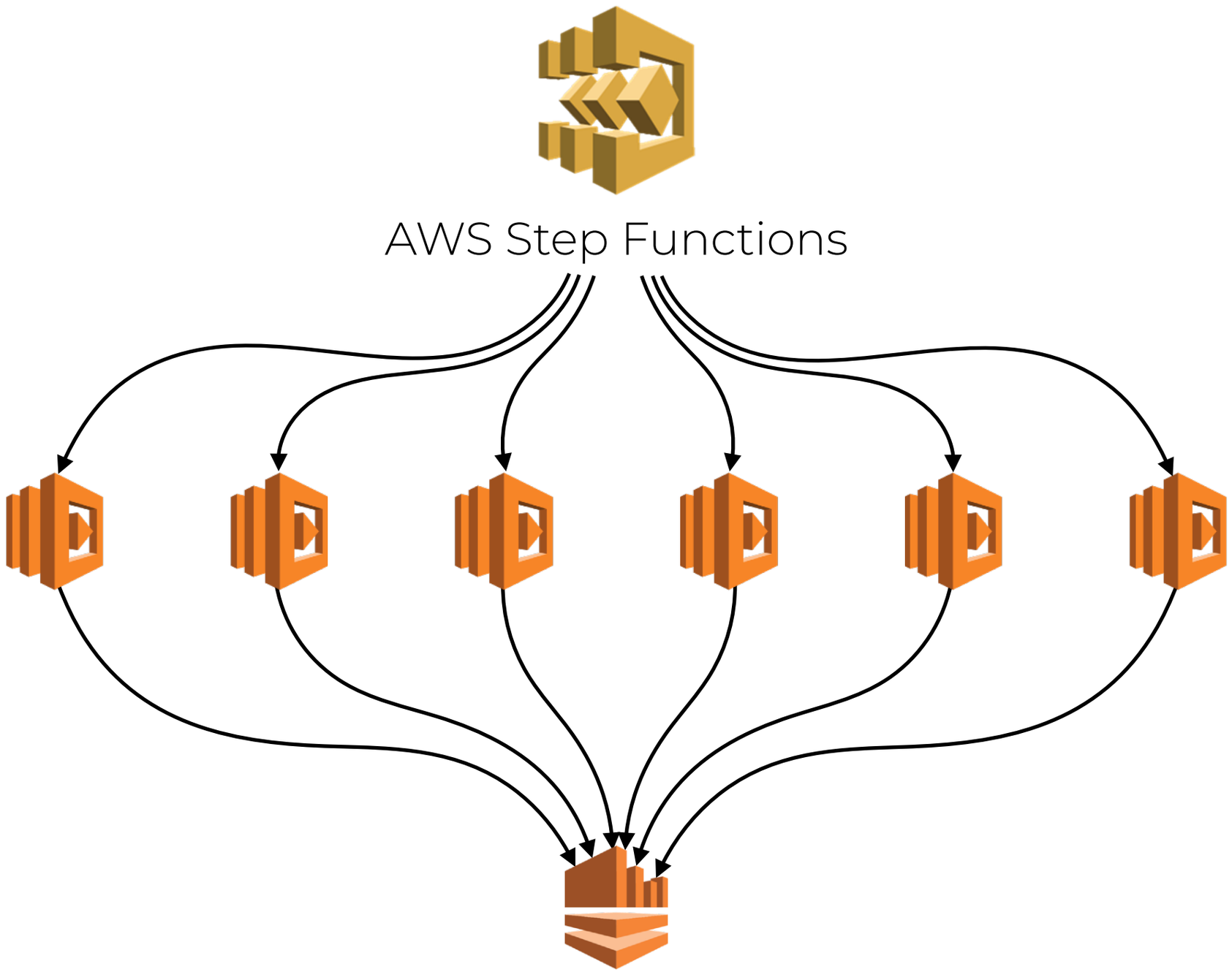

ced
user_notified
ed
DynamoDB
order_fulfilled
Kinesis Firehose
S3

AWS Step Functions

order flow



order flow



order flow