# AWS re:Invent

NOV. 29 – DEC. 3, 2021 | LAS VEGAS, NV

API304

# Building next-gen applications with event-driven architectures

Sam Dengler (he/him)
Principal Solutions Architect
AWS

Robbie Kohler (he/him)
Director, Software Engineering
Taco Bell

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Agenda

Enterprise integration patterns

Event-driven architecture

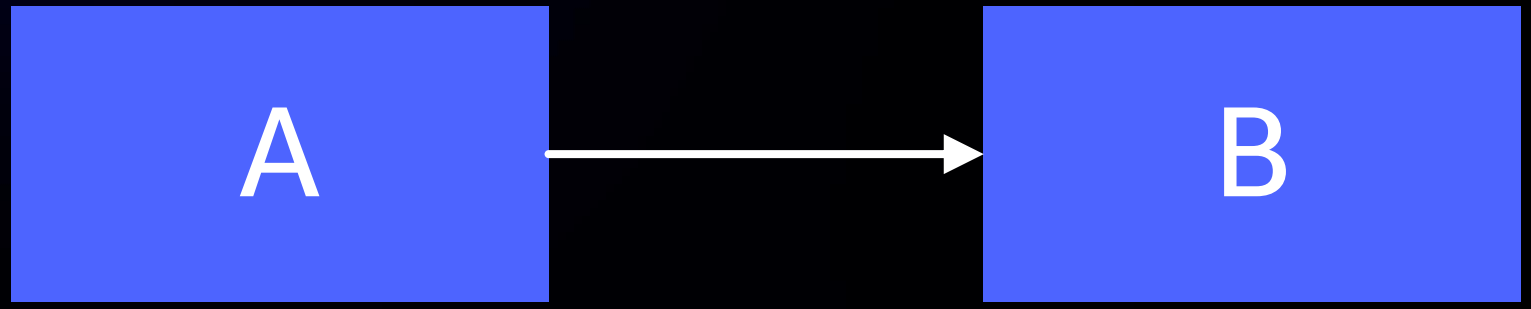
Taco Bell order middleware service

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Enterprise integration patterns

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Coupling: Integration's magic word




Coupling is a measure of independent variability between connected systems

Decoupling has a cost, both at design and run-time

Coupling isn't binary

Coupling isn't one-dimensional

Source: EnterpriseIntegrationPatterns.com

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# The many facets of coupling

- Technology dependency: Java vs. C++
- Location dependency: IP addresses, DNS
- Data format dependency: Binary, XML, JSON, ProtoBuf, Avro
- Data type dependency: int16, int32, string, UTF-8, null, empty
- Semantic dependency: Name, middle name, ZIP
- Temporal dependency: sync, async
- Interaction style dependency: messaging, RPC, query-style (GraphQL)
- Conversation dependency: pagination, caching, retries

Source: EnterpriseIntegrationPatterns.com

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

> **“The appropriate level of coupling depends on the level of control you have over the endpoints.”**

Gregor Hohpe

*Enterprise Integration Patterns*

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Synchronous request-response model

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Synchronous request-response model

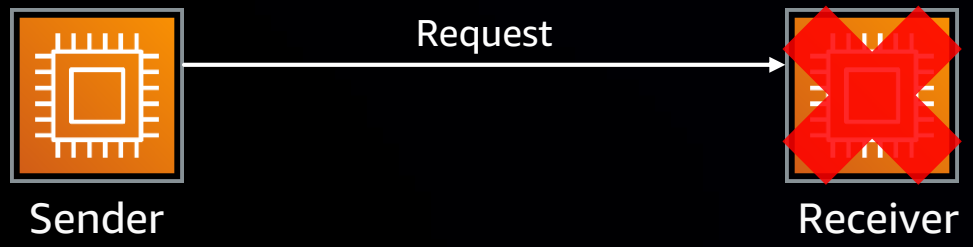



## Advantages

- Low latency
- Simple
- Fail fast

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Synchronous request-response model




## Advantages

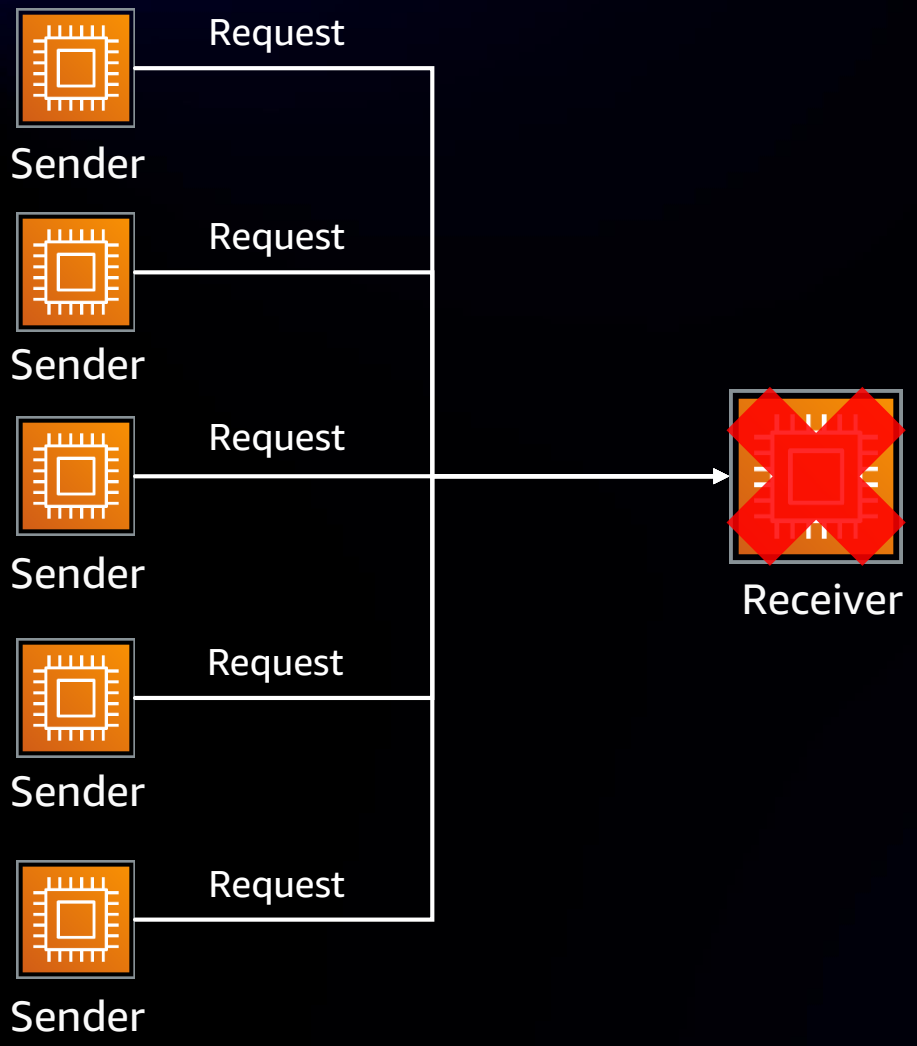
- Low latency
- Simple
- Fail fast

## Disadvantages

- Receiver failure

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Synchronous request-response model




## Advantages

- Low latency
- Simple
- Fail fast

## Disadvantages

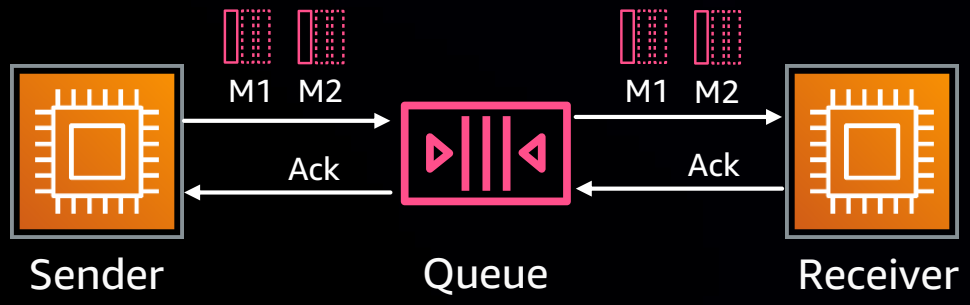
- Receiver failure
- Receiver throttled

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point (queue)

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (queue)



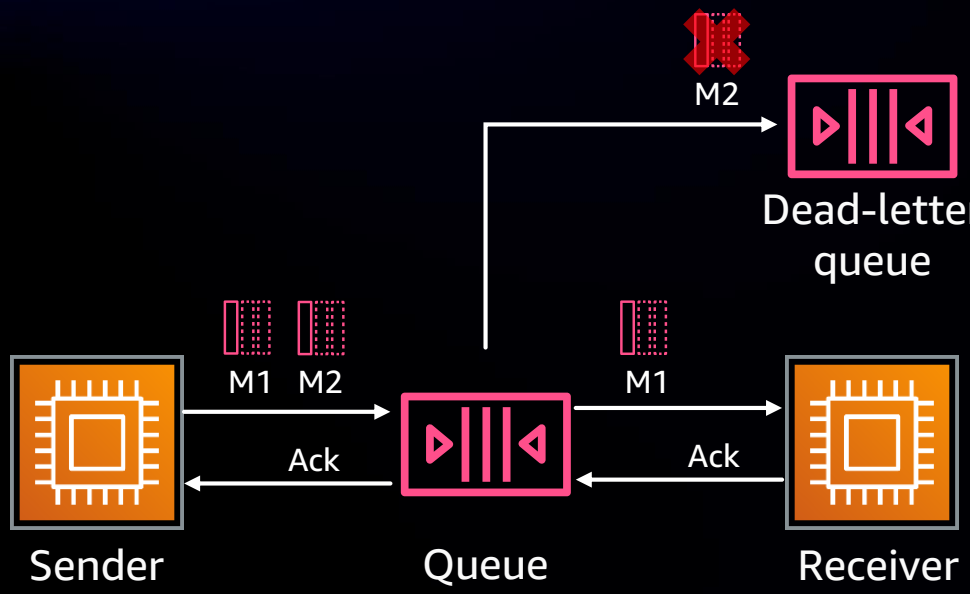

## Advantages

- Decreases temporal coupling
- Resilient to receiver failure
- Receiver controls consumption rate

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (queue)

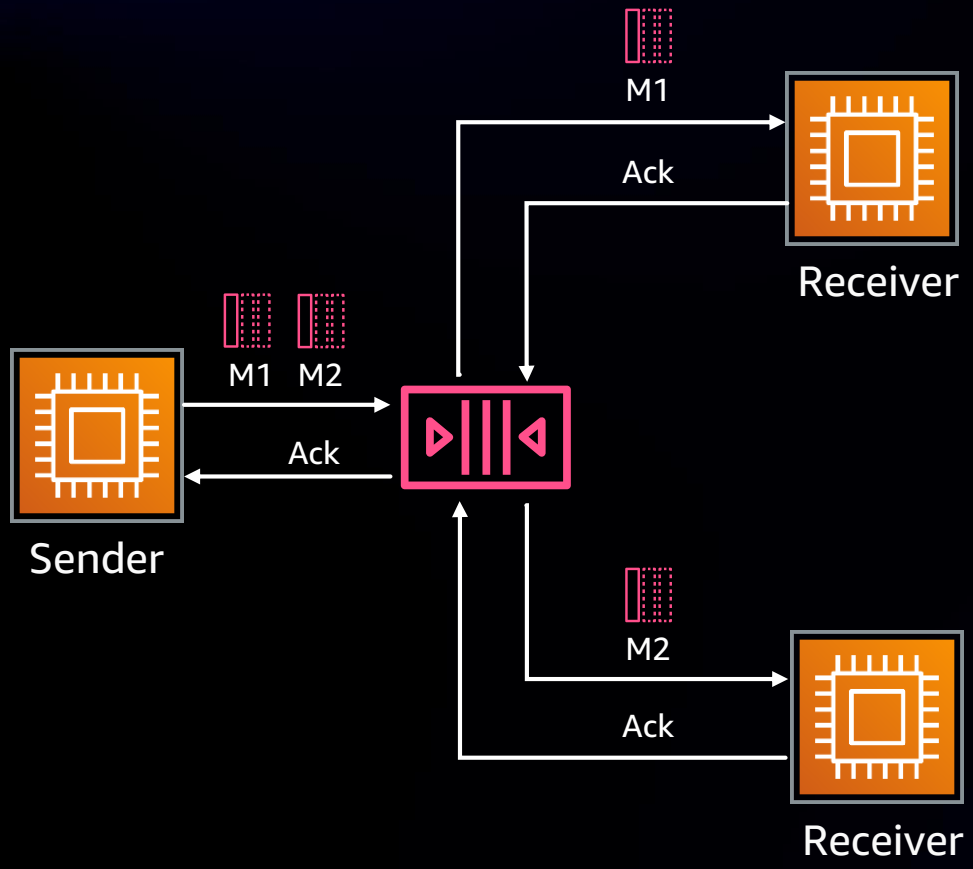



## Advantages

- …
- Dead-letter queue (DLQ) for errors

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (queue)

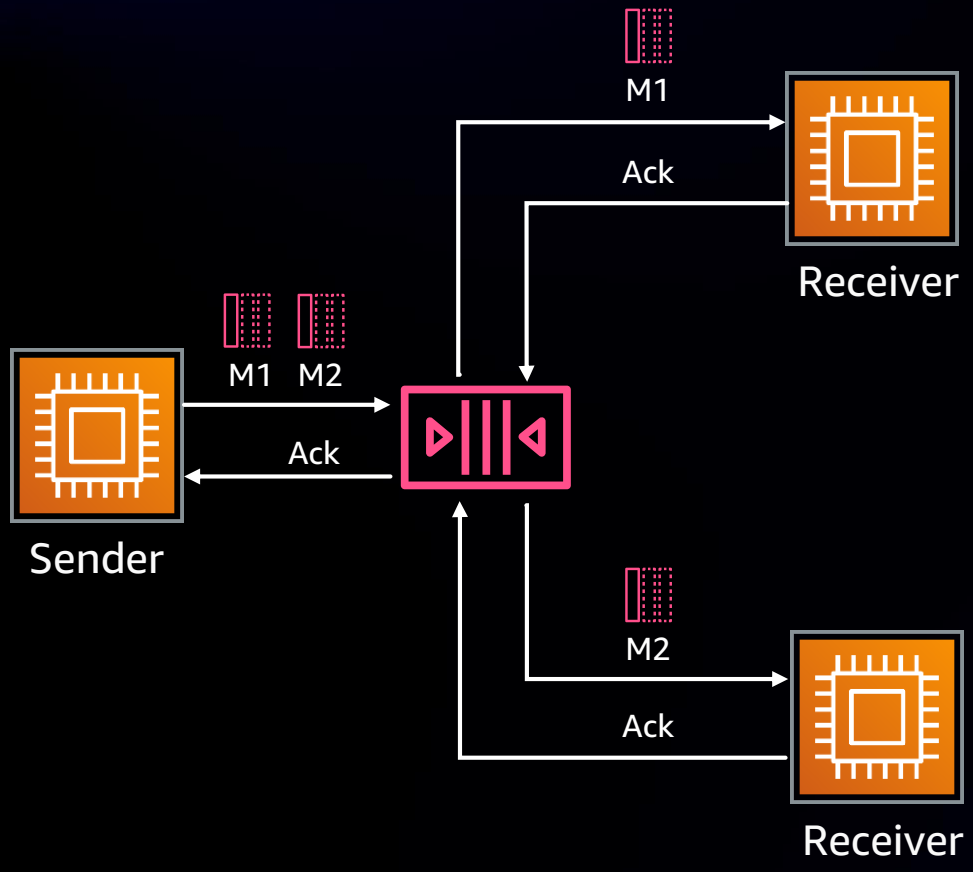



## Advantages

- …
- Only one receiver can consume each message

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (queue)




## Advantages

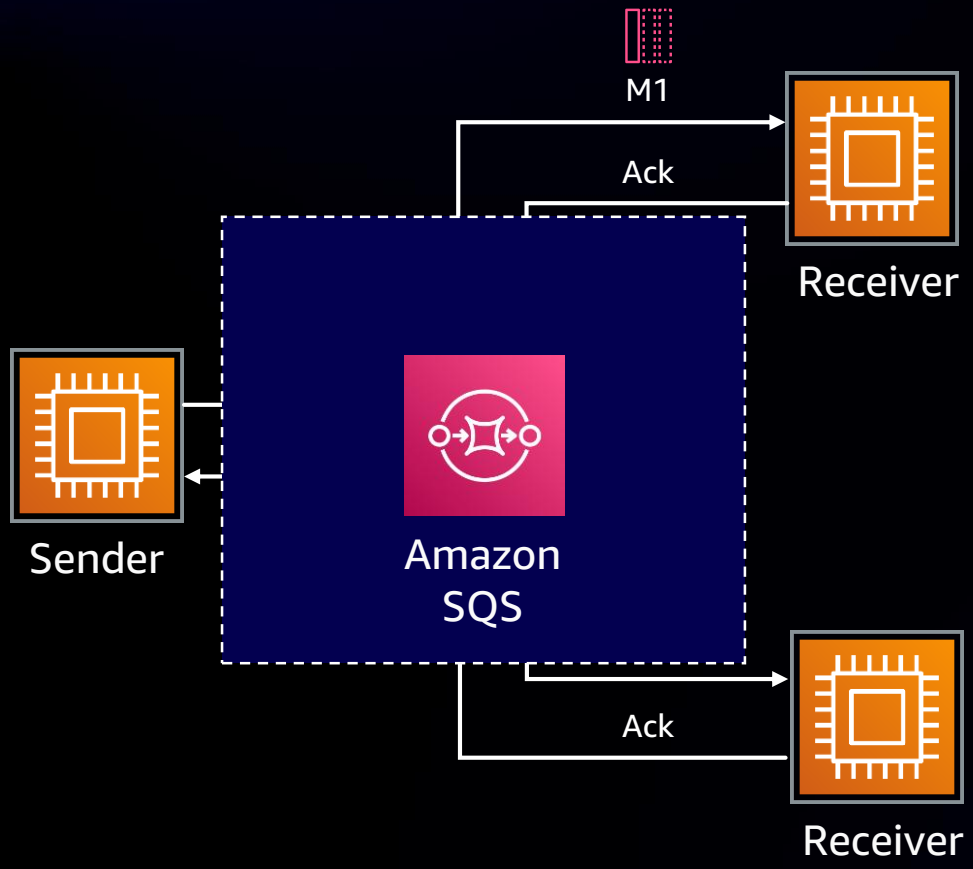
- …
- Only one receiver can consume each message

## Disadvantages

- Response correlation
- Backlog recovery time
- Fairness in multi-tenant systems

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (queue)

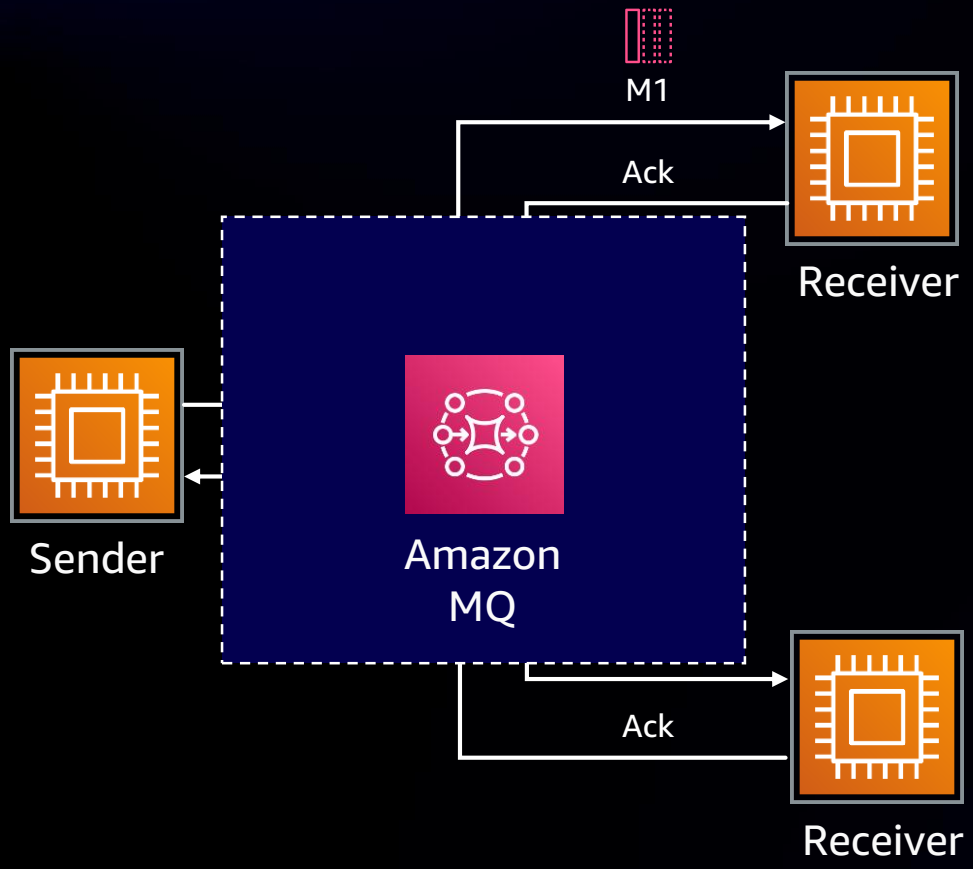



## Amazon Simple Queue Service (Amazon SQS)

- Fully managed message queue
- Scales almost infinitely
- Simple, easy-to-use API
- Dead-letter queue (DLQ) support
- Standard and FIFO options

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (queue)




## Amazon MQ

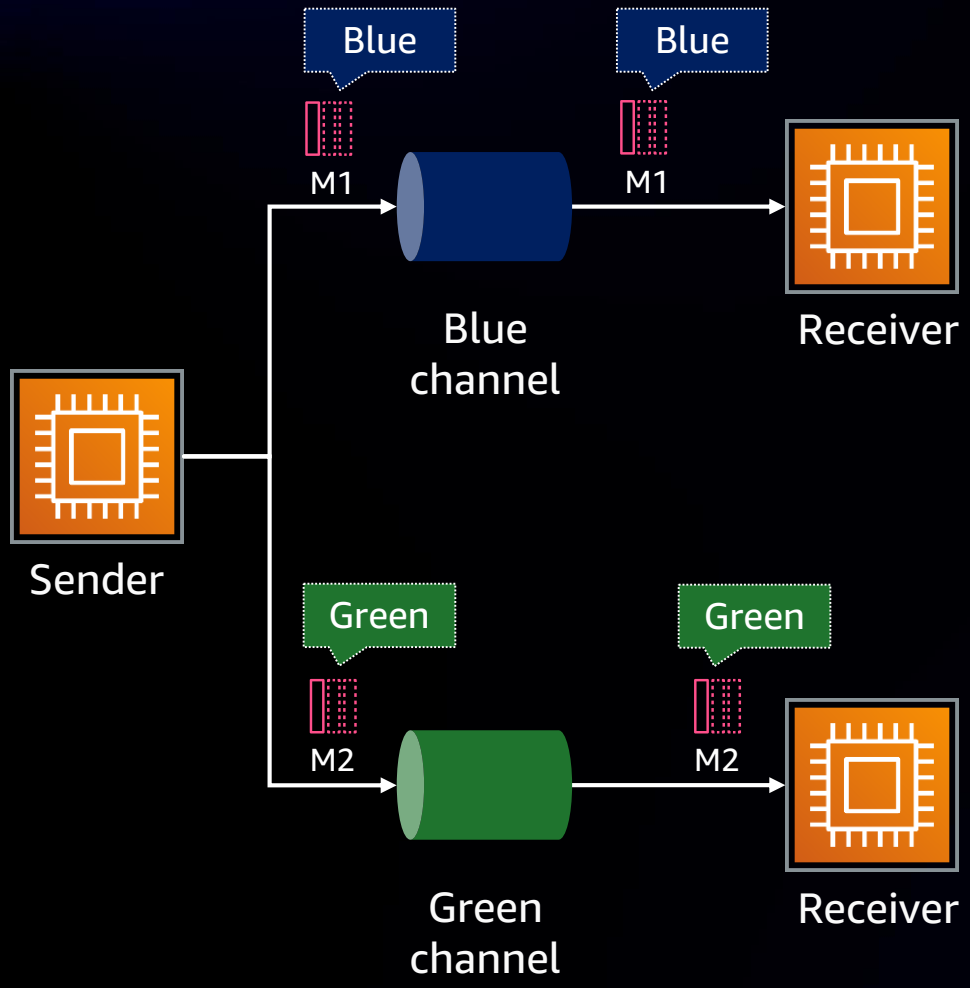
- Active MQ and RabbitMQ broker engine options
- Manages the provisioning, setup, and maintenance of message brokers
- Connects to current applications with industry-standard APIs and protocols

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (router)

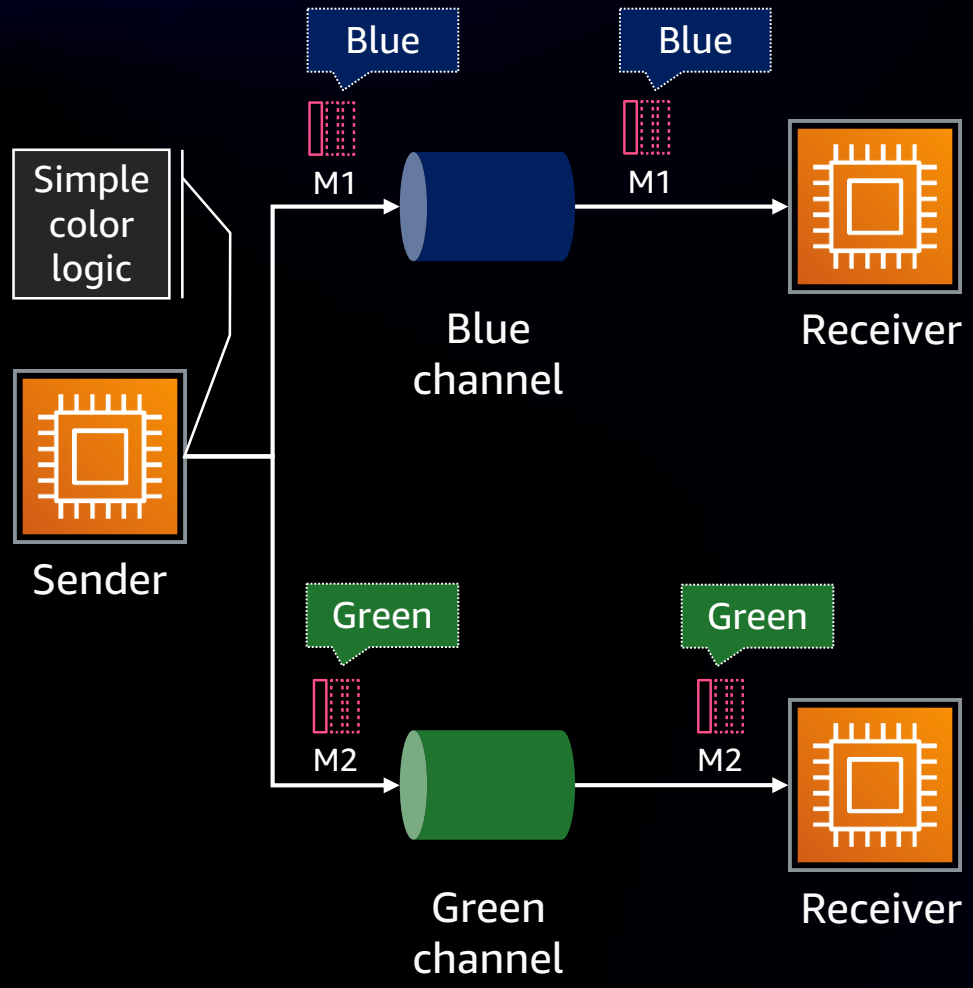
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (router)




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (router)

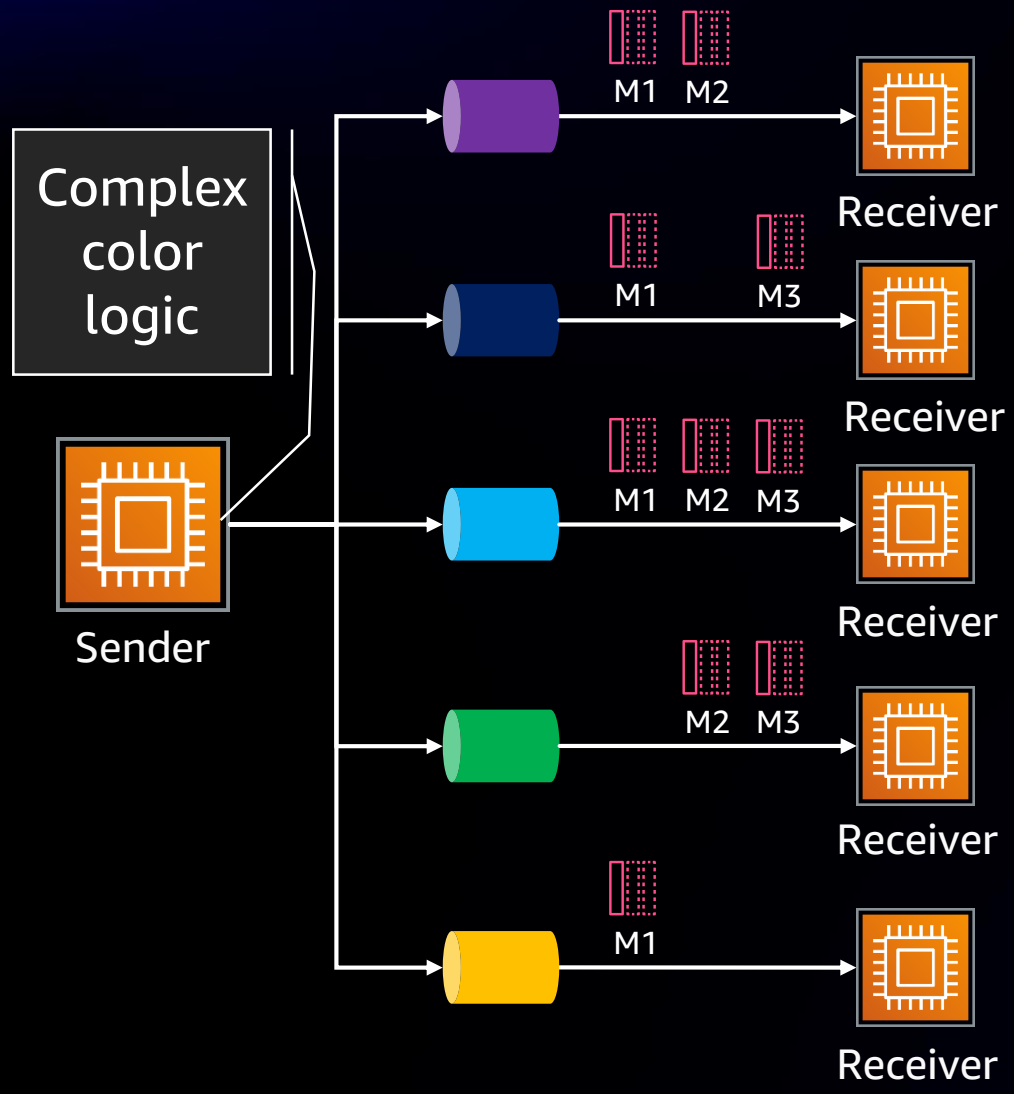



## Disadvantages

- Increases location coupling
- Sender maintains routing logic

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous point-to-point model (router)




## Disadvantages

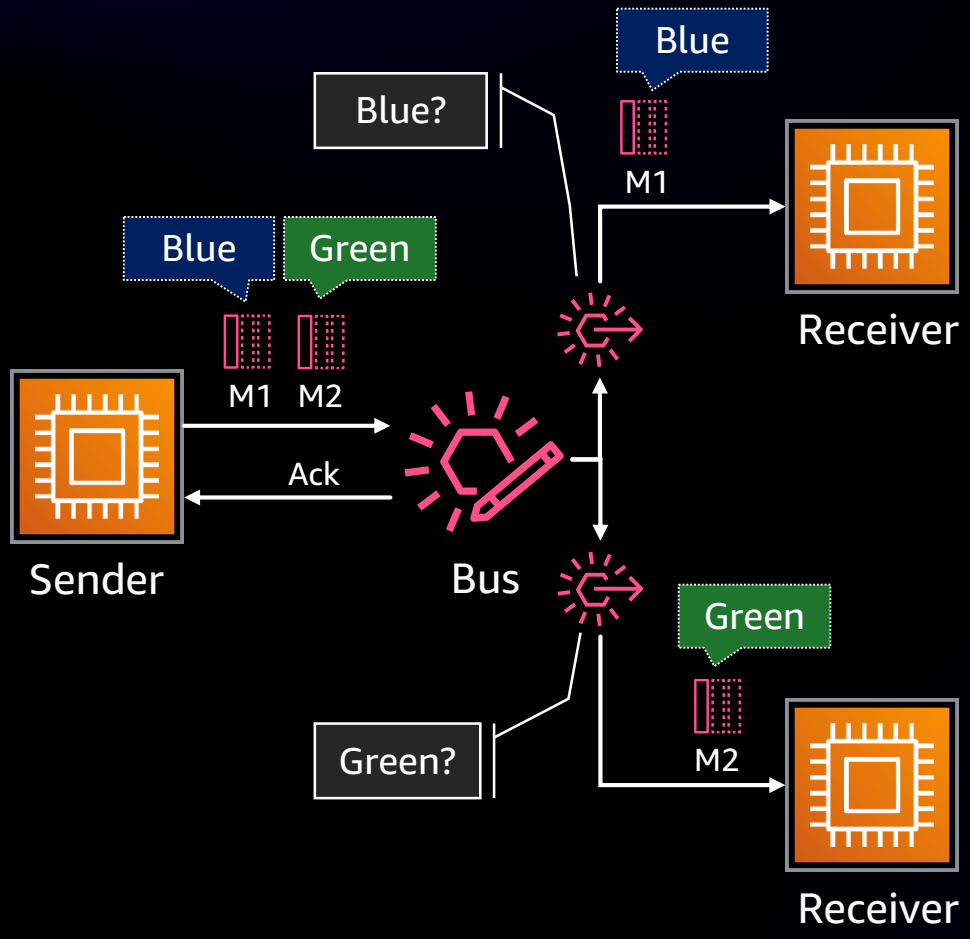
- Increases location coupling
- Sender maintains routing logic
- Sender complexity increases with time

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous message-router model (bus)

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous message-router (bus)

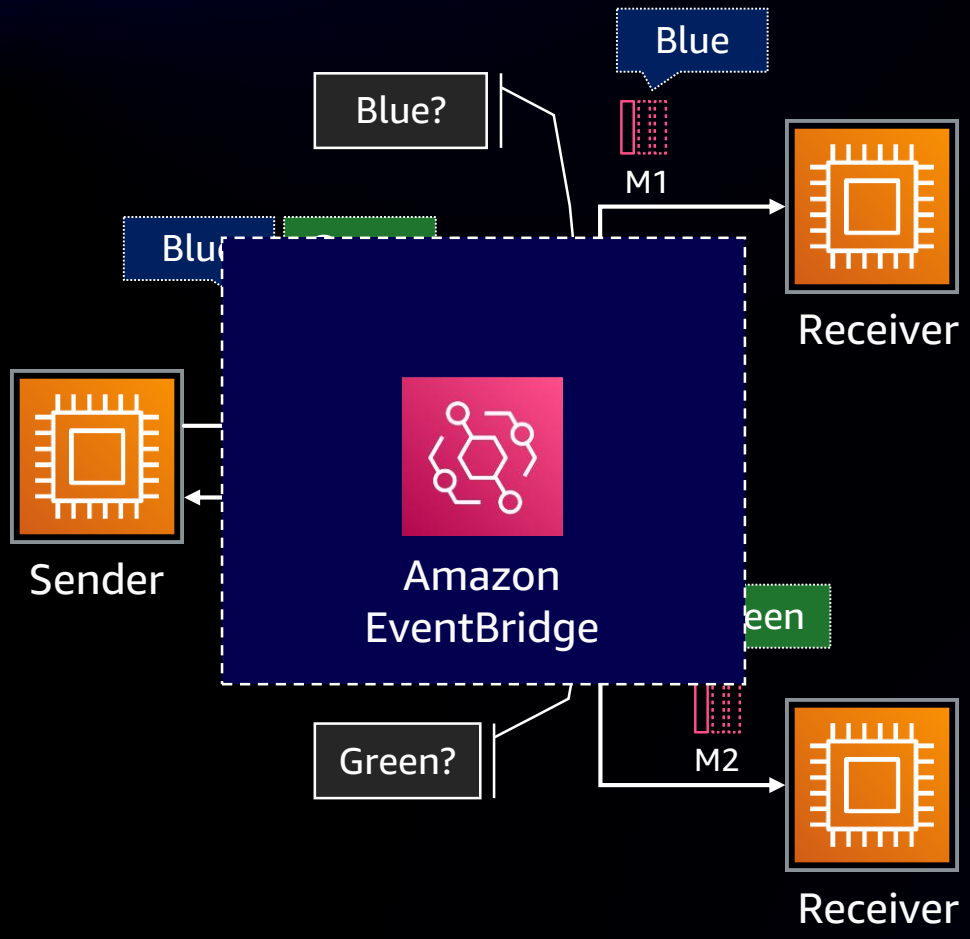



## Advantages

- Reduces location coupling
- Efficient for senders and receivers

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Asynchronous message-router (bus)



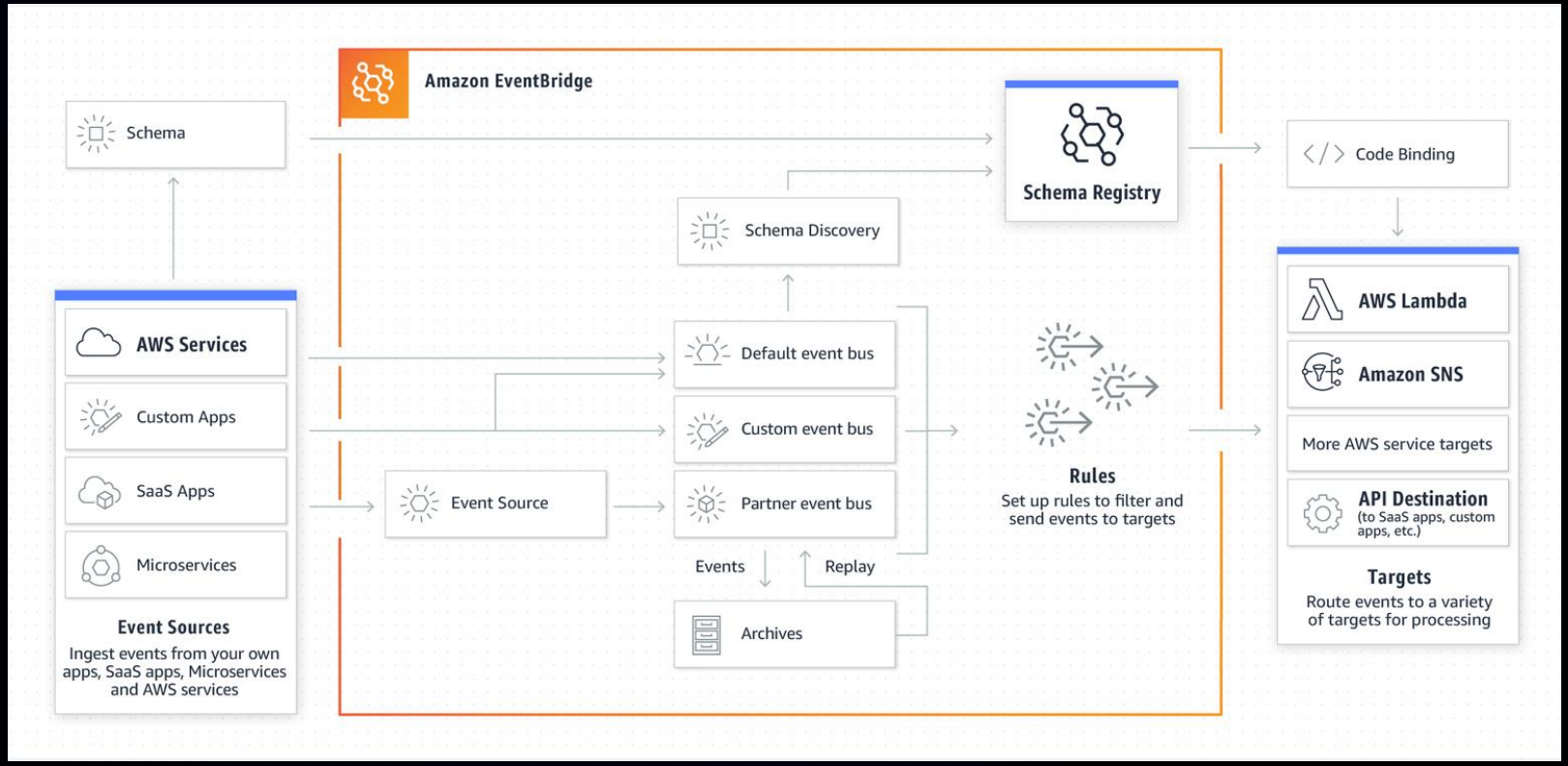

## Advantages

- Reduces location coupling
- Efficient for senders and receivers

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Amazon EventBridge

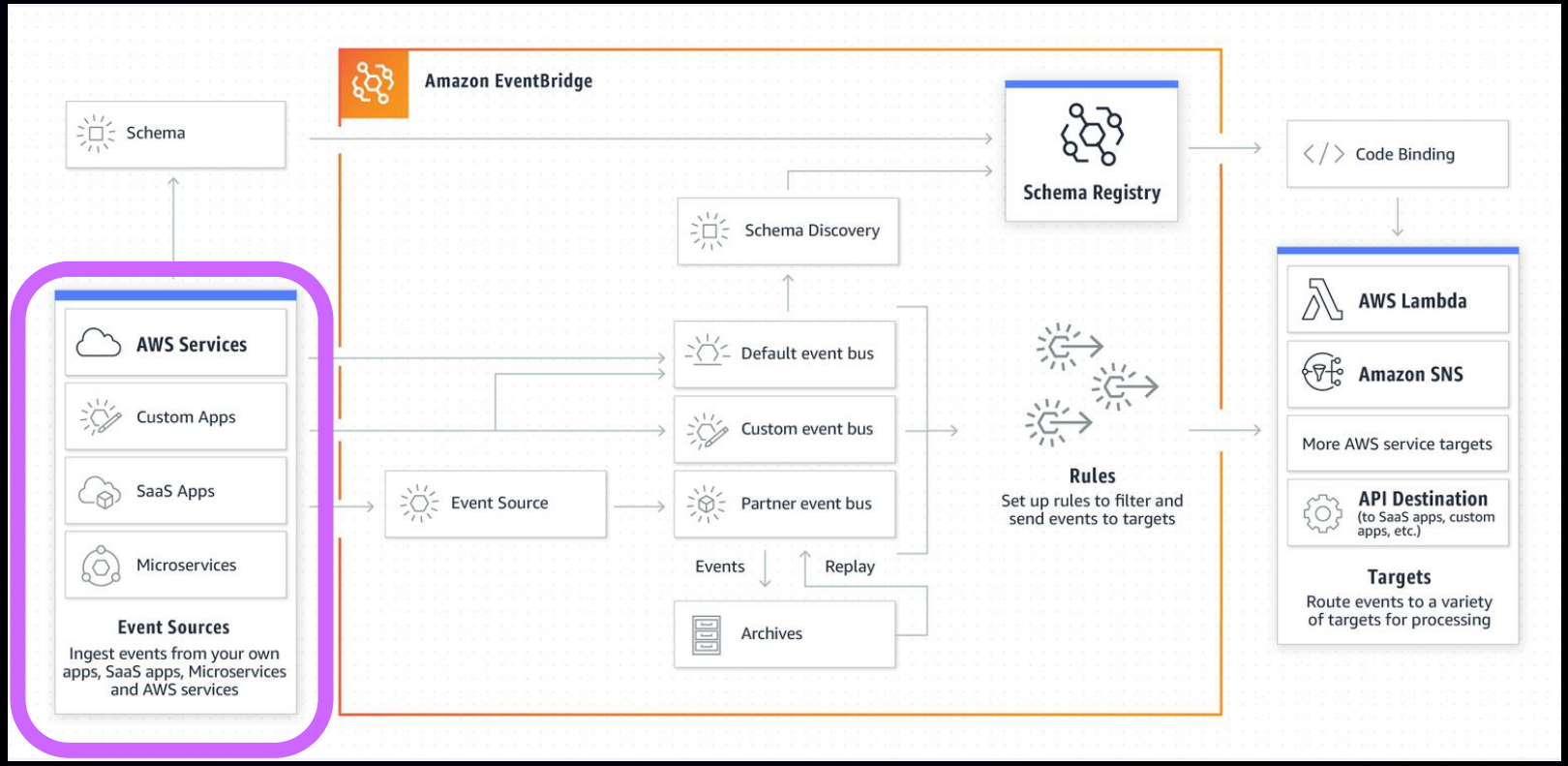
Amazon EventBridge is a simple, flexible, fully managed, pay-as-you-go, event bus service that makes it easy to ingest and process data from AWS services, your own applications, and SaaS applications




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Amazon EventBridge

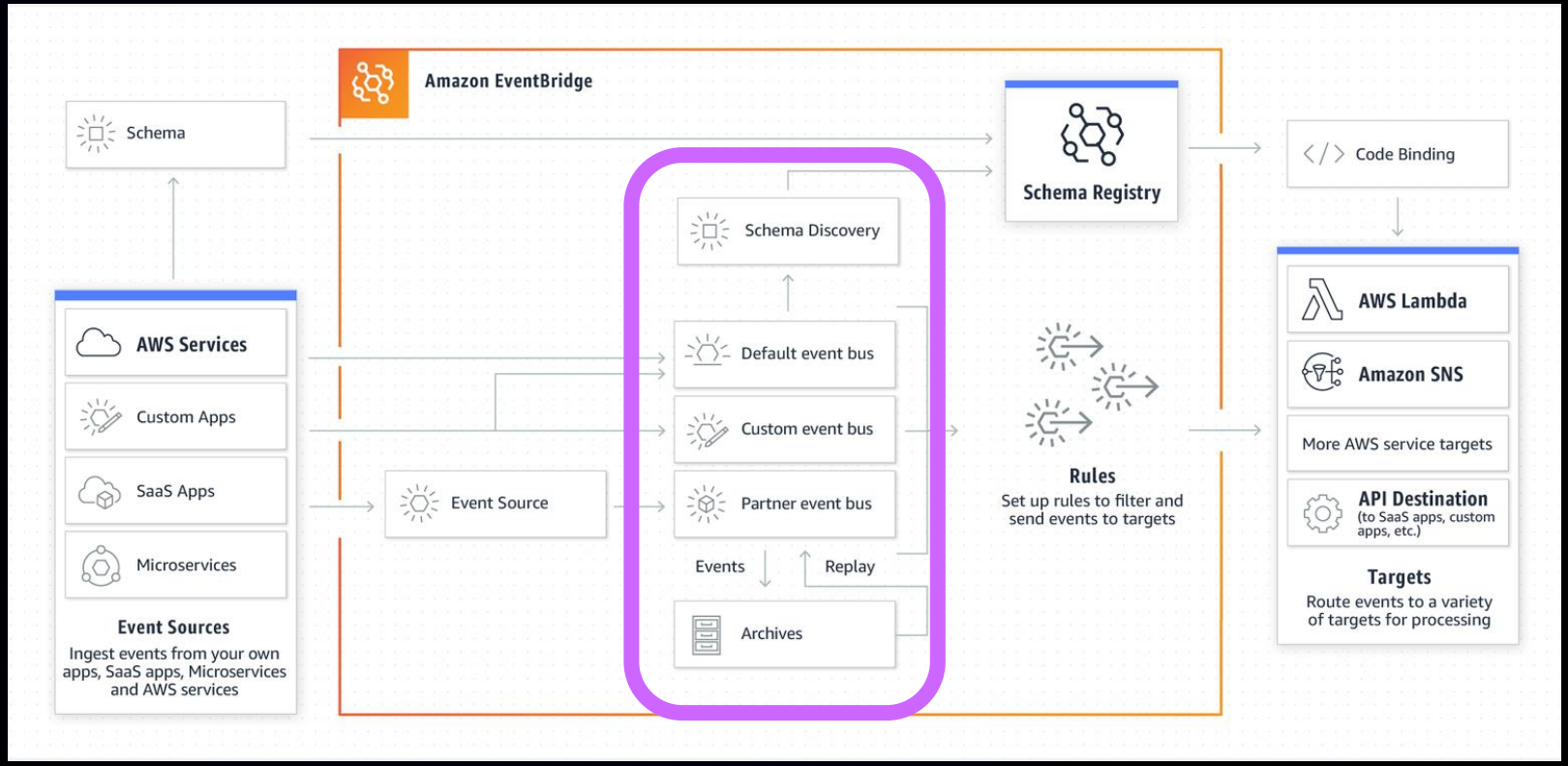
Amazon EventBridge is a simple, flexible, fully managed, pay-as-you-go, event bus service that makes it easy to ingest and process data from AWS services, your own applications, and SaaS applications




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Amazon EventBridge

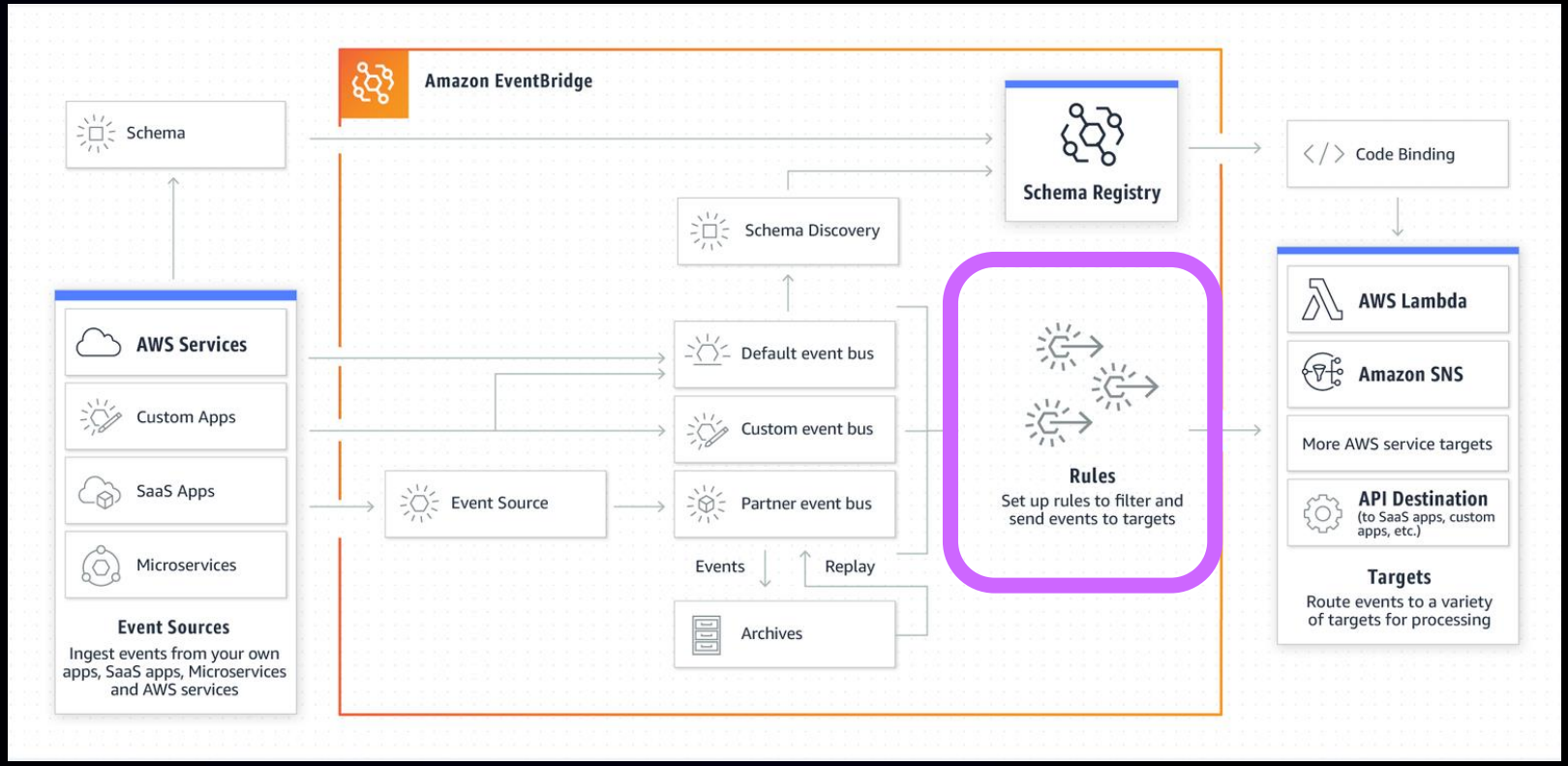
Amazon EventBridge is a simple, flexible, fully managed, pay-as-you-go, event bus service that makes it easy to ingest and process data from AWS services, your own applications, and SaaS applications




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Amazon EventBridge

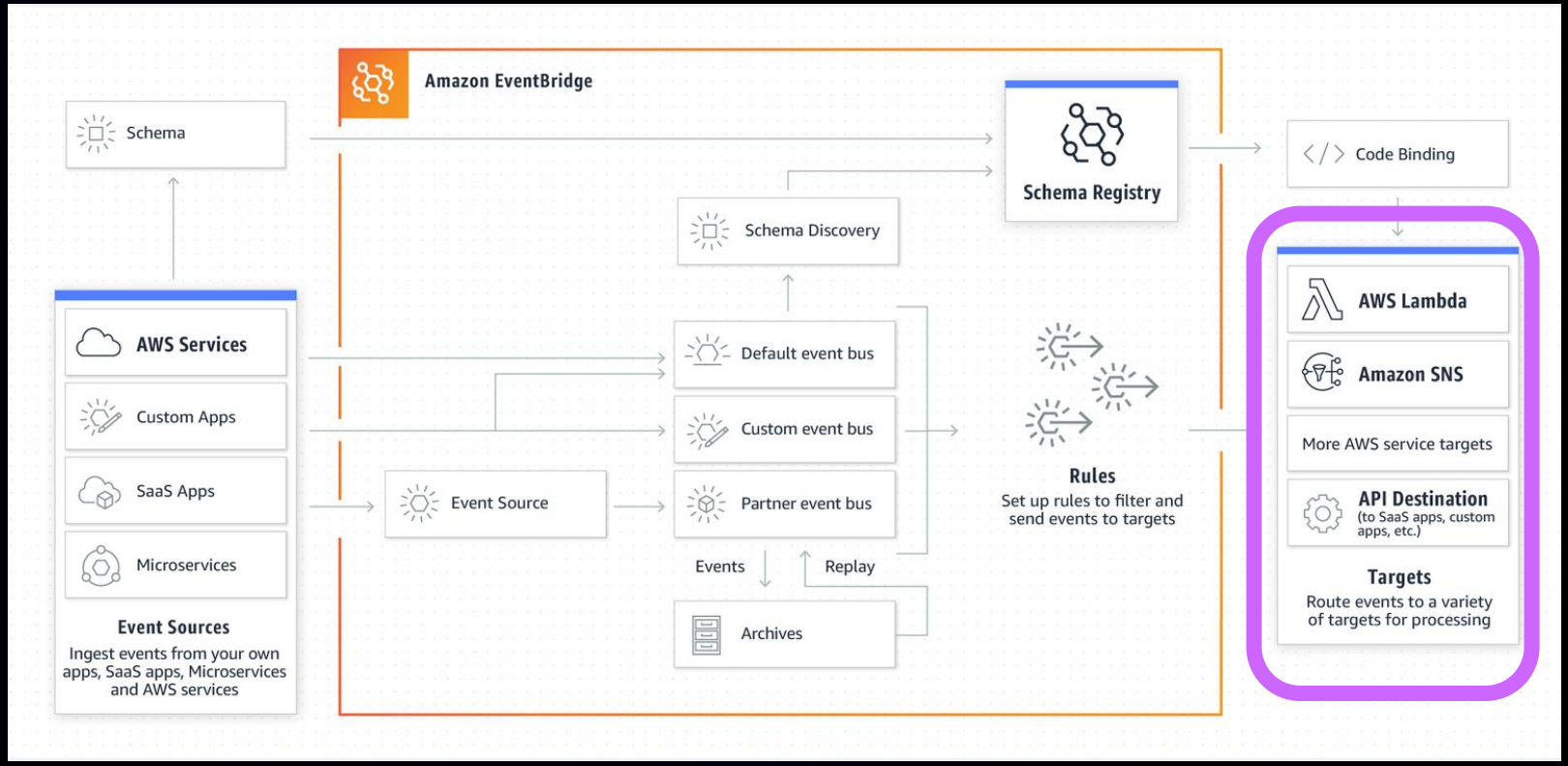
Amazon EventBridge is a simple, flexible, fully managed, pay-as-you-go, event bus service that makes it easy to ingest and process data from AWS services, your own applications, and SaaS applications




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Amazon EventBridge

Amazon EventBridge is a simple, flexible, fully managed, pay-as-you-go, event bus service that makes it easy to ingest and process data from AWS services, your own applications, and SaaS applications




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# EventBridge content-based routing rules

## EventBridge example event

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
    },
    "data": {
      "order-id": "1073459984",
      "created-at": "2021-11-26T16:05:09-04:00",
      "price": 24.62
      "currency": "AU",
    }
  }
}
```

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# EventBridge content-based routing rules

## EventBridge example event

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
    },
    "data": {
      "order-id": "1073459984",
      "created-at": "2021-11-26T16:05:09-04:00",
      "price": 24.62
      "currency": "AU",
    }
  }
}
```

## EventBridge example rule

```
{
  "detail": {
    "data":  {
      "currency": ["AU", "NZ"]
    }
  }
}
```

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# EventBridge content-based routing rules

## EventBridge example event

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
    },
    "data": {
      "order-id": "1073459984",
      "created-at": "2021-11-26T16:05:09-04:00",
      "price": 24.62
      "currency": "AU",
    }
  }
}
```

## EventBridge example rule

```
{
  "detail": {
    "data":  {
      "currency": ["AU", "NZ"]
    }
  }
}
```

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# EventBridge content-based routing rules

## EventBridge example event

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
    },
    "data": {
      "order-id": "1073459984",
      "created-at": "2021-11-26T16:05:09-04:00",
      "price": 24.62
      "currency": "AU",
    }
  }
}
```

## EventBridge example rule

```
{
  "detail": {
    "data":  {
      "currency": ["AU", "NZ"]
    }
  }
}
```



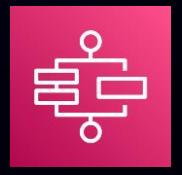
Match!

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Target 20+ AWS services and API destinations


AWS Lambda


AWS Step Functions


Amazon Kinesis Data Streams


Amazon CloudWatch


AWS CodePipeline


Amazon ECS


Amazon SNS


Amazon Kinesis Data Firehose


AWS Systems Manager


AWS CodeBuild


Amazon EC2

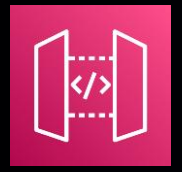

Amazon SQS


AWS Glue


AWS Systems Manager Incident Manager


Amazon SageMaker


AWS Batch


Amazon API Gateway


Amazon Redshift


Amazon Inspector


API destinations

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Event-driven architecture

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Properties of events

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
      "idempotency-key": "c1b95b88"
    },
    "data": {
      "order-id": "1073459984"
    }
}
```

- Events are signals that a system's state has changed
- Events occur in the past (e.g., OrderCreated)
- Events cannot be changed (immutable)
- Decrease coupling by restricting information to key data

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Sparse events vs. full state descriptions

Order 123 was created at 10:47 a.m. by customer 456



## Sparse events

Order 123 was created at 10:47 a.m. by customer 456. The current status is Open, the total was $237.51, …



## Full state description

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

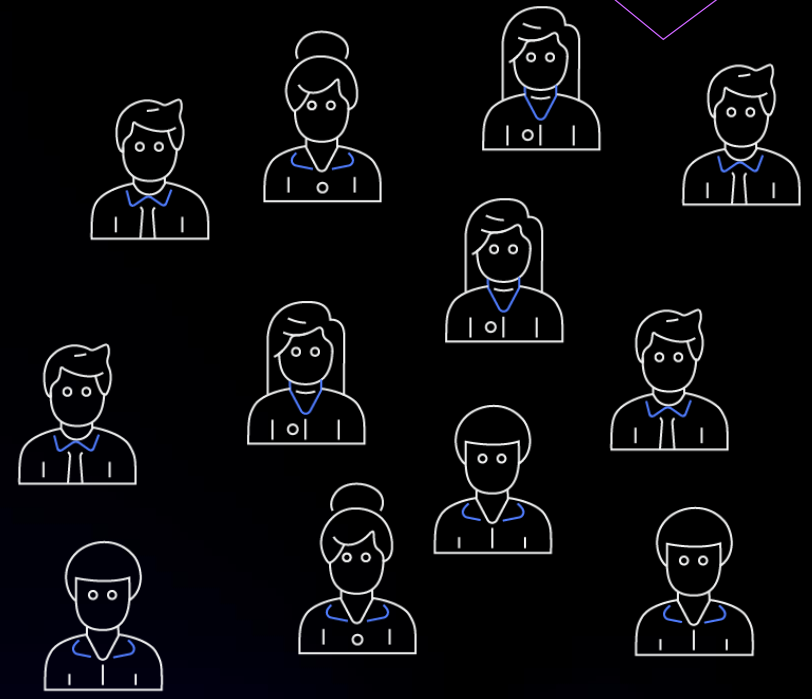
# Considerations with sparse events

Order 123 was created at 10:47 a.m. by customer 456



Events

What are the details for order 123?



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Considerations with full state descriptions

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
      "idempotency-key": "c1b95b88"
    },
    "data": {
      "order-id": "1073459984",
      "status": "Open",
      "total": "237.51"
    }
}
```

- Event schemas should be backwards compatible

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Considerations with full state descriptions

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
      "idempotency-key": "c1b95b88"
    },
    "data": {
      "order-id": "1073459984",
      "status": "Open",
      "total": "237.51"
    }
}
```

- Event schemas should be backwards compatible
- Cost to calculate values can increase over time




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Considerations with full state descriptions

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
      "idempotency-key": "c1b95b88"
    },
    "data": {
      "order-id": "1073459984",
      "status": "Open",
      "total": "237.51"
    }
}
```

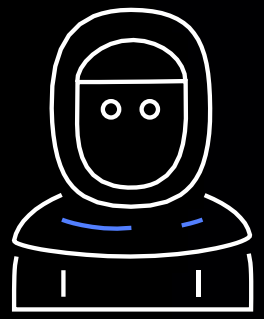
- Event schemas should be backwards compatible
- Cost to calculate values can increase over time




aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Choreograph events between domains using subscriptions



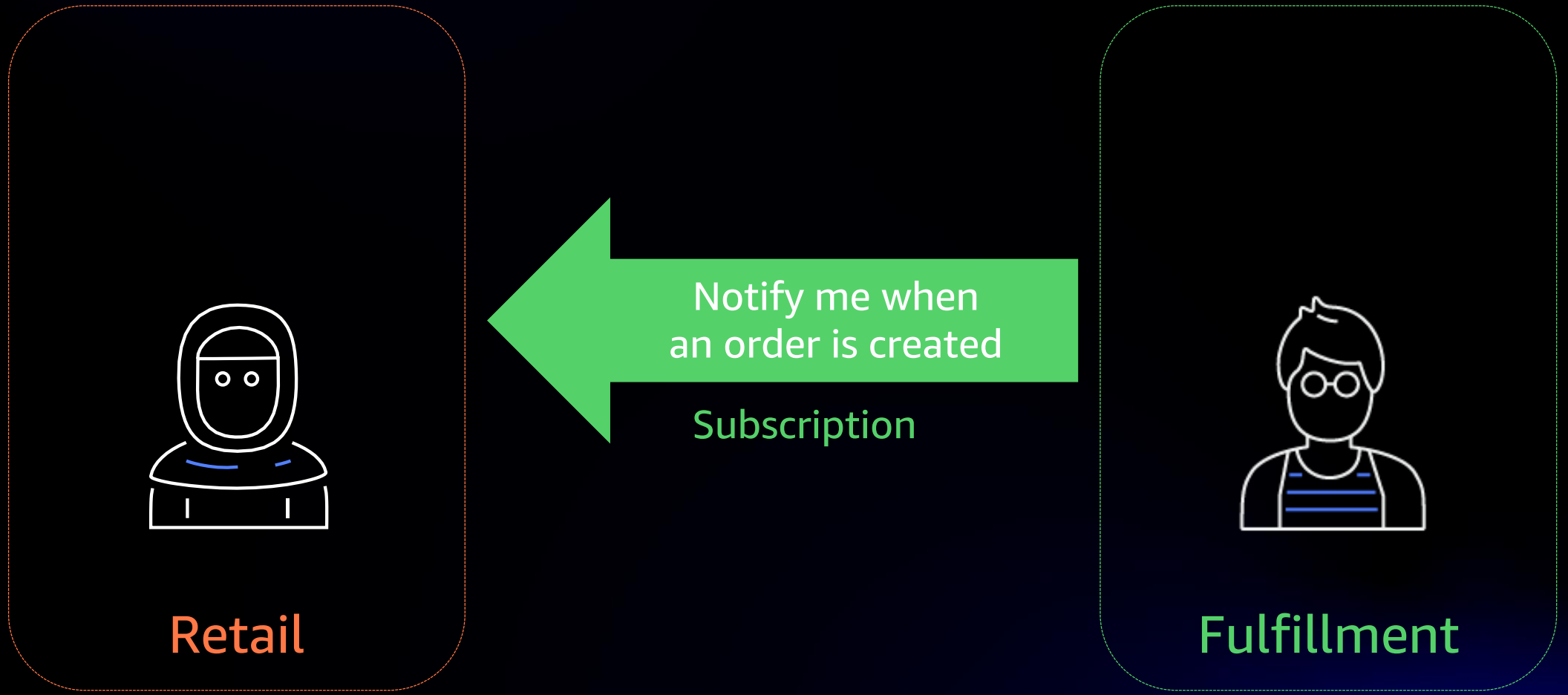
Retail



Fulfillment

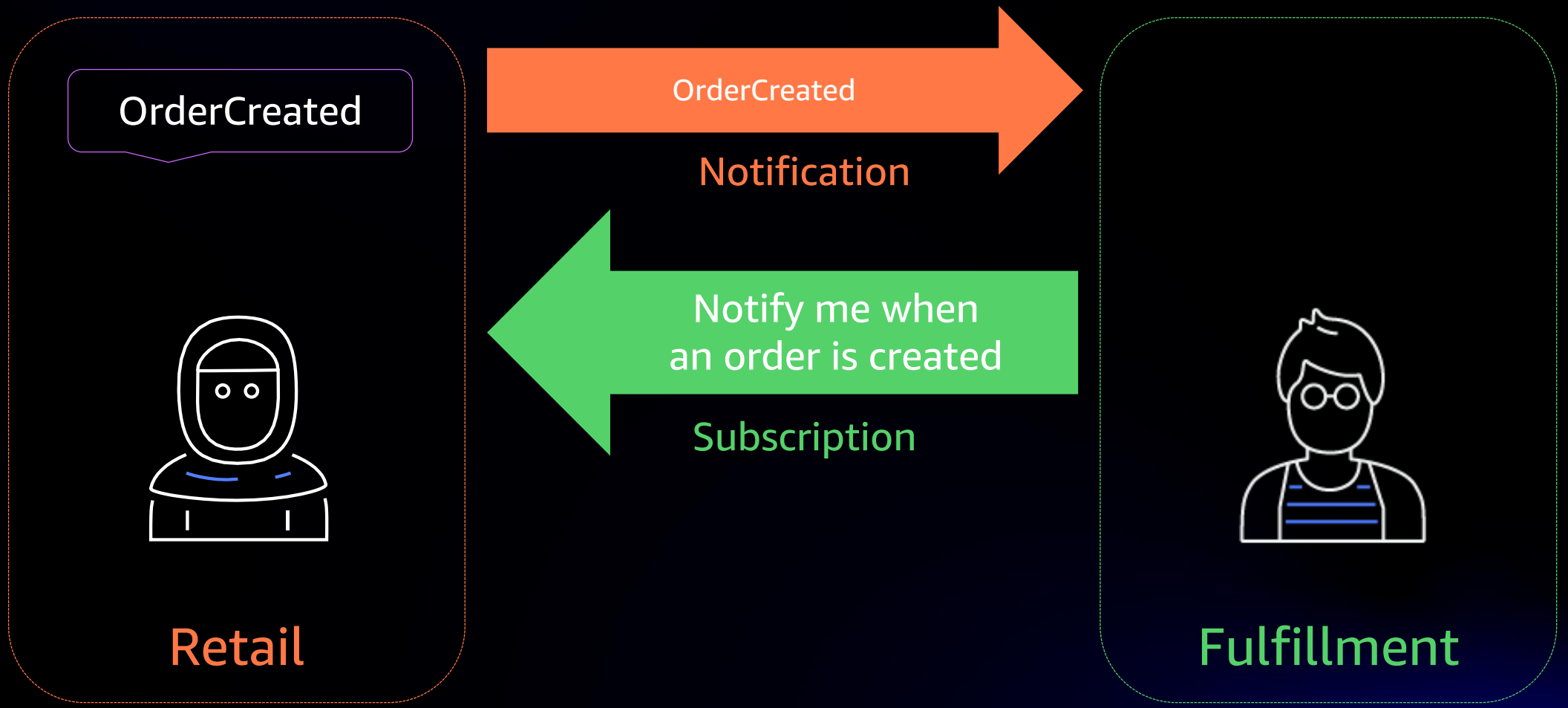
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Choreograph events between domains using subscriptions




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Choreograph events between domains using subscriptions




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Event uniqueness

## Idempotency

Operation will return the same results whether it is called once or multiple times (EIP)

## Idempotency key

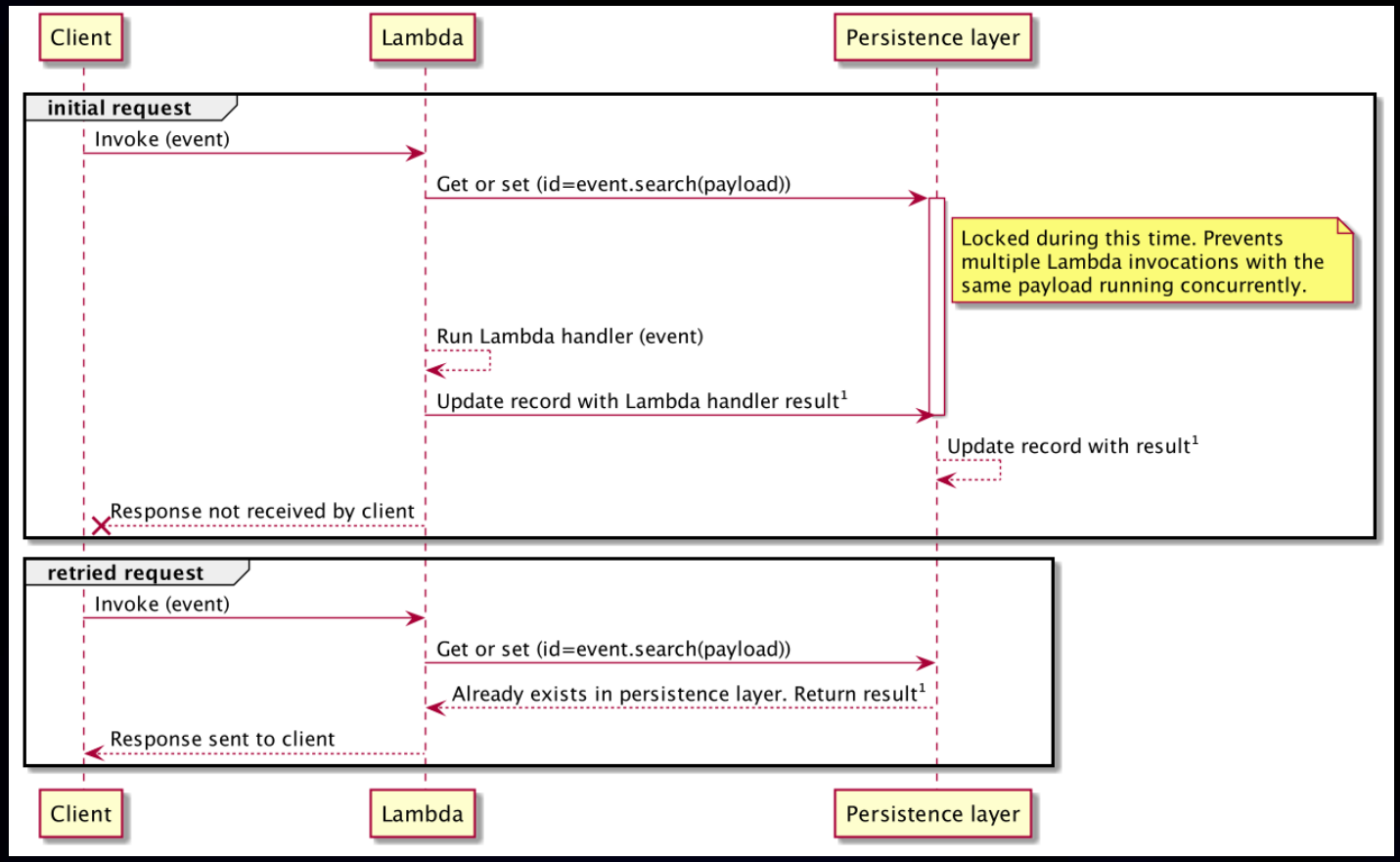
Assigned to the message by the sender to simplify deduplication by the receiver

```
{
  "source": "com.orders",
  "detail-type": "OrderCreated",
  "detail": {
    "metadata": {
      "idempotency-key": "c1b95b88"
    },
    "data": {
      "order-id": "1073459984"
    }
}
```

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Event deduplication strategies

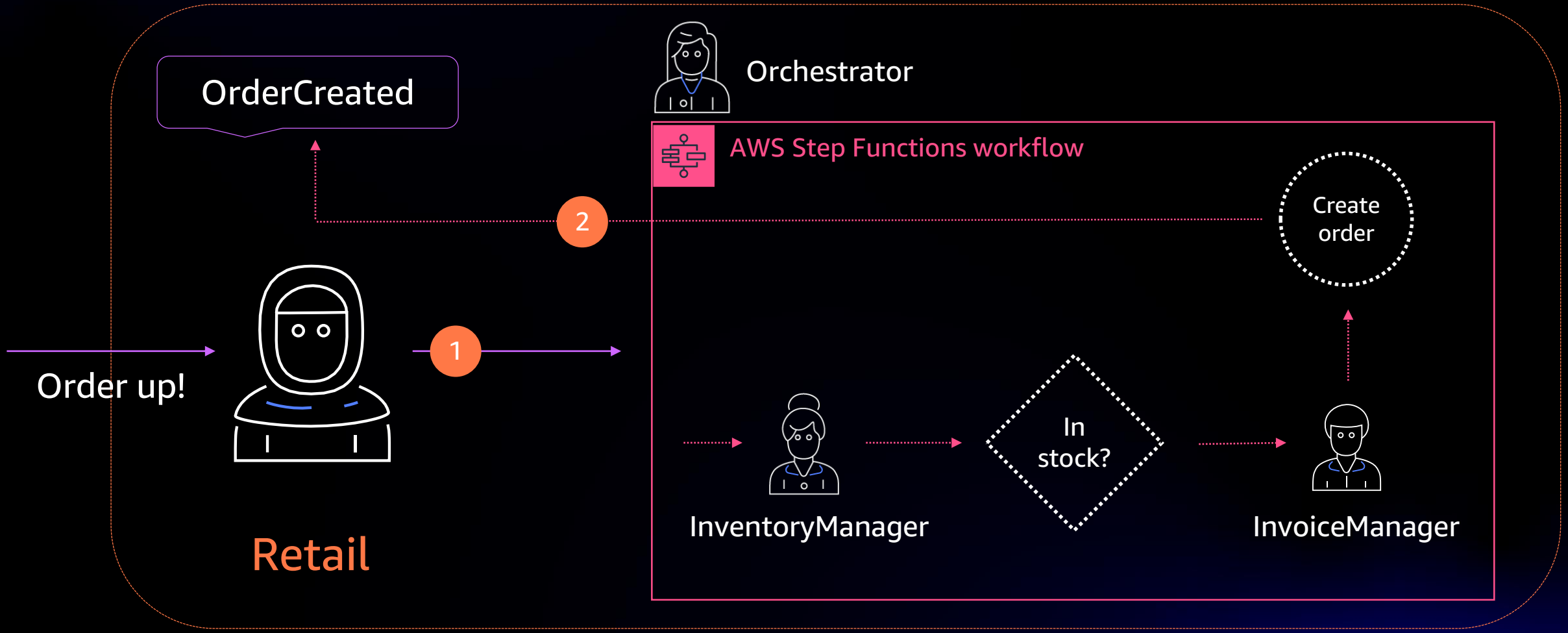
- Manage idempotent state in an external data store
- Specify a time window for deduplication
- Lambda Powertools Python Idempotency Library
- Making retries safe with idempotent APIs (AWS Builders' Library)




(Lambda Powertools Python Idempotency, https://bit.ly/3iWKu7z)

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Orchestrate a business process within a domain, resulting in a published event



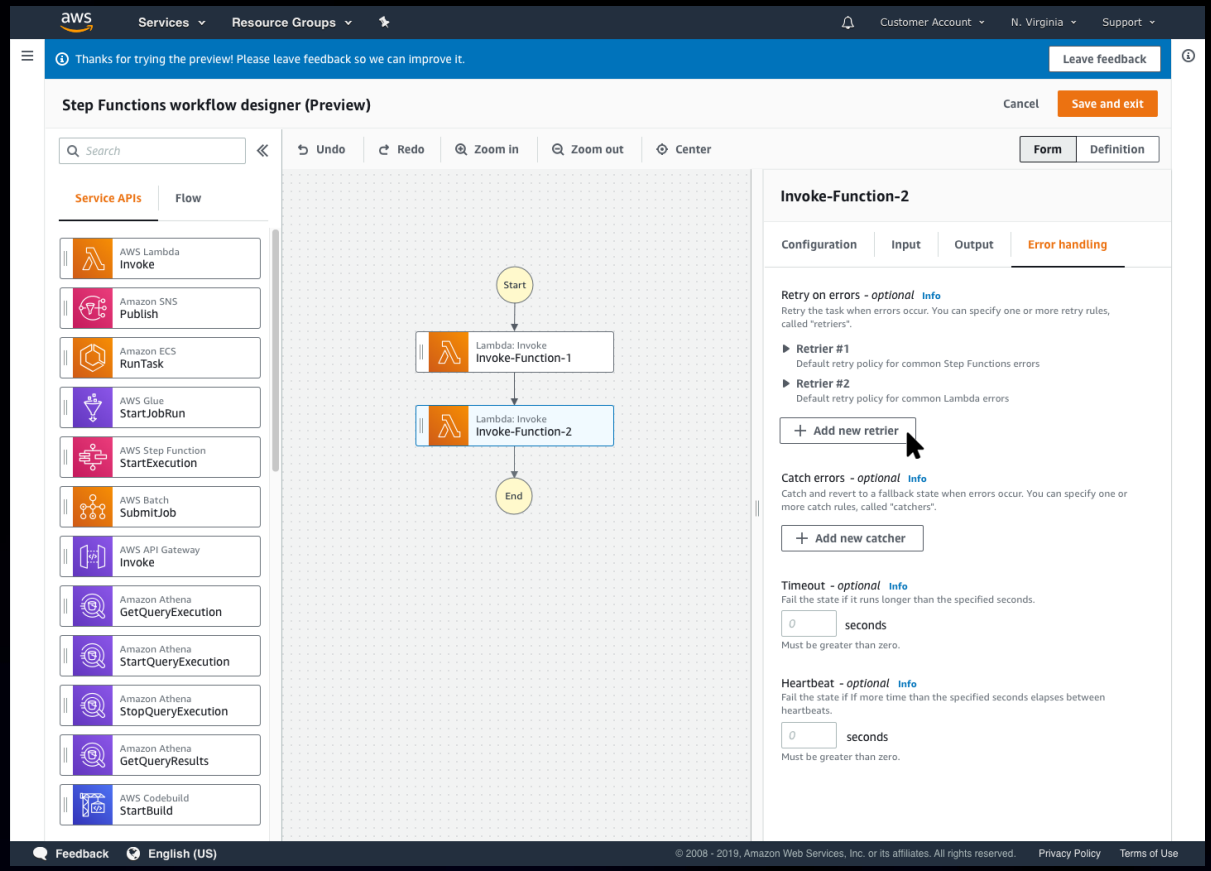

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# AWS Step Functions

The workflows you build with Step Functions are called state machines, and each step of your workflow is called a state

When you execute your state machine, each move from one state to the next is called a state transition

You can reuse components, easily edit the sequence of steps, or swap out the code called by task states as your needs change




AWS STEP FUNCTIONS WORKFLOW STUDIO

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Visual workflows

## Define

JSON – Amazon States Language

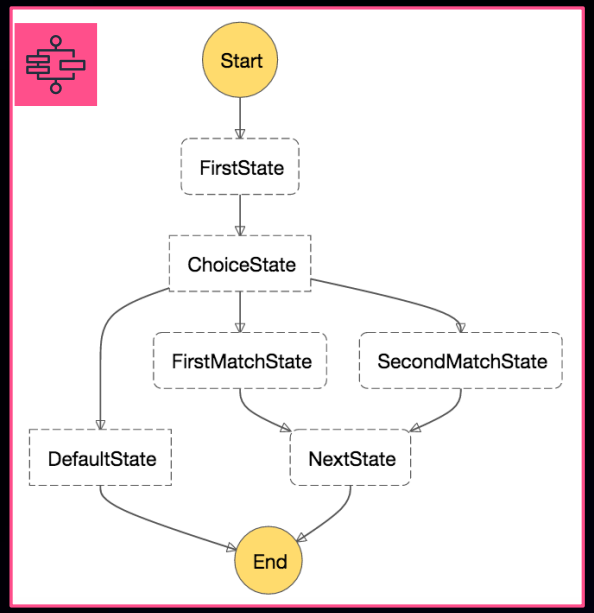
```
{
  "Comment": "An AWL example using a choice state.",
  "StartAt": "FirstState",
  "States": {
    "FirstState": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:REGION:ACCOUNT_ID:function:FUNCTION_NAME",
      "Next": "ChoiceState"
    },
    "ChoiceState": {
      "Type" : "Choice",
      "Choices": [
        {
```

**CDK** TypeScript, JavaScript, Python, Java, C#

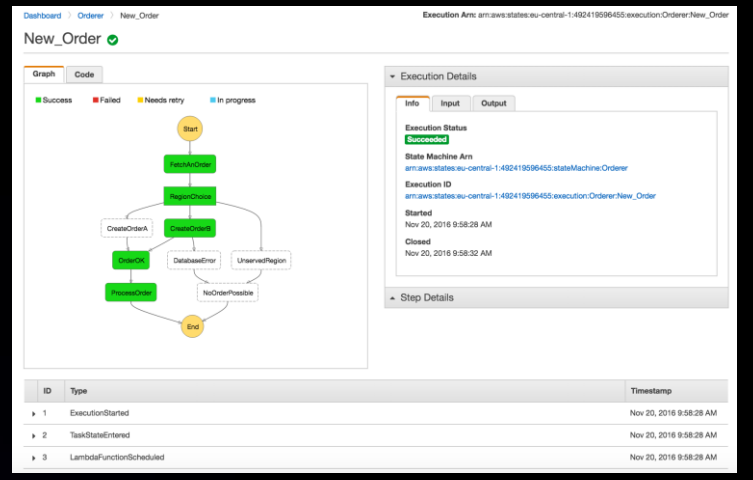
**Data Science SDK**

Python

## Visualize




## Execute and monitor







© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Step Functions integration types



## Optimized integrations

Customized to simplify the usage of 17 AWS services

Supported integration patterns:

- Request Response
- Wait for a Callback (.waitForTaskToken)
- Run a Job (.sync)




## AWS SDK integrations

Call 200 AWS services directly (9,000+ API actions)

Supported integration patterns:

- Request Response
- Wait for a Callback (.waitForTaskToken)

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Request Response

SERVICE INTEGRATION PATTERN

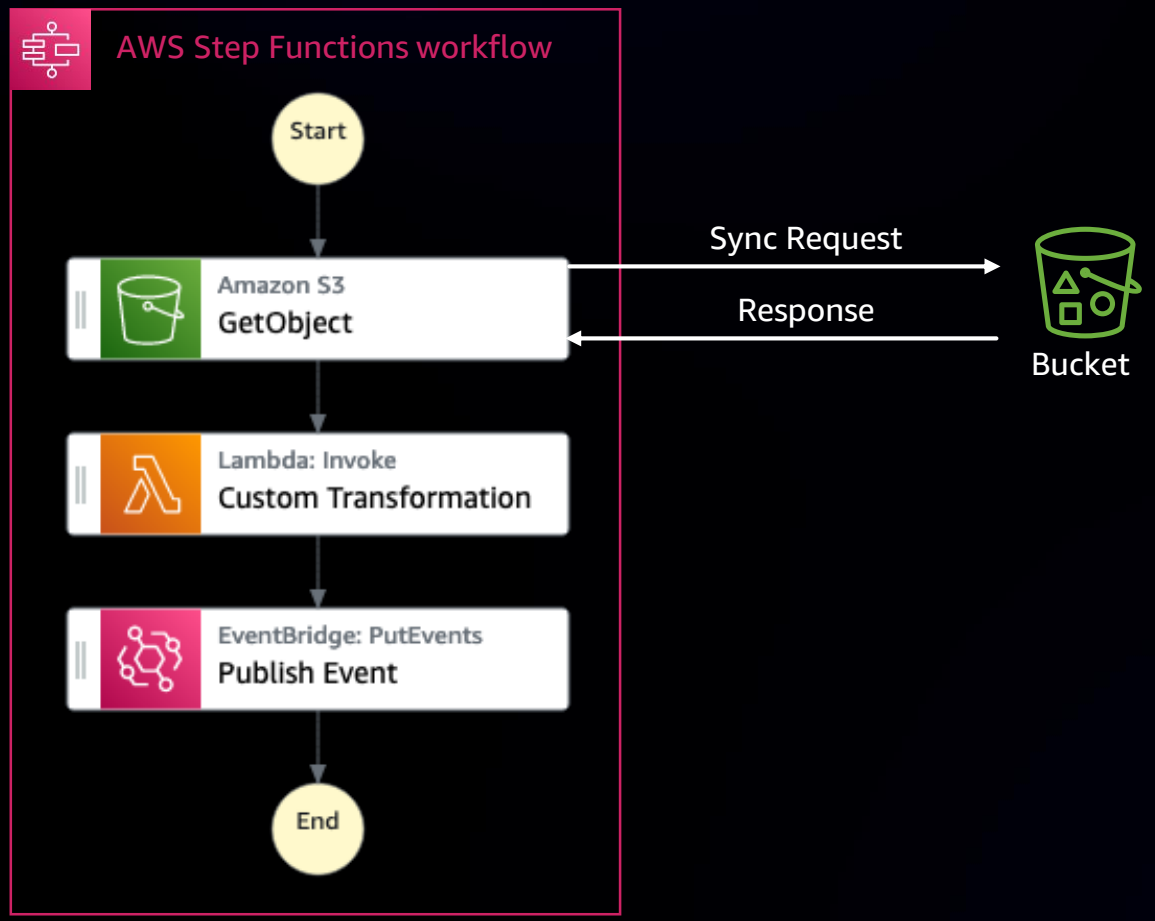



- Amazon EMR
- AWS Lambda
- Amazon SQS
- AWS Glue
- Amazon ECS/ AWS Fargate
- Amazon SNS
- Amazon Athena
- Amazon EKS
- AWS Step Functions
- Amazon SageMaker
- AWS Batch
- Amazon EventBridge
- Amazon DynamoDB
- AWS SDK
- Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Request Response

SERVICE INTEGRATION PATTERN

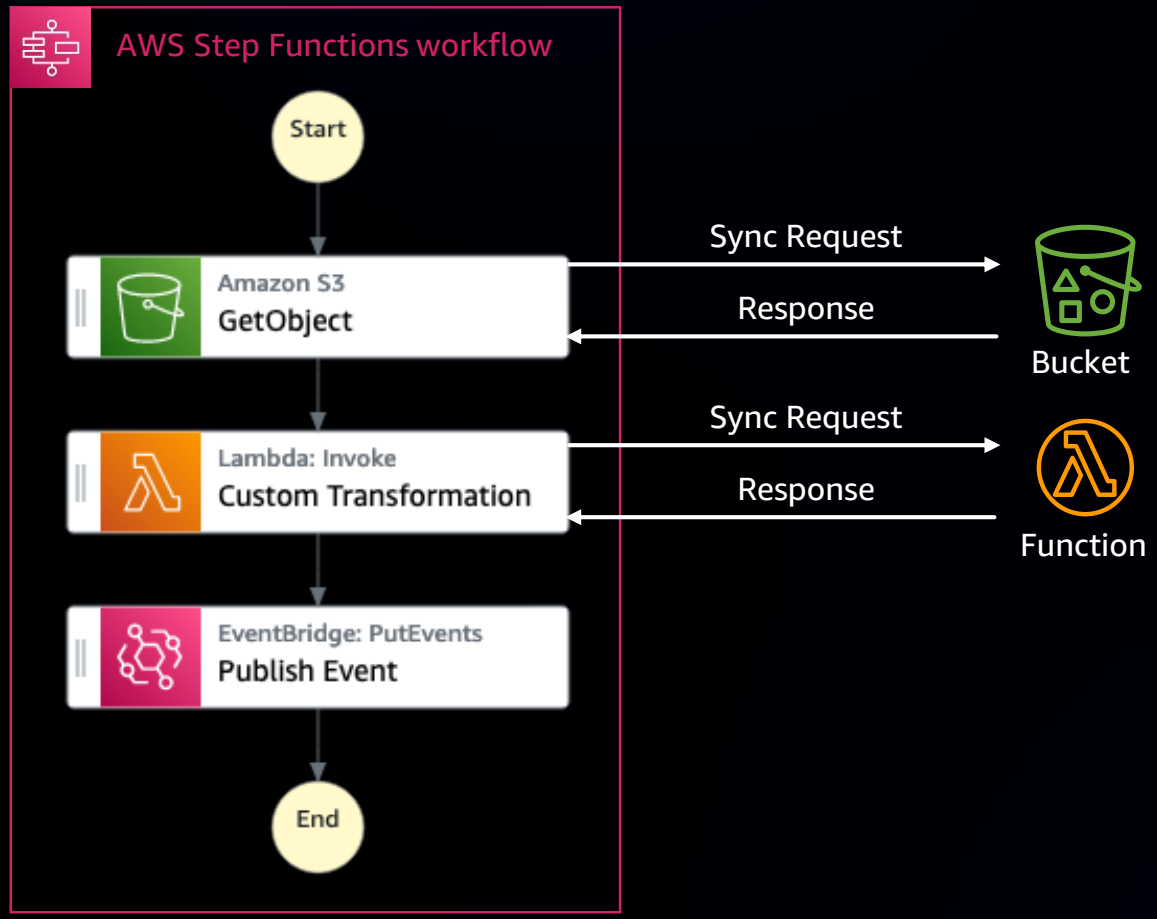



- Amazon EMR
- AWS Lambda
- Amazon SQS
- AWS Glue
- Amazon ECS/ AWS Fargate
- Amazon SNS
- Amazon Athena
- Amazon EKS
- AWS Step Functions
- Amazon SageMaker
- AWS Batch
- Amazon EventBridge
- Amazon DynamoDB
- AWS SDK
- Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Request Response

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Lambda

Amazon SQS

AWS Glue

Amazon ECS/
AWS Fargate

Amazon SNS

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

Amazon EventBridge

Amazon DynamoDB

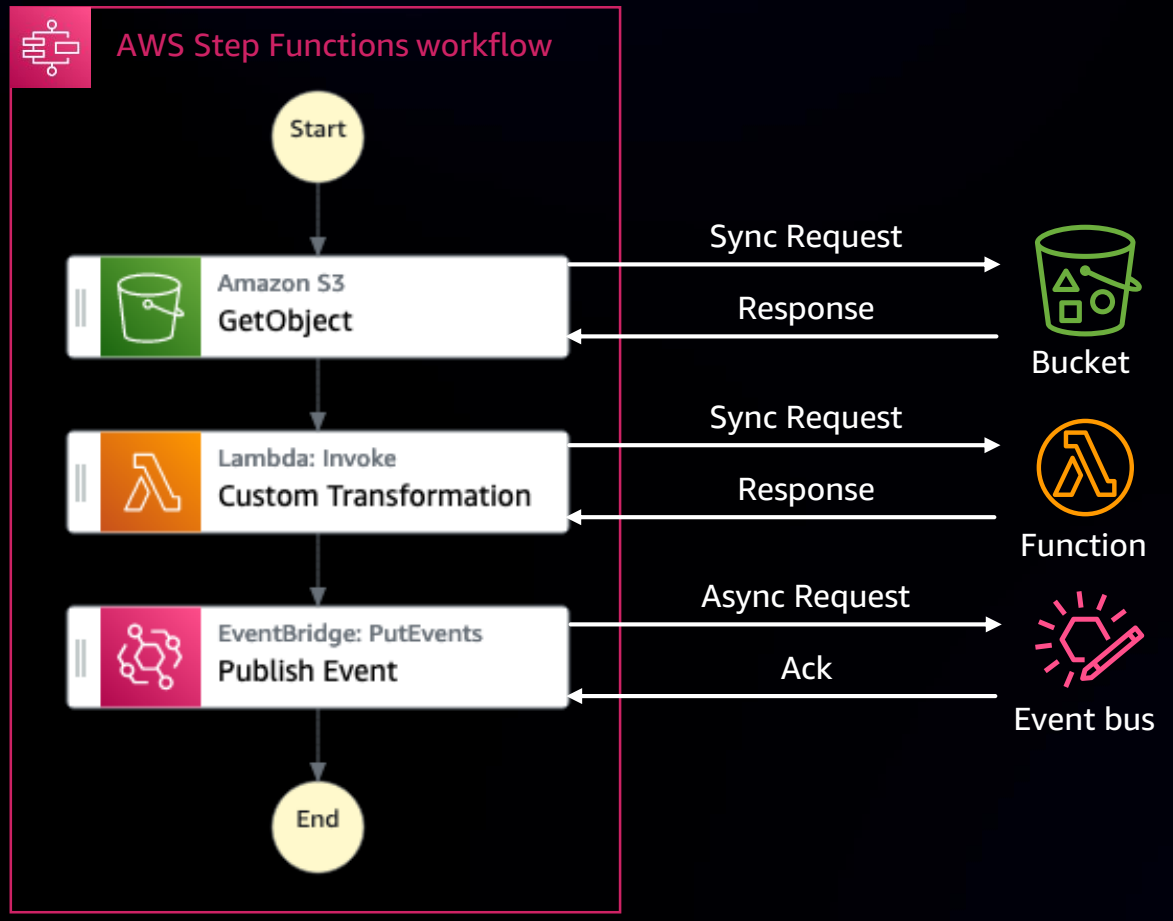
AWS SDK

Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Request Response

SERVICE INTEGRATION PATTERN

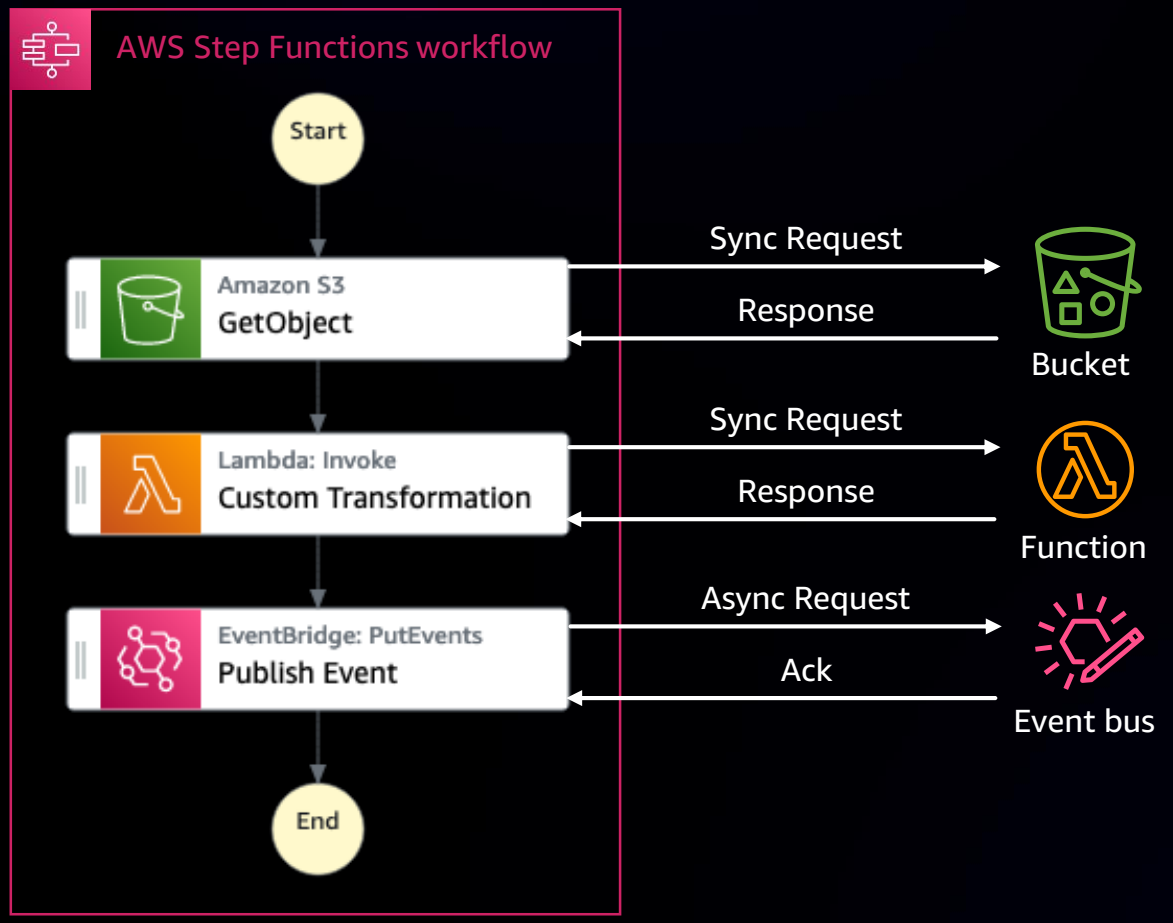



- Amazon EMR
- AWS Lambda
- Amazon SQS
- AWS Glue
- Amazon ECS/ AWS Fargate
- Amazon SNS
- Amazon Athena
- Amazon EKS
- AWS Step Functions
- Amazon SageMaker
- AWS Batch
- Amazon EventBridge
- Amazon DynamoDB
- AWS SDK
- Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Request Response

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Lambda

Amazon SQS

AWS Glue

Amazon ECS/
AWS Fargate

Amazon SNS

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

Amazon EventBridge

Amazon DynamoDB

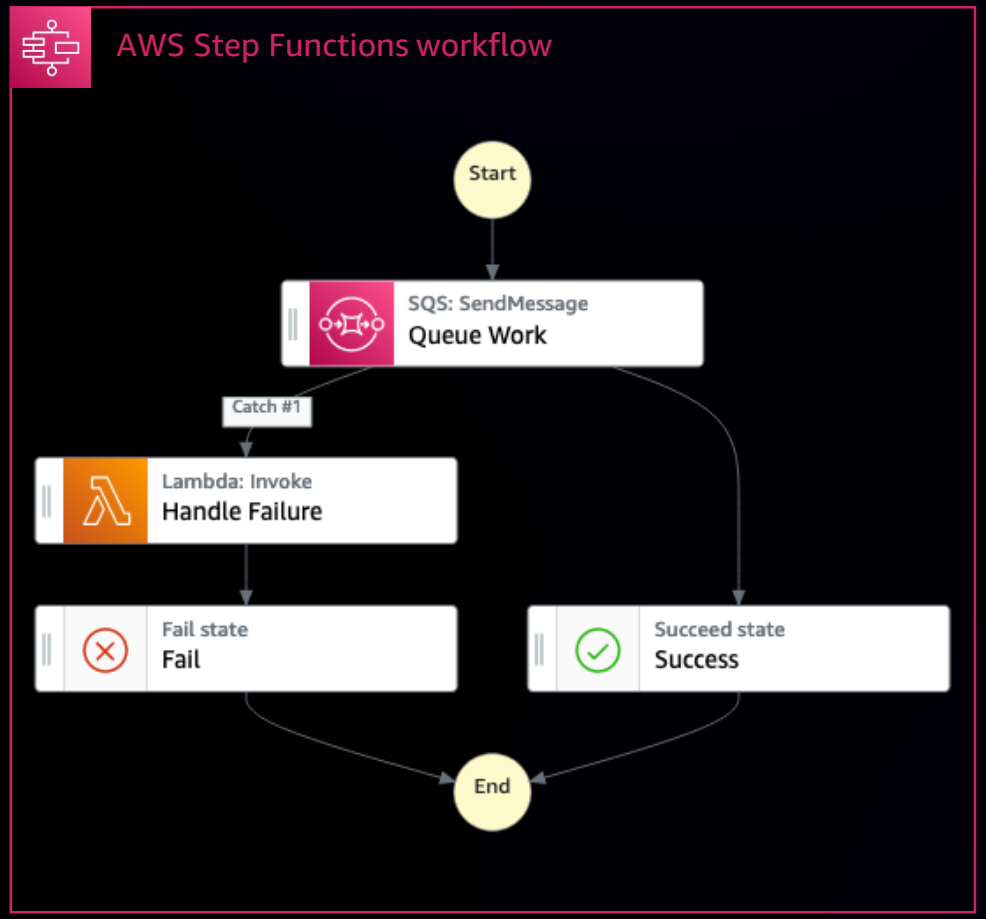
AWS SDK

Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Wait for a Callback (.waitForTaskToken)

SERVICE INTEGRATION PATTERN

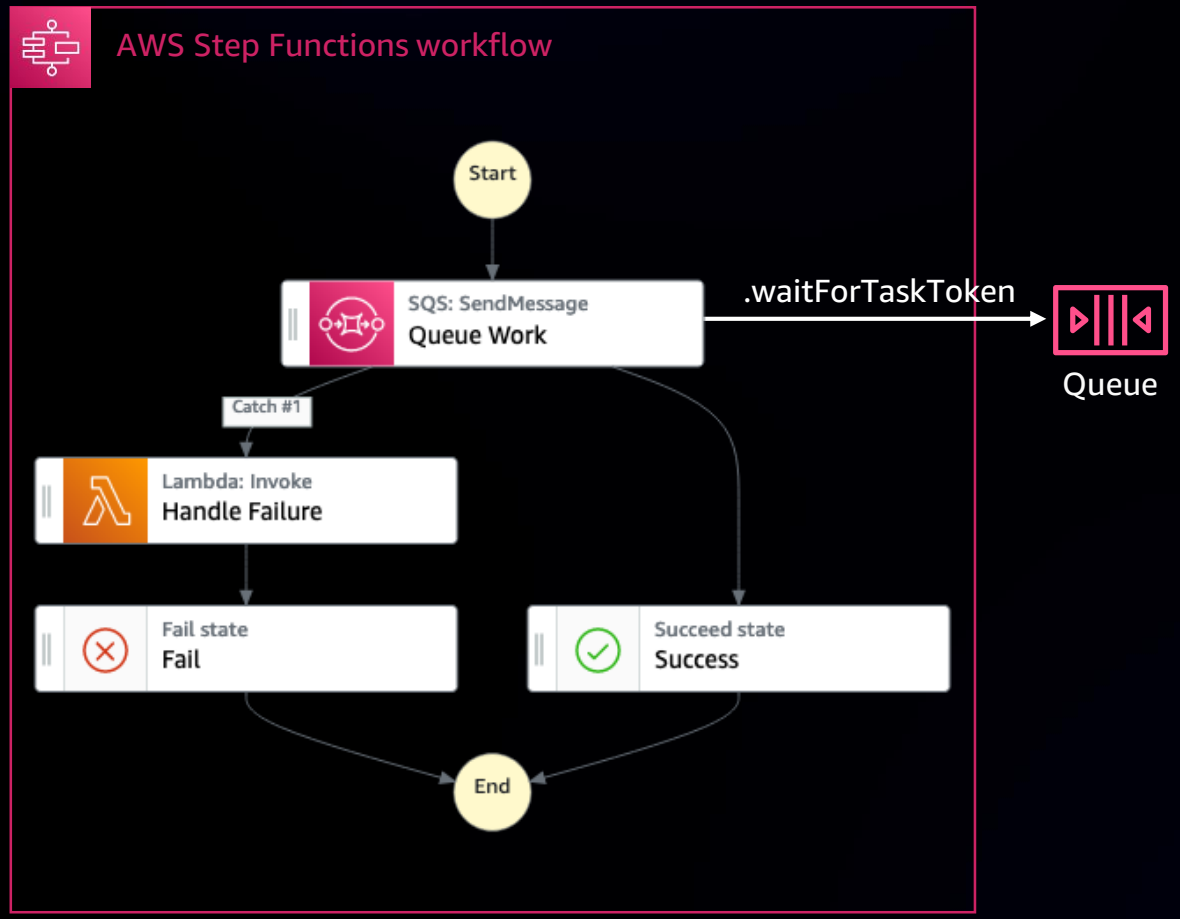



- AWS Lambda
- Amazon SQS
- Amazon ECS/ AWS Fargate
- Amazon SNS
- AWS Step Functions
- Amazon EventBridge
- AWS SDK
- **Amazon API Gateway**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Wait for a Callback (.waitForTaskToken)

SERVICE INTEGRATION PATTERN




AWS Lambda

Amazon SQS

Amazon ECS/
AWS Fargate

Amazon SNS

AWS Step Functions

Amazon EventBridge

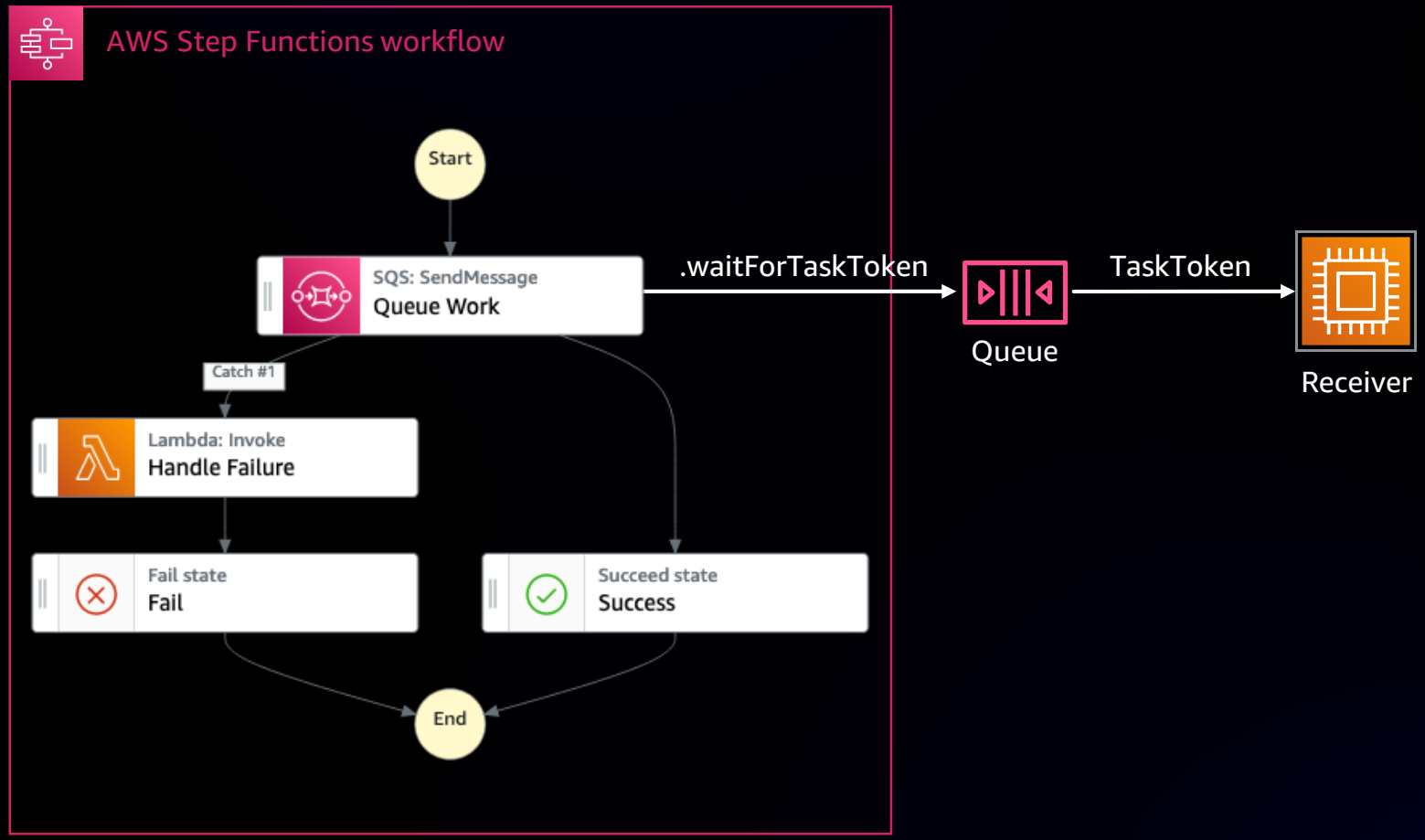
AWS SDK

**Amazon API Gateway**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Wait for a Callback (.waitForTaskToken)

SERVICE INTEGRATION PATTERN




AWS Lambda

Amazon SQS

Amazon ECS/
AWS Fargate

Amazon SNS

AWS Step Functions

Amazon EventBridge

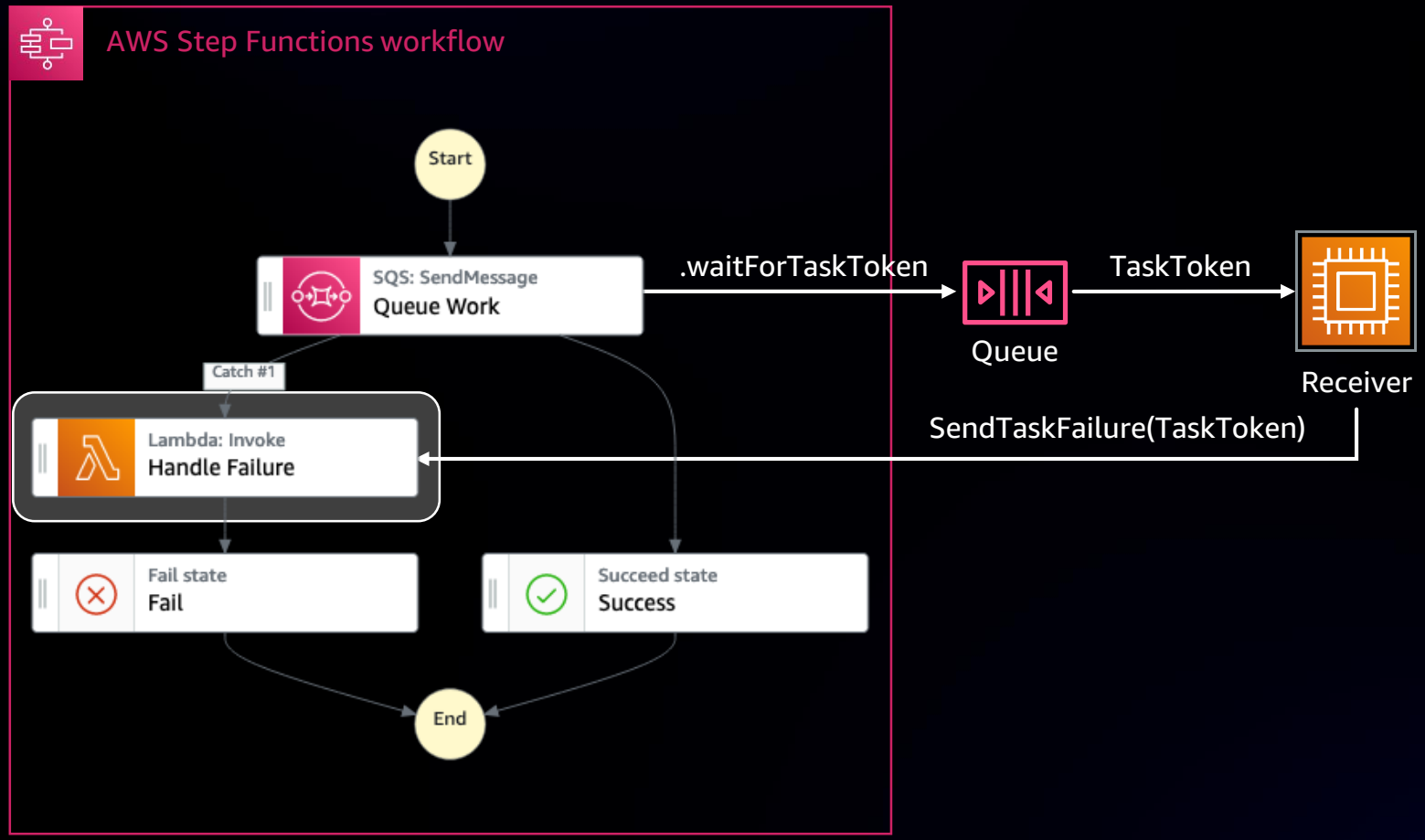
AWS SDK

**Amazon API Gateway**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Wait for a Callback (.waitForTaskToken)

SERVICE INTEGRATION PATTERN




AWS Lambda

Amazon SQS

Amazon ECS/
AWS Fargate

Amazon SNS

AWS Step Functions

Amazon EventBridge

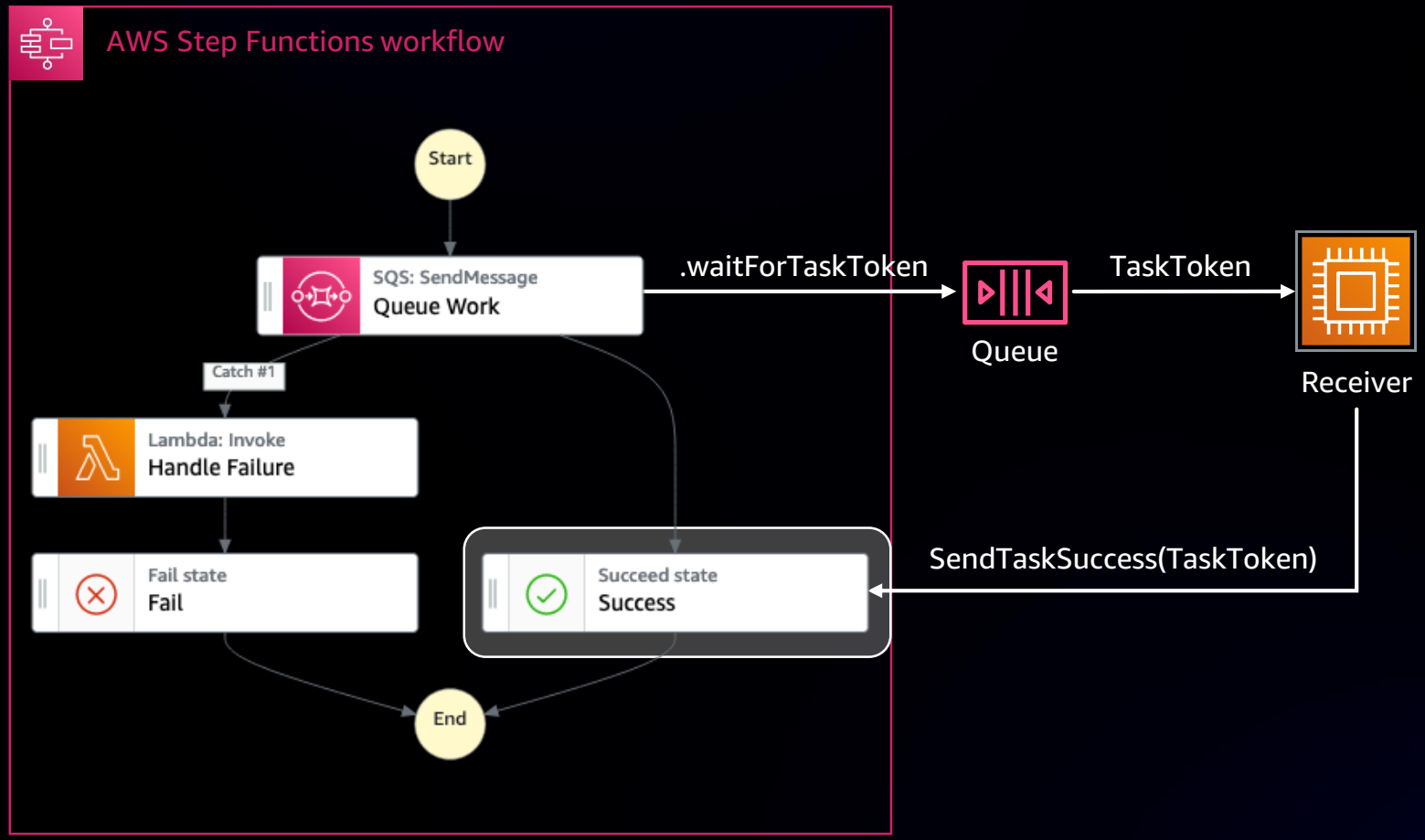
AWS SDK

**Amazon API Gateway**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Wait for a Callback (.waitForTaskToken)

SERVICE INTEGRATION PATTERN




AWS Lambda

Amazon SQS

Amazon ECS/
AWS Fargate

Amazon SNS

AWS Step Functions

Amazon EventBridge

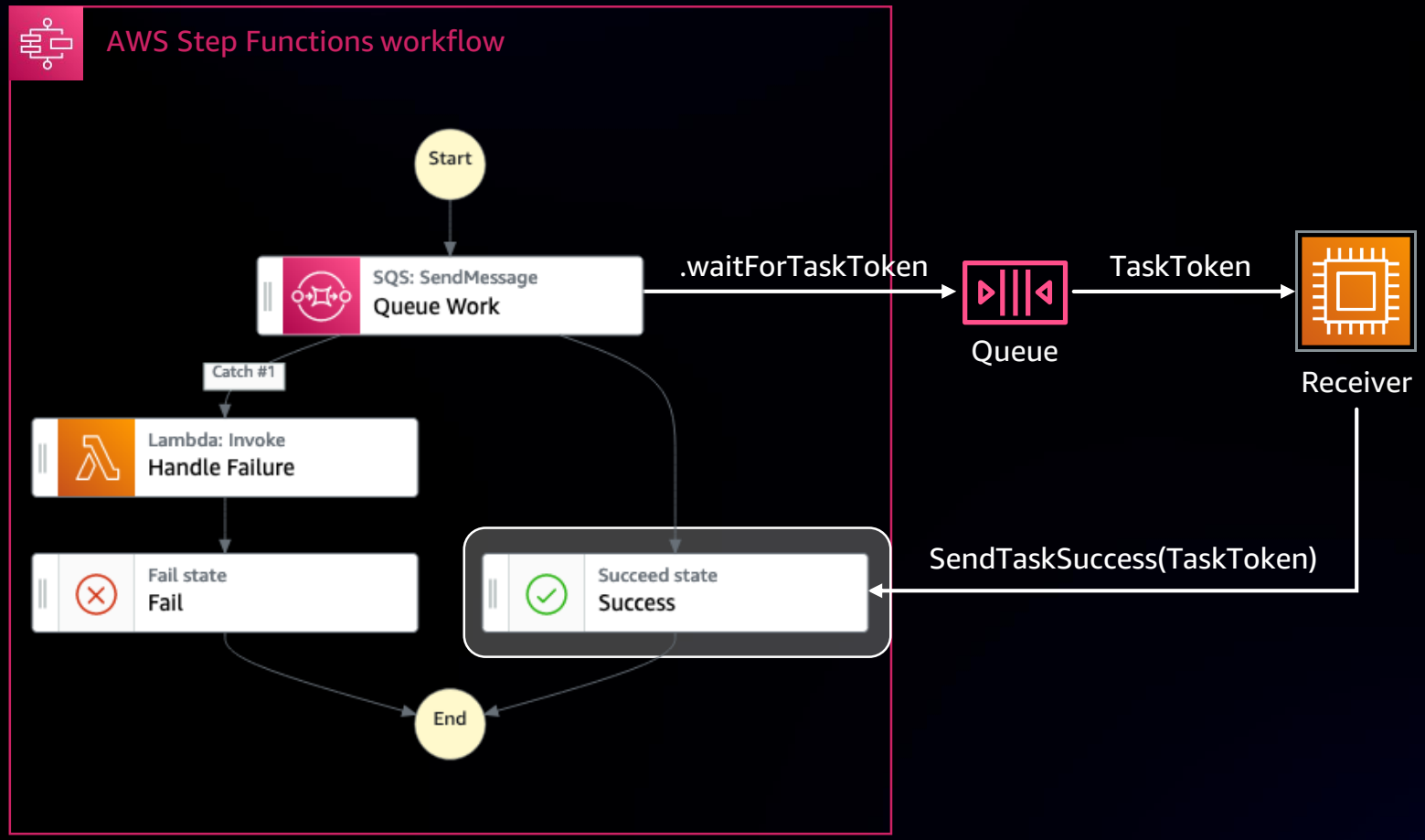
AWS SDK

**Amazon API Gateway**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Wait for a Callback (.waitForTaskToken)

SERVICE INTEGRATION PATTERN




AWS Lambda

Amazon SQS

Amazon ECS/
AWS Fargate

Amazon SNS

AWS Step Functions

Amazon EventBridge

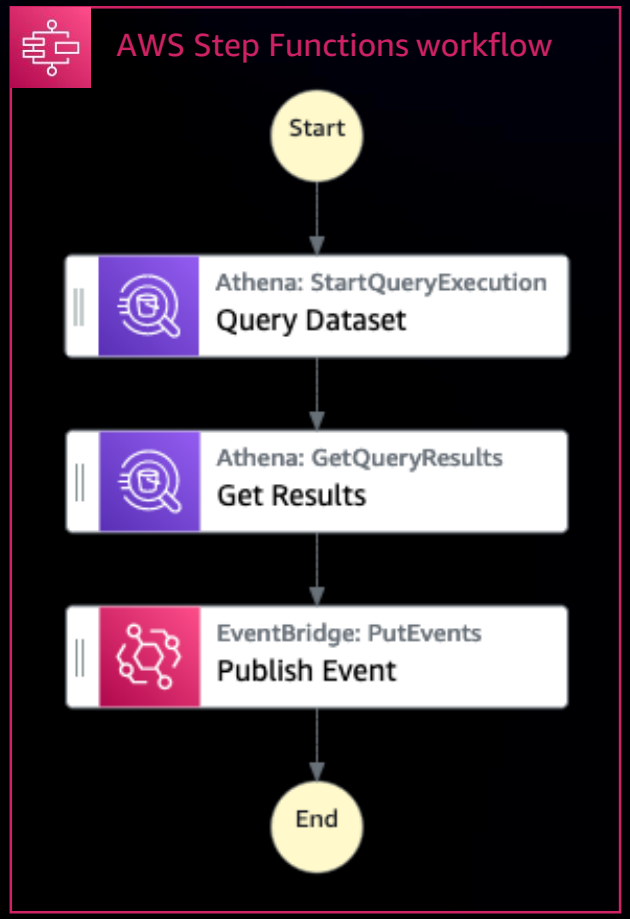
AWS SDK

**Amazon API Gateway**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Run a Job (.sync)

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Glue

Amazon ECS/AWS Fargate

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

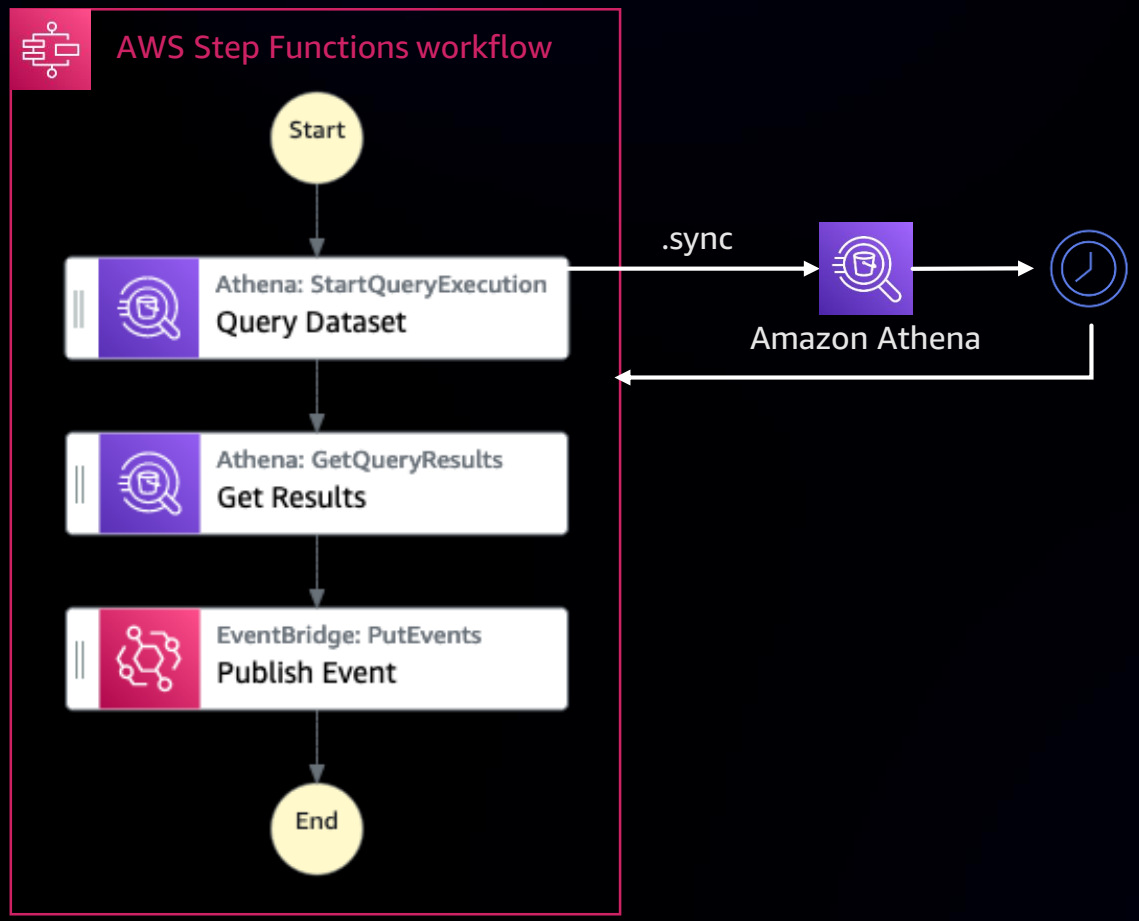
AWS CodeBuild

Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Run a Job (.sync)

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Glue

Amazon ECS/AWS Fargate

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

AWS CodeBuild

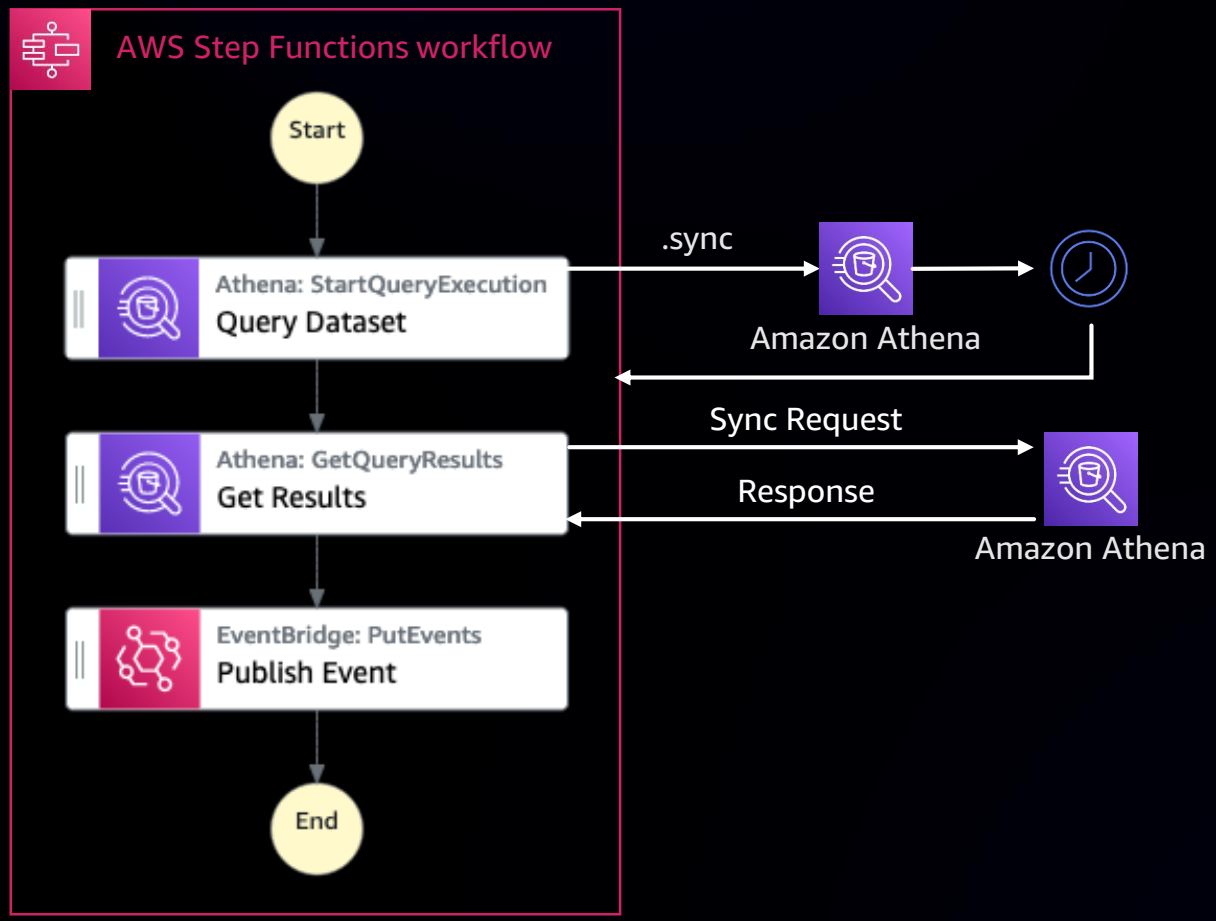
Amazon API Gateway

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Run a Job (.sync)

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Glue

Amazon ECS/AWS Fargate

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

AWS CodeBuild

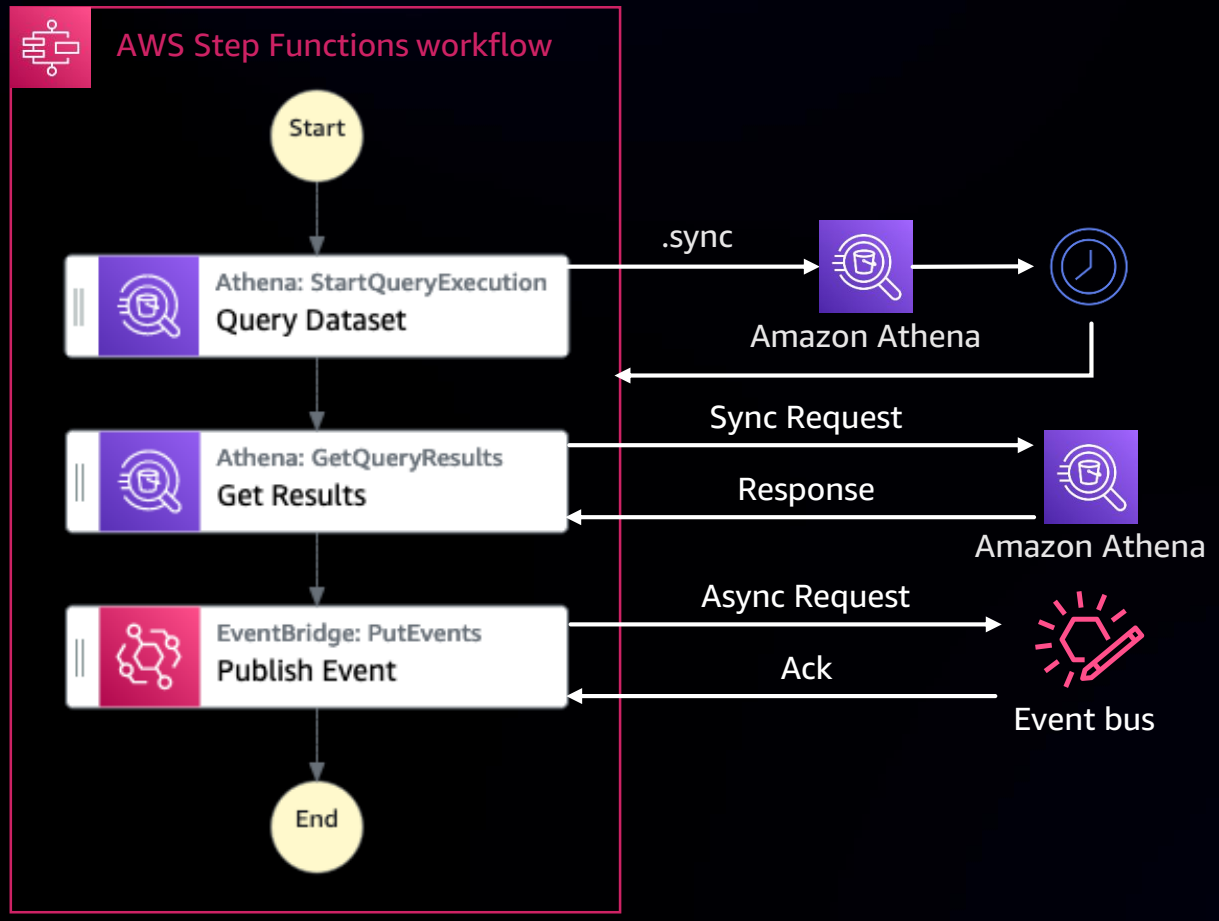
Amazon API Gateway

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Run a Job (.sync)

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Glue

Amazon ECS/AWS Fargate

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

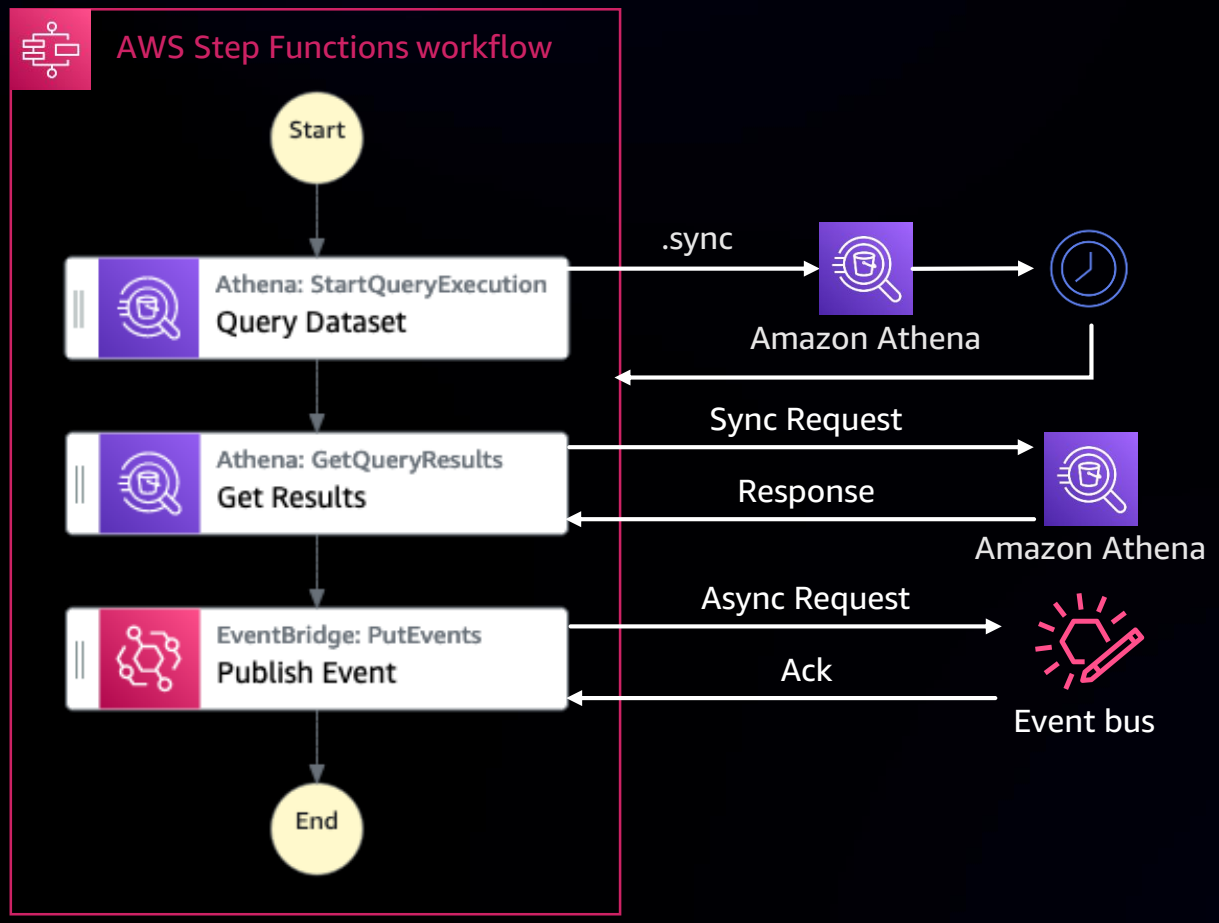
AWS CodeBuild

Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Run a Job (.sync)

SERVICE INTEGRATION PATTERN




Amazon EMR

AWS Glue

Amazon ECS/AWS Fargate

Amazon Athena

Amazon EKS

AWS Step Functions

Amazon SageMaker

AWS Batch

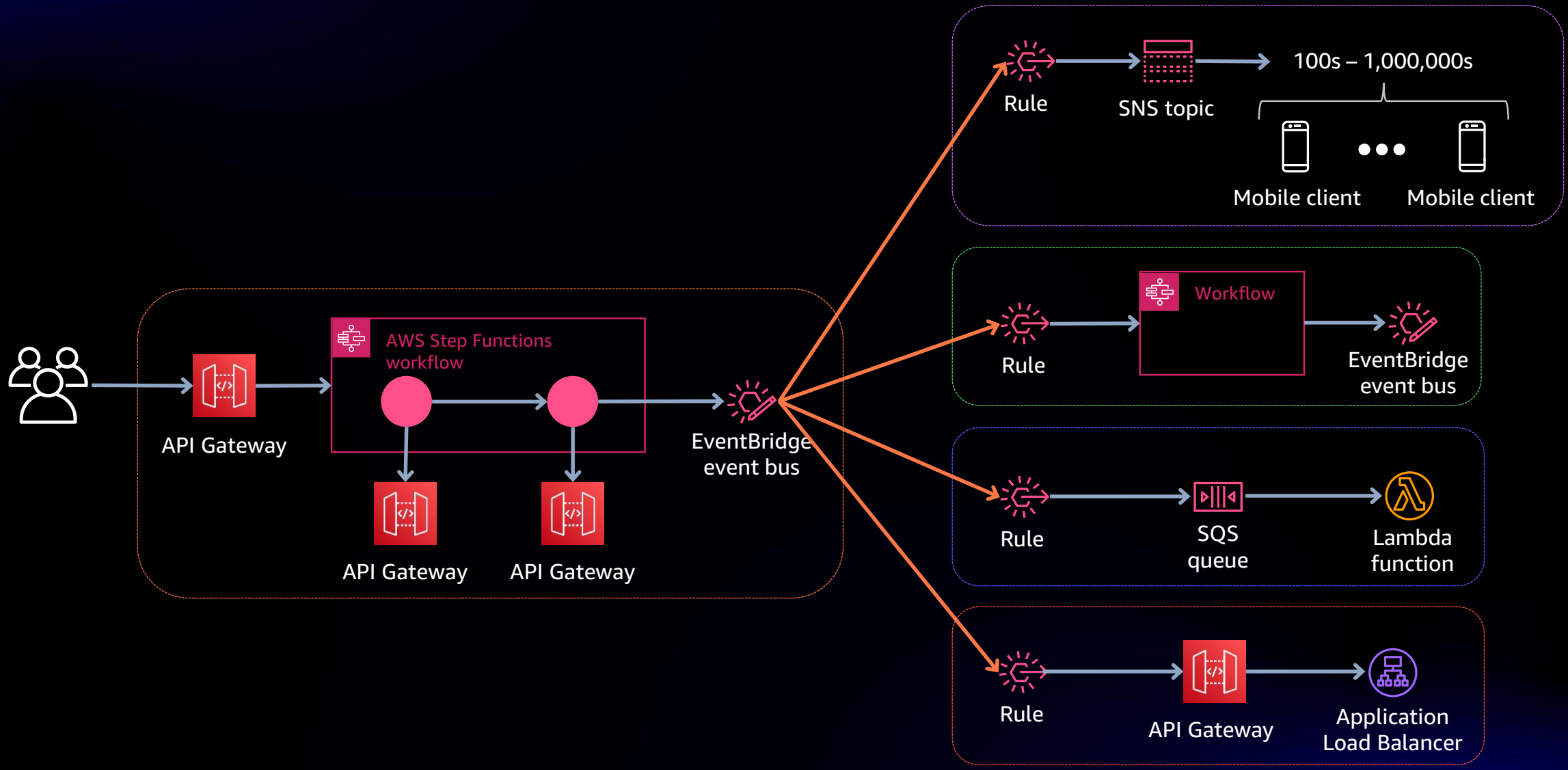
AWS CodeBuild

AWS SDK

Amazon API Gateway

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Better together: Orchestration + choreography




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Taco Bell

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# About Taco Bell

**7,000+**
US locations

**600+**
International locations

**$110M+**
Foundation awards

**42M+**
Customers per week

**1B+**
Orders per year






© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Las Vegas Taco Bell Cantina!





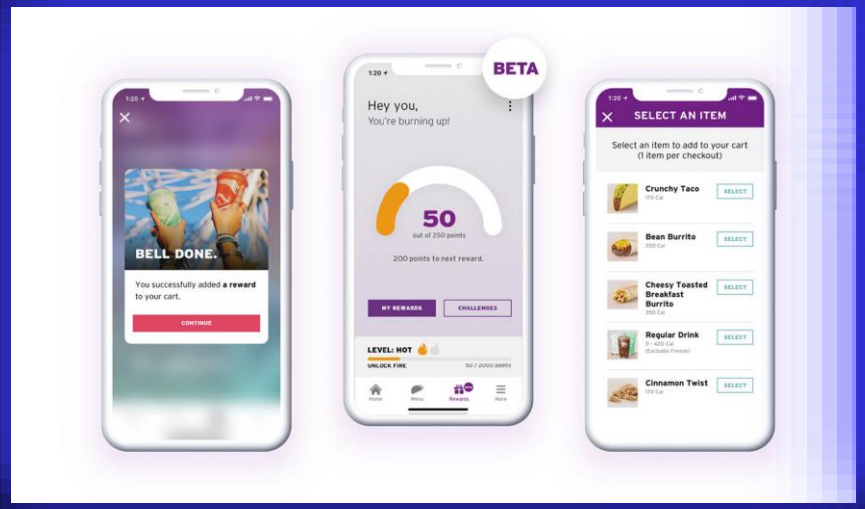

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Tech at Taco Bell















© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Digital commerce channels



ECOMMERCE MOBILE APPS



ECOMMERCE WEB APP



KIOSK



DELIVERY

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Digital commerce channels



ECOMMERCE MOBILE APPS



ECOMMERCE WEB APP



KIOSK

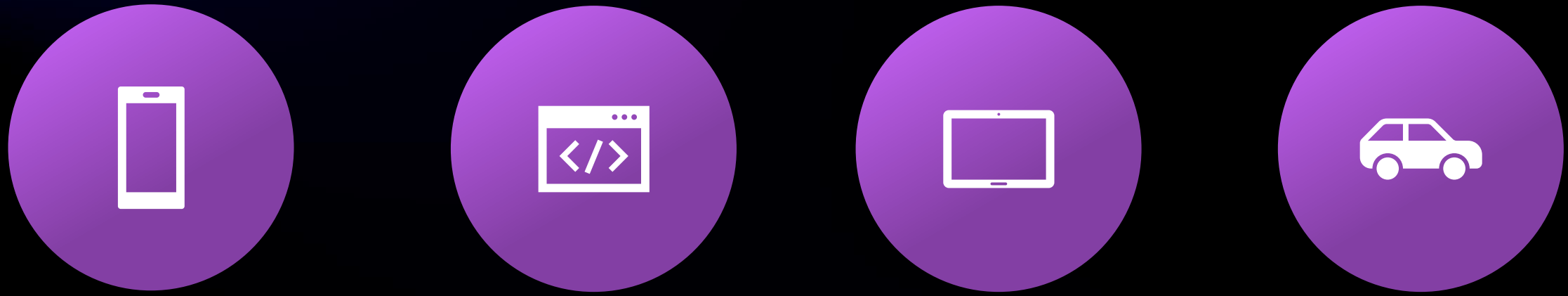


DELIVERY

***Plan is to grow to 50% of all orders through digital***

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Digital commerce channels



ECOMMERCE MOBILE APPS

ECOMMERCE WEB APP

KIOSK

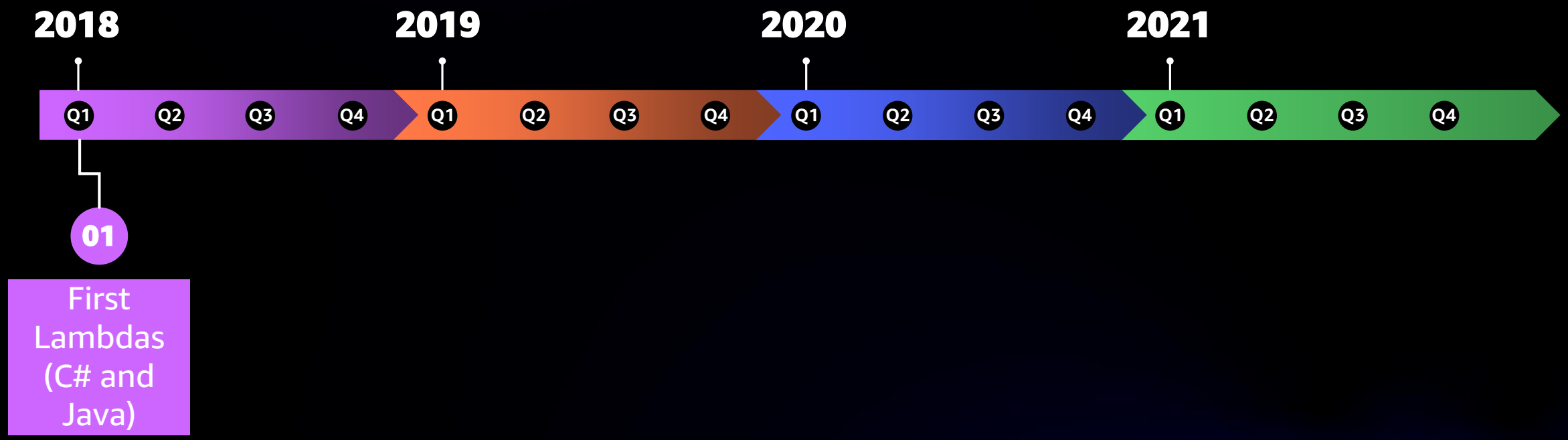
DELIVERY

***Plan is to grow to 50% of all orders through digital***

**Serverless-first**

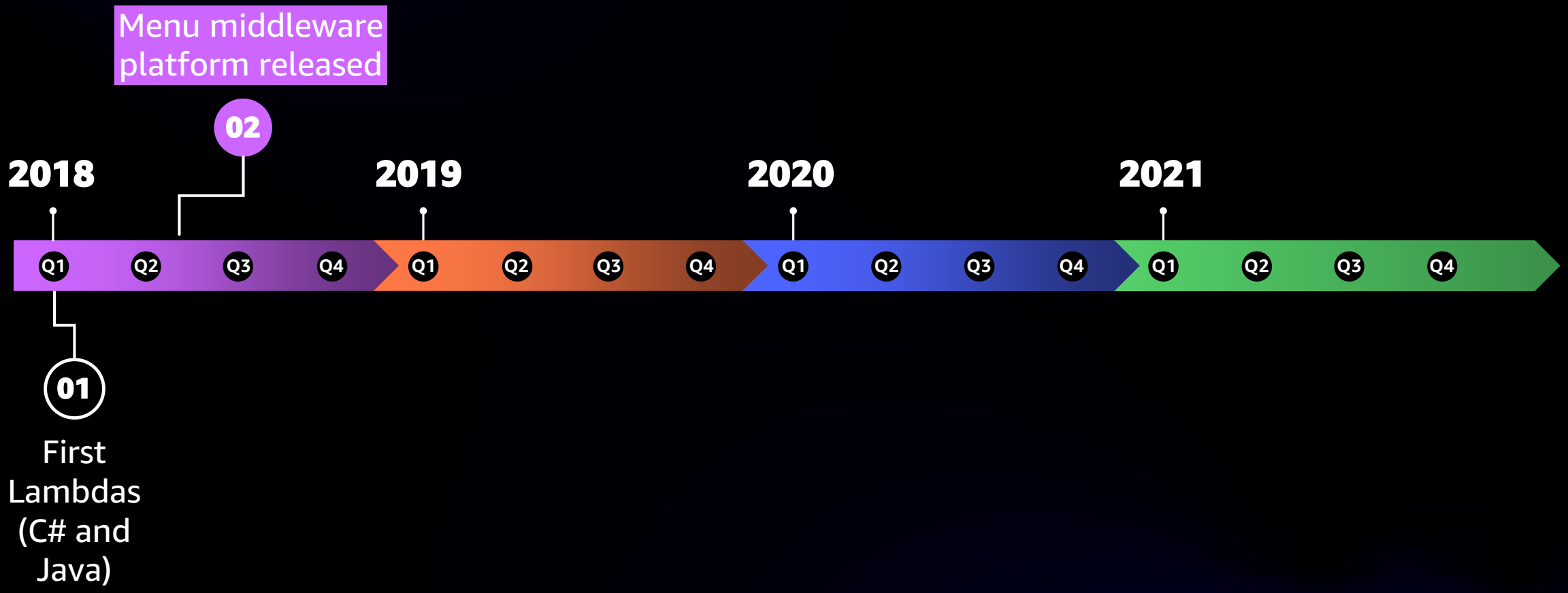
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Our serverless journey



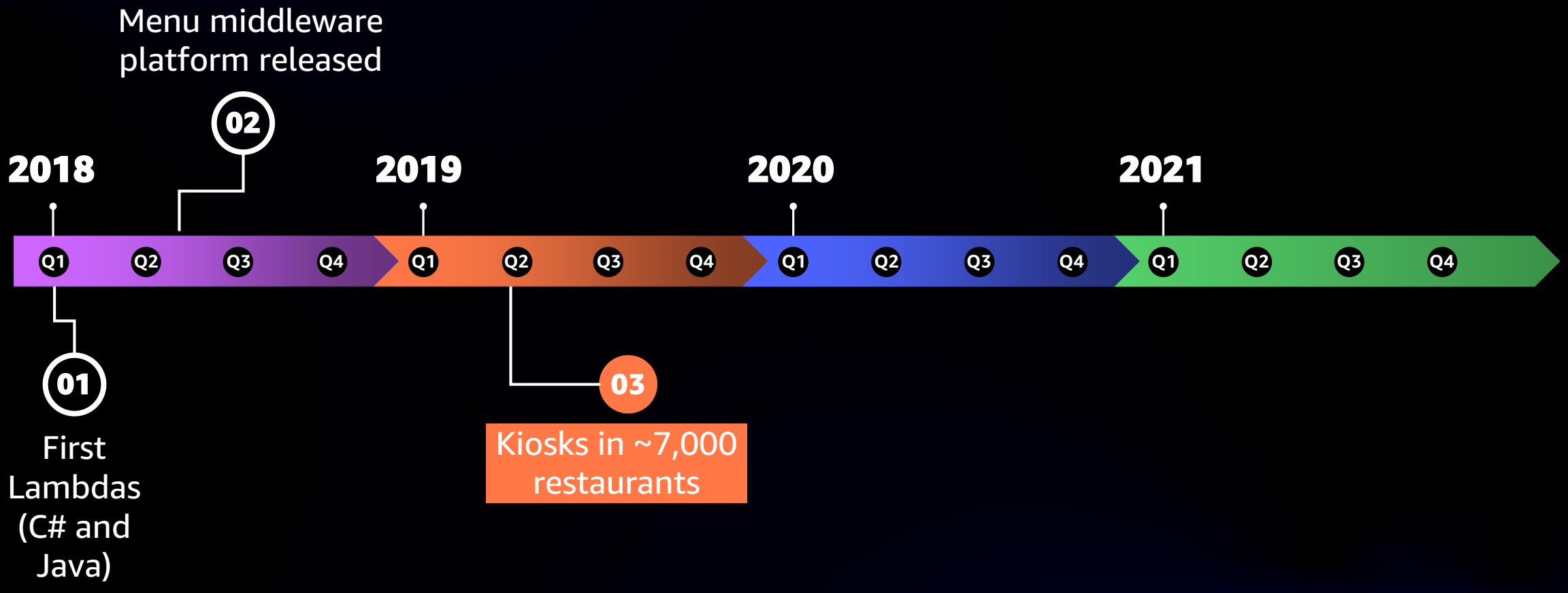

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Our serverless journey



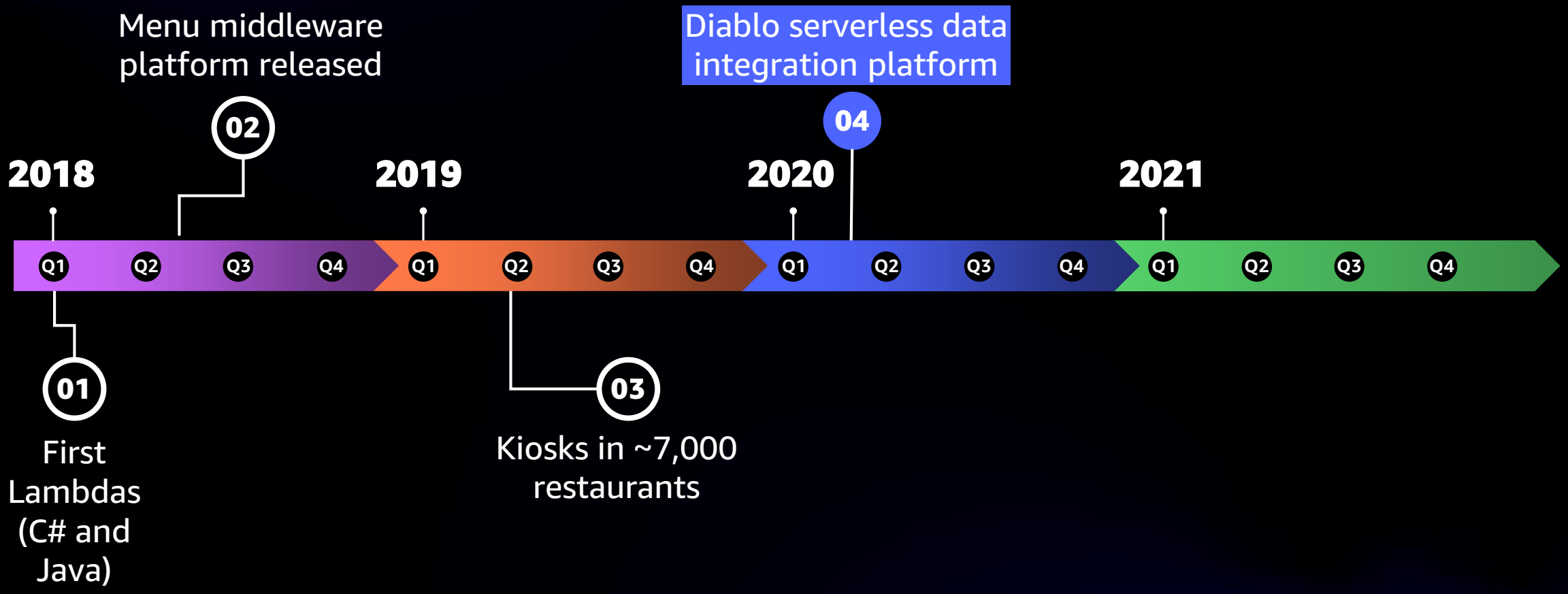

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Our serverless journey



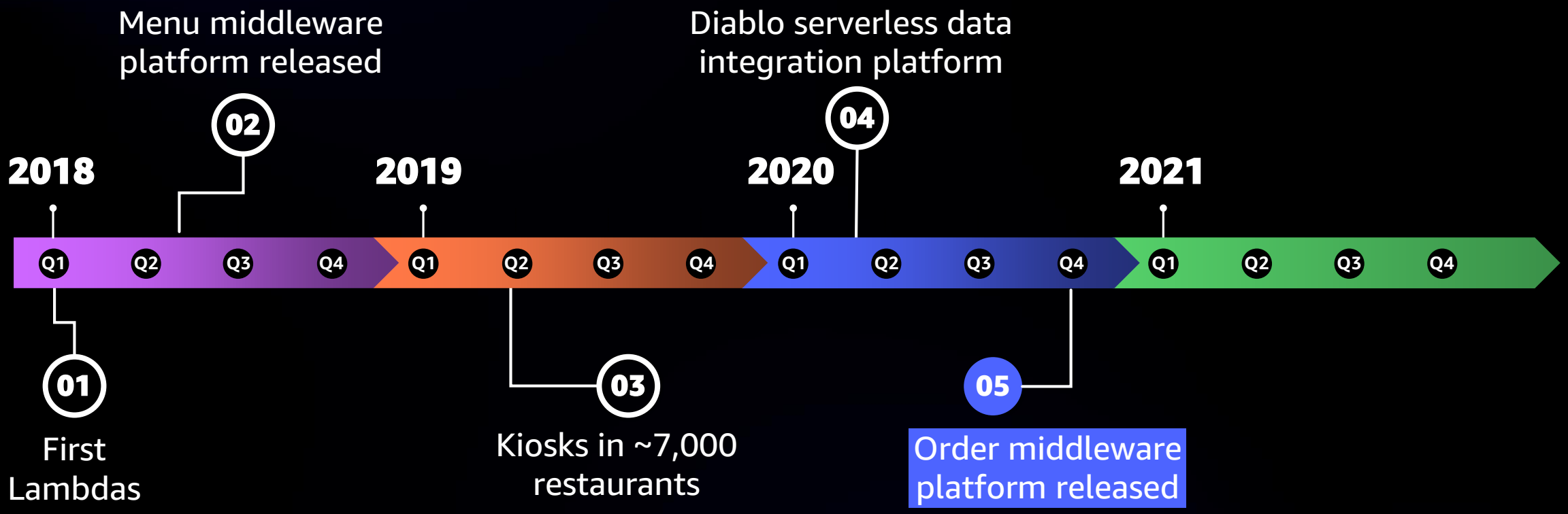

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Our serverless journey



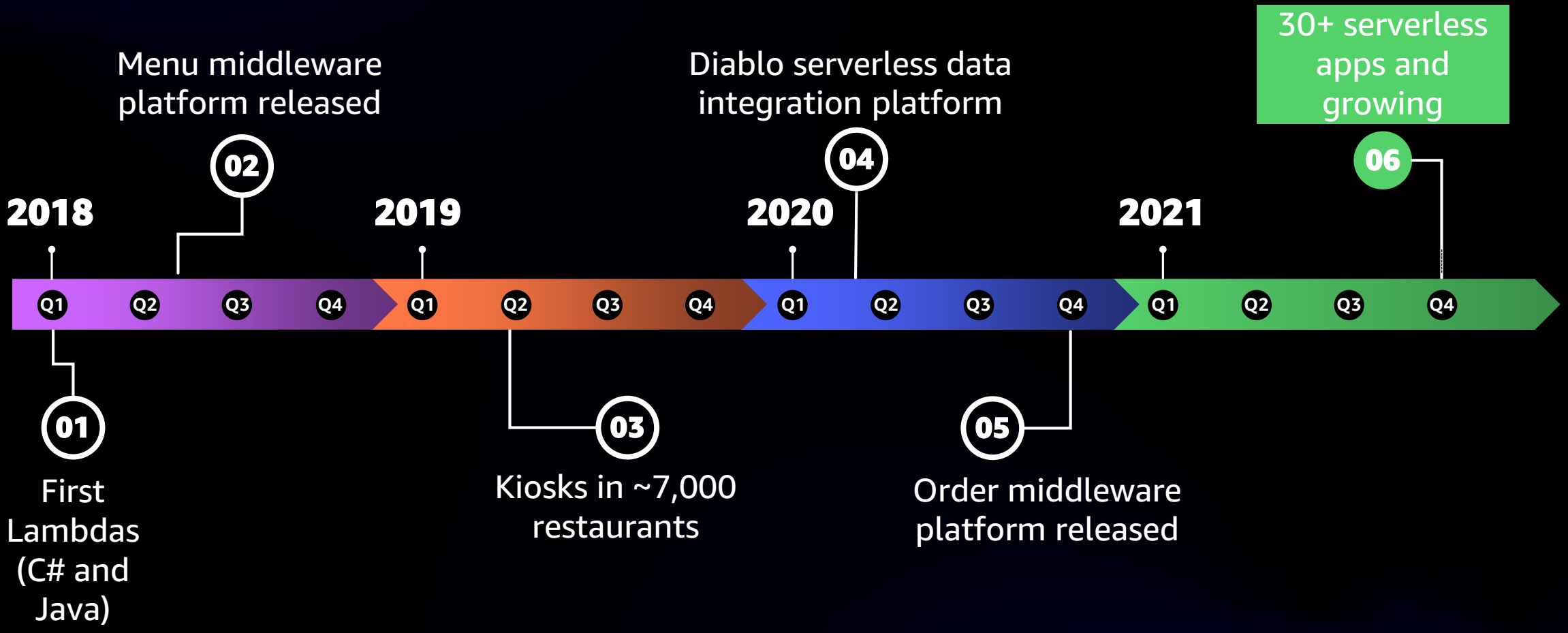

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Our serverless journey




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Our serverless journey




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Delivery integration using tablets

- Pandemic triggered massive demand for delivery

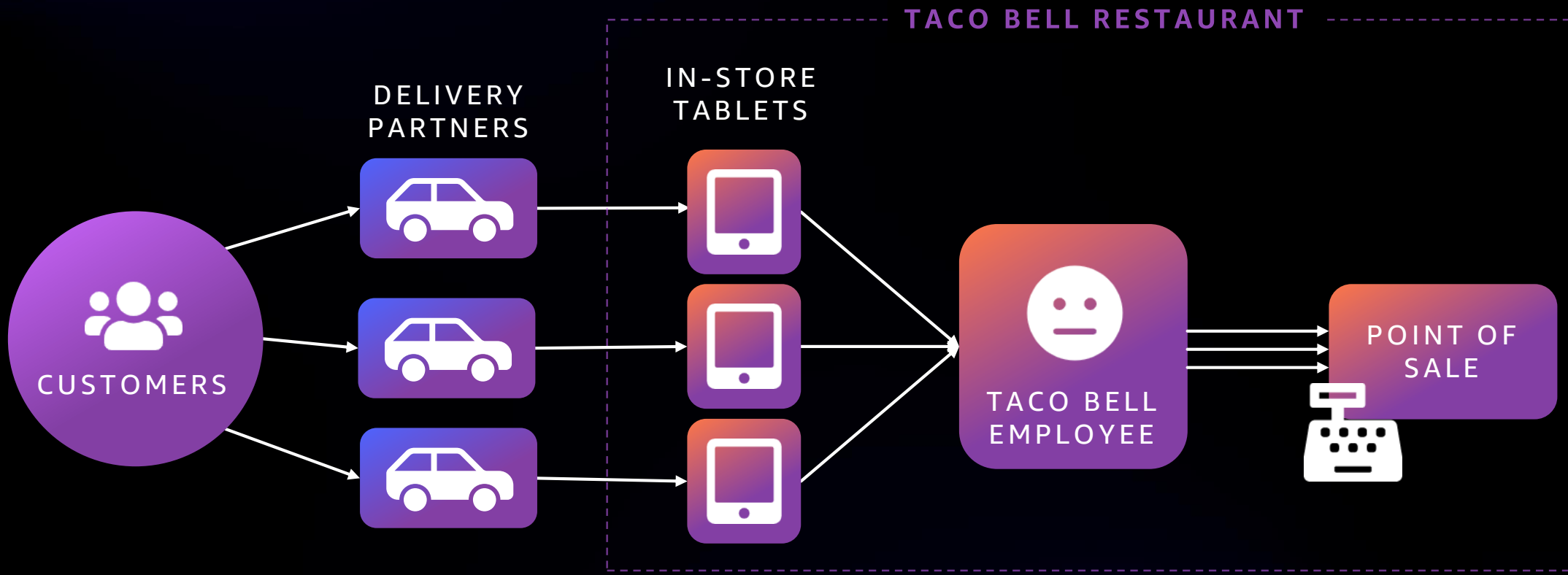
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Delivery integration using tablets

- Pandemic triggered massive demand for delivery
- To act fast, we used tablets to integrate with delivery partners

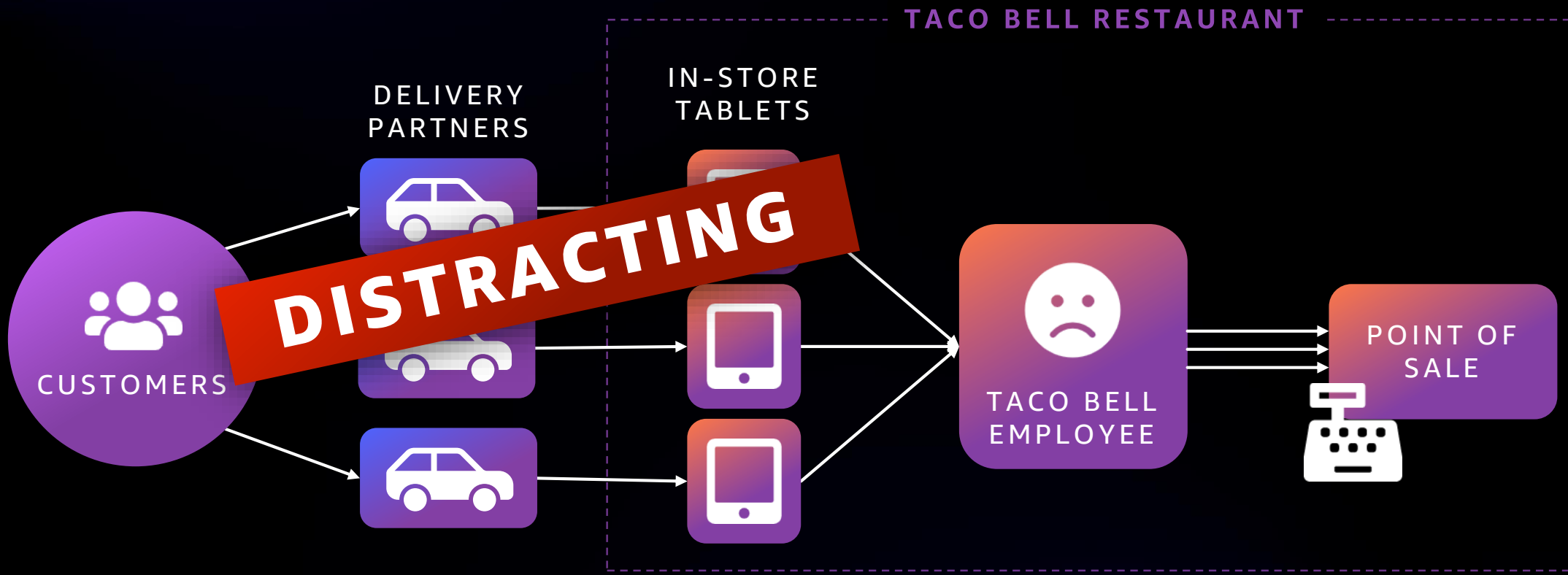
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Delivery integration using tablets



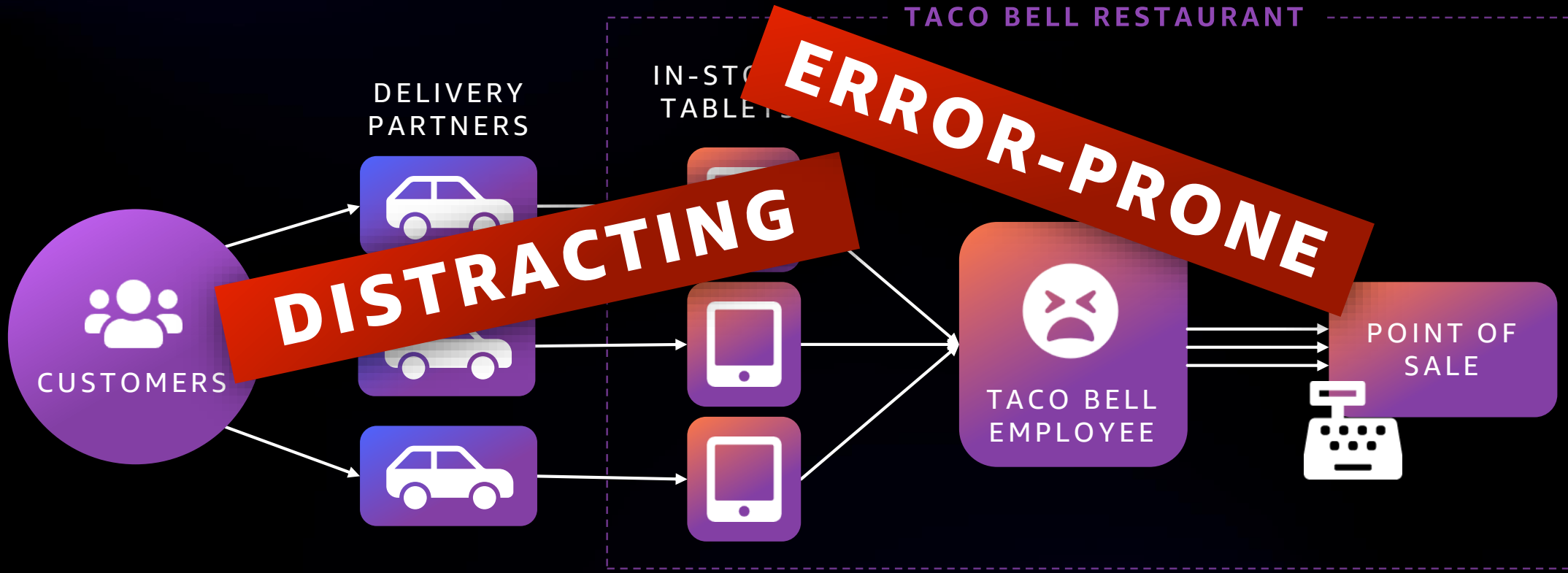

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Delivery integration using tablets



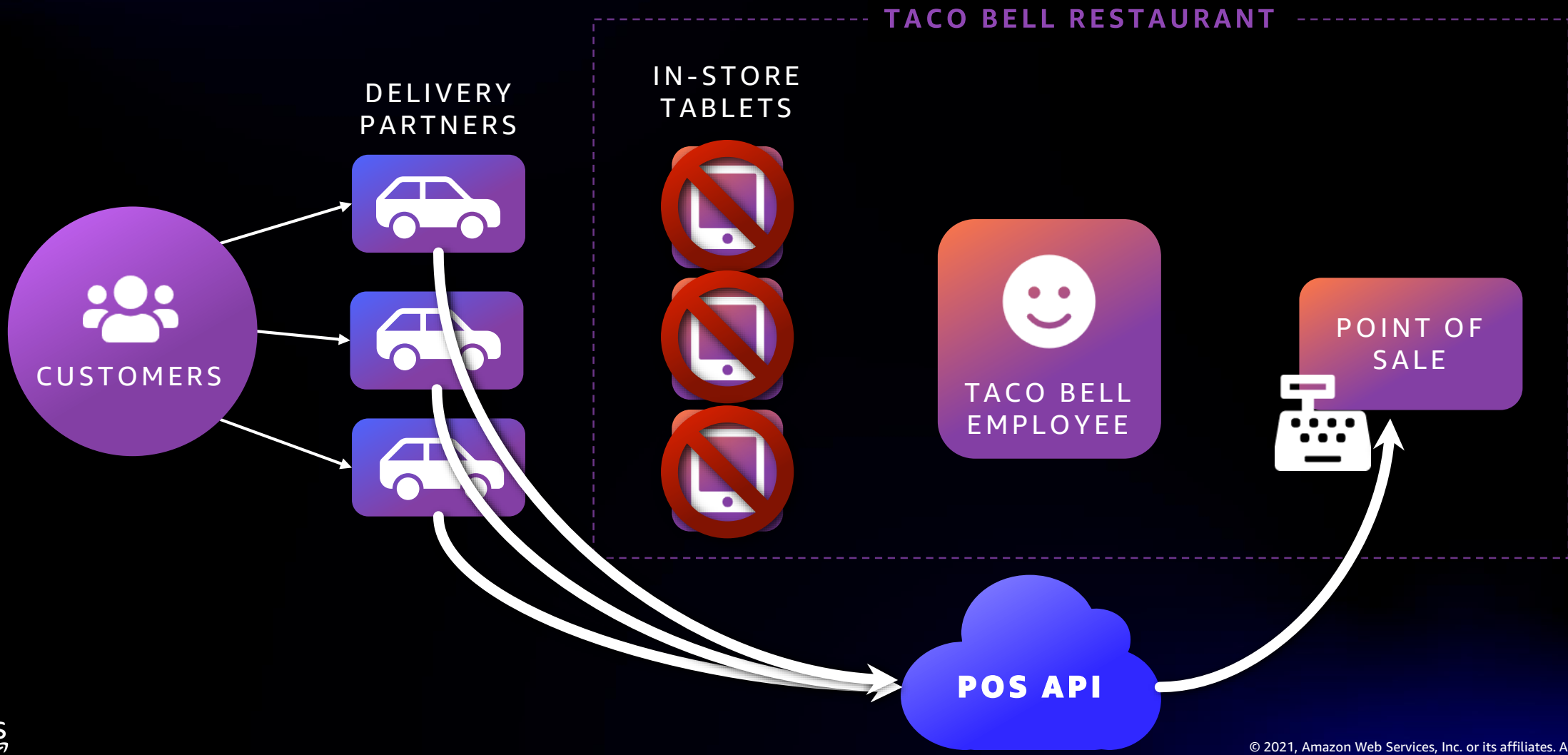

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Delivery integration using tablets




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Better solution: Integrate with point of sale

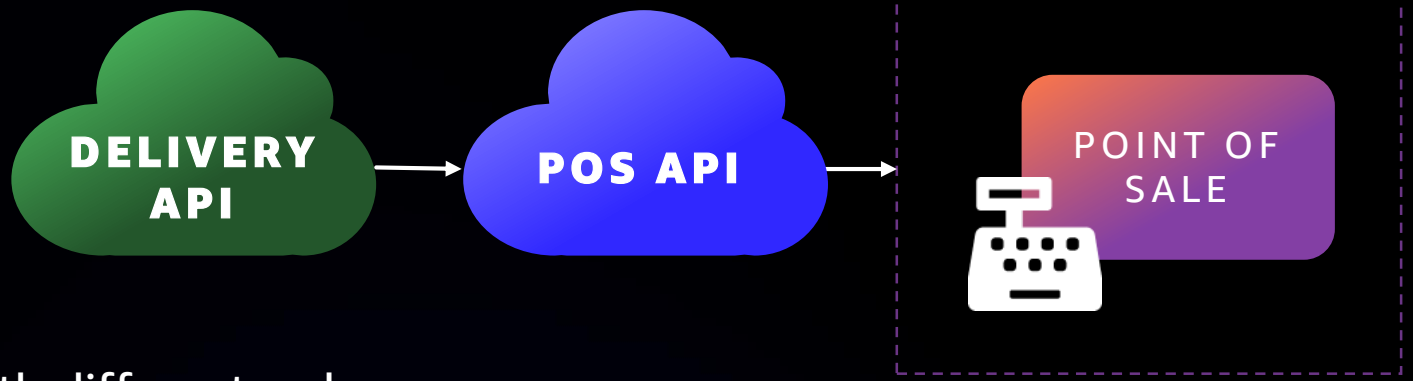



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Better solution: Integrate with point of sale

## Key challenges

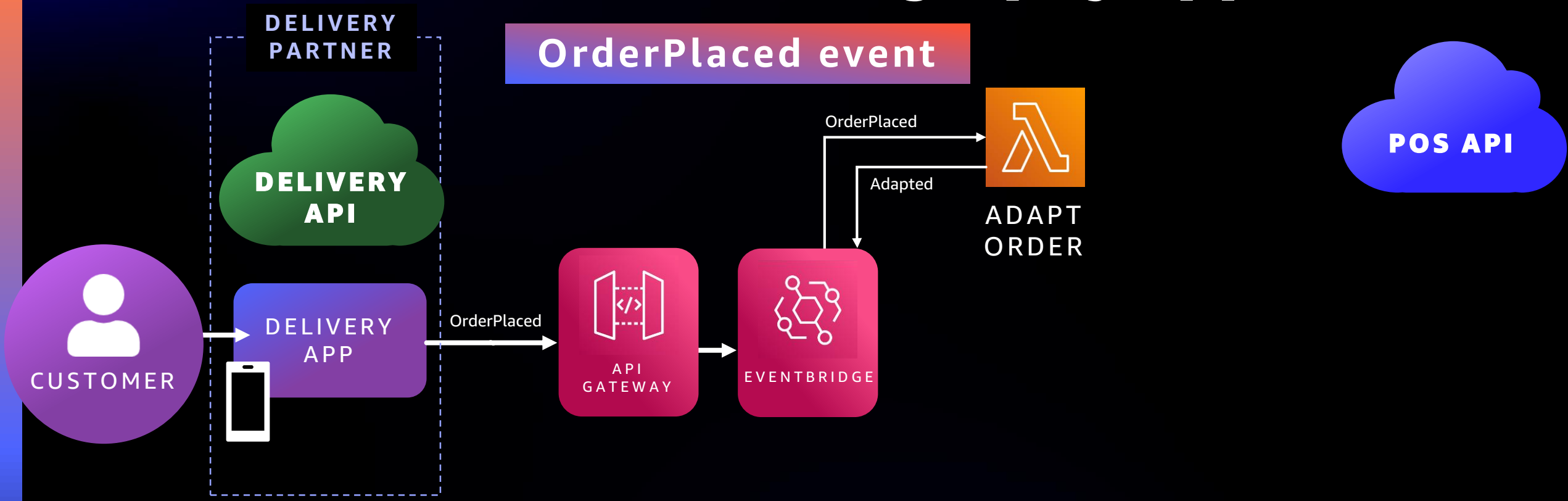
- Must scale to handle large bursts of orders (TV ad)
- Must handle retries – "*Everything fails, all the time*"
- Handle stores being offline
- Validate products being available
- Ability to cancel orders
- Handle multiple delivery providers with different order flows, APIs, and schemas




aws

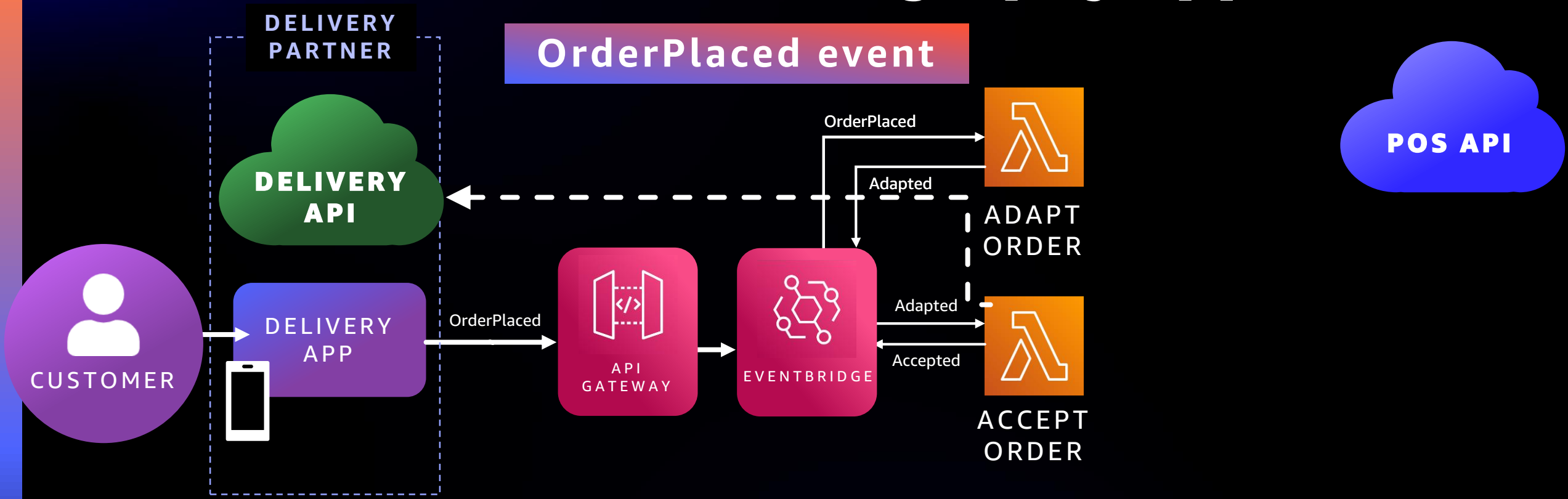
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Choreography approach



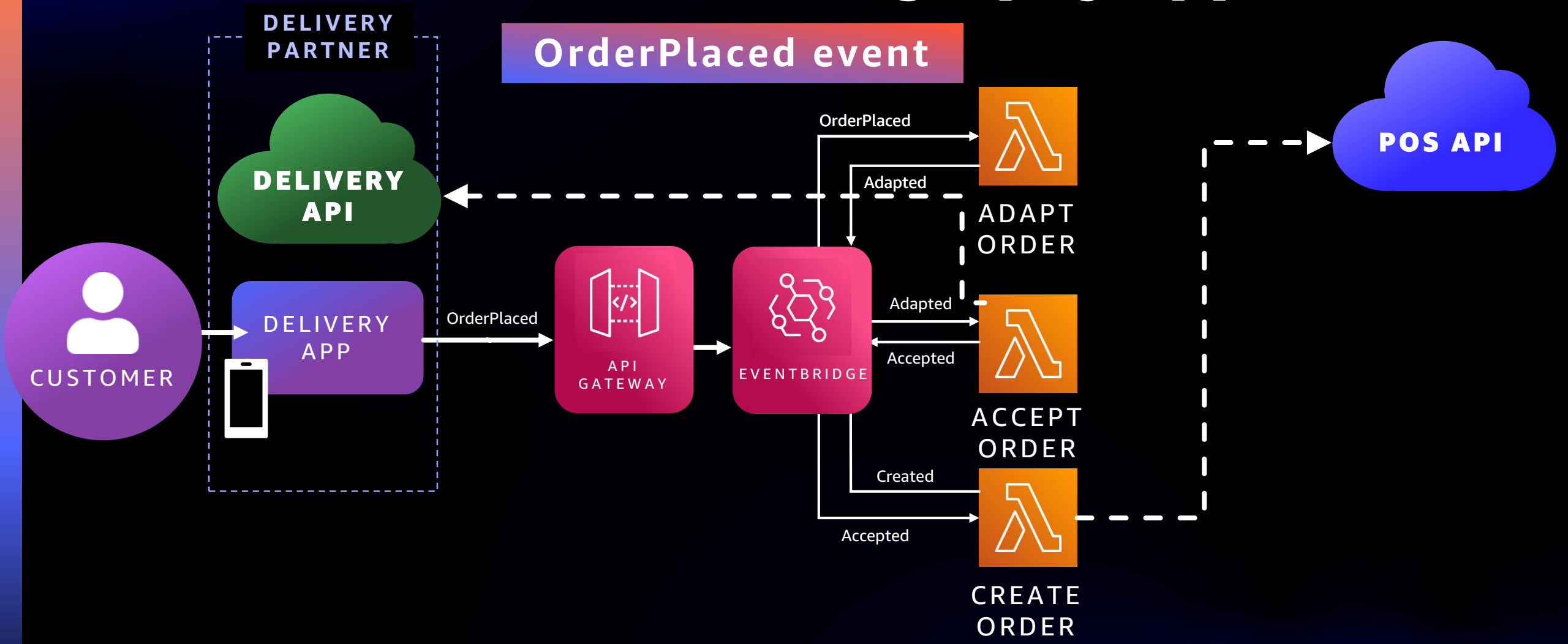

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Choreography approach



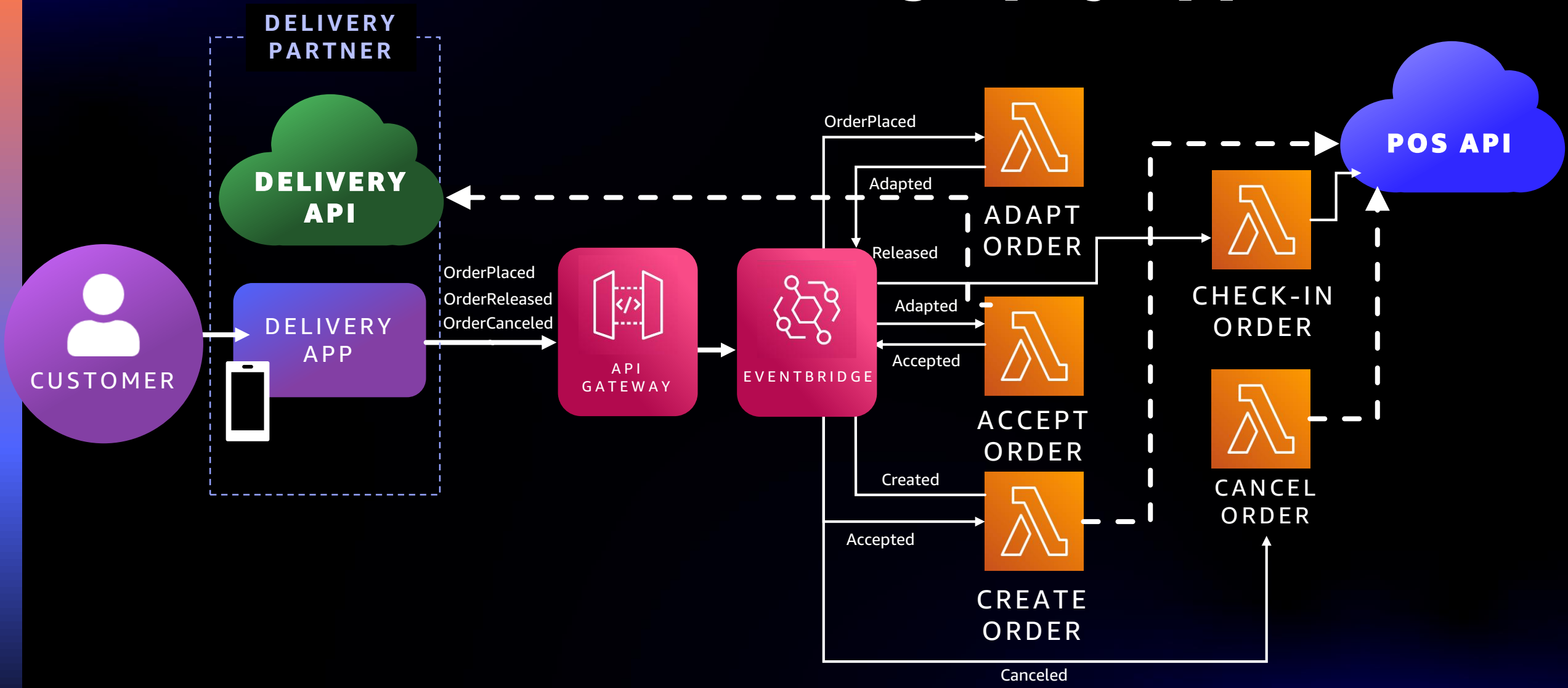

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Choreography approach



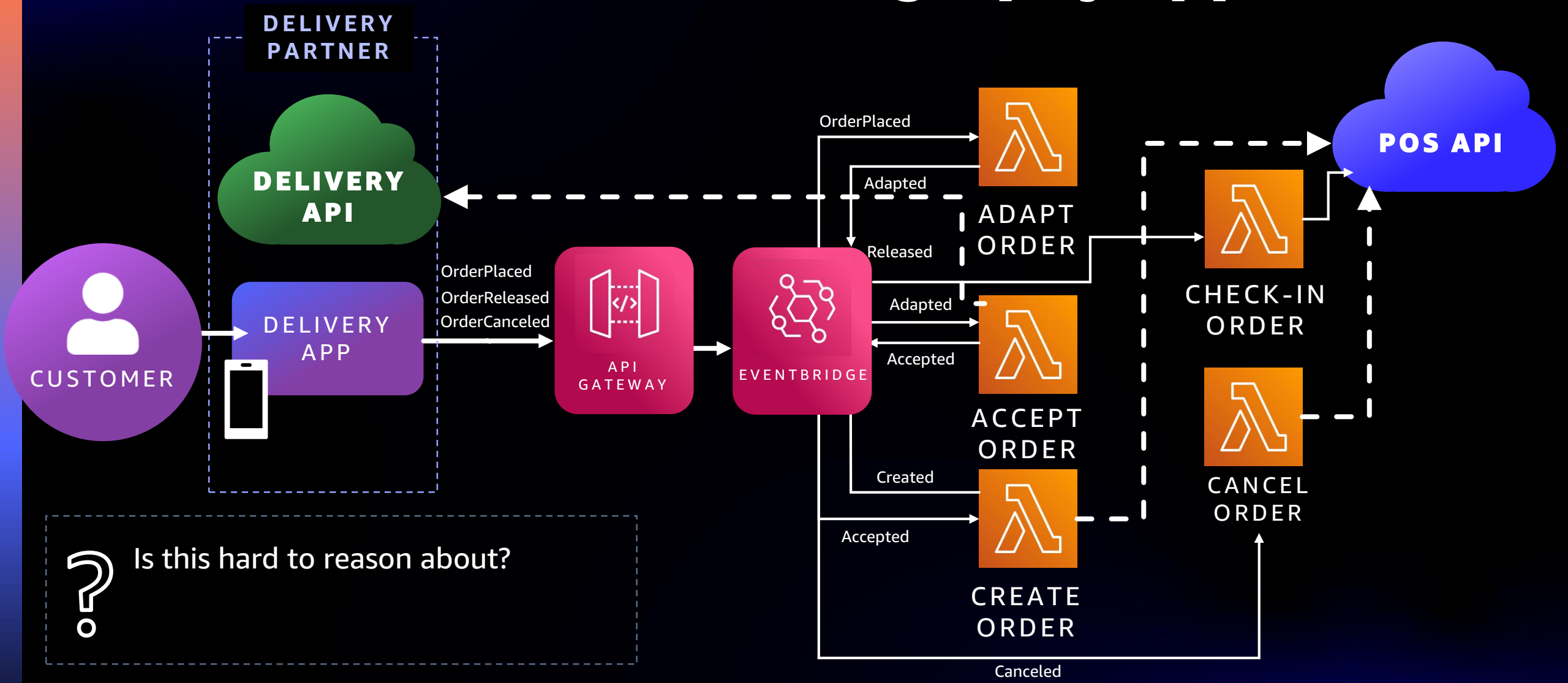

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Choreography approach




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

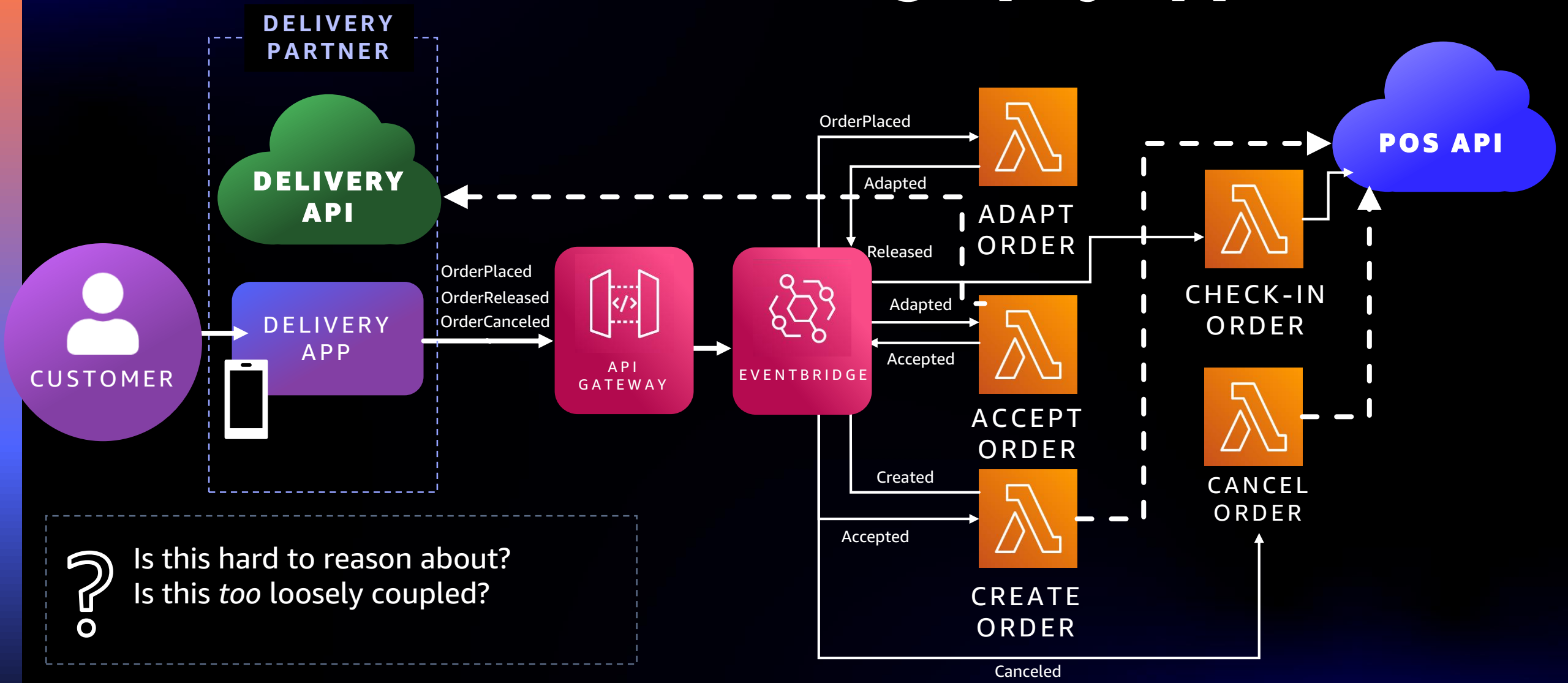
# Order middleware: Choreography approach




Is this hard to reason about?

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

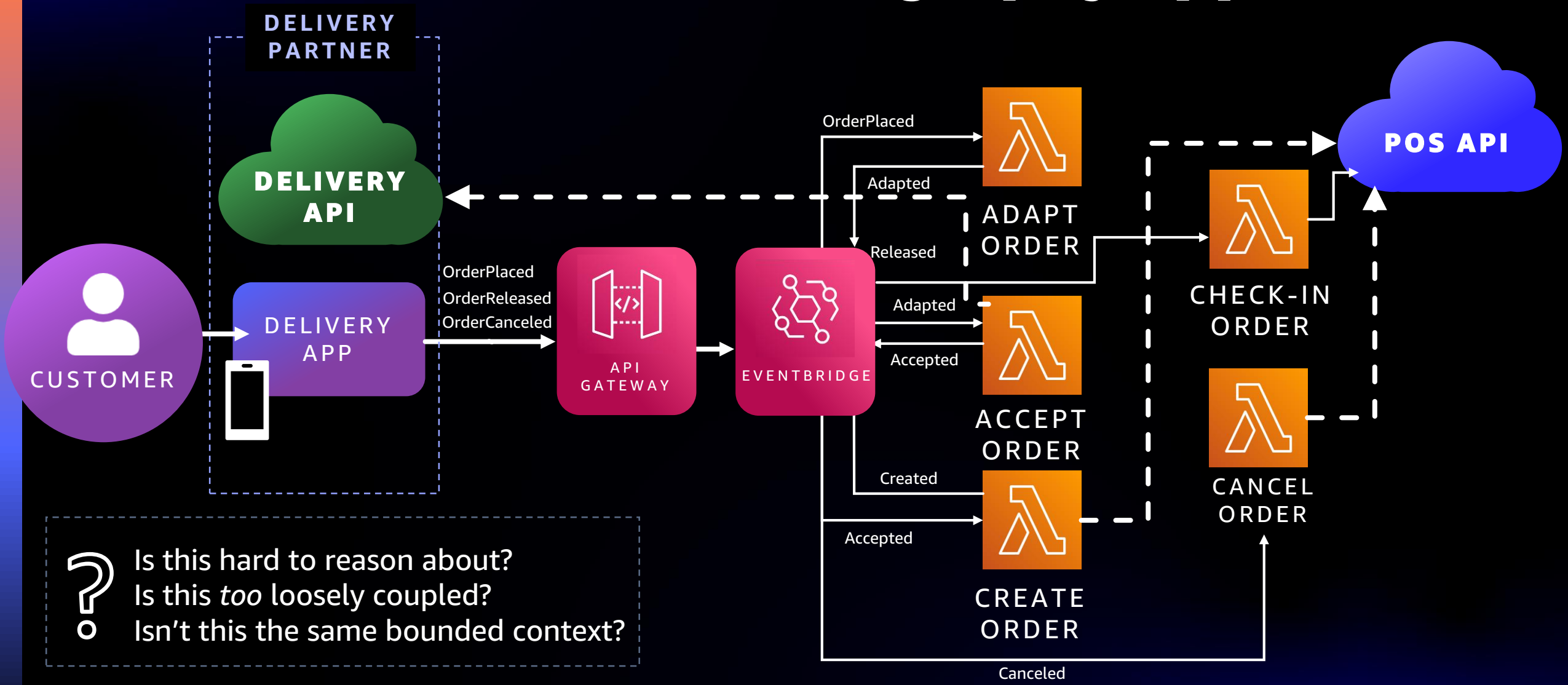
# Order middleware: Choreography approach




Is this hard to reason about?
Is this *too* loosely coupled?

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Choreography approach




Is this hard to reason about?
Is this *too* loosely coupled?
Isn't this the same bounded context?

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# There goes my serverless hero

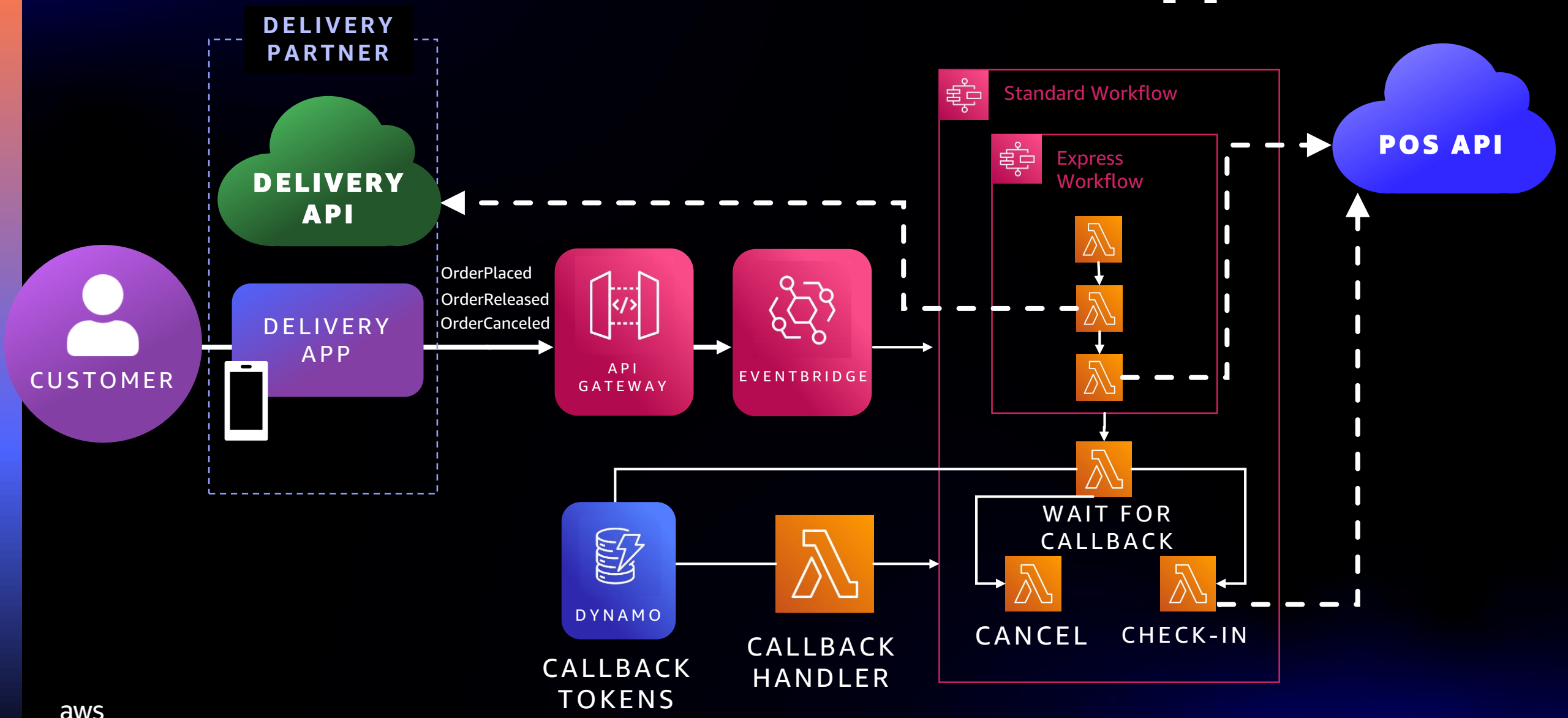
Choreography vs Orchestration in the land of serverless

AWS, EventBridge, Lambda, Serverless, SNS, Step Functions / August 7, 2020

https://bit.ly/3nUmfbL

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

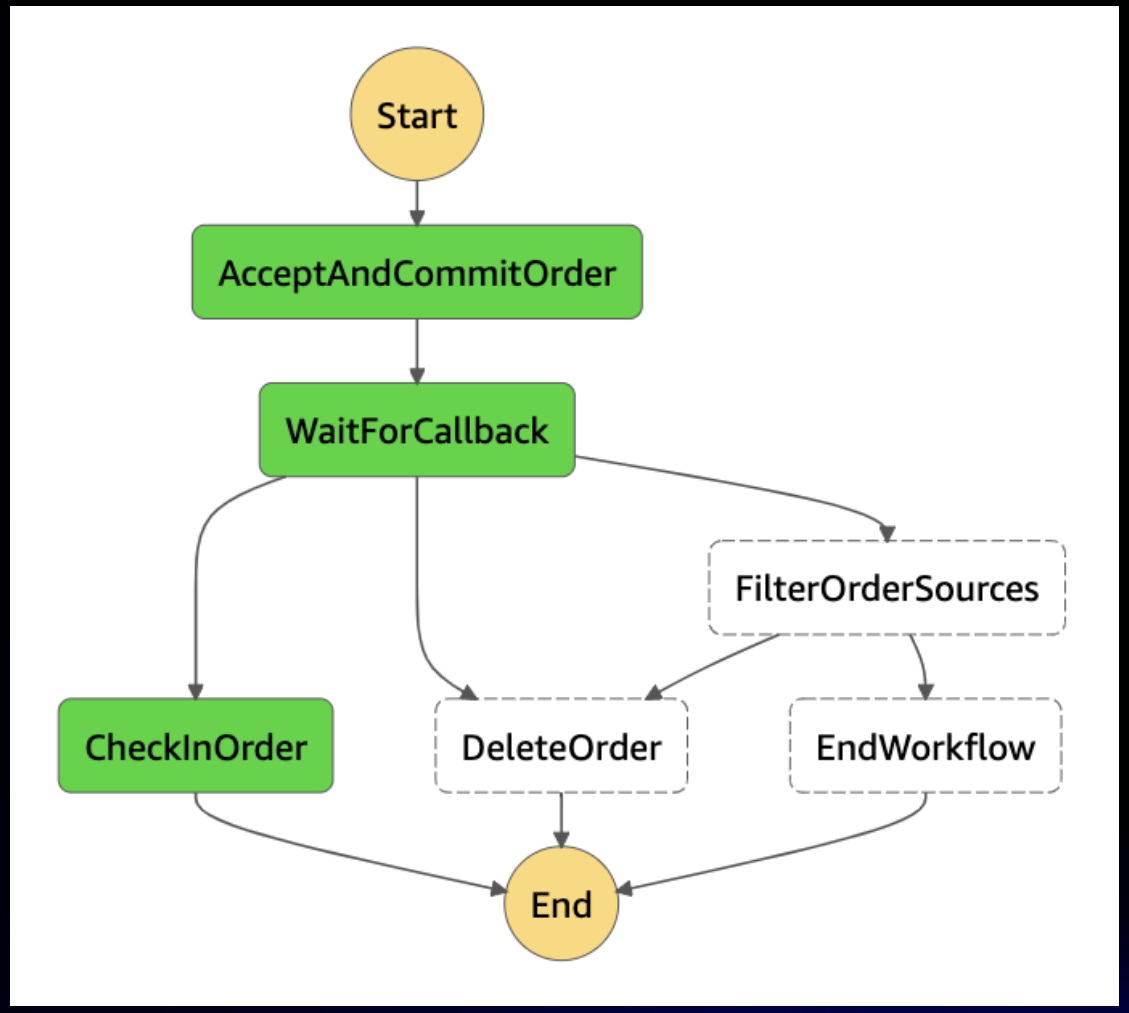
# Order middleware: Orchestration approach




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Orchestration design

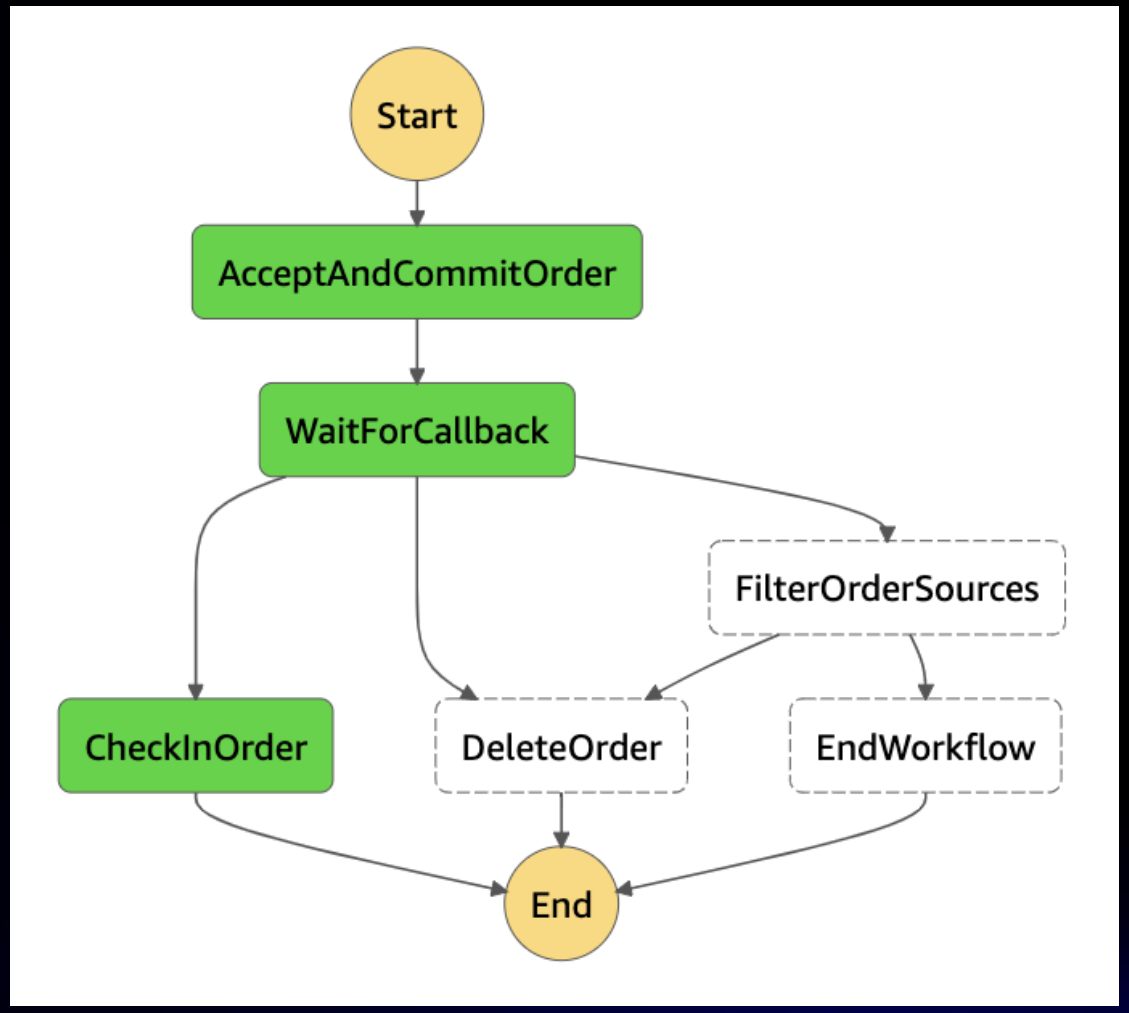
- Step Functions for orchestrating order workflow




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Orchestration design

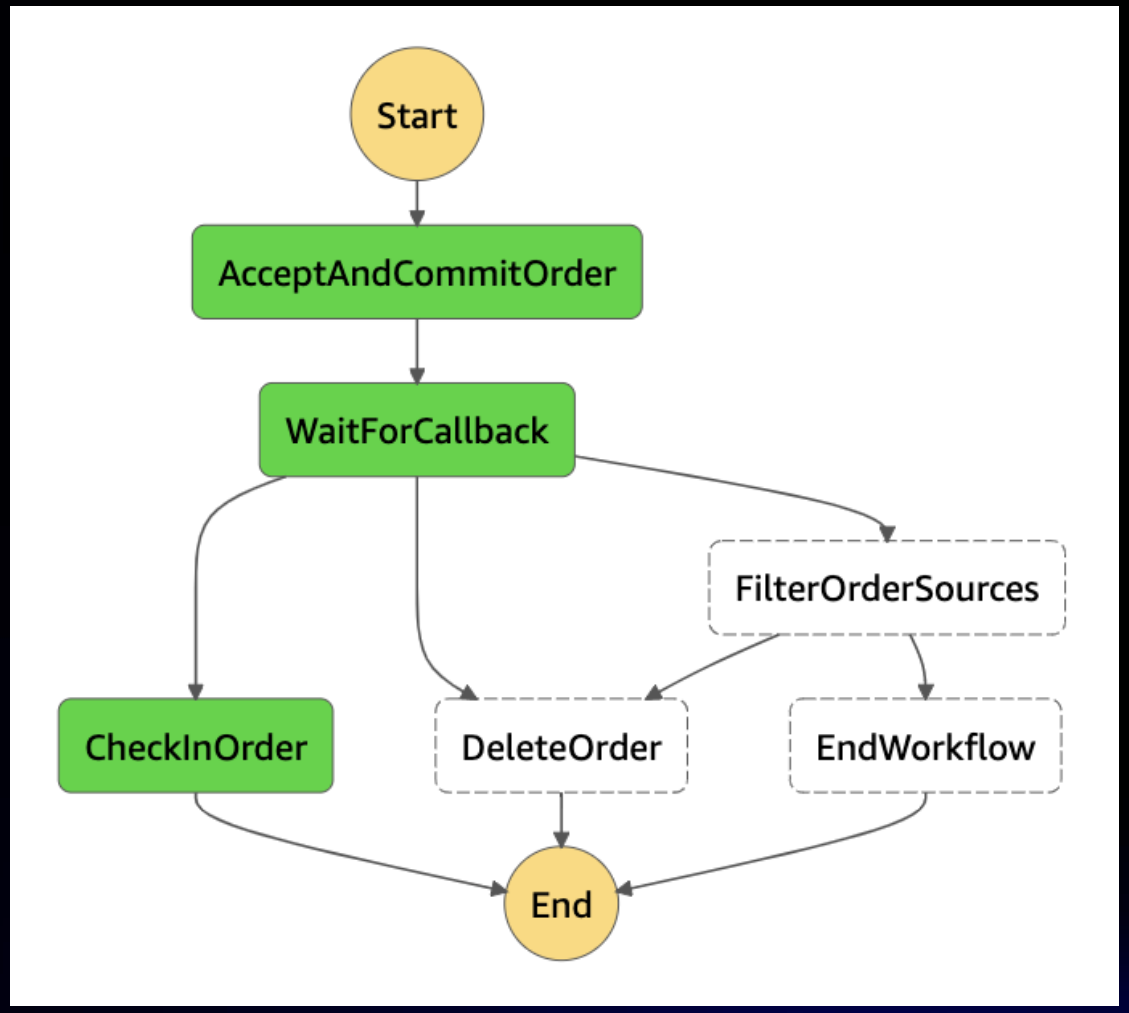
- Step Functions for orchestrating order workflow
- EventBridge used for handling inbound HTTP events (new order/release order)




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Orchestration design

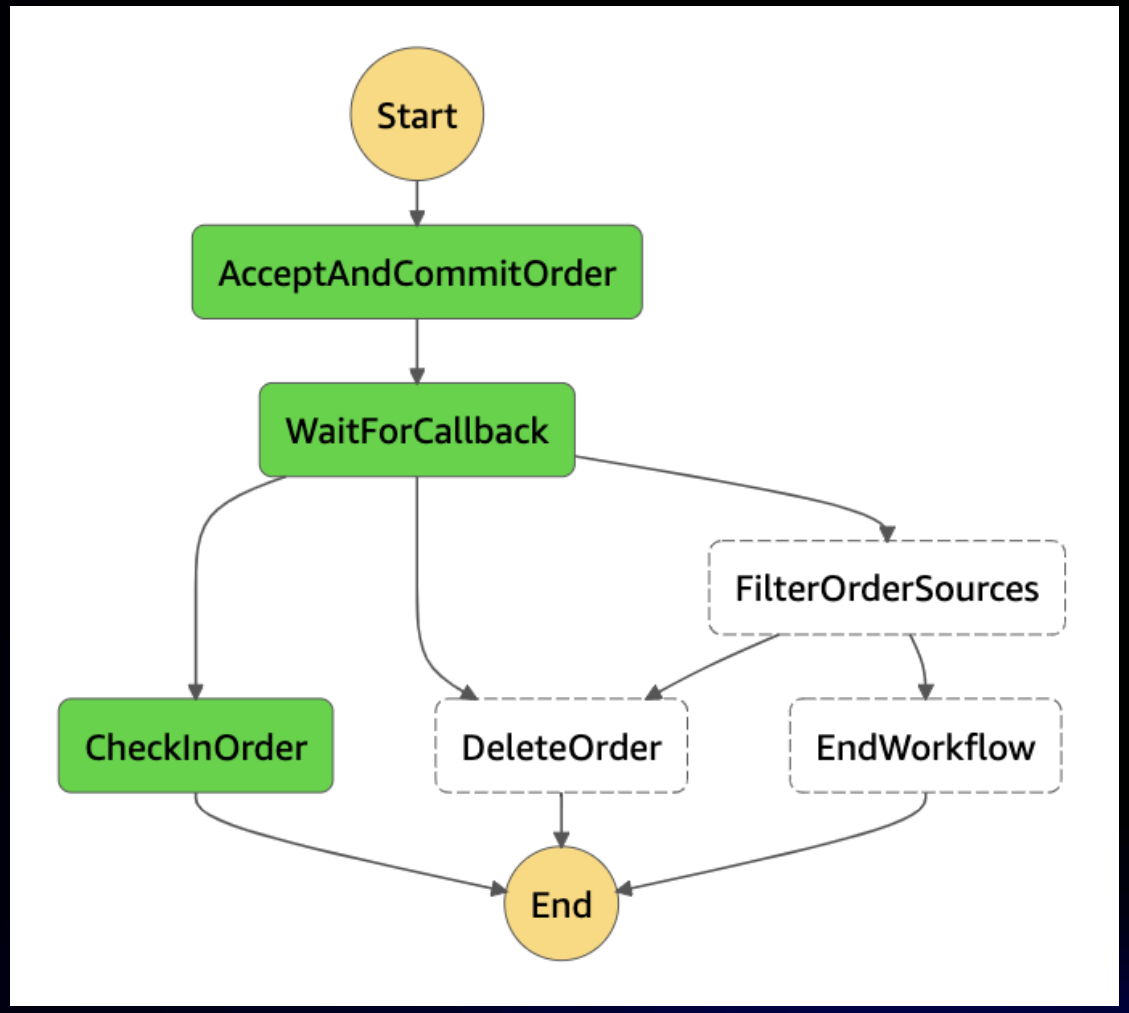
- Step Functions for orchestrating order workflow
- EventBridge used for handling inbound HTTP events (new order/release order)
- Easy to adapt from previous design




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Orchestration design

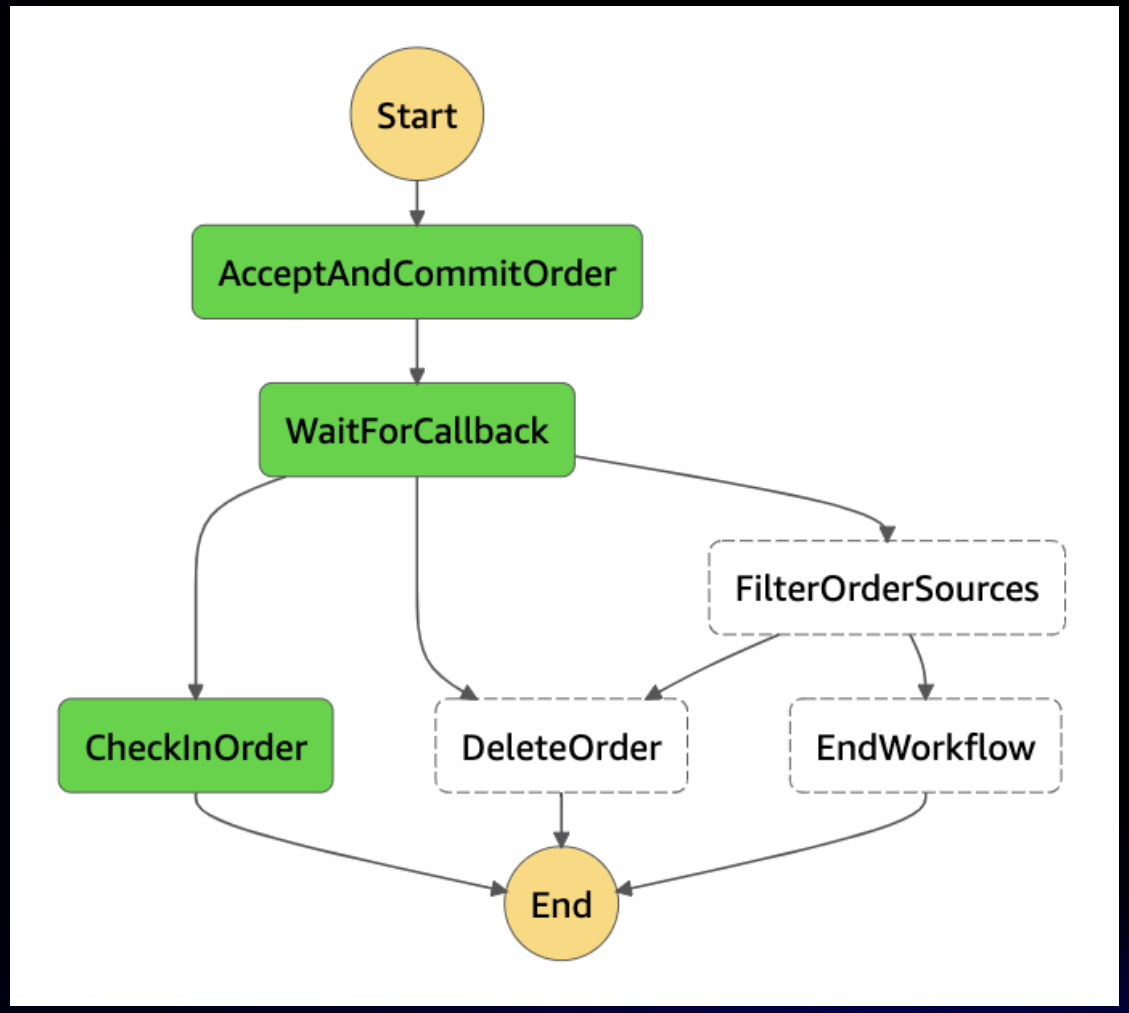
- Step Functions for orchestrating order workflow
- EventBridge used for handling inbound HTTP events (new order/release order)
- Easy to adapt from previous design
- Used Express Step Function subworkflow to save costs




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Orchestration design

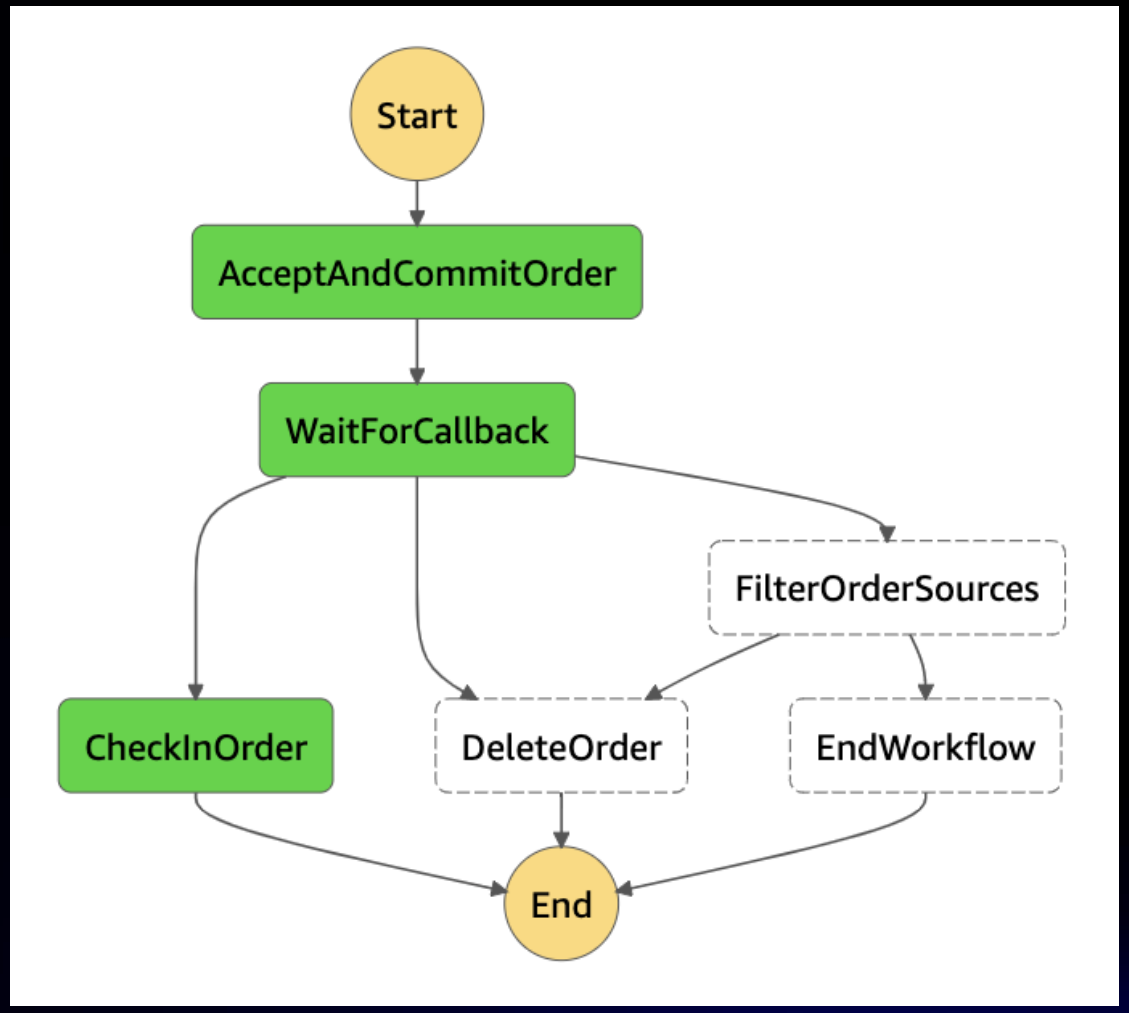
- Step Functions for orchestrating order workflow
- EventBridge used for handling inbound HTTP events (new order/release order)
- Easy to adapt from previous design
- Used Express Step Function subworkflow to save costs
- Retries easily handled




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Orchestration design

- Step Functions for orchestrating order workflow
- EventBridge used for handling inbound HTTP events (new order/release order)
- Easy to adapt from previous design
- Used Express Step Function subworkflow to save costs
- Retries easily handled
- Much easier to understand and troubleshoot!




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Callback Task Token pattern

1. Order workflow pauses to wait for delivery




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Callback Task Token pattern

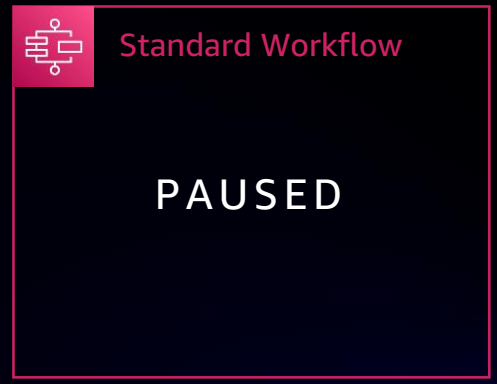
1. Order workflow pauses to wait for delivery
2. Delivery partner sends release event when driver nearby restaurant




CALLBACK TOKENS



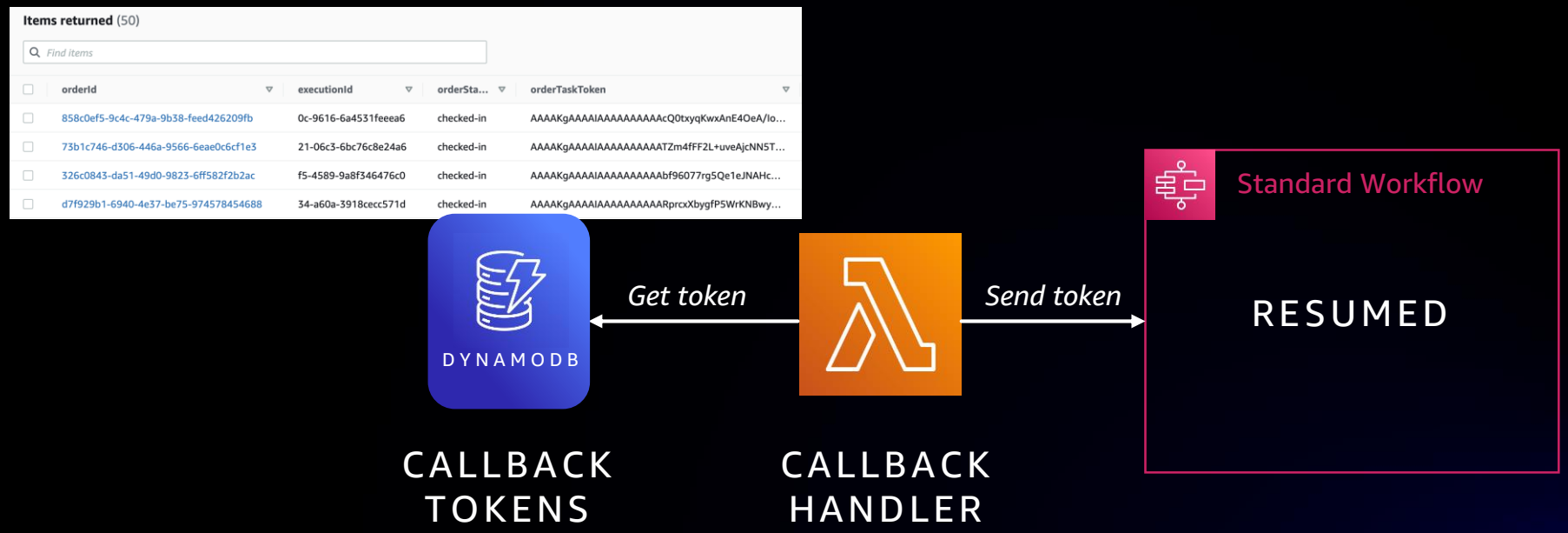
CALLBACK HANDLER




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Callback Task Token pattern

1. Order workflow pauses to wait for delivery
2. Delivery partner sends release event when driver nearby restaurant
3. Callback token handler sends task token to workflow to resume




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Dev process and CI/CD

## SERVERLESS FRAMEWORK

Plugins
- serverless-iam-roles-per-function
- serverless-step-functions

Lambdas and Step Function workflows in the same project

## MIDDY

Simple “middlewares” for Node.js

Fetching and caching parameters from SSM

Used for API keys and secrets

## GITLAB CI/CD

Each feature branch creates a new stack

Reduces collisions with other developers and main branches

No cost for additional stacks!

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Testing

Unit tests on core business logic

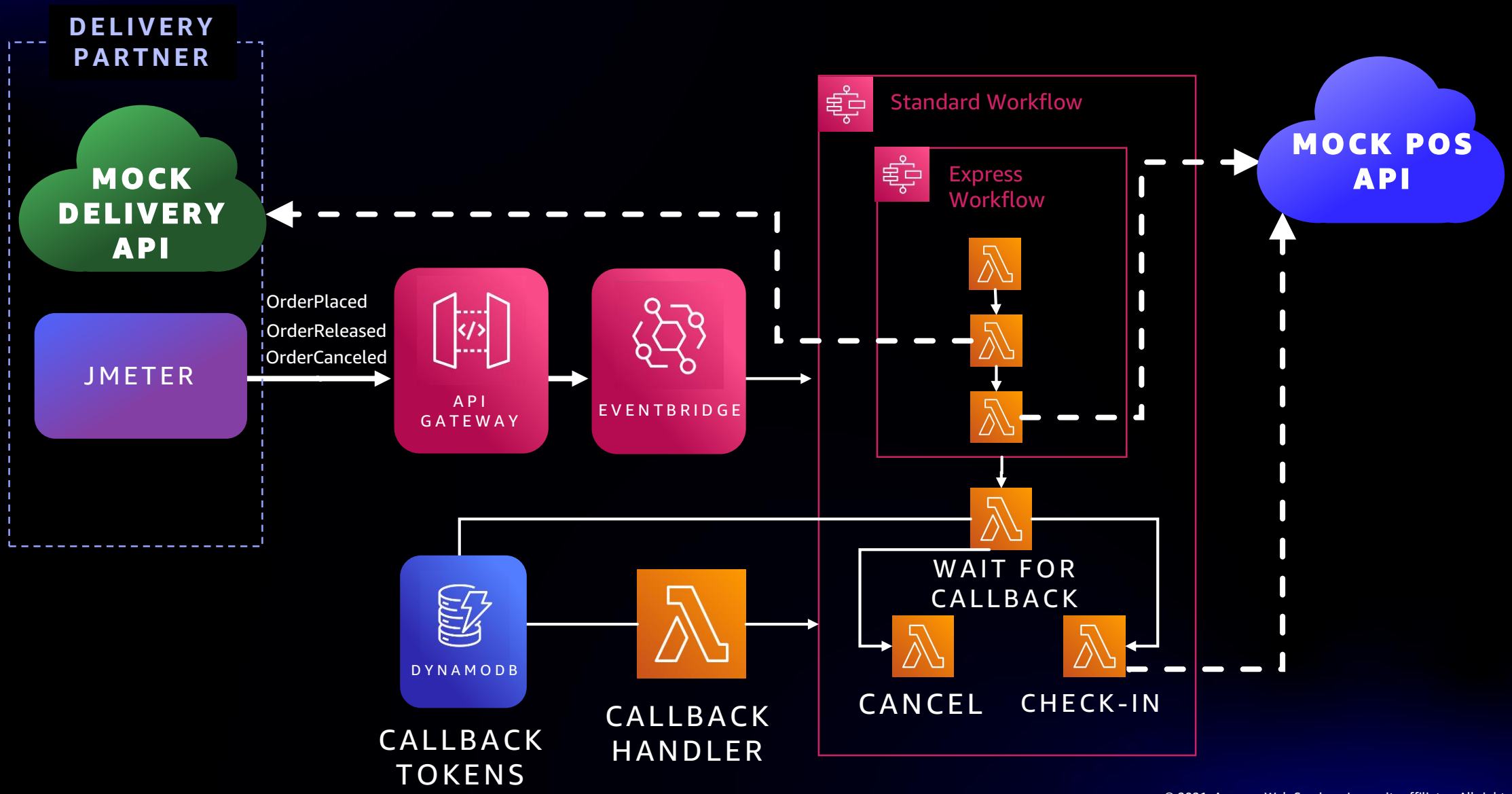
Integration tests against live AWS resources

Mocked out external API calls to facilitate load testing and E2E happy path testing

Test Step Function workflows run E2E and validate results

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Mocking external APIs




© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Monitoring/observability

## CLOUDWATCH LOGS

Started out logging a lot – tuned this down

CloudWatch Insights critical for querying logs at scale



CLOUDWATCH LOGS

## CLOUDWATCH ALARMS

Lambda failure alarms were firing too much at scale

Alarms on Step Functions were enough – retries smoothed out Lambda issues



CLOUDWATCH ALARMS

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Cost optimization

## Pay-as-you-go model with serverless

Great when scale is unknown – ability to calculate costs upfront

## Utilized Express Step Function nested workflow

Avoided monolithic Lambdas and saved money

## Removed unnecessary Step Function state transitions

Cost of transitions add up at scale

## Tuned Lambda memory configuration

Used lumigo-cli tool to right-size

aws

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Results



Functional POC within 2 weeks

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Results

- Functional POC within 2 weeks
- In production within 2 months

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Results

- Functional POC within 2 weeks
- In production within 2 months
- Scaled to millions of orders per hour in load testing

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Order middleware: Results

- Functional POC within 2 weeks
- In production within 2 months
- Scaled to millions of orders per hour in load testing
- Added 4 more integrations within 6 months

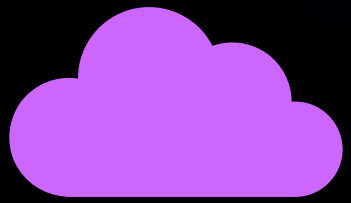
© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Taco Bell's serverless community

- Bi-weekly community of practice meeting
- Slack channel where ideas and serverless community news are shared
- Shared starter repos and documentation
- Serverless fights imposter syndrome
  - Deploy early and often
- Serverless democratizes
  - Entry level dev was lead developer on order middleware system
  - Bring everyone along for the ride!

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# What’s next?



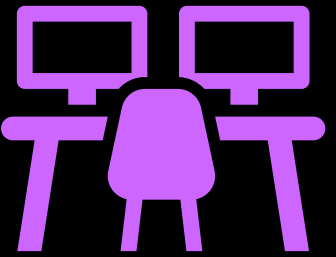
**Modernize with serverless**



**Work closer with business**



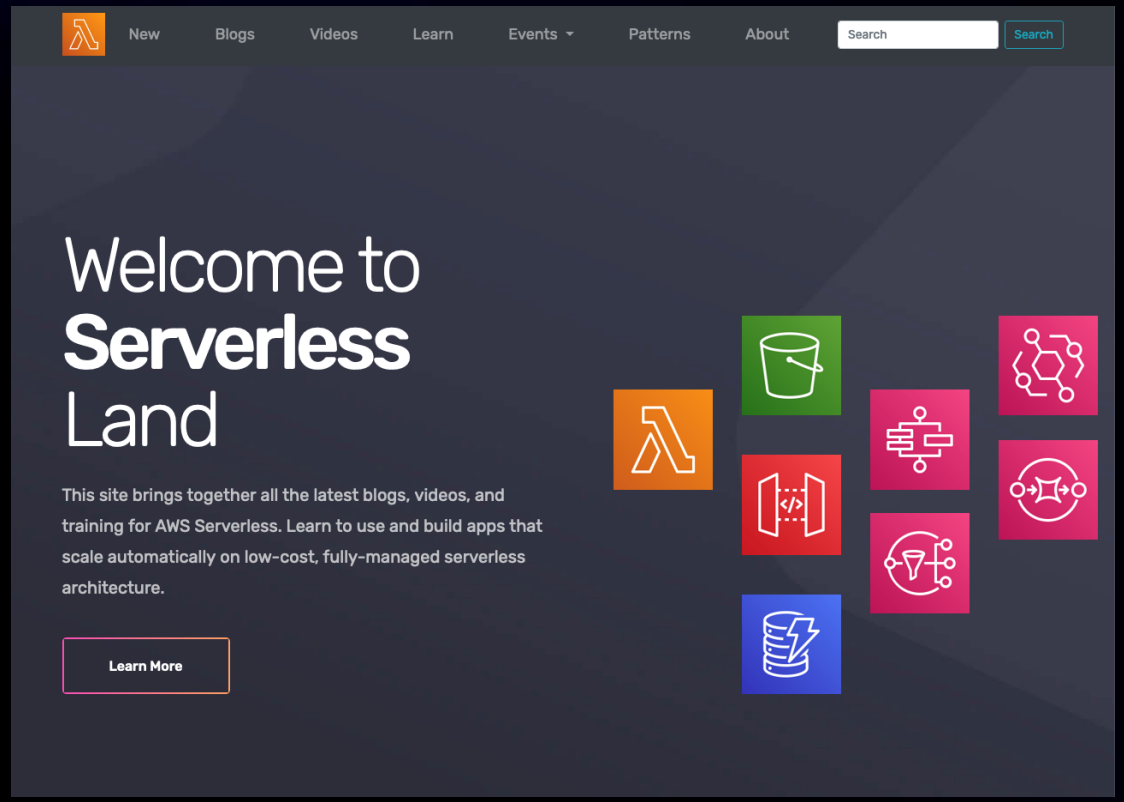
**Train and upskill team**



**Growing the team!**

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Want to learn more?




https://serverlessland.com/reinvent2021/

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Thank you!

Sam Dengler
@samdengler

Robbie Kohler
@robbie_kohler

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.