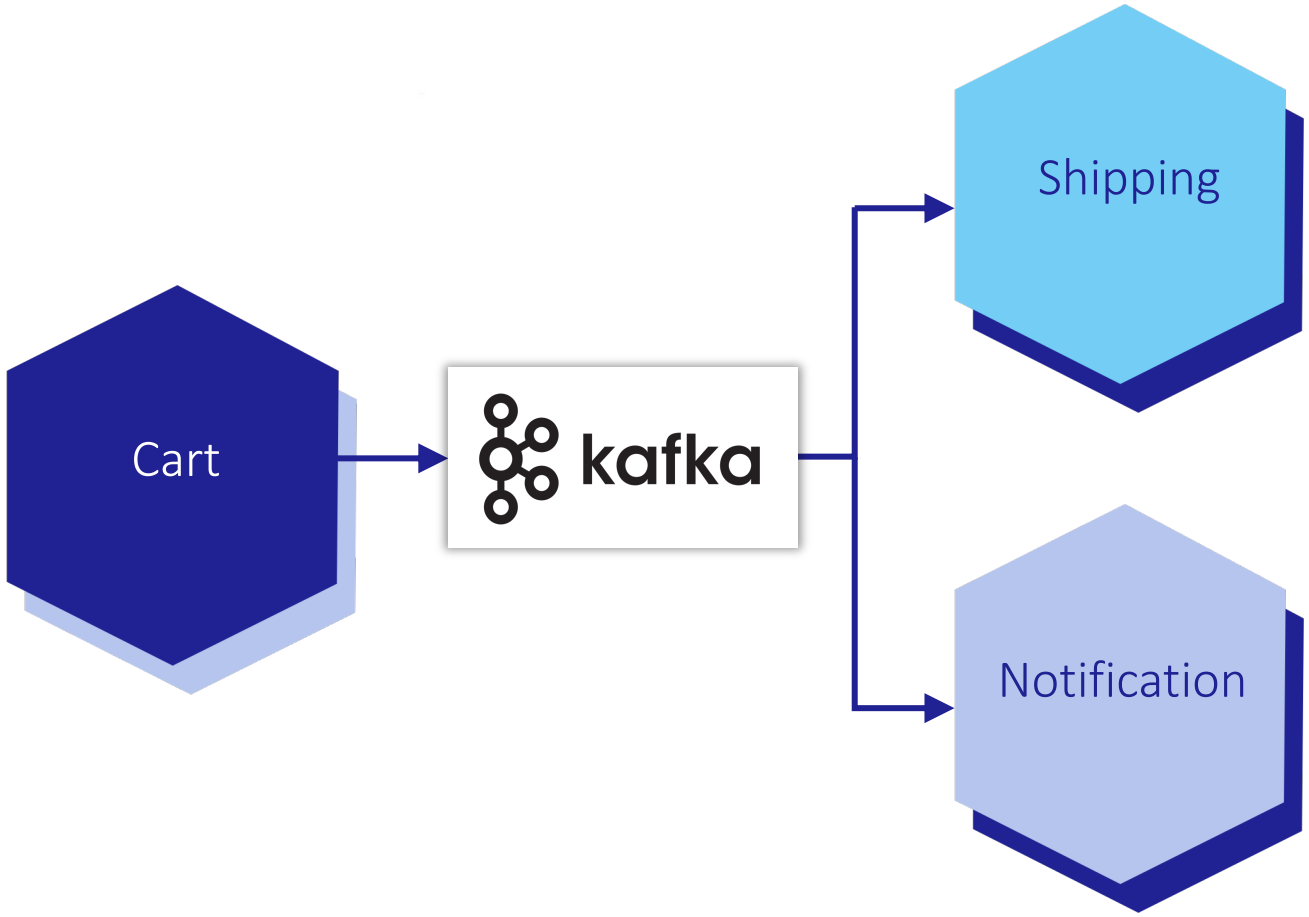

KubeCon
CloudNativeCon
Europe 2023
TiKV

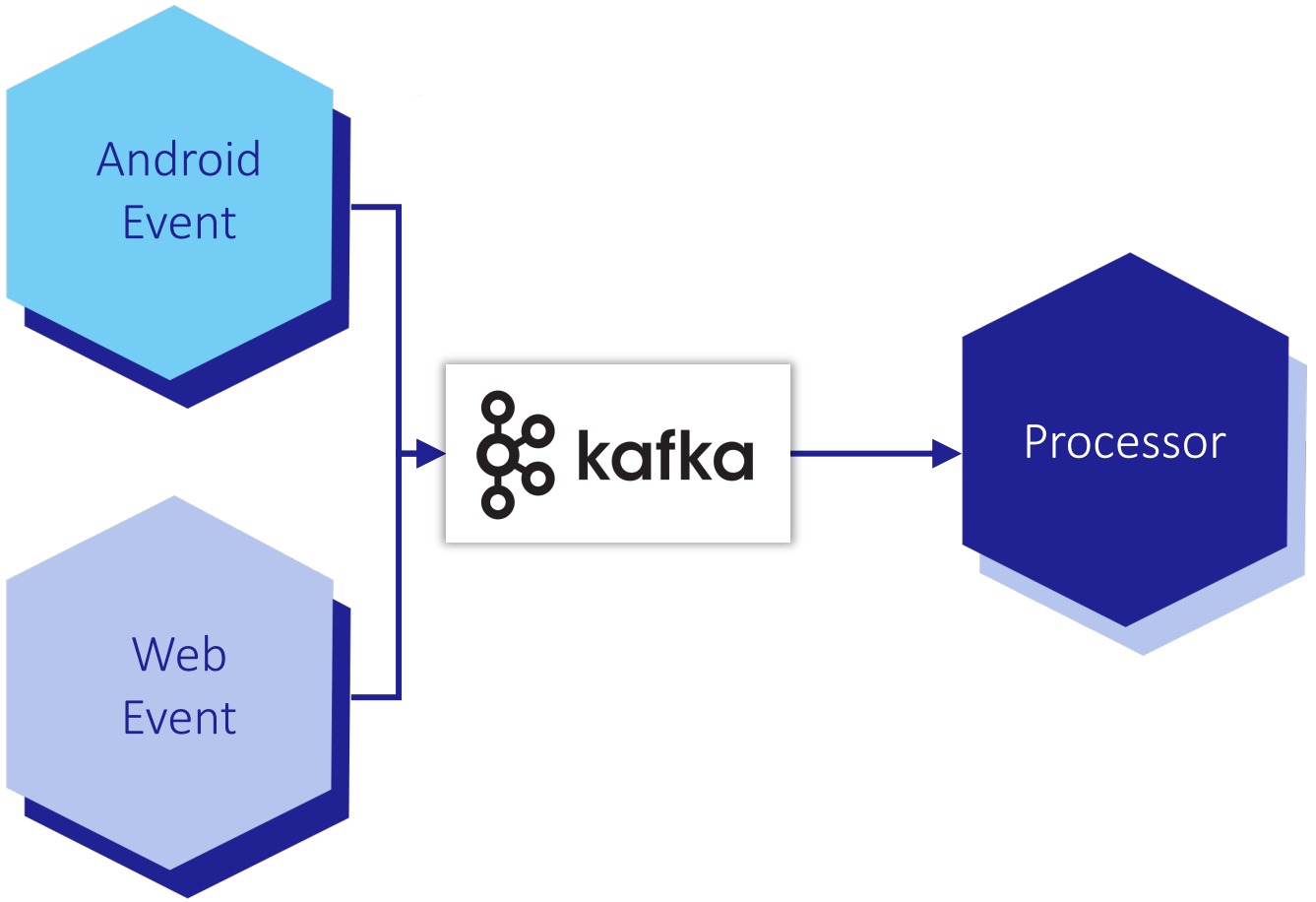
Building high-throughput applications with bulk-messaging in Dapr

Shubham Sharma (@shubham1172)
Dapr JS-SDK Maintainer
Software Engineer, Microsoft

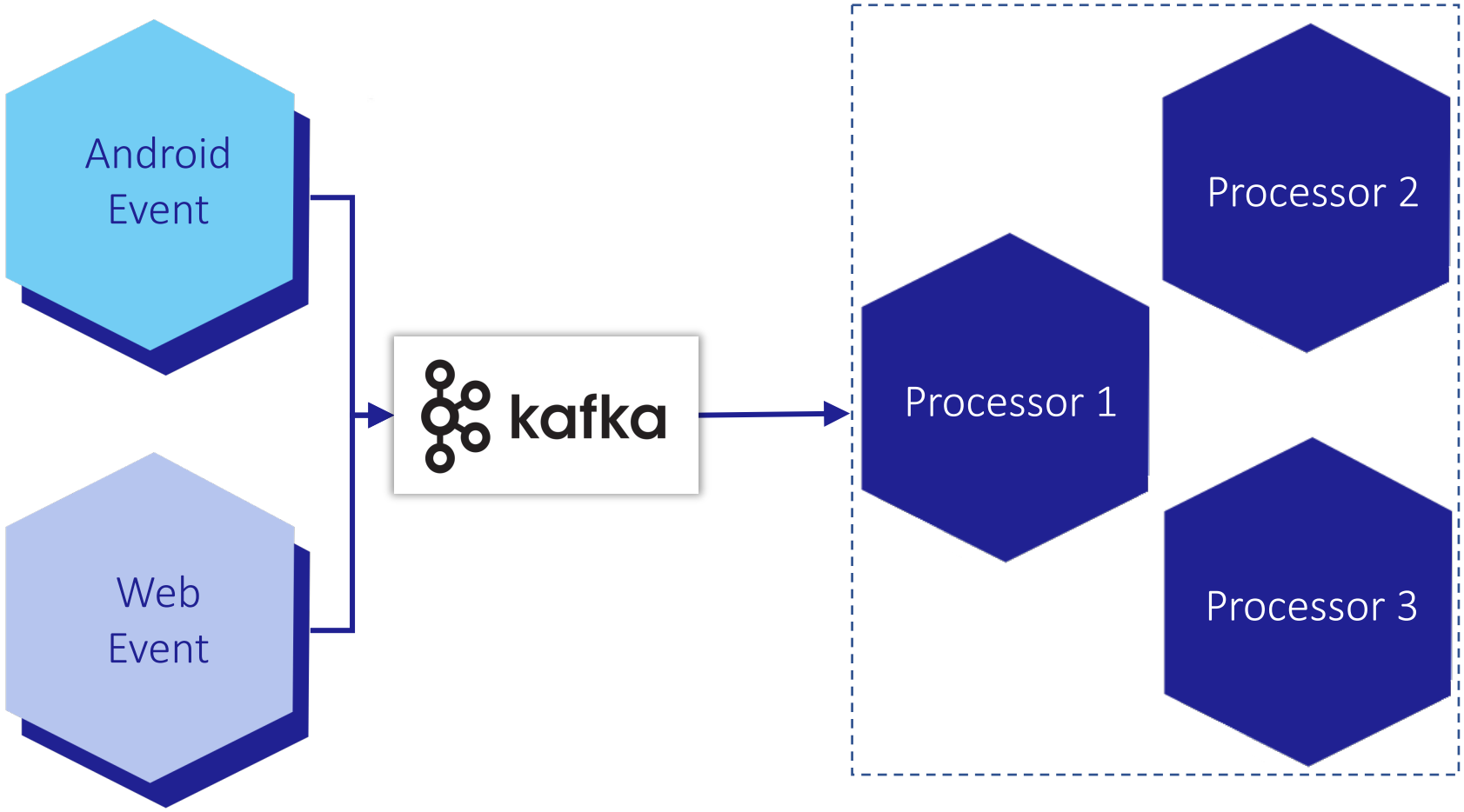
Asynchronous messaging



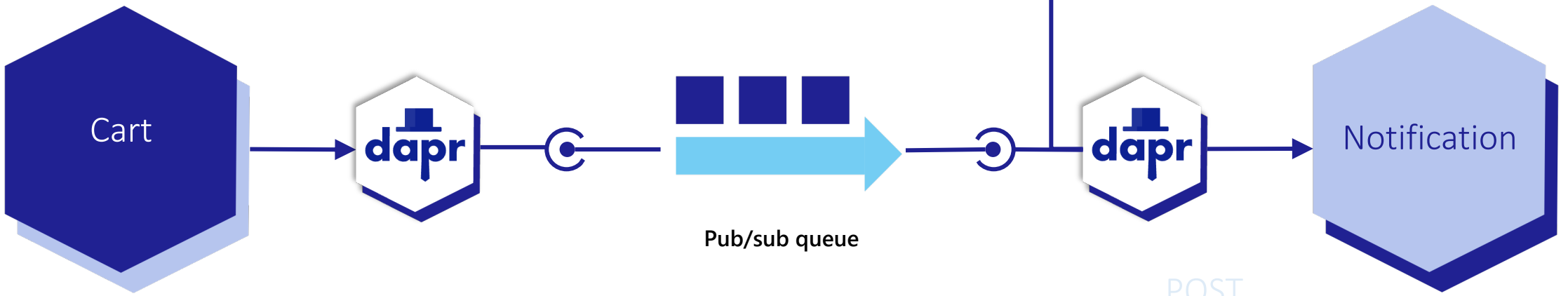
Asynchronous messaging



Asynchronous messaging



Asynchronous messaging with Dapr



POST
http://localhost:3500/v1.0/publish/kafka/order

{"data":"MyOrder"}

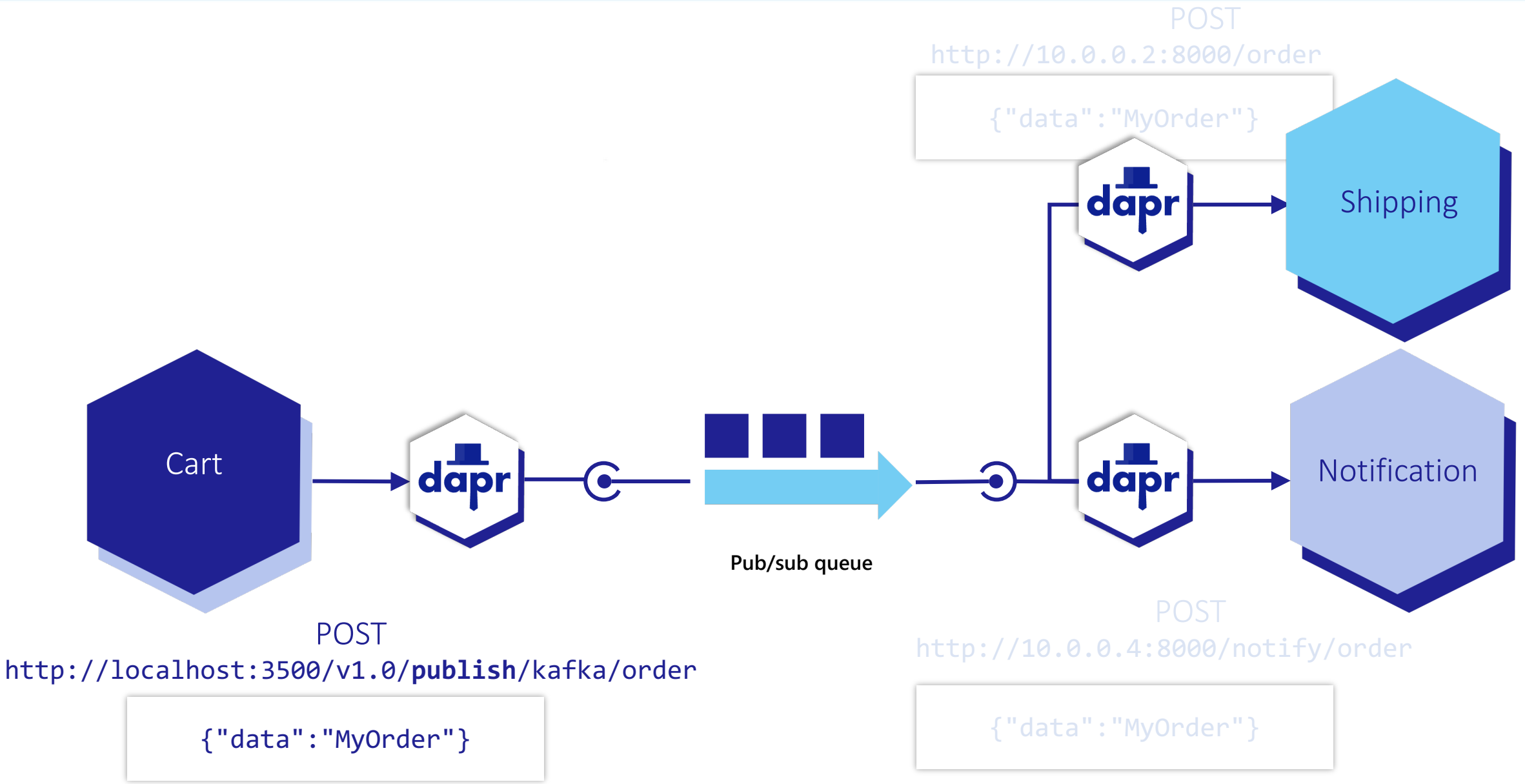
POST
http://10.0.0.2:8000/order

{"data":"MyOrder"}

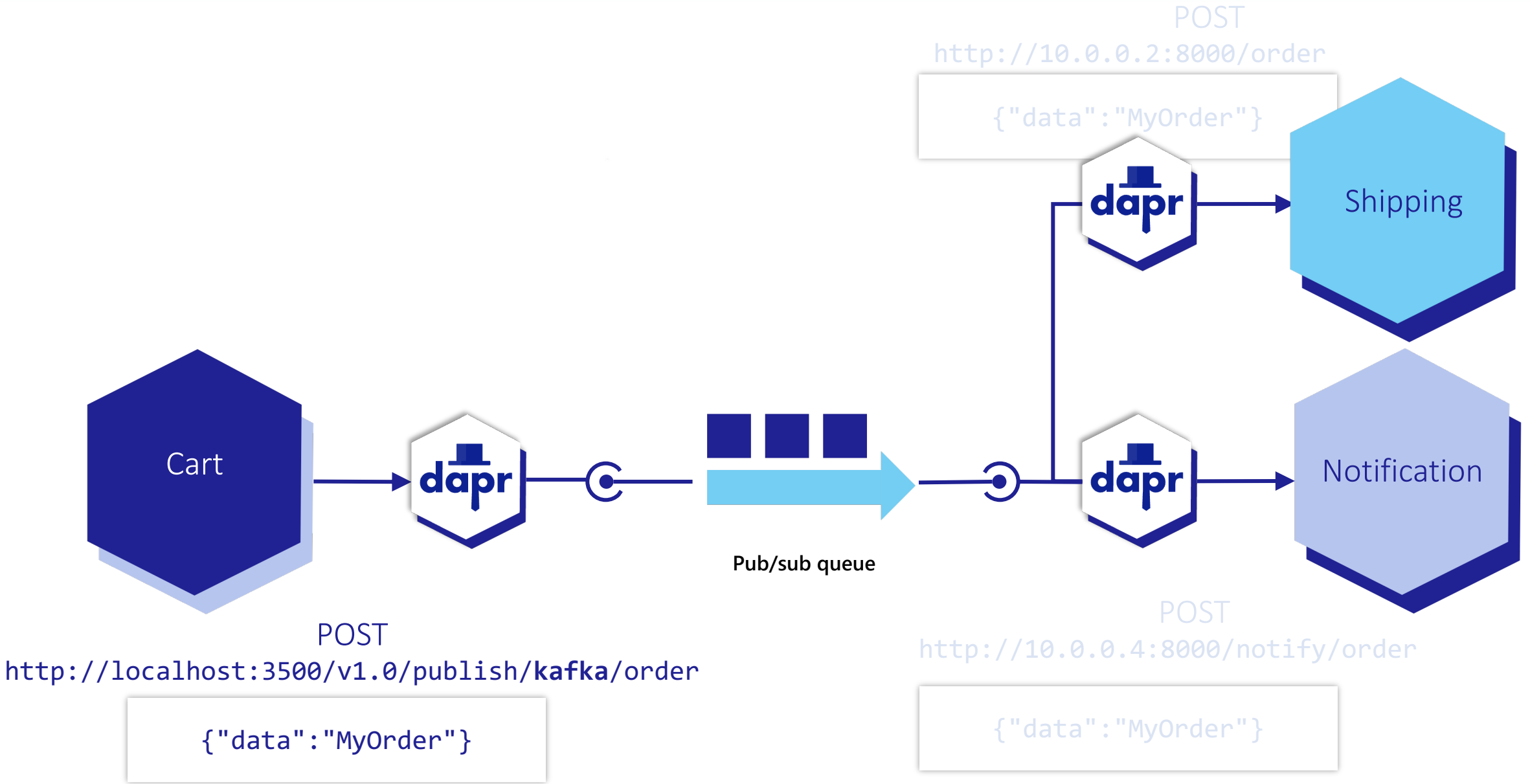
POST
http://10.0.0.4:8000/notify/order

{"data":"MyOrder"}

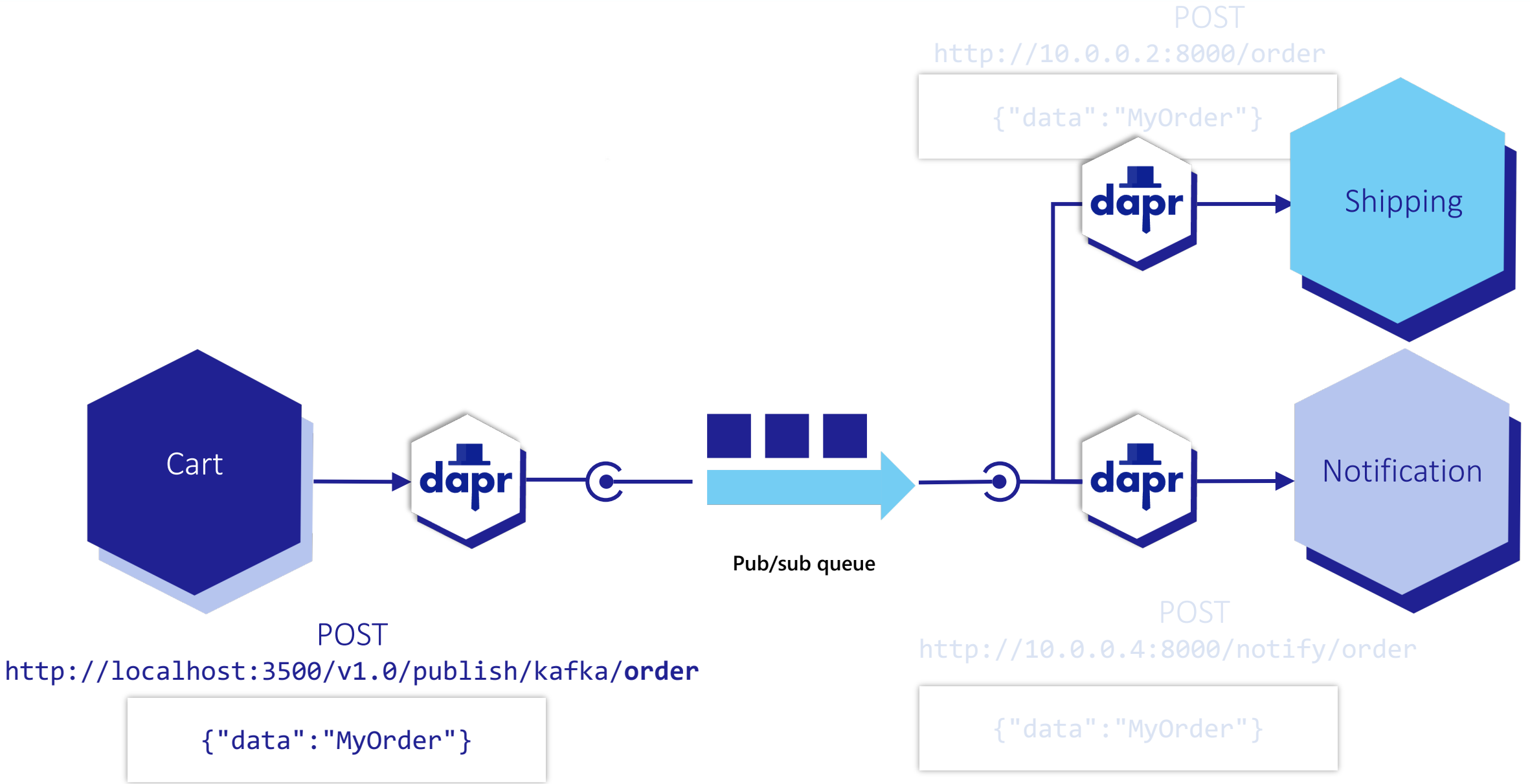
Asynchronous messaging with Dapr



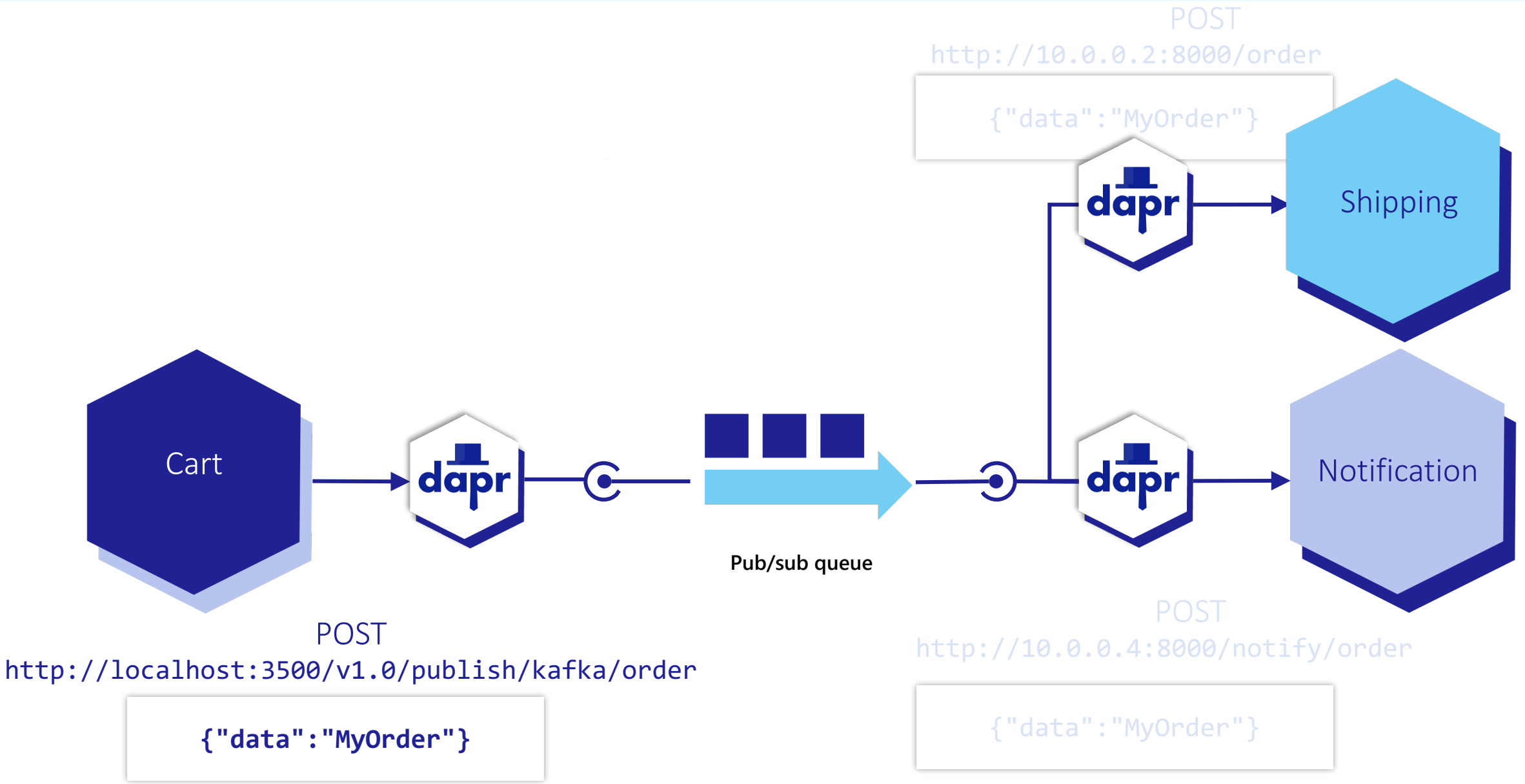
Asynchronous messaging with Dapr



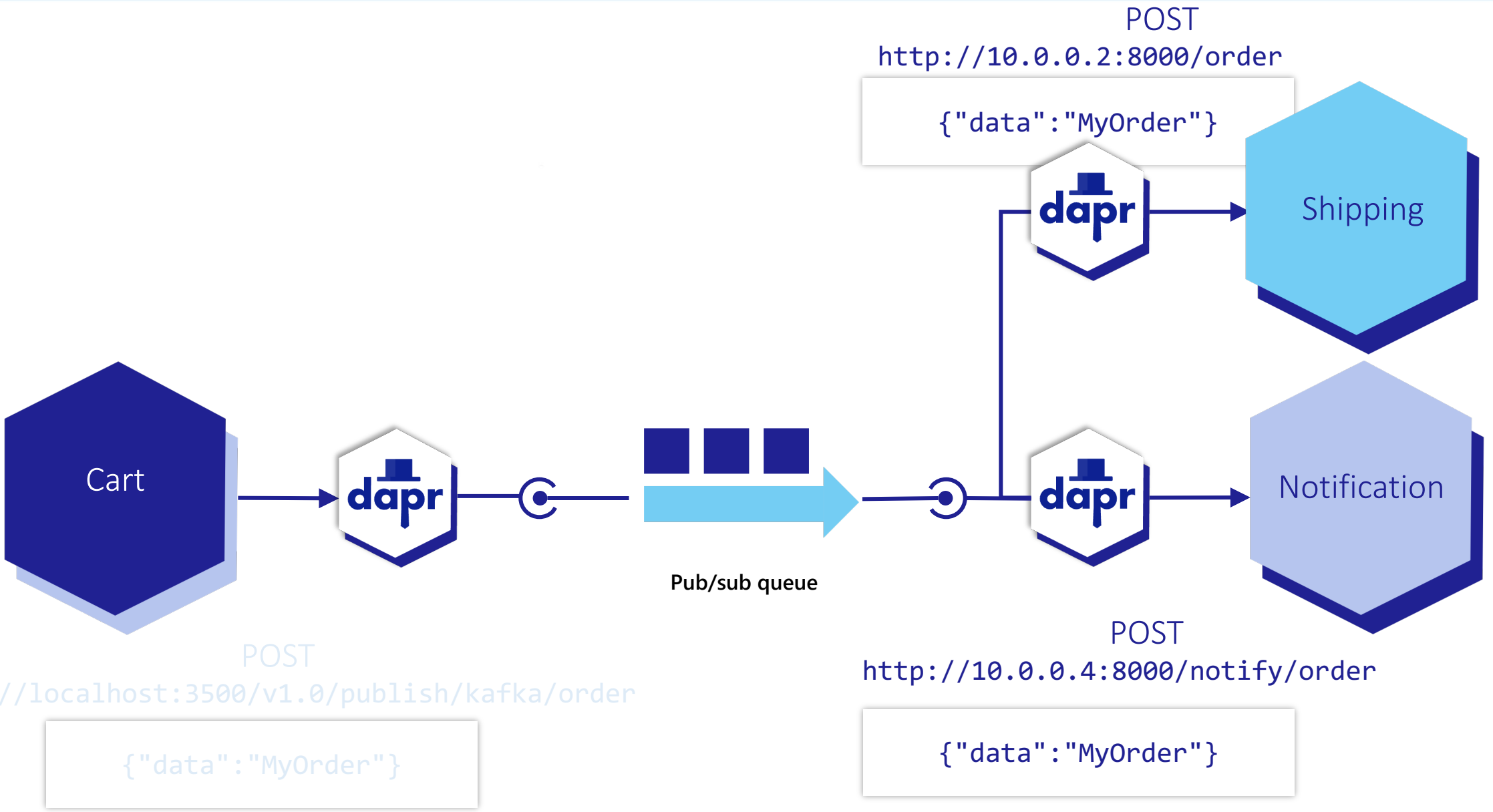
Asynchronous messaging with Dapr



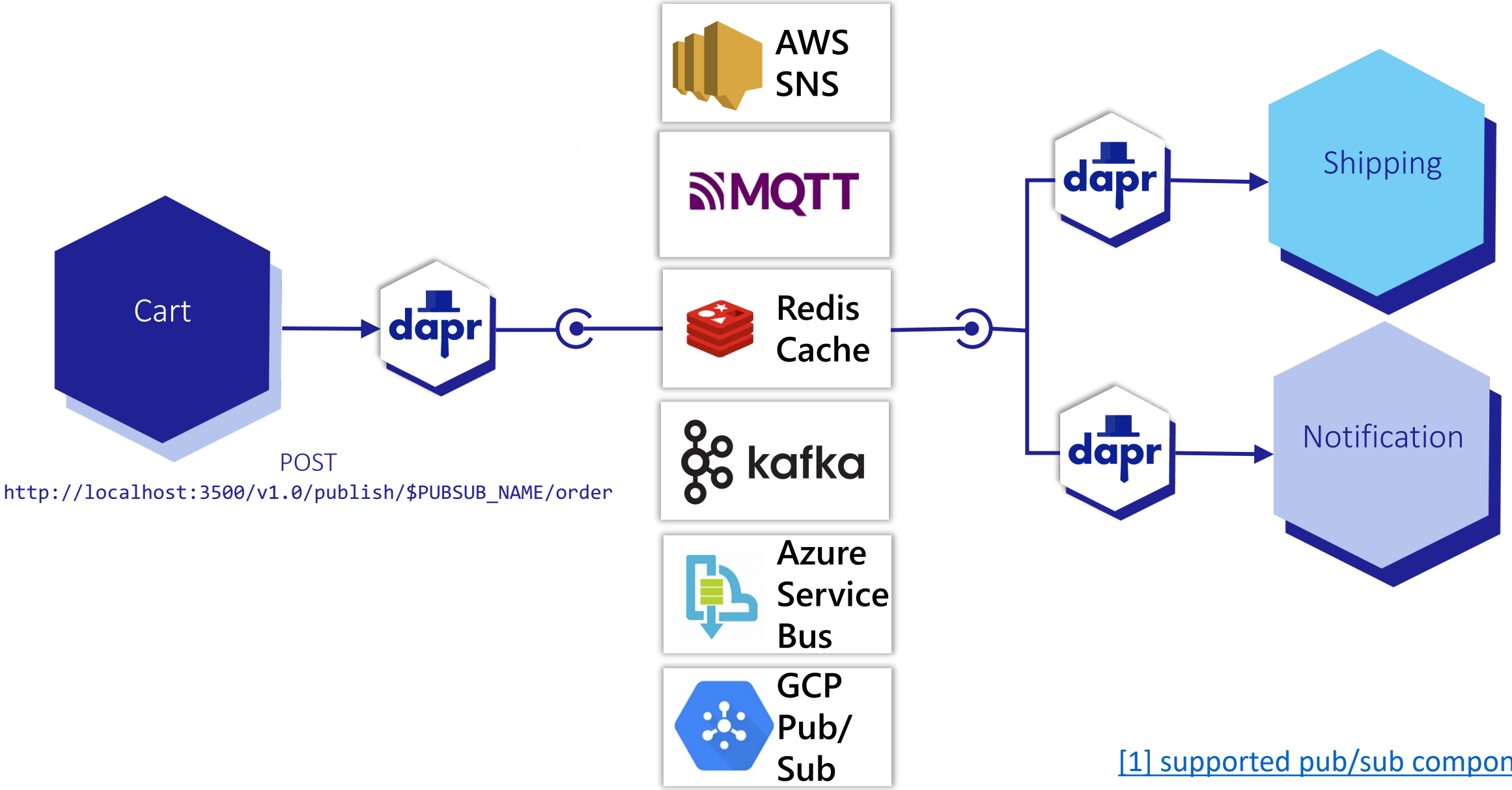
Asynchronous messaging with Dapr



Asynchronous messaging with Dapr



Asynchronous messaging with Dapr



[1] supported pub/sub components

Asynchronous messaging with Dapr

KubeCon CloudNativeCon Europe 2023

Dapr Pub/Sub highlights

1. At-least once delivery
2. Consumer groups and competing consumers patterns
3. Resilient publish and subscribe operations
4. Dead-letter topic support
5. Bulk messaging

```yaml
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: redis-pubsub
spec:
  type: pubsub.redis
  version: v1
  metadata:
  - name: redisHost
    value: localhost:6379
  - name: redisPassword
    value: "wachtwoord"
  - name: consumerID
    value: "myGroup"
  - name: enableTLS
    value: "false"
```

[2] Overview of pub/sub features

Asynchronous messaging with Dapr

Dapr Pub/Sub highlights

1. At-least once delivery
2. Consumer groups and competing consumers patterns
3. Resilient publish and subscribe operations
4. Dead-letter topic support
5. Bulk messaging

[2] Overview of pub/sub features

```yaml
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: redis-pubsub
spec:
  type: pubsub.redis
  version: v1
  metadata:
  - name: redisHost
    value: localhost:6379
  - name: redisPassword
    value: "wachtwoord"
  - name: consumerID
    value: "myGroup"
  - name: enableTLS
    value: "false"
```

Asynchronous messaging with Dapr

KubeCon CloudNativeCon Europe 2023

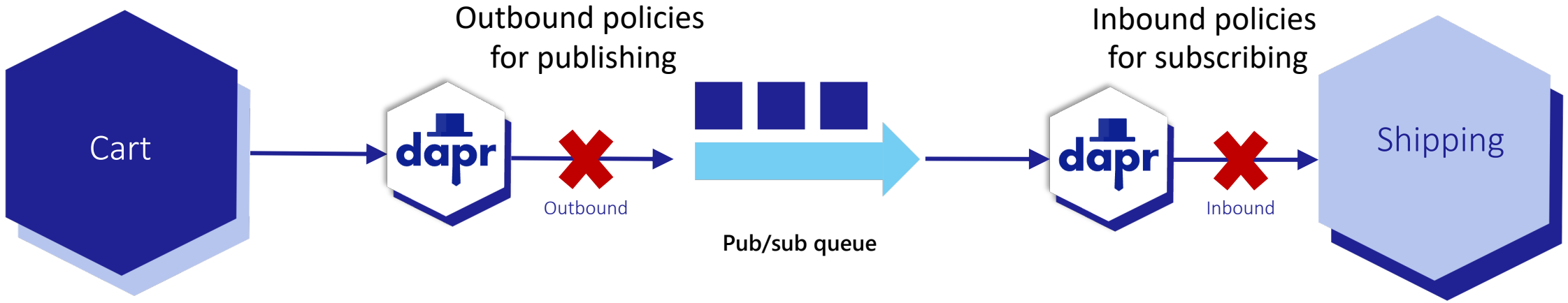
Dapr Pub/Sub highlights

1. At-least once delivery
2. Consumer groups and competing consumers patterns
3. Resilient publish and subscribe operations
4. Dead-letter topic support
5. Bulk messaging

```yaml
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: redis-pubsub
spec:
  type: pubsub.redis
  version: v1
  metadata:
  - name: redisHost
    value: localhost:6379
  - name: redisPassword
    value: "wachtwoord"
  - name: consumerID
    value: "myGroup"
  - name: enableTLS
    value: "false"
```

[2] Overview of pub/sub features

Asynchronous messaging with Dapr



Supported policies

1. Timeouts
2. Retries
3. Circuit Breakers

Asynchronous messaging with Dapr

Dapr Pub/Sub highlights

1. At-least once delivery
2. Consumer groups and competing consumers patterns
3. Resilient publish and subscribe operations
4. Dead-letter topic support
5. Bulk messaging

[2] Overview of pub/sub features

```yaml
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: redis-pubsub
spec:
  type: pubsub.redis
  version: v1
  metadata:
  - name: redisHost
    value: localhost:6379
  - name: redisPassword
    value: "wachtwoord"
  - name: consumerID
    value: "myGroup"
  - name: enableTLS
    value: "false"
```

Asynchronous messaging with Dapr

KubeCon CloudNativeCon Europe 2023

Dapr Pub/Sub highlights

1. At-least once delivery
2. Consumer groups and competing consumers patterns
3. Resilient publish and subscribe operations
4. Dead-letter topic support
5. Bulk messaging

```yaml
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: redis-pubsub
spec:
  type: pubsub.redis
  version: v1
  metadata:
  - name: redisHost
    value: localhost:6379
  - name: redisPassword
    value: "wachtwoord"
  - name: consumerID
    value: "myGroup"
  - name: enableTLS
    value: "false"
```

[2] Overview of pub/sub features

Asynchronous messaging with Dapr

Dapr Pub/Sub highlights

1. At-least once delivery
2. Consumer groups and competing consumers patterns
3. Resilient publish and subscribe operations
4. Dead-letter topic support
5. Bulk messaging

```yaml
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: eventhubs-pubsub
spec:
  type: pubsub.azure.eventhubs
  version: v1
  metadata:
    - name: eventHubNamespace
      value: "contosoNs"
    - name: storageAccountName
      value: "contosoStorage"
    - name: storageAccountKey
      value: "122333444455555666666"
    - name: storageContainerName
      value: "contosoContainer"
```

[2] Overview of pub/sub features

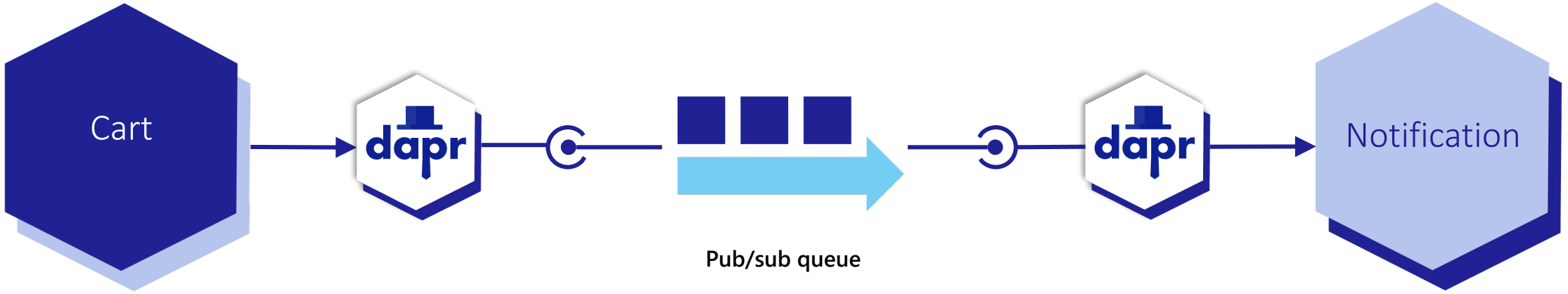
Demo (1 of 2)

Demo time

1. Simple publish and subscribe (HTTP and SDK)
2. Swappable components
3. Consumer groups and competing consumers patterns

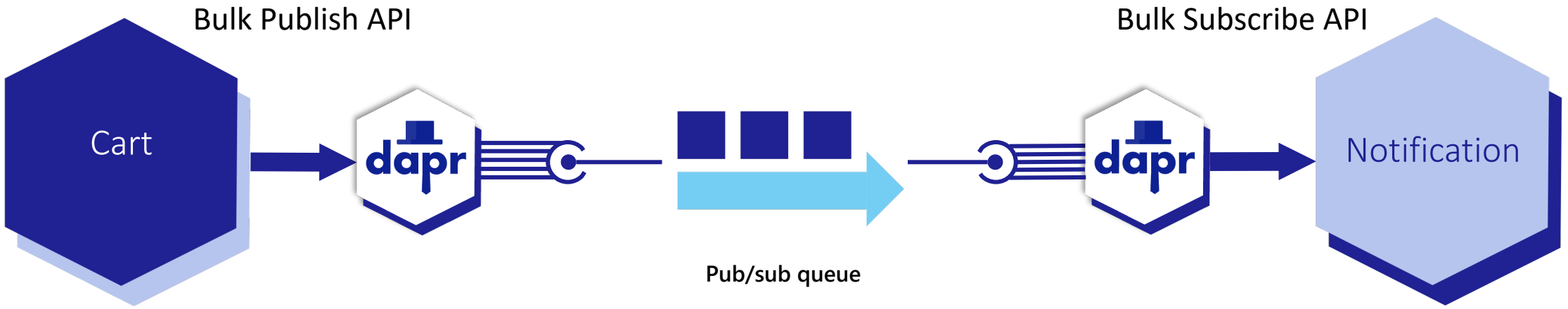
Asynchronous bulk messaging with Dapr

1. Bulk Pub/Sub APIs
2. Bulk support by Pub/Sub Dapr component



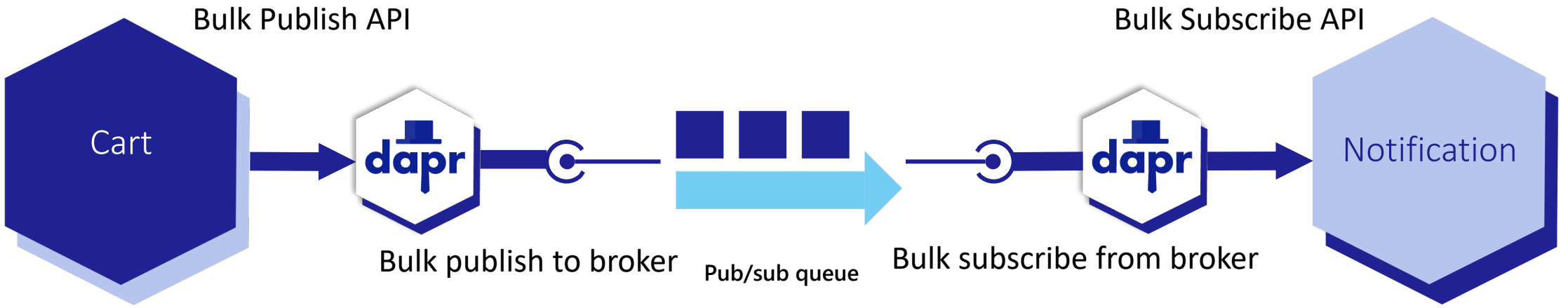
Asynchronous bulk messaging with Dapr

1. Bulk Pub/Sub APIs
2. Bulk support by Pub/Sub Dapr component



Asynchronous bulk messaging with Dapr

1. Bulk Pub/Sub APIs
2. Bulk support by Pub/Sub Dapr component



Asynchronous bulk messaging with Dapr

Bulk Pub/Sub highlights

1. Available as an **alpha API** from **Dapr v1.10**
2. **Highly performant** – optimizes App<=>Dapr and Dapr<=>Broker communication
3. **Non-transactional** – may partially fail
4. **No ordering guarantee** – messages are identifiable via unique IDs

[3] Publish and subscribe to bulk messages

Asynchronous bulk messaging with Dapr

Bulk Pub/Sub highlights

1. Available as an **alpha API** from **Dapr v1.10**
2. **Highly performant** – optimizes App<=>Dapr and Dapr<=>Broker communication
3. **Non-transactional** – may partially fail
4. **No ordering guarantee** – messages are identifiable via unique IDs

[3] Publish and subscribe to bulk messages

Asynchronous bulk messaging with Dapr

Bulk Pub/Sub highlights

1. Available as an **alpha API** from **Dapr v1.10**
2. **Highly performant** – optimizes App<=>Dapr and Dapr<=>Broker communication
3. **Non-transactional** – may partially fail
4. **No ordering guarantee** – messages are identifiable via unique IDs

[3] Publish and subscribe to bulk messages

Asynchronous bulk messaging with Dapr

Bulk Pub/Sub highlights

1. Available as an **alpha API** from **Dapr v1.10**
2. **Highly performant** – optimizes App<=>Dapr and Dapr<=>Broker communication
3. **Non-transactional** – may partially fail
4. **No ordering guarantee** – messages are identifiable via unique IDs

[3] Publish and subscribe to bulk messages

Demo (2 of 2)

Demo time

Bulk Publish and Subscribe APIs

Migrating application code to use bulk APIs

KubeCon CloudNativeCon Europe 2023

Publish one-by-one

```
const numMessages = 100;
for (let i = 0; i < numMessages; i++) {
    const message = {
        id: i,
        message: "Welcome to KubeCon + CloudNativeCon Europe 2023!"
    };

    await client.pubsub.publish(pubSubName, topicName, message);
}
```

Publish in bulk

```
const numMessages = 100;
const messages = [];
for (let i = 0; i < numMessages; i++) {
    messages.push({
        id: i,
        message: "Welcome to KubeCon + CloudNativeCon Europe 2023!"
    });
}
await client.pubsub.publishBulk(pubSubName, topicName, messages);
```

[4] JS-SDK PubSub API reference

Migrating application code to use bulk APIs

Receive one-by-one

```
daprServer.pubsub.subscribe(pubSubName, topicName, (message) => {
    console.log(`Received message (individually): ${JSON.stringify(message)}`);
});
```

Receive in bulk

```
daprServer.pubsub.subscribeBulk(pubSubName, topicName, (message) => {
    console.log(`Received message (in bulk): ${JSON.stringify(message)}`);
});
```

[4] JS-SDK PubSub API reference

How fast is it?

Publishing **100** messages of size **1 KiB** with **in-memory** broker over **HTTP**

Publish

max	225.37
p90	168.96
p95	**176.2**
p50 (avg)	**148.02**
min	0.42
med	146.90

Bulk Publish

max	168.84
p90	45.13
p95	**55.29**
p50 (avg)	**20.14**
min	0.78
med	13.59

All units are in milliseconds

How fast is it?

Send and receive **100 messages** of size **100 bytes** with **Kafka** over **HTTP**

Subscribe

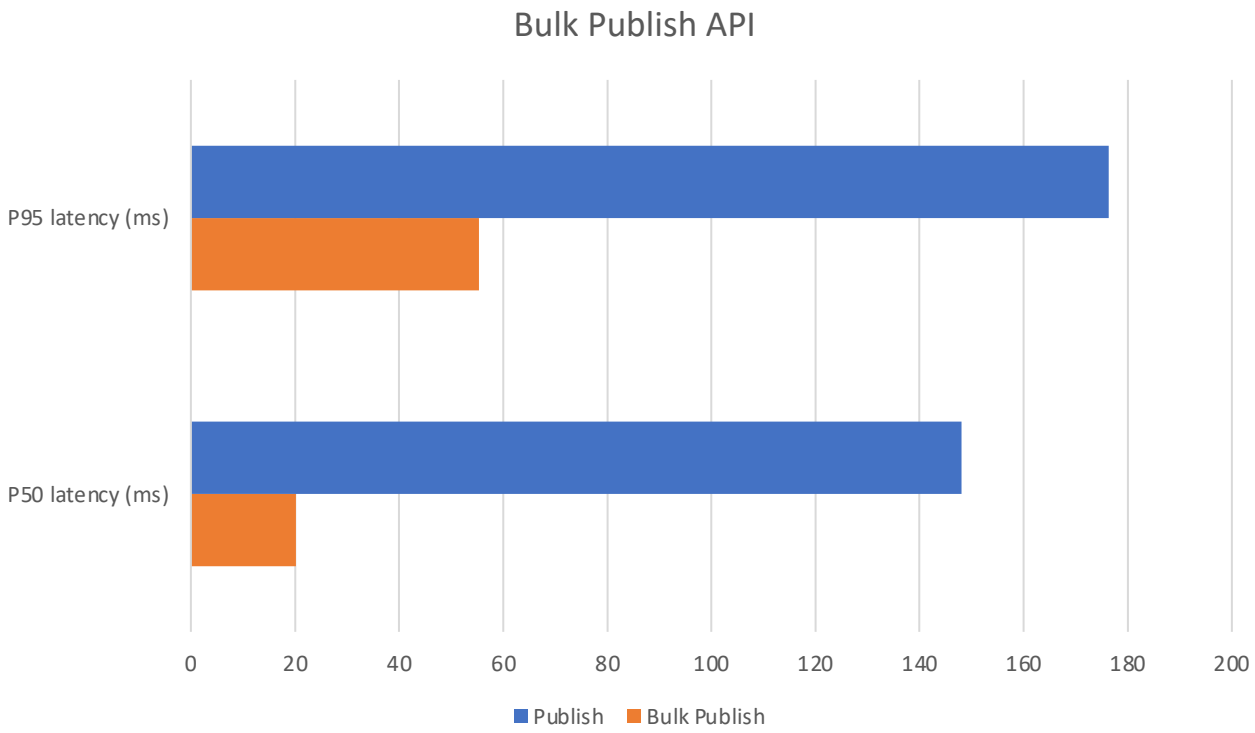
max	1176.94
p90	986.71
p95	**1066.04**
p50 (avg)	**502.75**
min	0.309
med	475.46

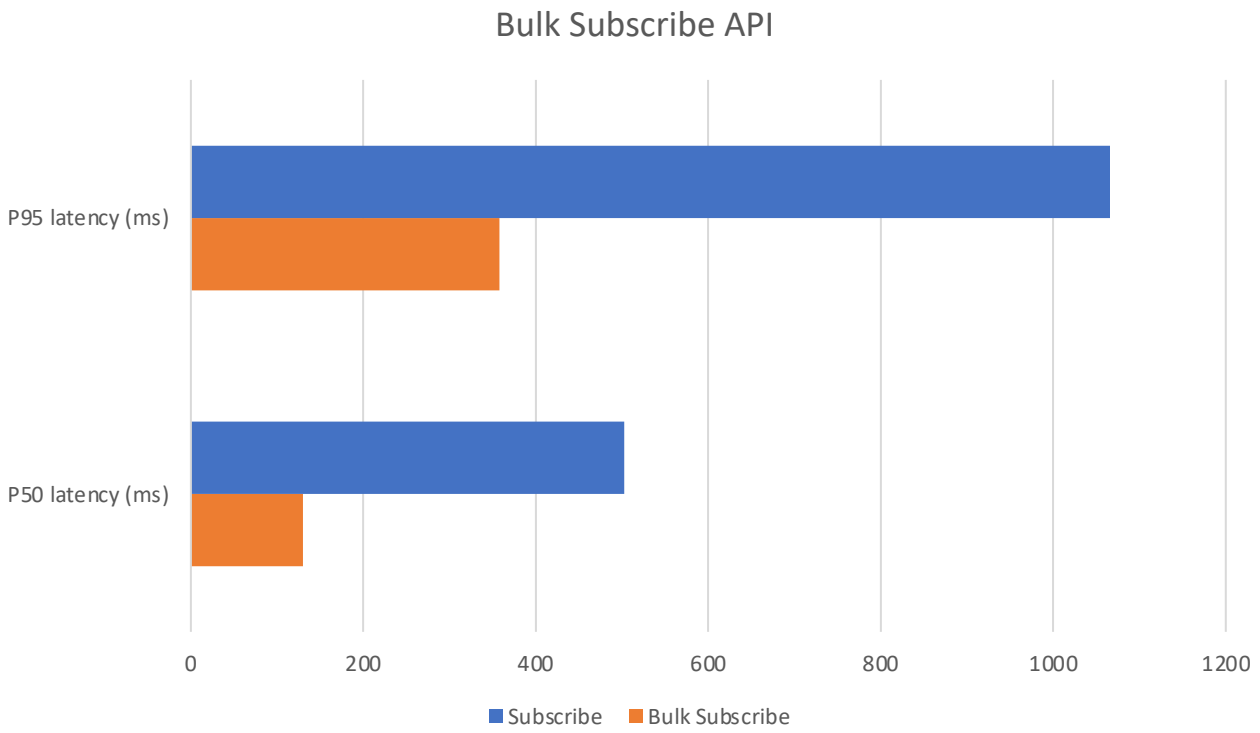
Bulk Subscribe

max	470.40
p90	248.49
p95	**357.75**
p50 (avg)	**129.71**
min	0.56
med	105.97

All units are in milliseconds

How fast is it?





Thank you

Get started at http://dapr.io

Join the Discord community aka.ms/dapr-discord

Join the community calls aka.ms/dapr-community

Follow on Twitter @daprdev

Contribute at github.com/dapr

Links

[1] https://docs.dapr.io/reference/components-reference/supported-pubsub/

[2] https://docs.dapr.io/developing-applications/building-blocks/pubsub/pubsub-overview/#features

[3] https://docs.dapr.io/developing-applications/building-blocks/pubsub/pubsub-bulk/

[4] https://v1-11.docs.dapr.io/developing-applications/sdks/js/js-client/#publish-messages

KubeCon CloudNativeCon
Europe 2023
North America 2021

Please scan the QR Code above
to leave feedback on this session