
KubeCon
CloudNativeCon
North America 2019

cloudevents

KubeCon | CloudNativeCon

North America 2019

CloudEvents
Intro, Deep-Dive and More!

Clemens Vasters, Microsoft
Klaus Deissner, SAP
Doug Davis, IBM
Vladimir Bacvanski, PayPal

Agenda

- CloudEvents
  - History & Summary
  - Demo
  - Status & Plans post v1.0

- CloudEvents in Production
  - Microsoft - Clemens
  - SAP - Klaus
  - Knative - Doug
  - PayPal - Vladimir

Serverless WG Overview

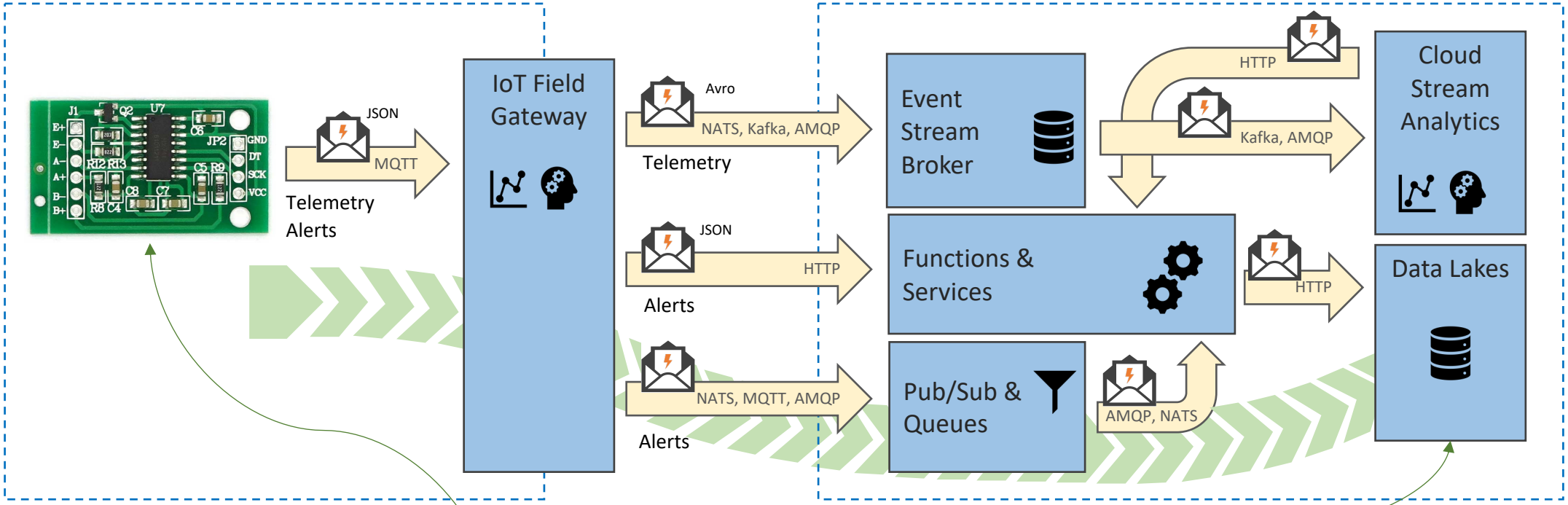
- Technical Oversight Committee initiated (mid-2017)
  - Whitepaper
    - Overview of technology
    - State of ecosystem
    - Recommendations for possible CNCF next steps
  - Landscape
- CloudEvents
  - Project started Dec 2017
  - CNCF Sandbox project approved May 2018
  - CNCF Incubator project approved Oct 2019
- Function workflow - orchestration of Functions

CNCF CloudEvents

- Event Protocol Suite developed in CNCF Serverless WG
  - Common metadata attributes for events
  - Flexibility to innovate on event semantics
  - Simple abstract type system mappable to different encodings
- Transport options
  - HTTP(S) 1.1 Webhooks, also HTTP/2
  - MQTT 3.1.1 and 5.0
  - AMQP 1.0
- Encoding options
  - JSON (required for all implementations)
  - Extensible for binary encodings: Avro, AMQP, etc.

cloudevents

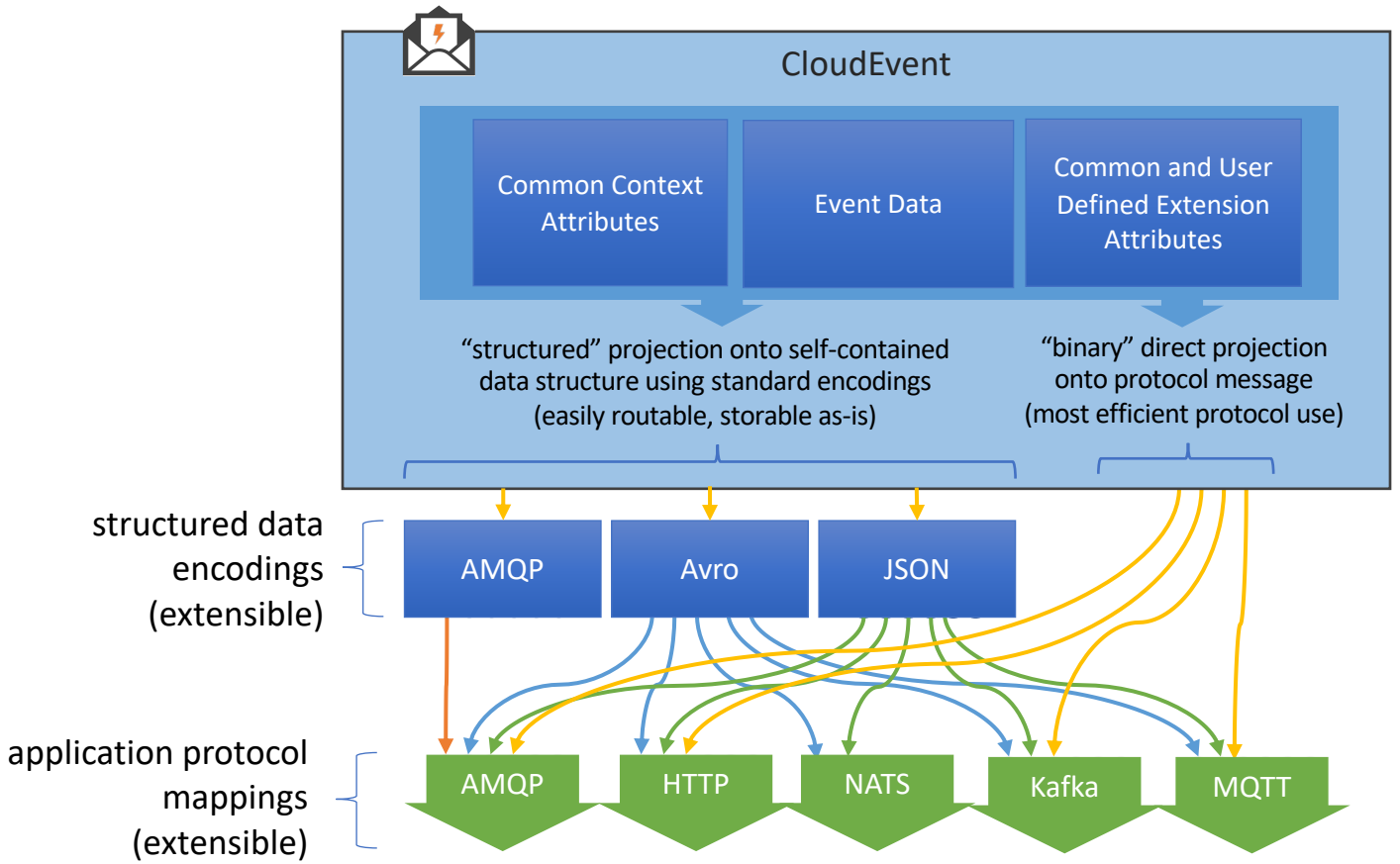
Why CloudEvents?



- Event data is often routed via multiple hops and often using different protocols
- How is what gets sent here easily routed to and stored here in hybrid edge/cloud and multi-cloud systems?

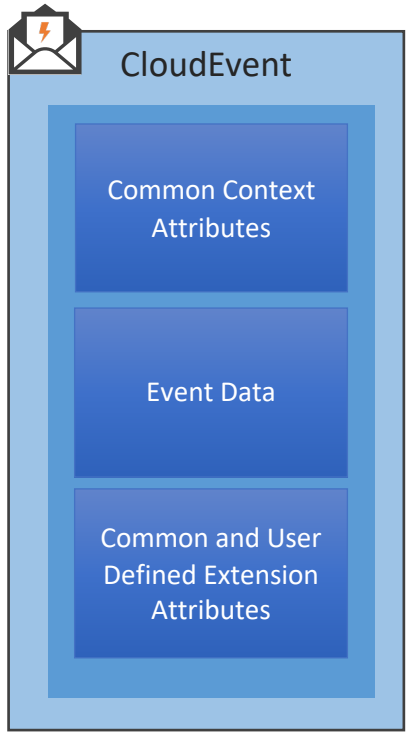
Why CloudEvents?

- Binds to existing standard application protocols
- Does not try to abstract away protocols but leverages each for its strengths
- Integrates with existing messaging and eventing stacks
- Leverages existing data encodings and is easy to adapt to new ones (Protobuf, CBOR, MsgPack, etc.)
- Allows for protocol switching and transcoding on multi-hop routes



CloudEvents - Base Specification

- CloudEvents is a lightweight common convention for events.
- It's *intentionally* not a messaging model* to keep complexity low.
  - No reply-path indicators, no message-to-message correlation, no target address indicators, no command verbs/methods.
- Metadata for handling of events by generic middleware and/or dispatchers
  - What kind of event is it? **type, specversion**
  - When was it sent? **time**
  - What context was it sent out of? **source, subject**
  - What is this event's unique identifier? **id**
  - What's the shape of the carried event data? **datacontenttype**, **schema**
- Event data may be text-based (esp. JSON) or using some binary encoding

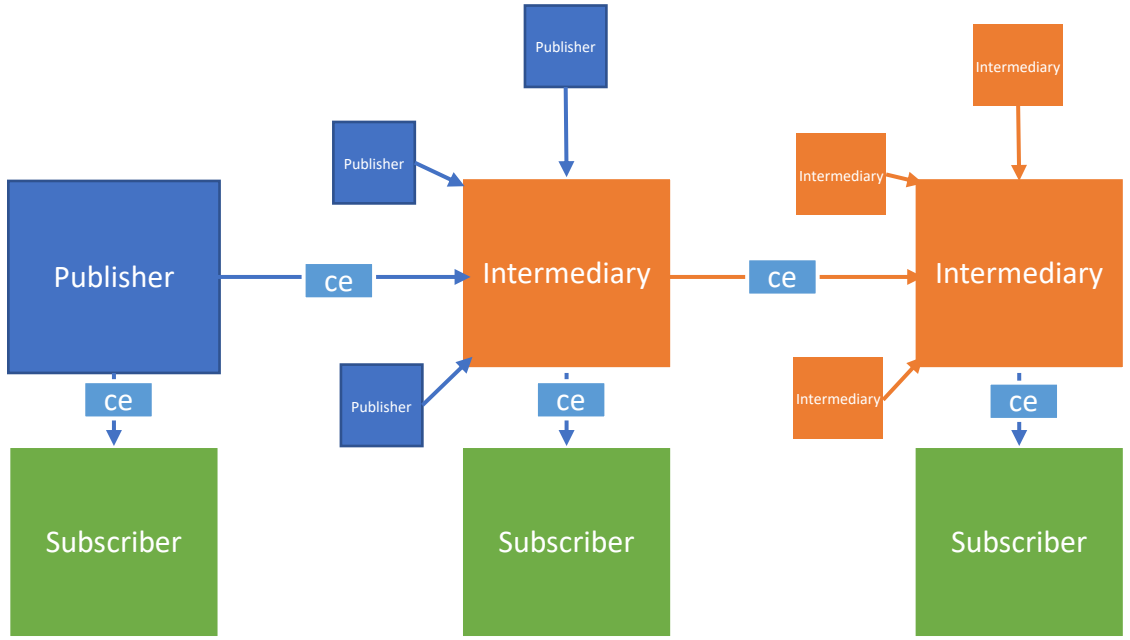


Eventing vs Messaging

- Events and messages are both mailing envelopes for data, decorated by metadata – but they are different.
  - Events carry facts. They report things that have happened.
    - State transitions, observed conditions, objects having been created, …
  - Messages carry intents. The sender expects something to happen.
    - Command execution, job handling, workflow progress, …
- Events are published as an information option for interested subscribers. Audience size may be zero or many.
- Messages are directed to handlers. There may be delivery and handling status feedback, replies, conversations, or complex control flows like Workflows and Sagas. Audience size is often one handler per message.

CloudEvents is Eventing

- Carry facts: “Something happened”
- Publish/subscribe distribution
  - Subscribe at source or distributor
- Some publish/subscribe principles:
  - Publishers don’t know/care who will subscribe or is currently subscribed
  - Subscribers might tap into single-sourced or consolidated event streams
  - Event flow is unidirectional
    - network or disk
    - one or more routing intermediaries
    - radio broadcast
    - …



Attributes CloudEvents does not define

- “to”
  - There’s no “to” attribute because events aren’t aimed at and routed to a specific consumer. Subscribers get to decide which events they pick.
- “reply-to”
  - There’s no “reply-to” attribute because events aren’t job assignments that ought to require a reply *and* because a subscriber can’t be expected to be capable of reaching any given reply destination
- “topic” / “queue”
  - CloudEvents defines the origin context as “source”, but does not include the name of any specific intermediary construct in the event metadata because events might travel through multiple intermediary hops

CloudEvents - Event Formats

- Event formats bind the abstract CloudEvents information model to specific wire encodings.
- All implementation must support JSON. Avro is a supported binary format.
- Further compact binary event format candidates might be CBOR, or Protobuf.

```
{
  "specversion" : "1.0",
  "type" : "myevent",
  "source" : "uri:example-com:mydevice",
  "id" : "A234-1234-1234",
  "time" : "2018-04-05T17:31:00Z",
  "datacontenttype" : "text/plain",
  "data" : "Hello"
}
```

JSON Representation

CloudEvents - Example

HTTP - Binary

```
POST /event HTTP/1.0
Host: example.com
Content-Type: application/json
ce-specversion: 1.0
ce-type: com.bigco.newItem
ce-source: http://bigco.com/repo
ce-id: 610b6dd4-c85d-417b-b58f-3771e532

{
  "action": "newItem",
  "itemID": "93"
}
```

HTTP - Structured

```
POST /event HTTP/1.0
Host: example.com
Content-Type: application/cloudevents+json

{
  "specversion": "1.0",
  "type": "com.bigco.newItem",
  "source": "http://bigco.com/repo",
  "id": "610b6dd4-c85d-417b-b58f-3771e532",
  "datacontenttype": "application/json",
  "data": {
    "action": "newItem",
    "itemID": "93"
  }
}
```

CloudEvents - Transport Bindings

- HTTP 1.1, HTTP/2, HTTP/3:
  - Binds to the HTTP message
  - Binary and structured modes
- AMQP:
  - Binds event to the AMQP message
  - Binary and structured modes
- MQTT:
  - Binds event to MQTT PUBLISH frame.
  - Binary and Structured for MQTT v5
  - Structured mode only for MQTT v3.1.1
- NATS:
  - Binds event to the NATS message.
  - Structured mode only
- Apache Kafka:
  - Binds to the Kafka message
  - Structured and binary mode

Protocol bindings directly map onto the protocol's message structure and using protocol semantics. Accepts that protocols are different.

Binary mode: Event metadata projected onto the protocol message metadata, event data onto the protocol message payload

Structured mode: Event is self-contained as an encoded byte stream, metadata may be promoted (duplicated) into protocol message metadata.

Complicated Matters Scoped out of v1.0

- Signatures
  - Symmetric: Who holds the signing keys?
  - Asymmetric: Who distributes the verification keys?
  - Whose directory/directories/vaults is the subscriber trusting?
  - Which subscribers does a directory/vault grant access?
  - How/when does who rotate signing keys? How do subscribers know?
  - How to keep track of key history (archived events, events in logs)?
- End-to-End Encryption
  - Same as above but with encryption keys
- Encrypting/signing multicast datagrams doesn't allow for peer-to-peer session keys, which means that "master" keys must be rotated far more frequently than when those are only used for session-key exchange

- Hardest: Agree on **ONE WAY OF DOING ALL THIS:** APIs, Algos, Hints, Versioning

Project Status

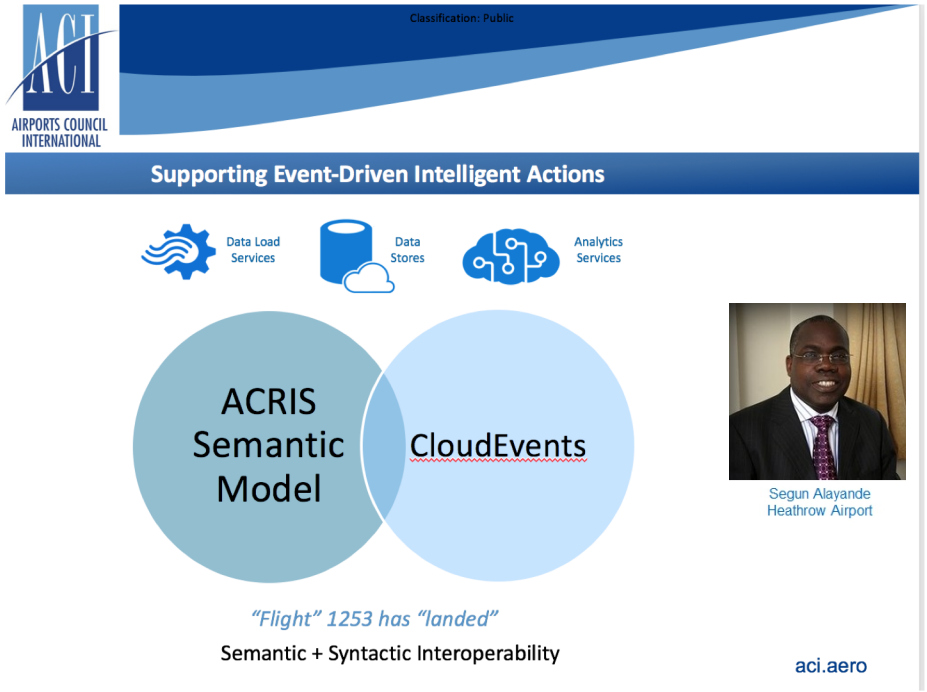
- V1.0 - Approved 10/24/2019
- Incubator status – Approved 10/24/2019

- What's next?

Airport Demo - ACRIS Semantic Model

“The Semantic Model, coupled with the CloudEvents format, can support interoperable, event-driven systems that can re-act intelligently to real-time state changes such as an aircraft landing or fulfilled order. With the use of event brokers, edge-to-cloud computing and AI, these events can be detected faster and analyzed in greater detail.”

Segun Alayande
Heathrow Airport



Airport Demo

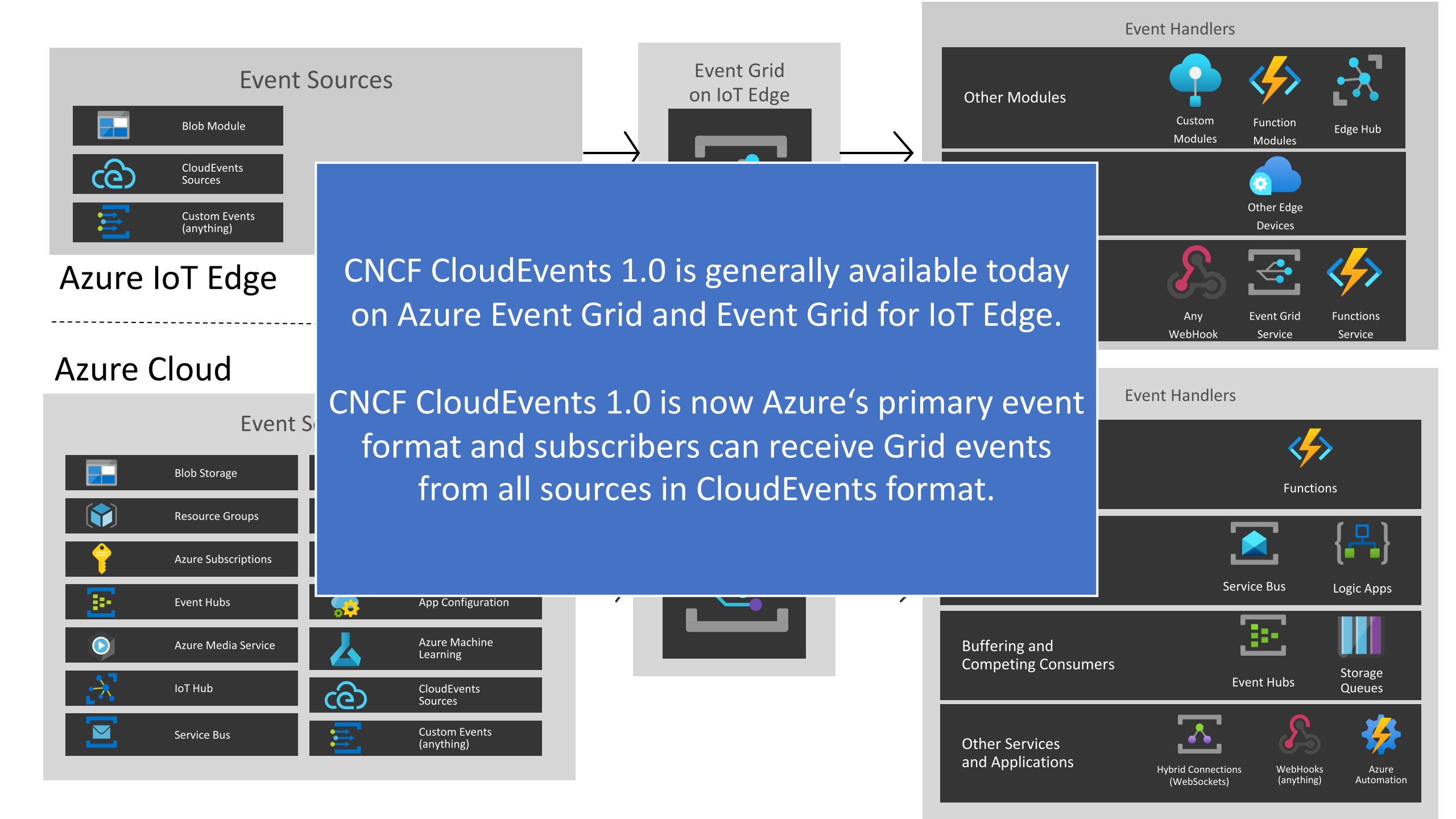
Scan this code to get FREE coffee* →

http://srcdog.com/airport

*the coffee is virtual.


KubeCon
CloudNativeCon
North America 2019
CloudEvents in Production


KubeCon
CloudNativeCon
North America 2019
Microsoft
Clemens Vasters

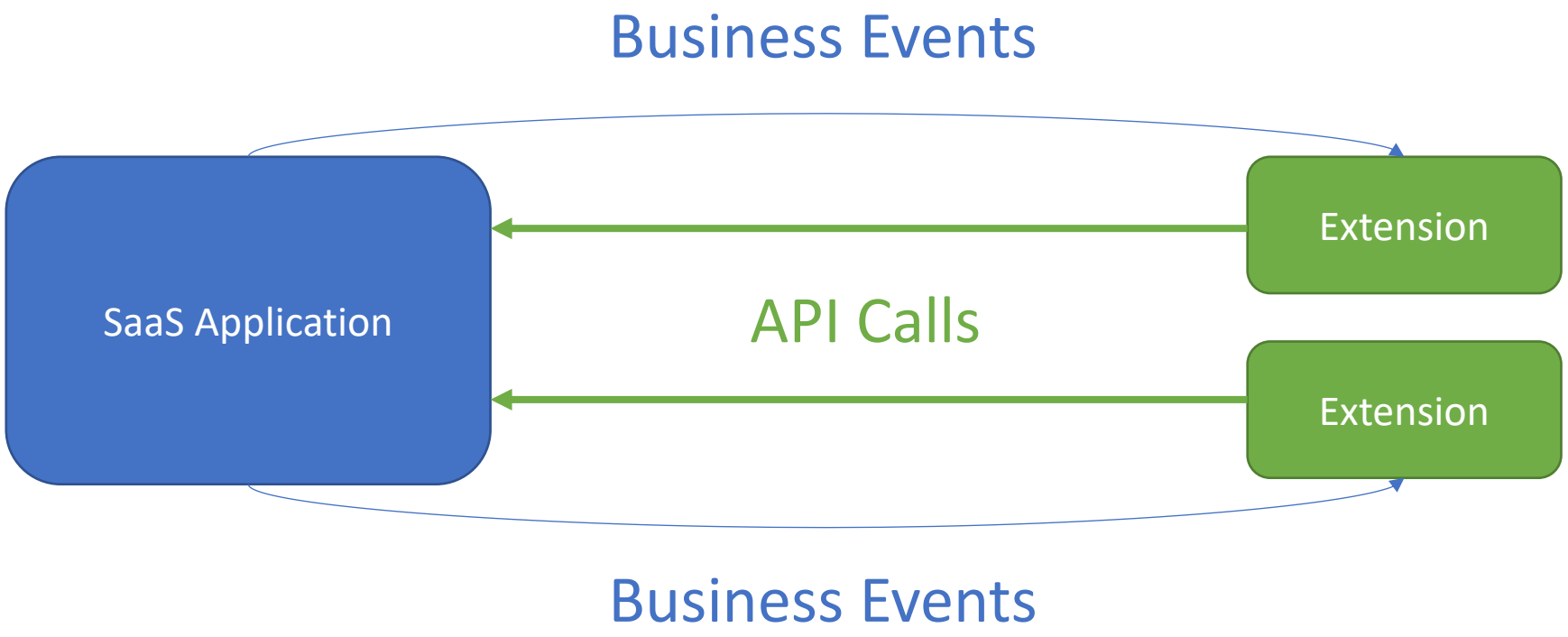

Event Sources
Blob Module
CloudEvents Sources
Custom Events (anything)
Azure IoT Edge
Azure Cloud
Event Grid on IoT Edge
Event Handlers
Other Modules
Custom Modules
Function Modules
Edge Hub
Other Edge Devices
Any WebHook
Event Grid Service
Functions Service
CNCF CloudEvents 1.0 is generally available today on Azure Event Grid and Event Grid for IoT Edge.
CNCF CloudEvents 1.0 is now Azure‘s primary event format and subscribers can receive Grid events from all sources in CloudEvents format.
Blob Storage
Resource Groups
Azure Subscriptions
Event Hubs
Azure Media Service
IoT Hub
Service Bus
App Configuration
Azure Machine Learning
CloudEvents Sources
Custom Events (anything)
Event Handlers
Functions
Service Bus
Logic Apps
Buffering and Competing Consumers
Event Hubs
Storage Queues
Other Services and Applications
Hybrid Connections (WebSockets)
WebHooks (anything)
Azure Automation

SAP

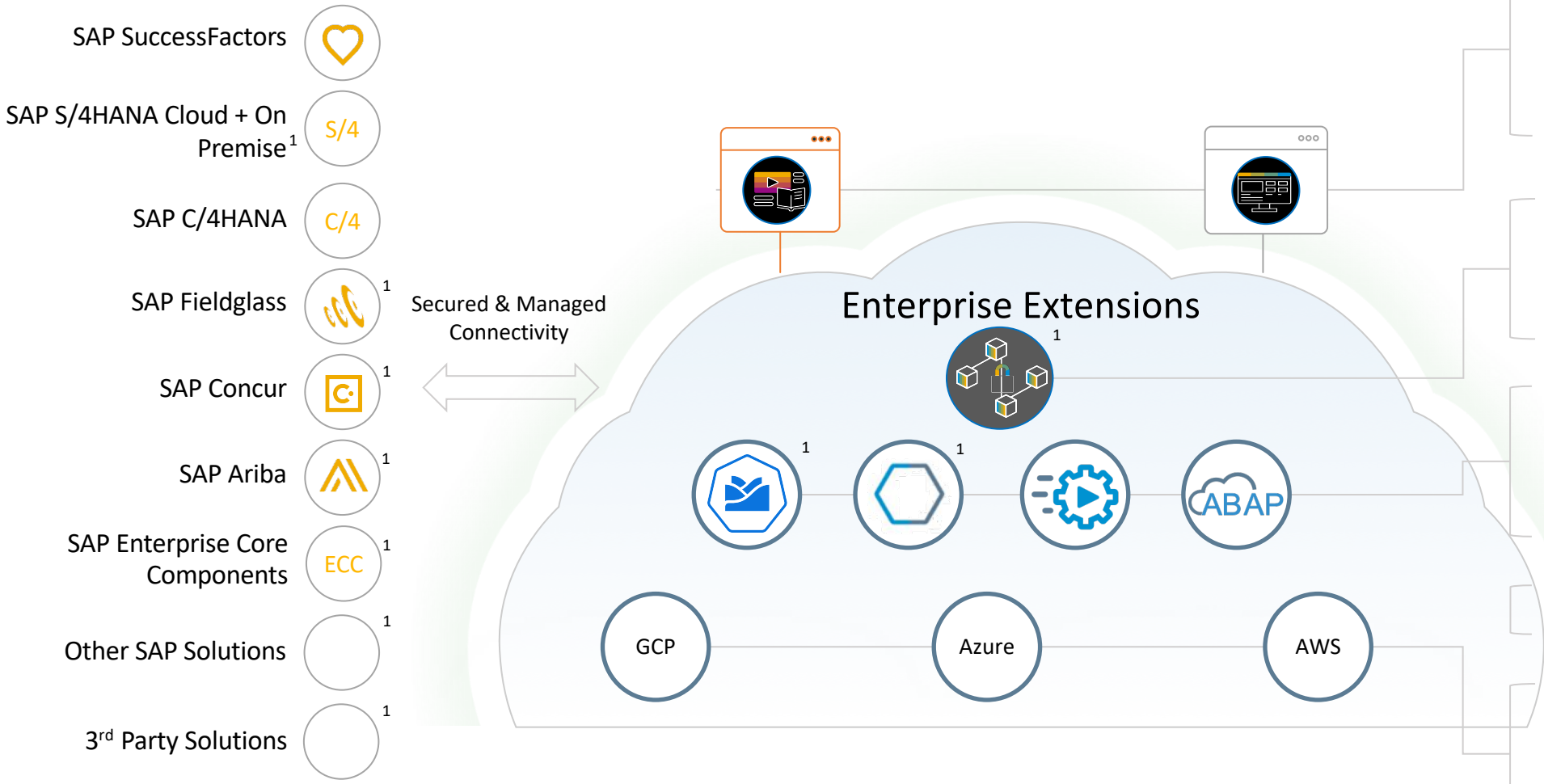
Klaus Deissner

Motivation

Extensibility is crucial for business applications



SAP Cloud Platform Extension Factory



Multiple Administration UIs
- SAP Cloud Platform Cockpit
- SAP C/4HANA Cockpit
- SAP SuccessFactors Extension Center [1]
- …

Central Management Plane
- Secured App Connectivity
- Central Apps APIs & Events Registry
- Standardized Service Mgmt & Consumption

Runtime of choice
- SAP Cloud Platform Kyma runtime
- SAP Cloud Platform Extension Factory, serverless runtime
- Application Runtime
- ABAP

One Event Bus
- SAP Cloud Platform Enterprise Messaging

Infrastructure of choice
- Google Cloud Platform
- Amazon Web Services
- Microsoft Azure
- others

1 Planned innovation

Benefits of using CloudEvents

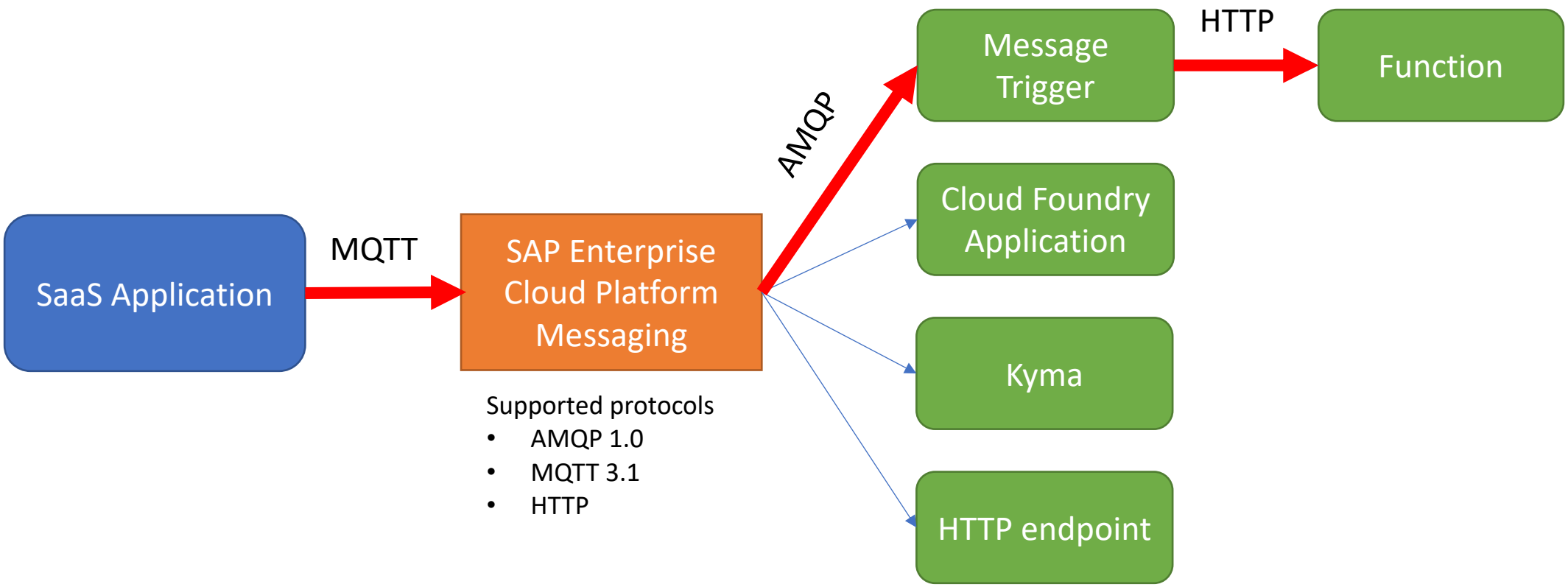
- Interoperability
  - Multiple products originating from multiple companies
  - Infrastructure partners support CloudEvents

- Standardization
  - Development tools can rely on CloudEvents
  - Developers extending SAP solutions may already know CloudEvents
  - Eventing infrastructure

Example – SAP Subscription Billing


```
{
   "specversion" : "1.0",
   "type" : "sap.billing.sb.subscription.created.v1",
   "source" : "urn:sap:topicNS:sap/billing/sb",
   "id" : "2EBDF073-19F1-4D46-9050-FF8CB4AFEE30",
   "time" : "2019-11-07T13:22:48.093Z",
   "datacontenttype" : "application/json",
   "data" : {
     "subscriptionId" : "231352e6-7474-4d56-b06c-3707fdd34061",
     "eventType" : "created",
     "eventLogEntryId" : "d16e2ebd-1268-48ec-8513-a2488bbb92ac",
     "changedSubscriptionAspects" : [ ]
   }
}
```

SAP Cloud Platform Enterprise Messaging

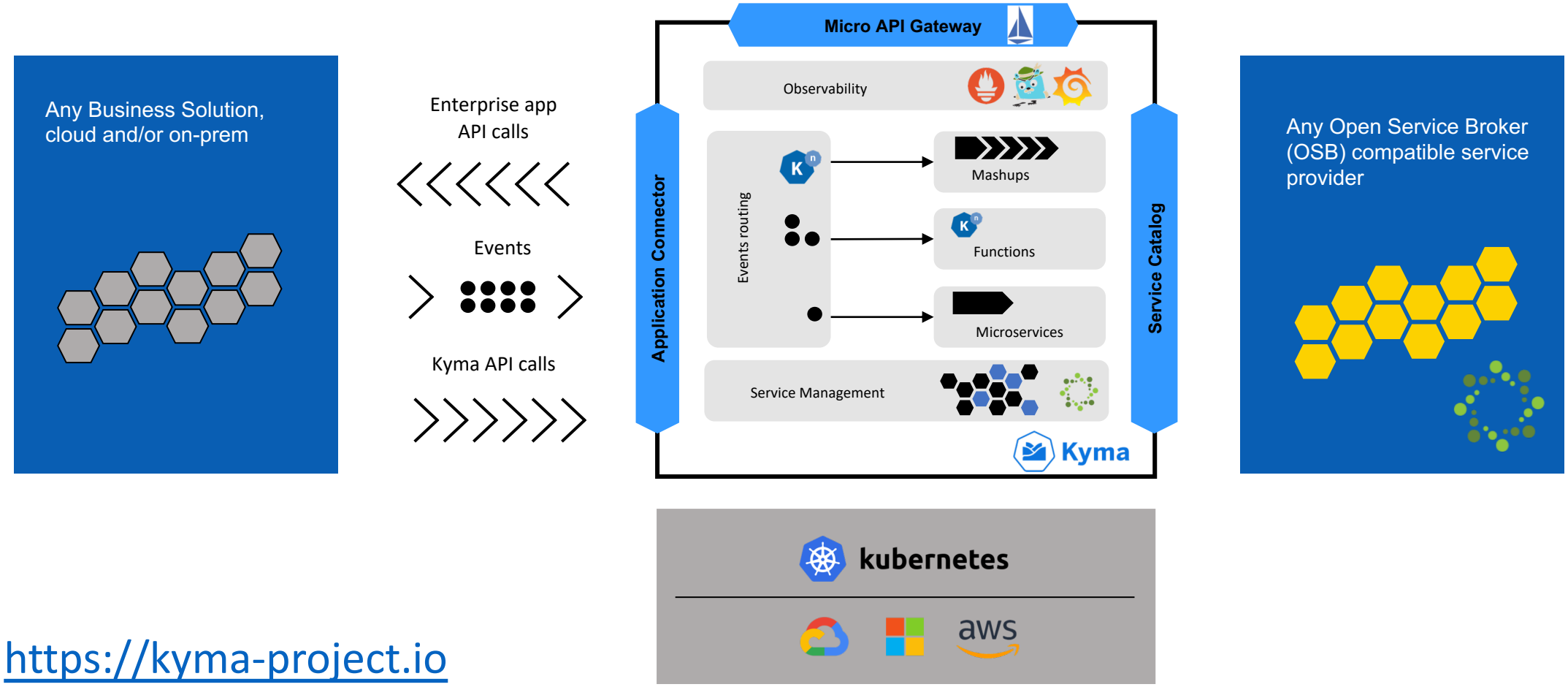


CloudEvents in the serverless runtime


```
module.exports = function (event, context) {
    switch (event.ce.type) {
        case 'Reset':
            ...
        case 'Disconnect':
            ...
        case 'Order.OrderStatus.OrderReleased':
            ...
        case 'Offer.Product':
            if (event.ce.source === 'Controller' &&
                event.ce.subject === me) {
                connState.update(event);
            } else if (!connState.isRegistered()) {
                connState.resetState();
                registerSupplier();
            }
            break;
    }
}
```

```
function registerSupplier() {
    event.sendResponseEvent({
        id: uuid(),
        source: me,
        time: Date.now(),
        type: "Connection",
        data: {
            system: me,
            organization: meOrg,
            logo: meLogo
        }
    });
}
```

Kyma



https://kyma-project.io

Conclusion

- Routing CloudEvents with existing messaging infrastructure is hard
- Think carefully
  - what `source` means in your organization
  - what `type` means in your organization
- JMS and CloudEvents is not a good fit

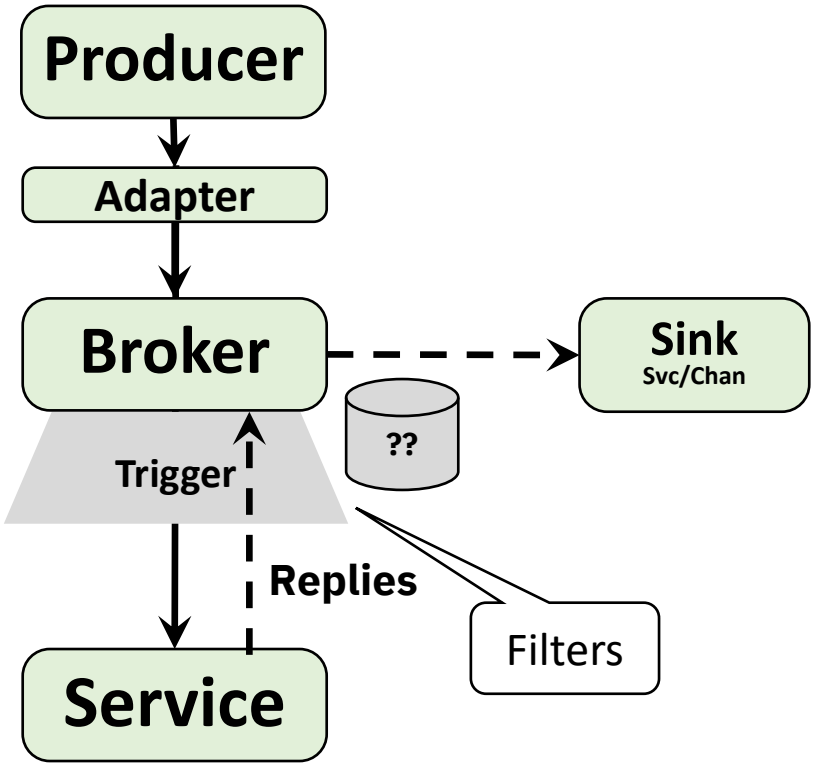
- Interested in future activities
  - Subscription API and protocol
  - Event catalogs

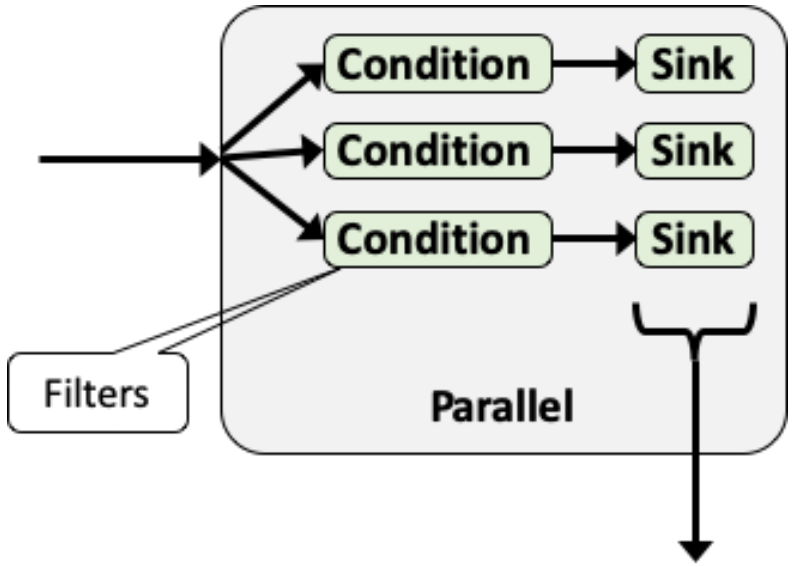
Knative / IBM

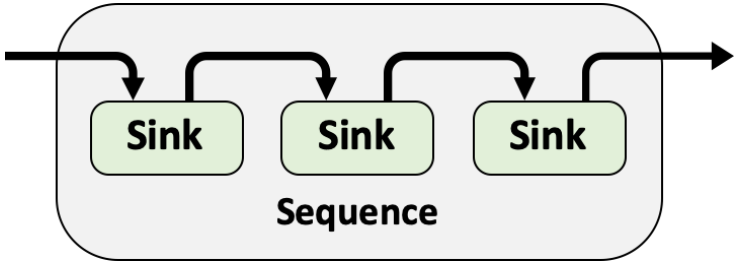
Doug Davis

Knative / IBM

- Eventing Model leverages CloudEvents
  - Events are normalized into CloudEvents and passed around via HTTP
  - Enables filtering/routing without the need to understand/parse the business logic
  - Mix-n-match eventing components to build an "eventing workflow"





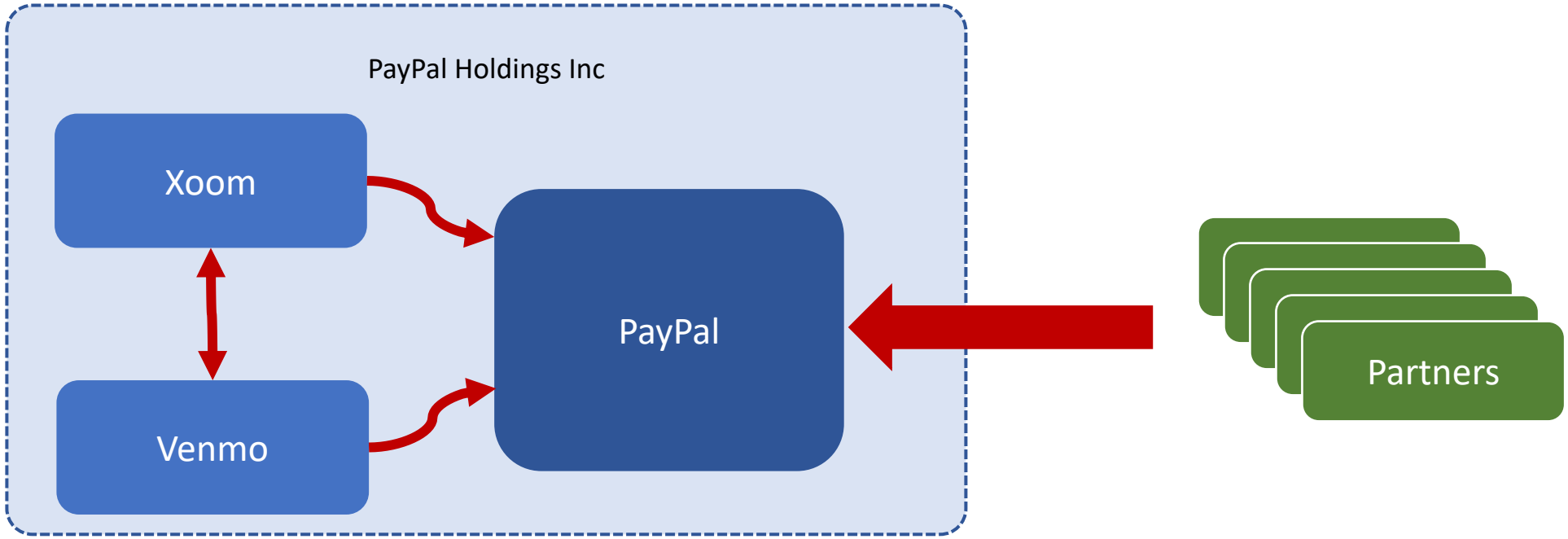


IBM includes Knative as an IBM Cloud Kubernetes Service add-on

PayPal

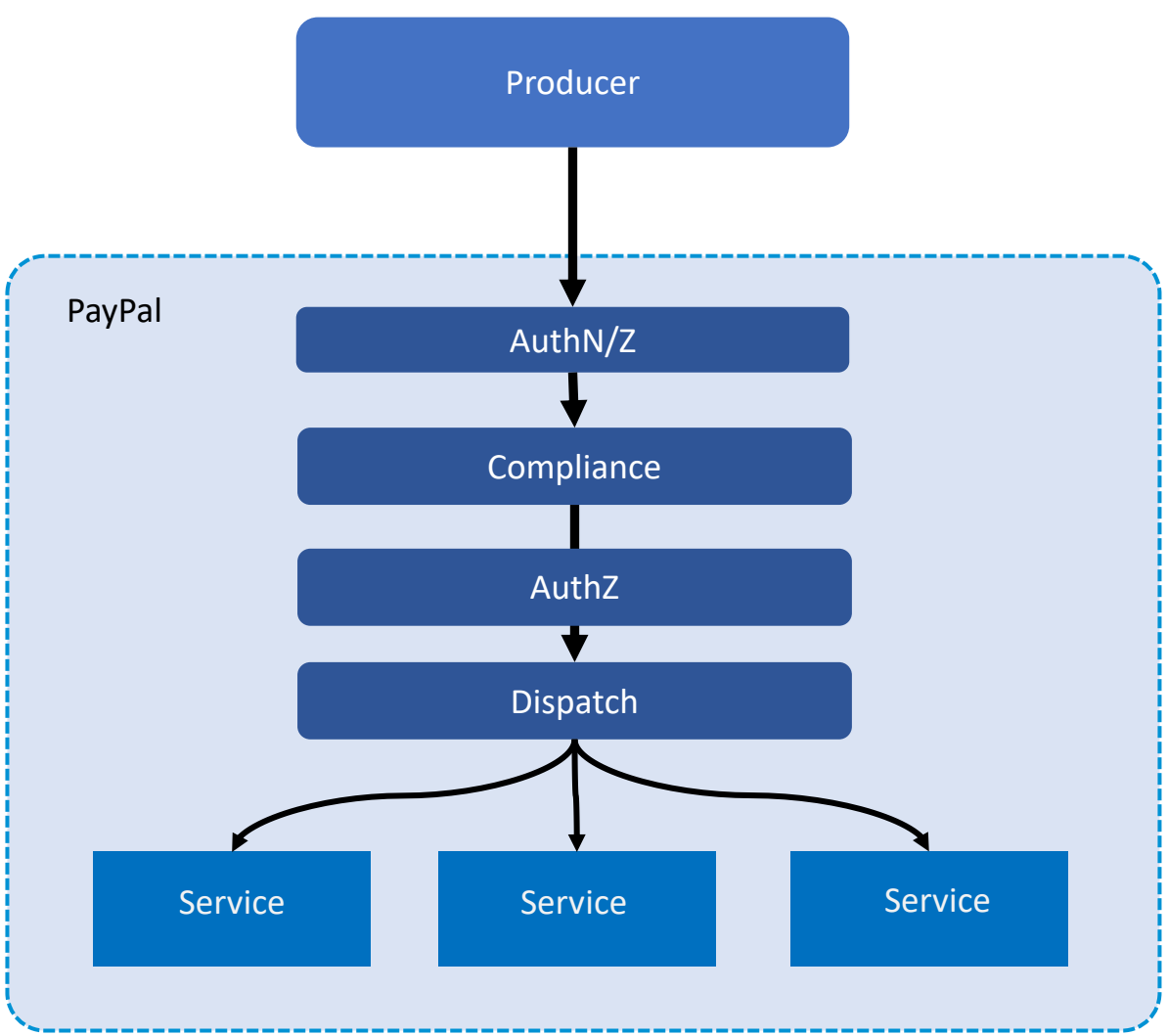
Vladimir Bacvanski

PayPal CloudEvents Adoption



- Live for some inter-brand notifications
- Gradual adoption with some strategic partners

PayPal Event Gateway – Logical Architecture



HTTP

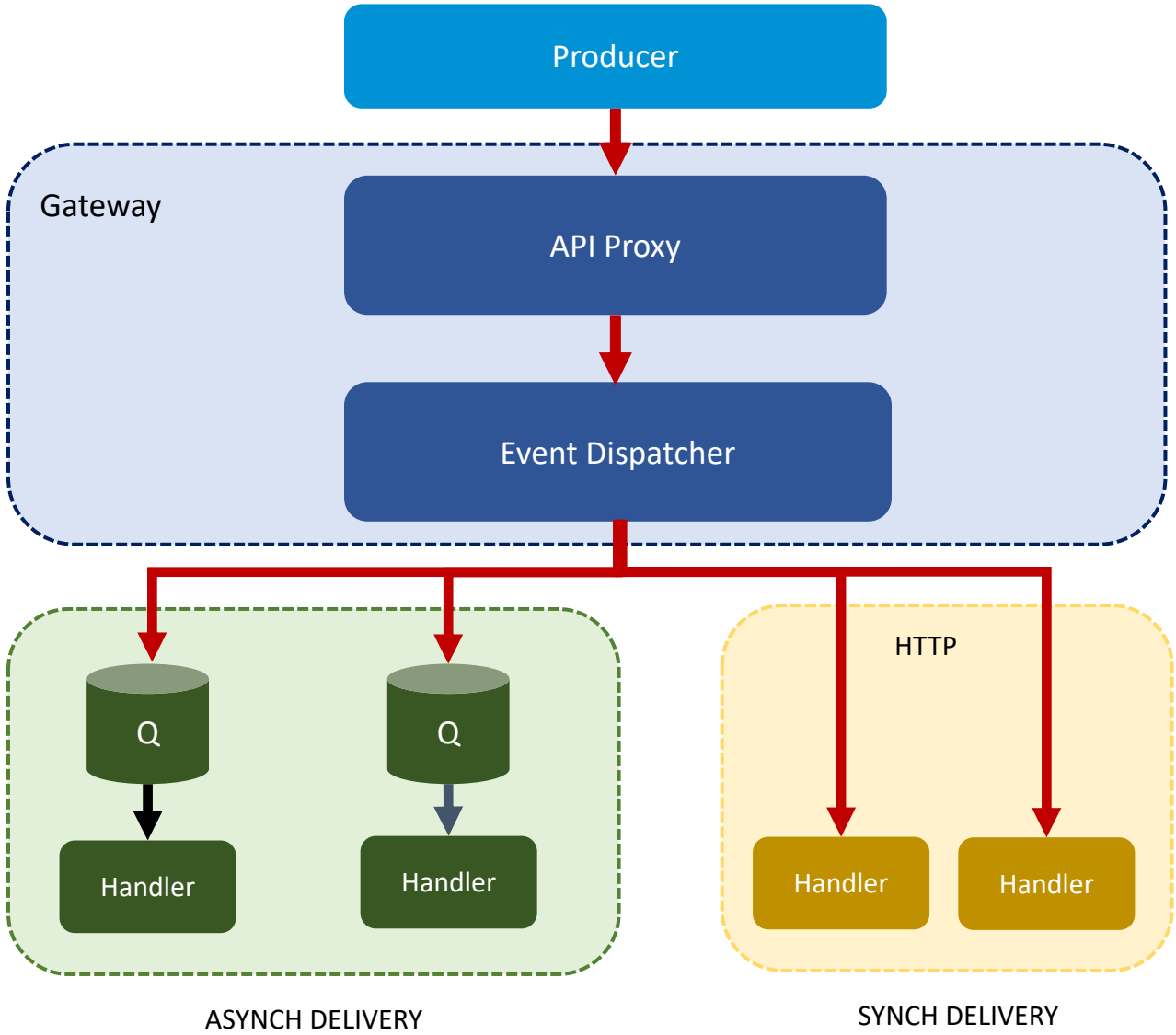
Endpoint AuthN/AuthZ

Well-Formed check and potentially finer-grained AuthZ

Rule based Dispatch

Event Handler

PayPal Event Gateway Architecture



- https://XXX.paypal.com/v1/notification/events

- Existing API endpoint AuthN/Z and routing

- Well-Formedness check
- Context based AuthN
- Endpoint Selection
- Transport specific dispatch

- Business Logic

Thank You!

- CloudEvents : https://cloudevents.io/
  - Org : https://github.com/cloudevents
  - Spec repo : https://github.com/cloudevents/spec
  - SDKs : https://github.com/cloudevents/sdk-...

- Weekly calls : Thursdays at 12pm ET

- Questions?