Baltic Microsoft Developers Community

Queues in Azure: what, why and how

Aleksandr Surkov

Senior Cloud Solution Architect VISEO, France

Microsoft MVP, RD, MCT

What will we talk about?

Queue theory

- What is a queue?
- Delivery types (FIFO, At most once, At least once, Exactly once)
- Real delivery process (peek data, delete data and so on)

Queues in Azure

- Messages vs Events
- Azure queues overview
- Azure queues comparison

Demo – queues in IoT scenario

- Using EventHub and EventGrid in IoT Scenario

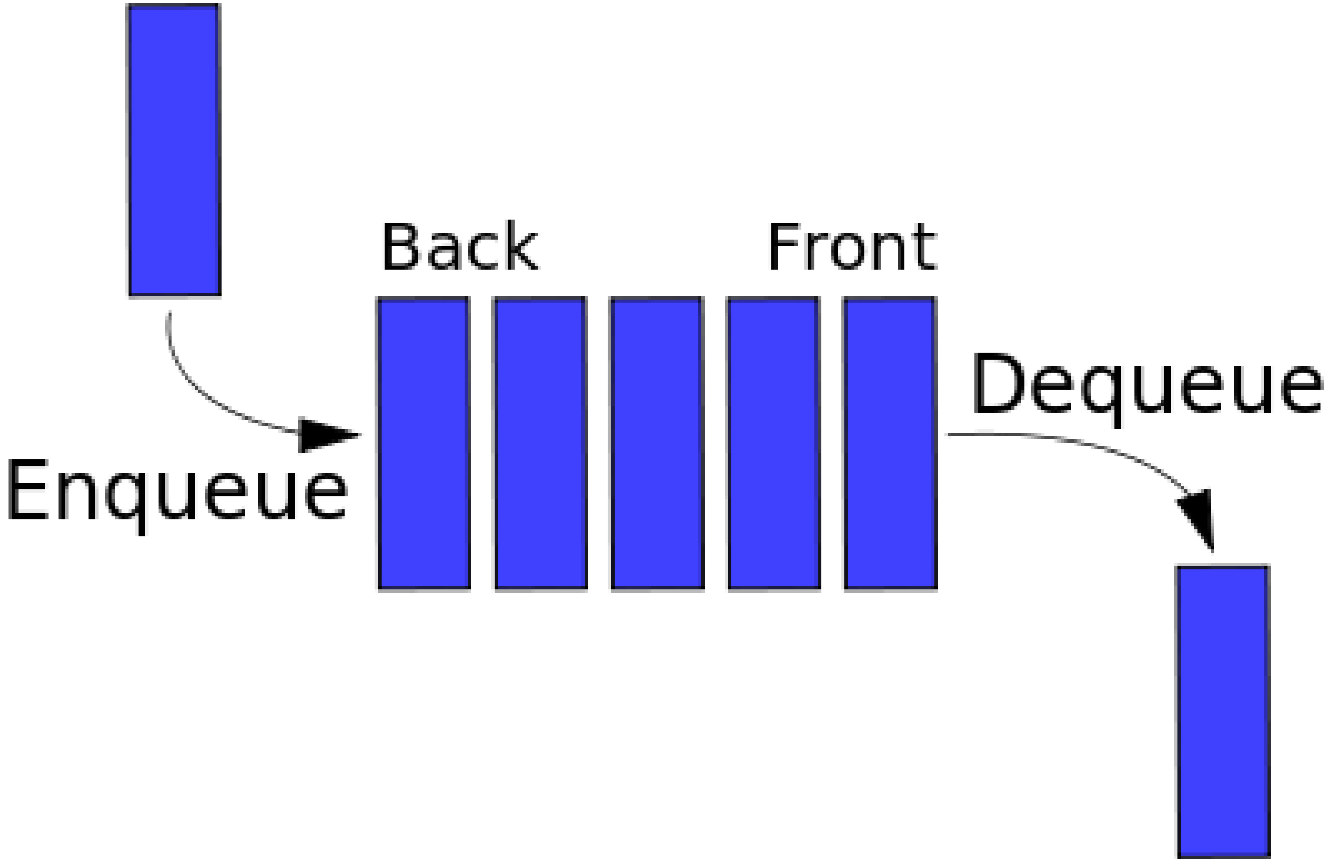
About me

- 10+ years of development experience
- Managed creation of IoT service in Yandex.Cloud from scratch
- Certified Cloud Solution Architect
- Microsoft Most Valuable Professional
- Microsoft Regional Director
- Microsoft Certified Trainer

What is a queue

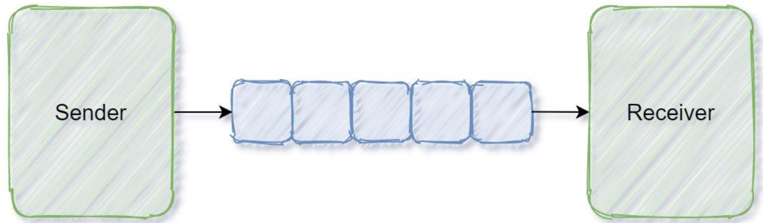
In computer science, a **queue** is a collection of entities that are maintained in a sequence and can be modified by the addition of entities at one end of the sequence and the removal of entities from the other end of the sequence.

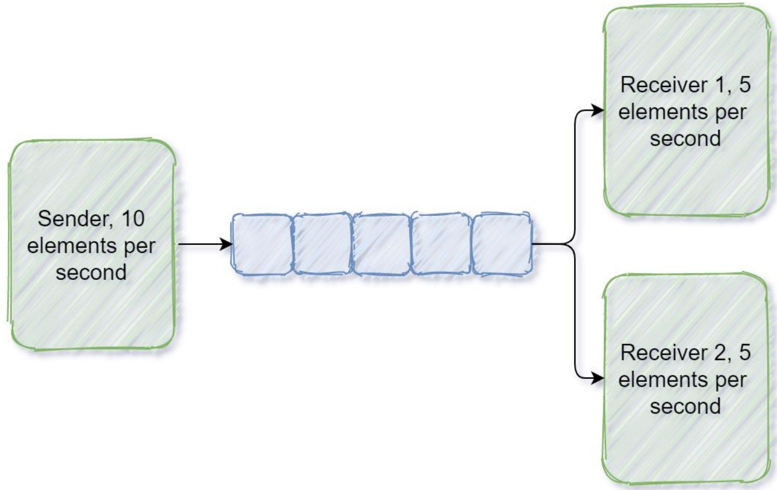
Wikipedia

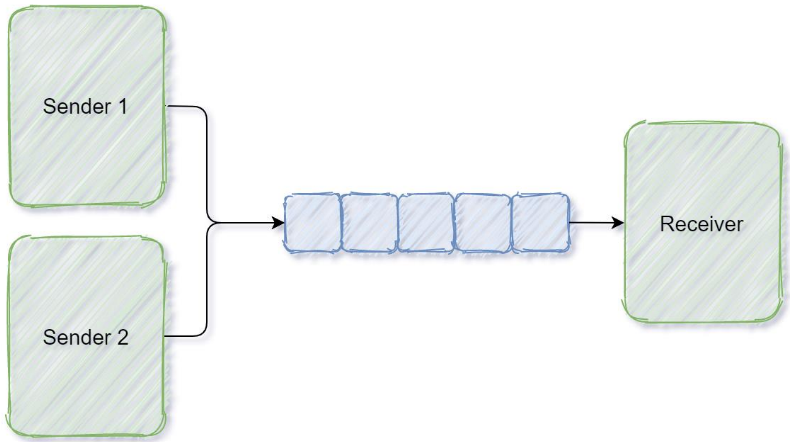


Why to use queues?

- **Better Performance**
  - Message queues enable asynchronous communication, which means that the endpoints that are producing and consuming messages interact with the queue, not each other
- **Increased Reliability**
  - Queues make your data persistent and reduce the errors that happen when different parts of your system go offline.
- **Granular Scalability**
  - Message queues make it possible to scale precisely where you need to.
- **Simplifed Decoupling**
  - Message queues remove dependencies between components and significantly simplify the coding of decoupled applications.

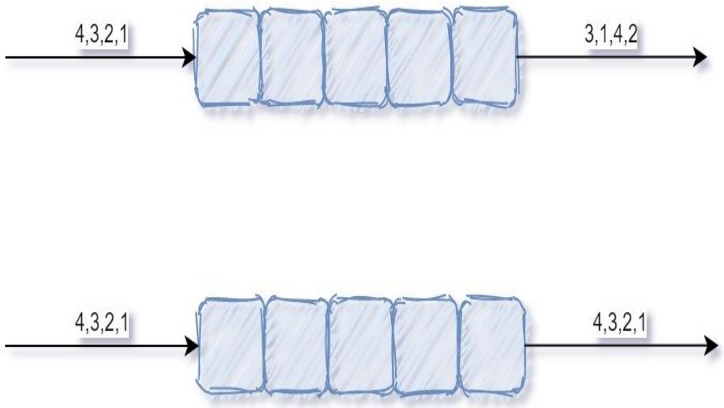



Sender, 10 elements per second
Receiver 1, 5 elements per second
Receiver 2, 5 elements per second



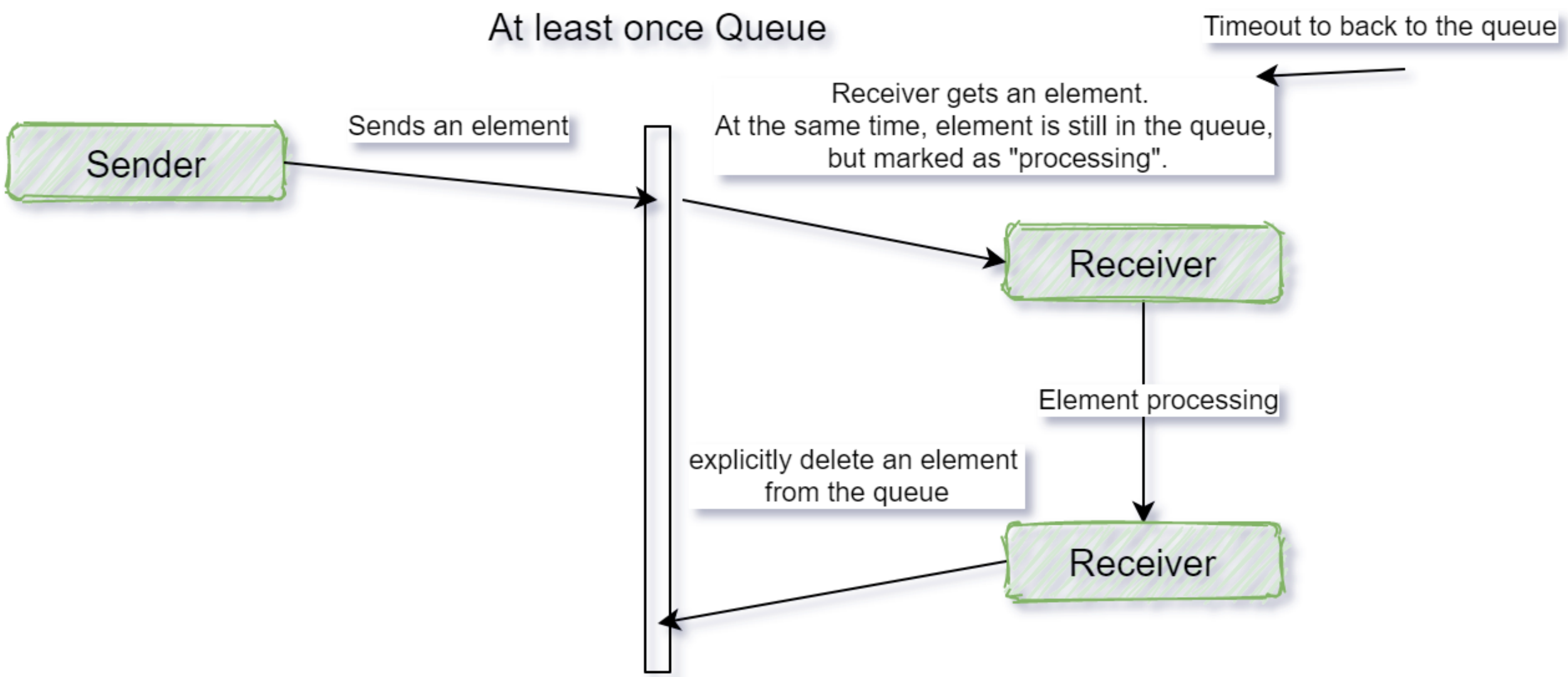
Queue types

- **At-most once**
  - Once delivered, there is no chance of delivering again. If the consumer is unable to handle the message due to some exception, the message is lost.
- **At-least once**
  - message will be delivered at least once. There is high chance that message will be delivered again as duplicate.
- **Exactly-once**
  - there will be only one and once message delivery. It difficult to achieve in practice.



FIFO queue

Example of the queue


At least once Queue
Timeout to back to the queue
Receiver gets an element.
At the same time, element is still in the queue,
but marked as "processing".
Sends an element
Sender
Receiver
Element processing
explicitly delete an element
from the queue
Receiver

The most known Queues

- The **Advanced Message Queuing Protocol** (**AMQP**) is an open standard application layer protocol for message-oriented middleware. The defining features of AMQP are message orientation, queuing, routing (including point-to-point and publish-and-subscribe), reliability and security.
- **MQTT** (**Message Queuing Telemetry Transport**) is an open lightweight, publish-subscribe network protocol that transports messages between devices. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited.
- **Apache Kafka** is a framework implementation of a software bus using stream-processing. It is an open-source software platform developed by the Apache Software Foundation. The project aims to provide a unified, high-throughput, low-latency platform for handling real-time data feeds.

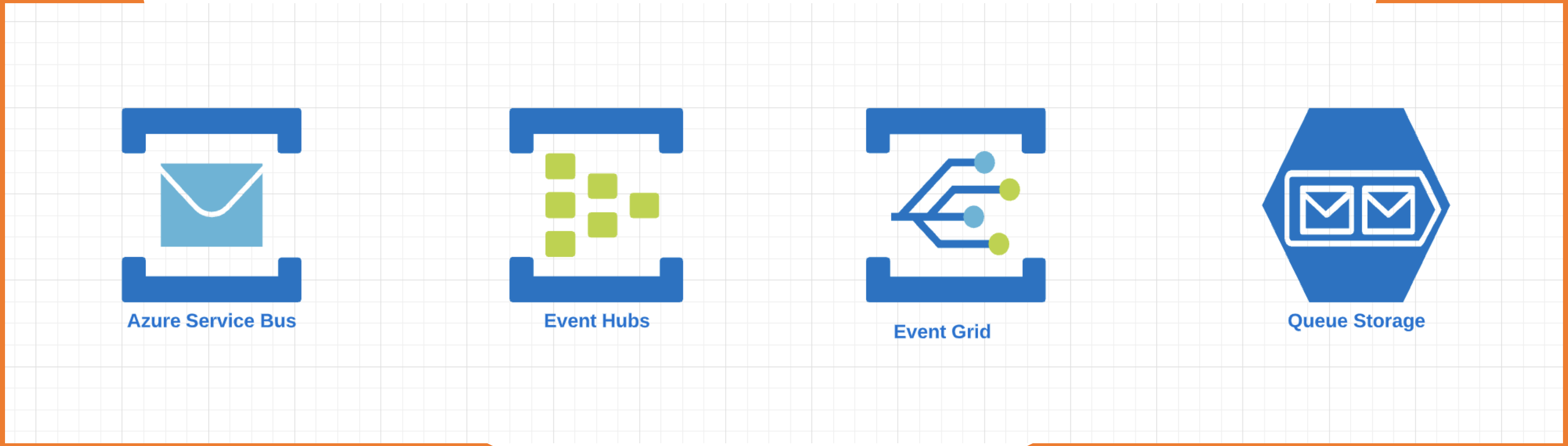
The difference

- **AMQP** – is standard with many realizations (RabbitMQ, StormMQ…). It is more complex but has more features
- **MQTT** – is standard with many different realizations (Mosquitto, JoramMQ...). It is simple and lightweight
- **Kafka** – is a software (may be even service), that not only sends data, but also store a lot of data inside the queue and can be scaled as a service to support highload.

All of them use brokers.

AMQP vs MQTT: https://www.educba.com/amqp-vs-mqtt/

Queues in Azure



https://medium.com/slalom-technology/azure-messaging-when-to-use-what-and-why-post-2-81d164cc838e

Event vs Message

Event

- An event is a lightweight notification of a condition or a state change. The publisher of the event has no expectation about how the event is handled. The consumer of the event decides what to do with the notification.

Message

- A message is raw data produced by a service to be consumed or stored elsewhere. The publisher of the message has an expectation about how the consumer handles the message. A contract exists between the two sides. For example, the publisher sends a message with the raw data, and expects the consumer to create a file from that data and send a response when the work is done.

https://docs.microsoft.com/en-us/azure/event-grid/compare-messaging-services

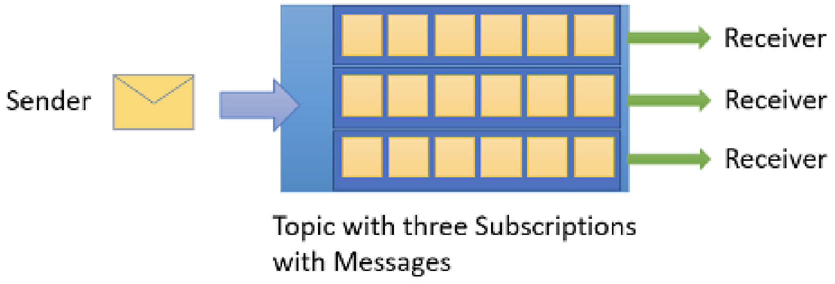
Azure Event and Messaging Services

Service	Purpose	Type	When to use
Event Grid	Reactive programming	Event distribution (discrete)	React to status changes
Event Hubs	Big data pipeline	Event streaming (series)	Telemetry and distributed data streaming
Service Bus	High-value enterprise messaging	Message	Order processing and financial transactions
Azure Storage Queues	Simple, reliable, persistent messaging within and between services	Message	Very large message stores (over 80 GB), unreliable consumers

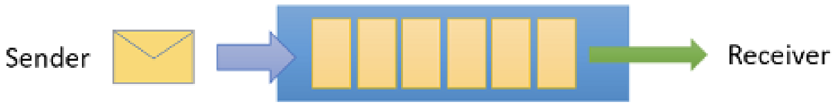
Comparing cloud event and messaging options

Requirement	Simple queuing	Eventing and PubSub	Big data streaming	Enterprise messaging
Product	Queue storage	Event Grid	Event Hubs	Service Bus
Supported advantages	• Communication within an app • Individual message • Queue semantics / polling buffer • Simple and easy to use • Pay as you go	• Communication between apps / orgs • Individual message • Push semantics • Filtering and routing • Pay as you go • Fan out	• Many messages in a Stream (think in MBs) • Ease of use and operation • Low cost • Fan in • Strict ordering • Works with other tools	• Instantaneous consistency • Strict ordering • Java Messaging Service • Non-repudiation and security • Geo-replication and availability • Rich features (such as deduplication and scheduling)
Weaknesses	• Ordering of messaging • Instantaneous consistency	• Ordering of messaging • Instantaneous consistency	• Server-side cursor • Only once	• Cost • Simplicity
Type	Serverless	Serverless	Big data	Enterprise

Azure Service Bus



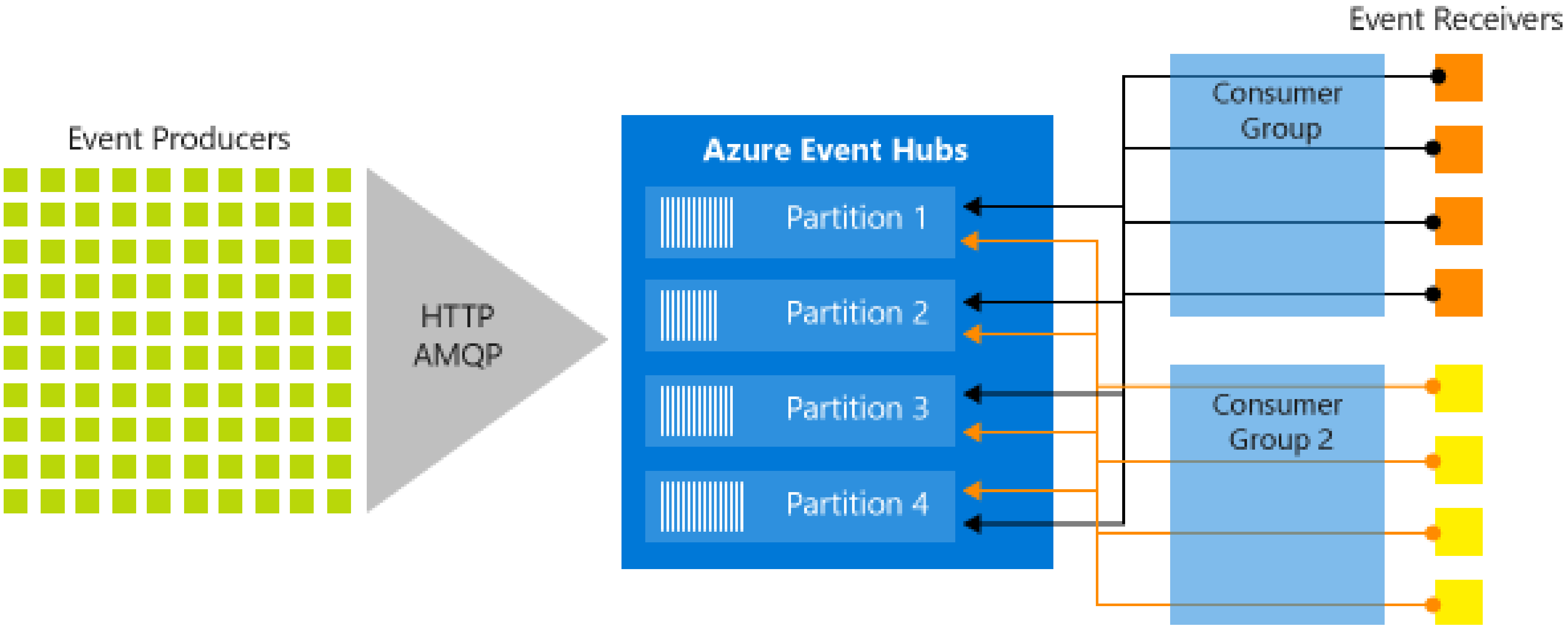
Topic with three Subscriptions with Messages



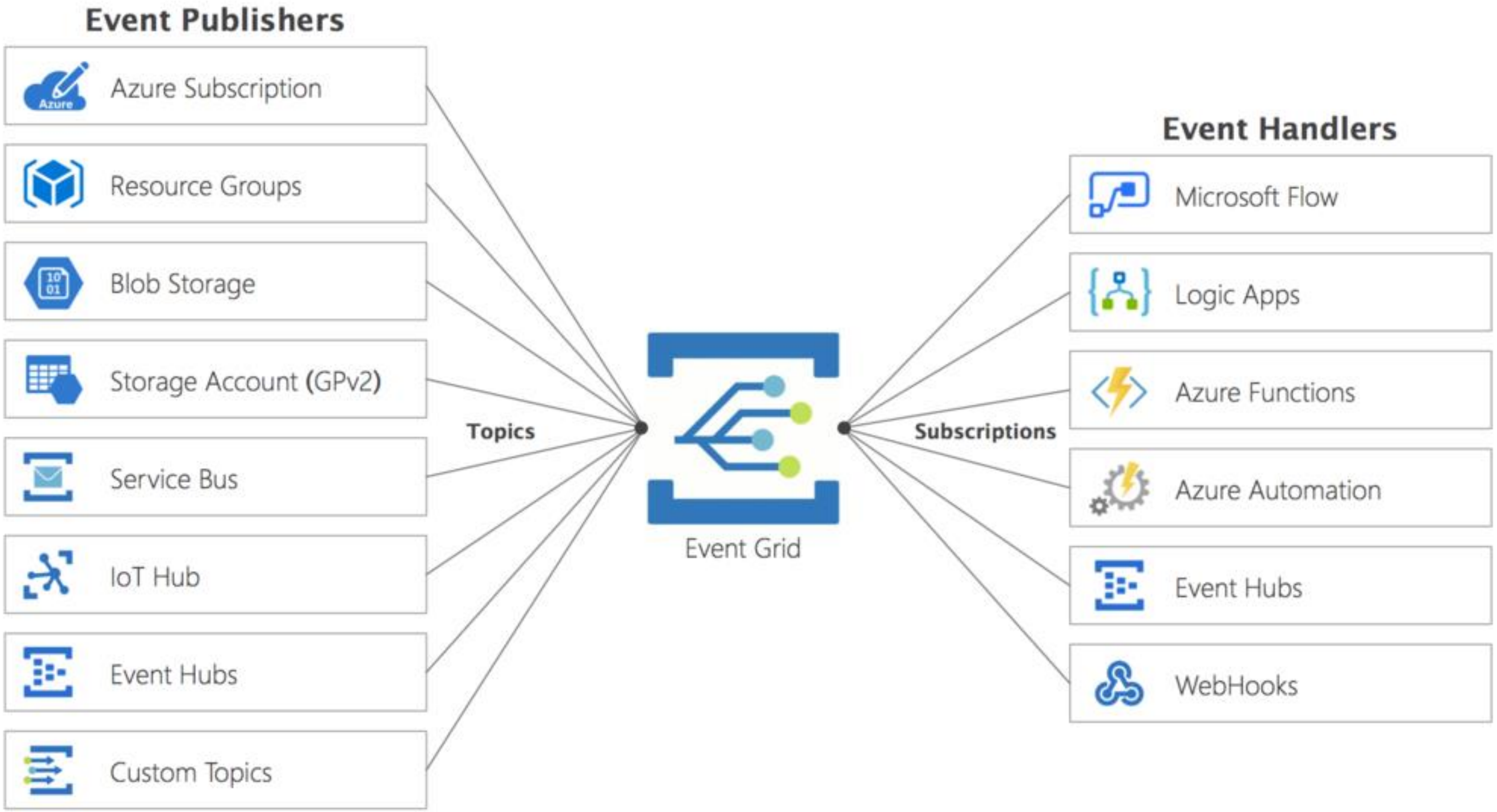
Message Queue with Messages

- **Dead Letter Queue:** Azure Service Bus creates DLQ sub-queue to hold messages that could not be delivered or processed.
- **Consumption Mode:** Azure Service Bus supports several consumption modes: pub/sub with a pull model, competing consumers, and partitioning can be achieved with the use of topics, subscriptions, and actions.
- **Duplicate Detection:** Azure Service Bus is the only message broker that supports duplicate detection natively.
- **Delivery Guarantee:** Azure Service Bus supports three delivery guarantees: At-least-once, At-most-once, and Effectively once.
- **Message Ordering:** Azure Service Bus can guarantee first-in-first-out using sessions.

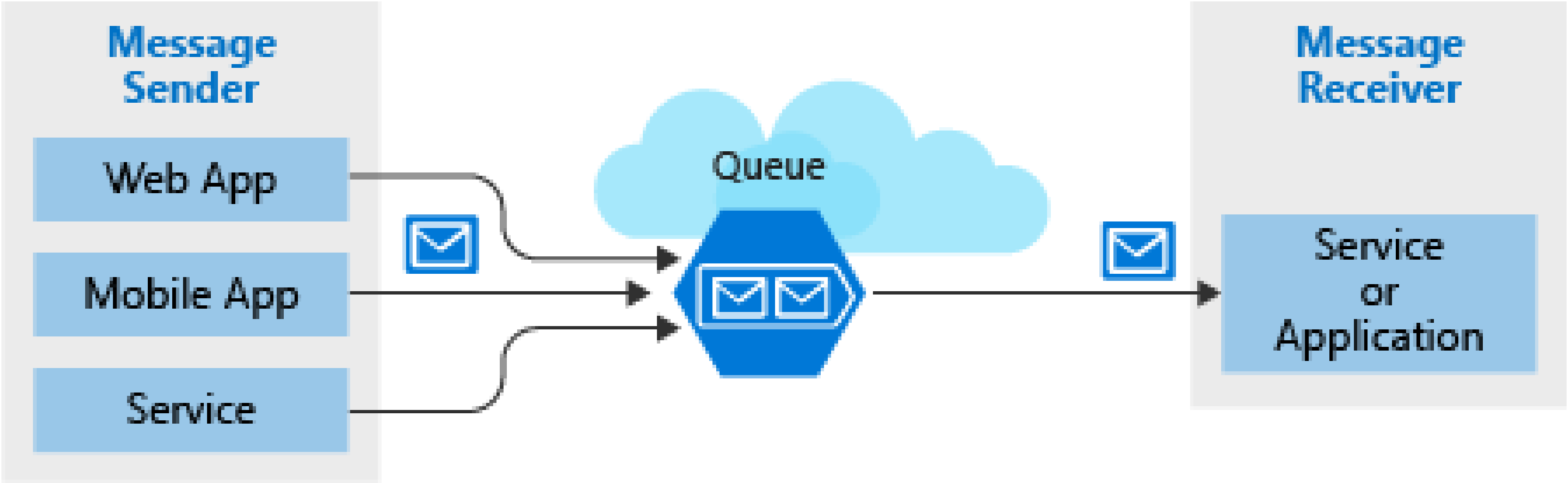
Azure Event Hubs



Azure Event Grid



Azure Queue Storage

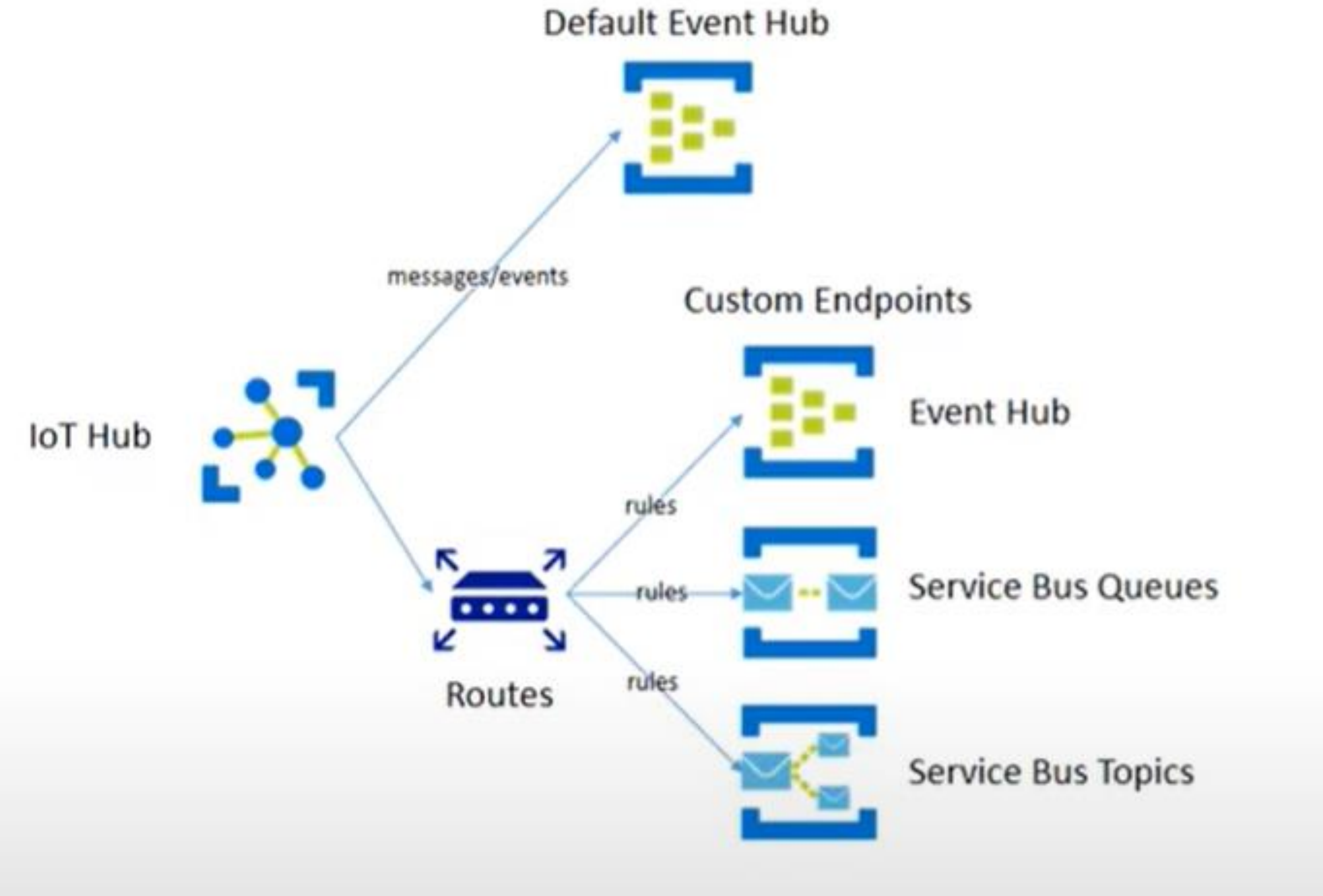


HTTP/S only. General ability is to store large amount of data

Messaging example: IoT Hub

Azure IoT Hub

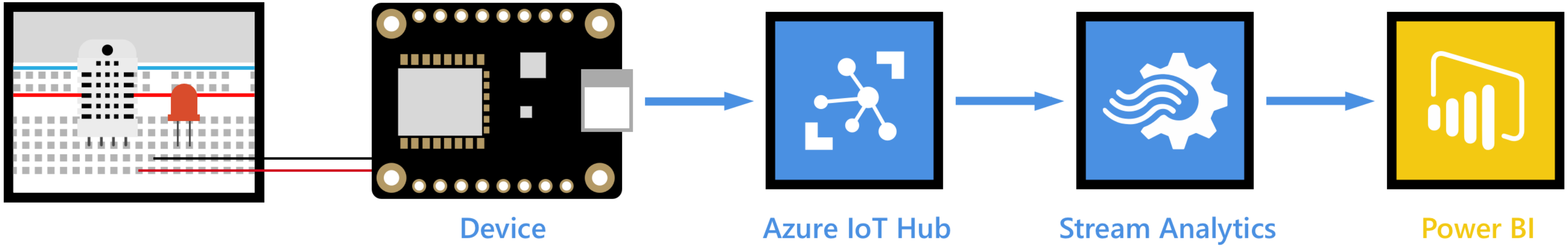
- MQTT - Use on all devices that do not require to connect multiple devices (each with its own per-device credentials) over the same TLS connection.
- AMQP - Use on field and cloud gateways to take advantage of connection multiplexing across devices.
- HTTPS - Use for devices that cannot support other protocols.



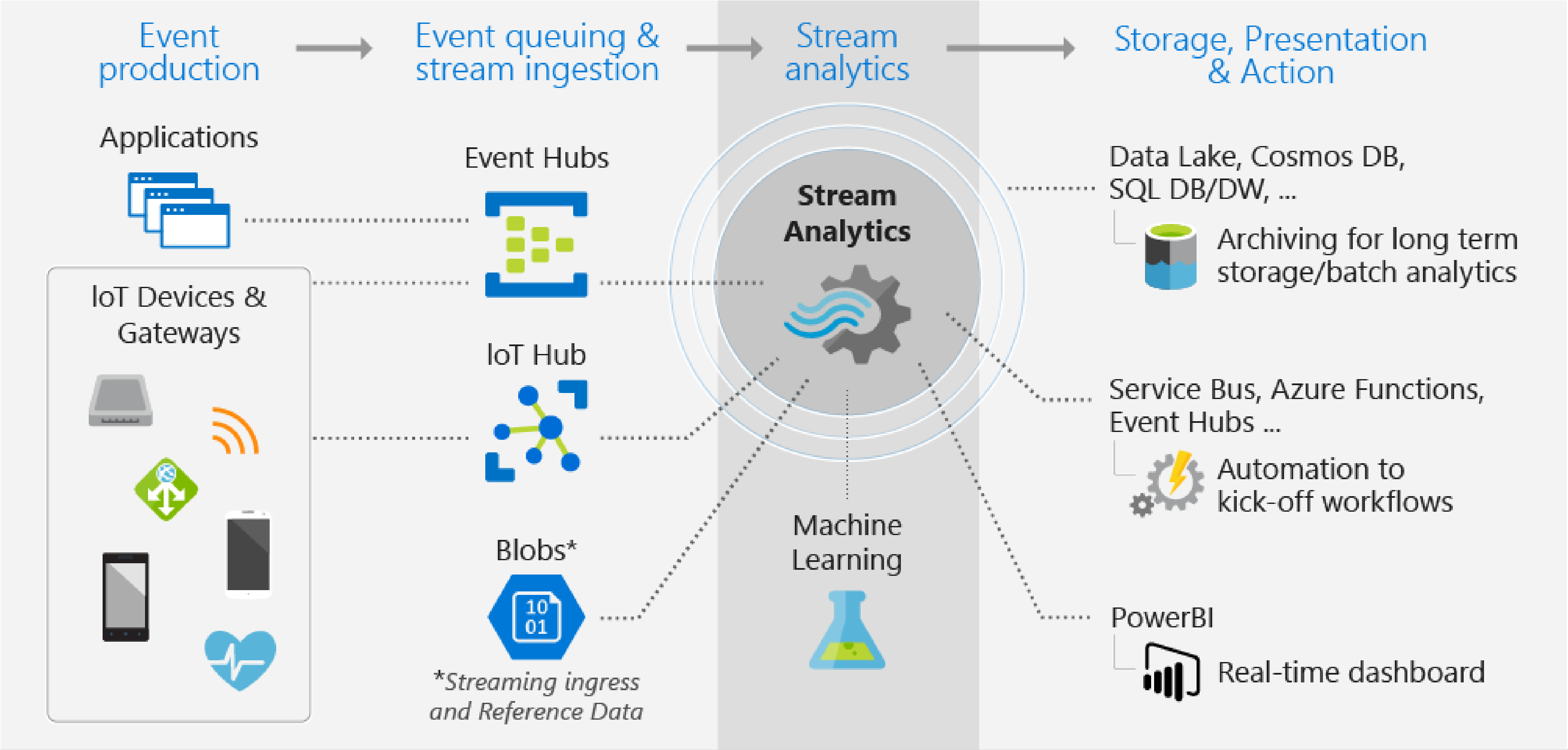
IoT Hub: Message Routing or Event Grid Integration

Decision Point	Message Routing	Event Grid
Data type	Telemetry data, potentially with enhancement	Filtered unenhanced telemetry data and device lifecycle events
Next step endpoint	Limited number built-in with support for some connectors	Azure Functions, Logic Apps (next slide), Storage Accounts, Service Bus queues, webhooks
Ordering	Ordering maintained	Ordering not guaranteed

Demo

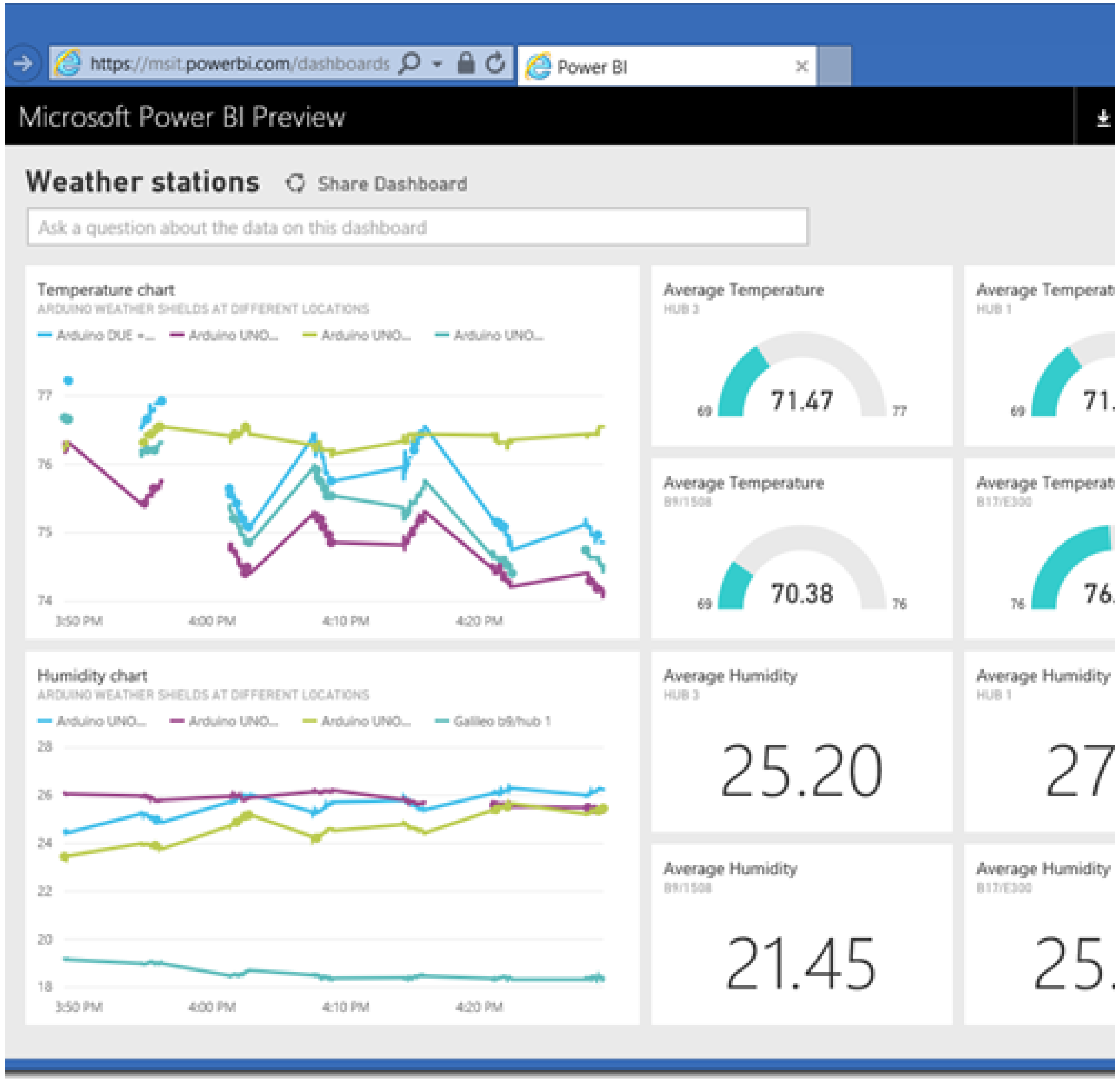


What is Azure Stream Analytics


Event production
Event queuing & stream ingestion
Stream analytics
Storage, Presentation & Action
Applications
IoT Devices & Gateways
Event Hubs
IoT Hub
Blobs*
*Streaming ingress and Reference Data
Stream Analytics
Machine Learning
Data Lake, Cosmos DB, SQL DB/DW, …
Archiving for long term storage/batch analytics
Service Bus, Azure Functions, Event Hubs …
Automation to kick-off workflows
PowerBI
Real-time dashboard

What is Power BI

- Power BI can:
    - Connect to data sources
    - Transform and append data
    - Create visual representations of data
    - Create reports
    - Create dashboards

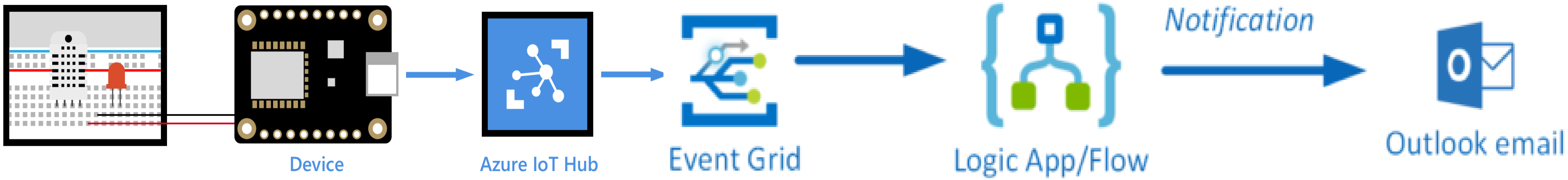


Links

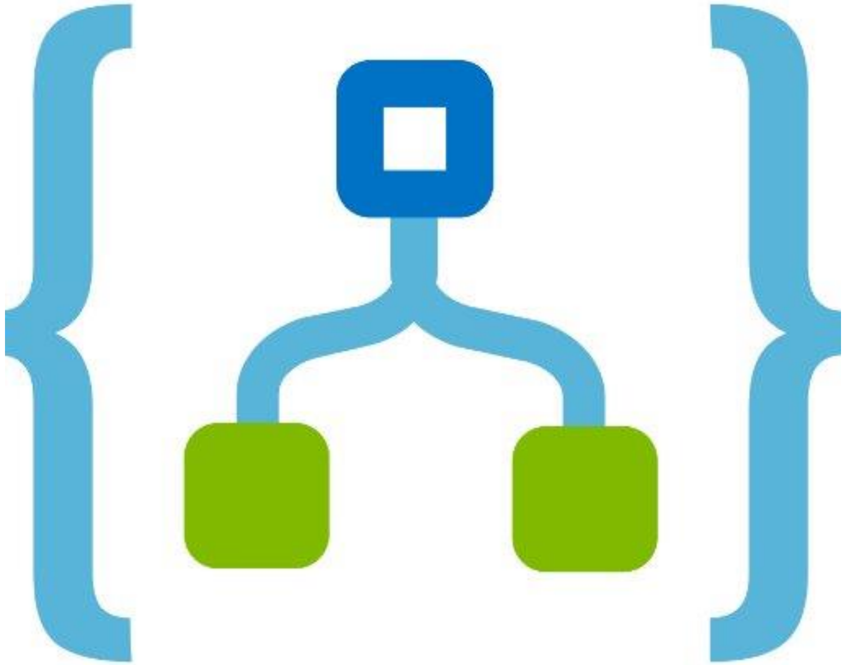
- Raspberry PI Simulator: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-raspberry-pi-web-simulator-get-started

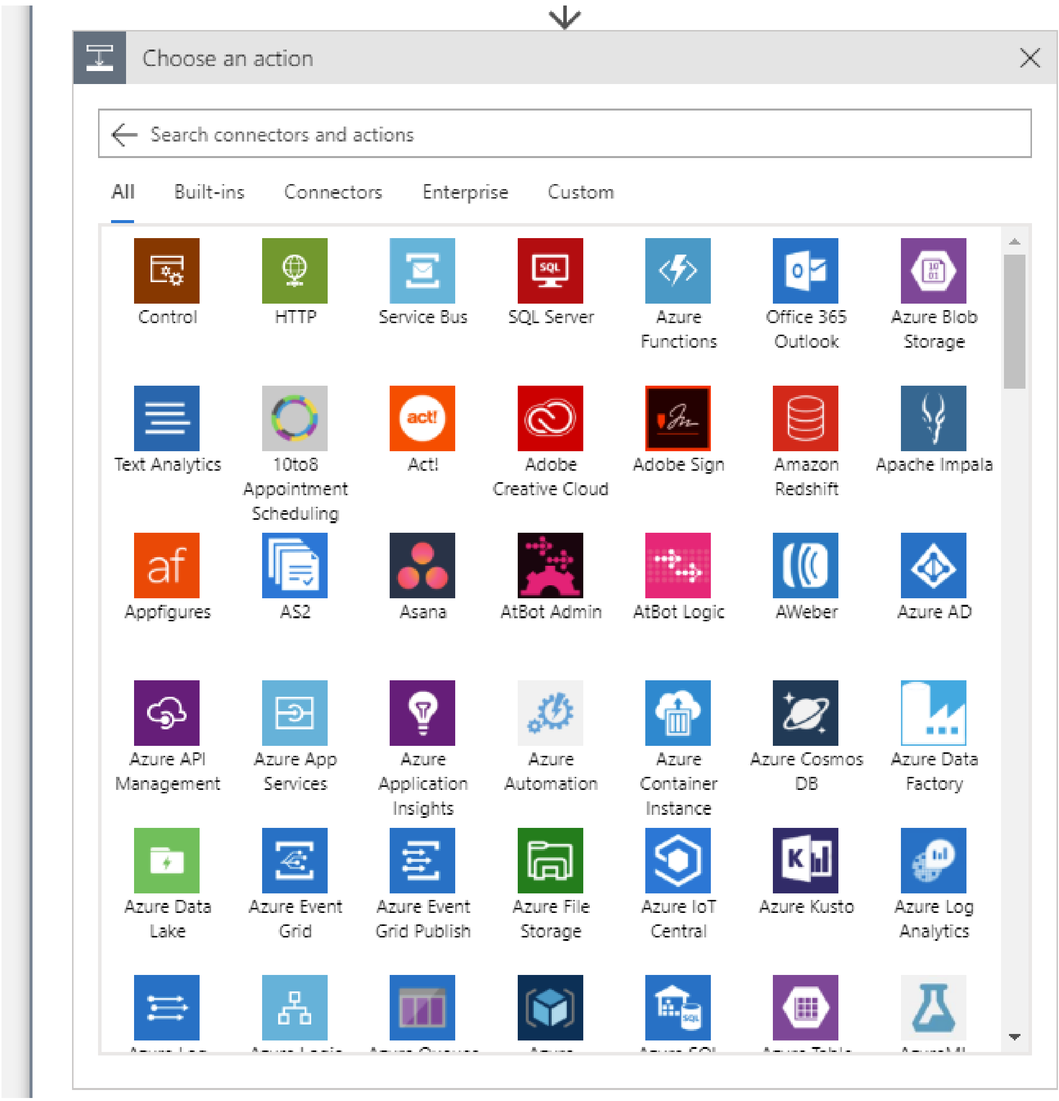
- Power BI demo: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-live-data-visualization-in-power-bi

Demo


Device
Azure IoT Hub
Event Grid
Logic App/Flow
Notification
Outlook email

Azure Logic App





Choose an action

Search connectors and actions

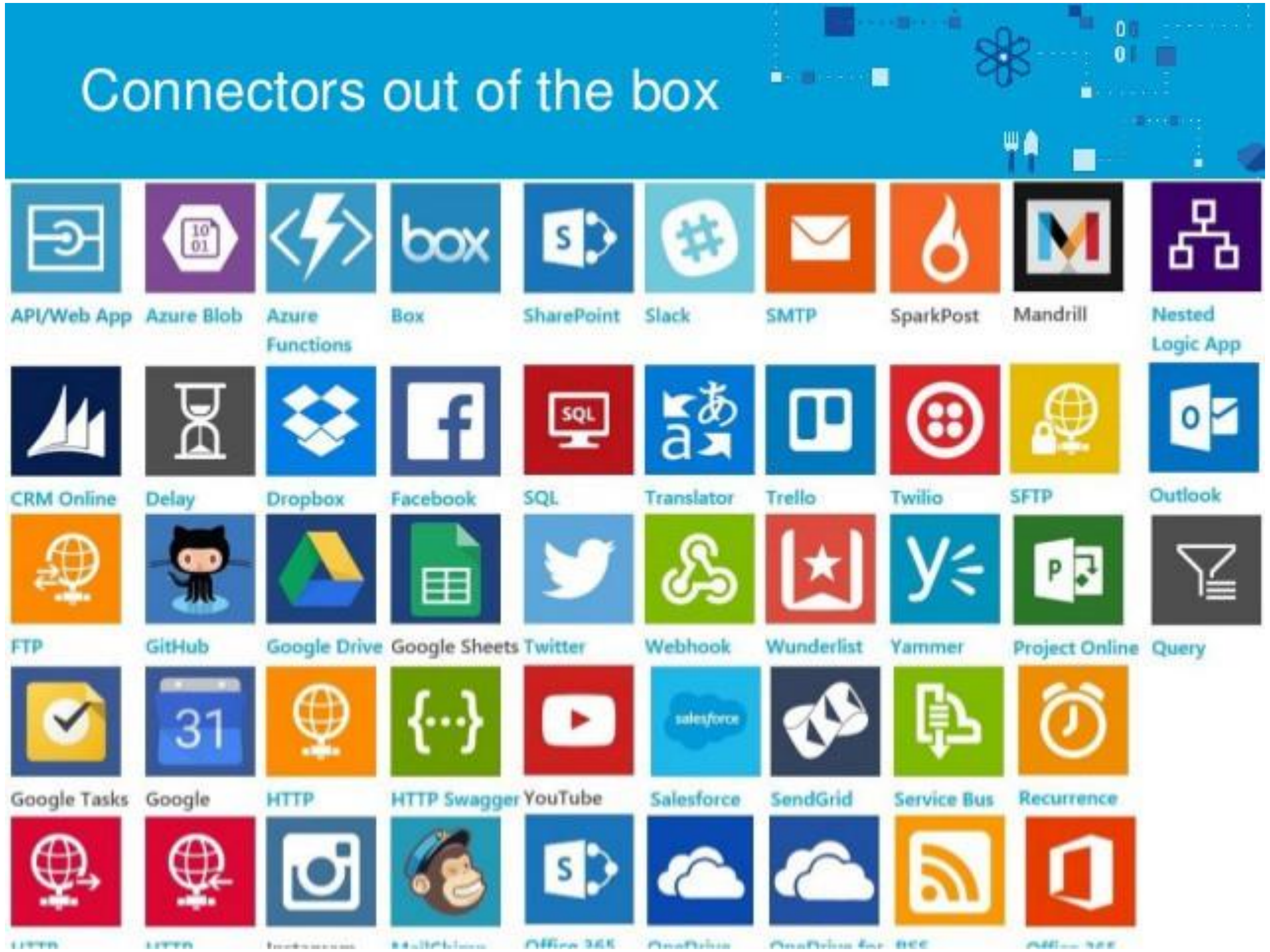
All | Built-ins | Connectors | Enterprise | Custom

Control, HTTP, Service Bus, SQL Server, Azure Functions, Office 365 Outlook, Azure Blob Storage, Text Analytics, 10to8 Appointment Scheduling, Act!, Adobe Creative Cloud, Adobe Sign, Amazon Redshift, Apache Impala, Appfigures, AS2, Asana, AtBot Admin, AtBot Logic, AWeber, Azure AD, Azure API Management, Azure App Services, Azure Application Insights, Azure Automation, Azure Container Instance, Azure Cosmos DB, Azure Data Factory, Azure Data Lake, Azure Event Grid, Azure Event Grid Publish, Azure File Storage, Azure IoT Central, Azure Kusto, Azure Log Analytics

Azure Logic App

- How does it work?

This is a trigger that fires on some event and triggers the reaction to process this event.



Links

- Logic Apps demo: https://docs.microsoft.com/en-us/azure/event-grid/publish-iot-hub-events-to-logic-apps