

Get the best out of Live Sessions HOW?

e!



Check your Internet Connection

Log in 10 mins before, and check your internet connection to avoid any network issues during the LIVE session.

Speak with the Instructor

By default, you will be on mute to avoid any background noise. However, if required you will be **unmuted by instructor**.



Clear Your Doubts

Feel free to clear your doubts. Use the “Questions” tab on your webinar tool to interact with the instructor at any point during the class.

Let us know if you liked our content

Please share feedback after each class. It will help us to enhance your learning experience.



edureka!



Microsoft Azure Developer Associate (AZ-204)

COURSE OUTLINE

MODULE 12

Introduction to Azure IaaS Compute Solutions

Implementing Azure Batch Service and Disk Encryption

Designing and Developing Applications That Use Containers

Implementing Azure App Service Web Apps and Mobile Apps

Implementing Azure App Service API Apps and Azure Functions

Developing Solutions That Use Azure Table Storage and Cosmos DB

Developing Solutions That Use Relational Database and Azure Blob Storage

Implementing Authentication and Access Control in Azure

Implementing Secure Data Solutions and Integrate Caching & CDN

Instrument Monitoring, Logging and Scalability of Apps & Services

Connecting to and Consuming Azure and Third-party Services

Developing Event-based and Message-based Solutions in Azure





Module 12 – Developing Event-based and Message-based Solutions in Azure

Topics

- Azure Messaging Service
- Types Of Azure Messaging Service
- Azure Storage Queue
- Azure Service Bus Queue and Topics
- Azure Event Hubs
- Azure IoT Hub
- Azure Event Grid
- Azure Notification Hubs
- Azure Service Bus Messaging Patterns
- Service Bus Communication Mechanisms

Objectives

After completing this module, you should be able to:

- Know the uses of Azure Event Grid
- Implement solutions that use Azure Event Hubs
- Understand the working of Azure Notification Hubs
- Implement solutions that use Azure Service Bus
- Understand the use-cases of Azure Queue Storage queues

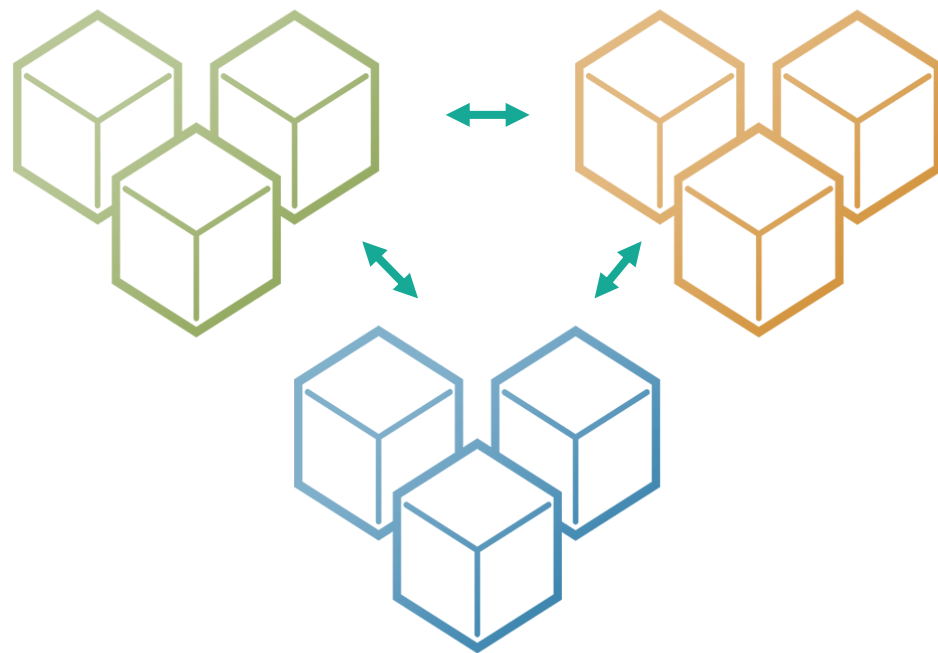




Azure Messaging Service

Introduction to Cloud Based Messaging Service

There are many different scenarios where applications would use Messaging. This could be in Microservices, Internet of Things (IoT), Inter-Application communications, and many other scenarios where messaging makes sense to use. As a result of it there are many different use cases for messaging, there are a number of messaging services that can be used within the Microsoft Azure cloud platform.



Types of Azure Messaging Service

01

Azure Storage Queue

02

Azure Service Bus Queue

03

Azure Service Bus Topic

04

Azure Event Hubs

05

Azure IoT Hub

06

Azure Event Grid

07

Azure Notification Hubs

Azure Storage Queue

01

Azure Storage Queue

02

03

04

05

06

The Azure Storage Queue is one of the original Message Queue services within the Microsoft Azure platform. It's been around since the initial General Availability of Azure back in 2010. The Azure Storage Queues are a great option to start with for using a cloud-based Message Queue service. It's fairly simple to integrate, and it's also fairly cost-effective among the other key features.



Azure Service Bus Queue and Topics

01

02

Azure Service Bus Queue and Topics

03

04

05

06

The Azure Service Bus is a set of features in Microsoft Azure that are centred around inter & intra-application messaging. Within Azure Service Bus are 2 message queue services: Queues and Topics. The Azure Service Bus Queues are simply a more robust message queue service than what is provided by Azure Storage Queues. Azure Service Bus Topics take the same general features and scalability of Azure Service Bus Queues, & adds on a few more advanced features.



Azure Event Hubs

01

02

03

Azure Event Hubs

04

05

06

Azure Event Hubs is a slightly different type of message queue service when compared to Service Bus Queues, Service Bus Topics, and Azure Storage Queues. Instead of working on 1 message in, then 1 message out at a time, Azure Event Hubs works as the front door to an event stream pipeline. The difference with Azure Event Hubs being the front door to an event stream pipeline enables it to handle a much larger scale of messages.



Azure IoT Hub

01

02

03

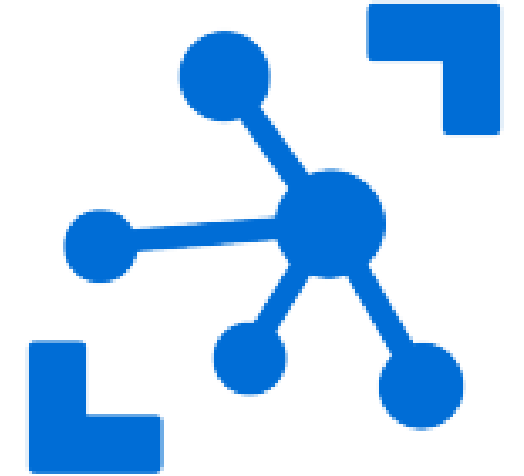
04

Azure IoT Hub

05

06

Microsoft added the Azure IoT Hub service to the Azure platform, and it's built on the foundation of Azure Event Hubs with additional capabilities built specifically for the Internet of Things. Azure IoT Hub adds additional IoT specific features for managing and securing large amounts of individual IoT devices as they connect and send events through the service.



Azure Event Grid

01

02

03

04

05

Azure Event Grid

06

Azure Event Grid is a messaging service that enables event-based architecture to be built more easily. This is a messaging service that's built with a few target uses in mind. It is similar to Azure Service Bus Topics that enables a publish / subscribe messaging model. However, the similarity pretty much ends there. Azure Event Grid is a cloud-native messaging service that enables event-based architectures like Microservices and Event-Driven systems to be built more easily.



Azure Notification Hubs

01

02

03

04

05

06

Azure Notification Hubs




Azure Notification Hubs is a different kind of messaging service. Instead of using send messages between apps or with a microservices architecture; this service is used to send Native Mobile Push Notifications from any services to native apps running on mobile devices . You can have a single endpoint to send notifications to, then this service automatically figures out how to send those notifications devices you're targeting.





Comparison of Services

Event Grid Vs. Event Hub Vs. Service Bus

Event Grid	Event Hubs	Service Bus
Purpose Reactive Programming	Purpose Big Data Pipeline	Purpose High Value enterprise messaging
Type Event Distribution (Discrete)	Type Event Streaming (Series)	Type Message
When to Use React to status changes	When to Use Telemetry and distributed data streaming	When to Use Order processing and financial transactions
		

Azure Service Bus

Introduction to Azure Service Bus

Microsoft Azure Service Bus is a fully managed enterprise integration message broker. Service Bus is most commonly used to decouple applications and services from each other, and is a reliable and secure platform for asynchronous data and state transfer.

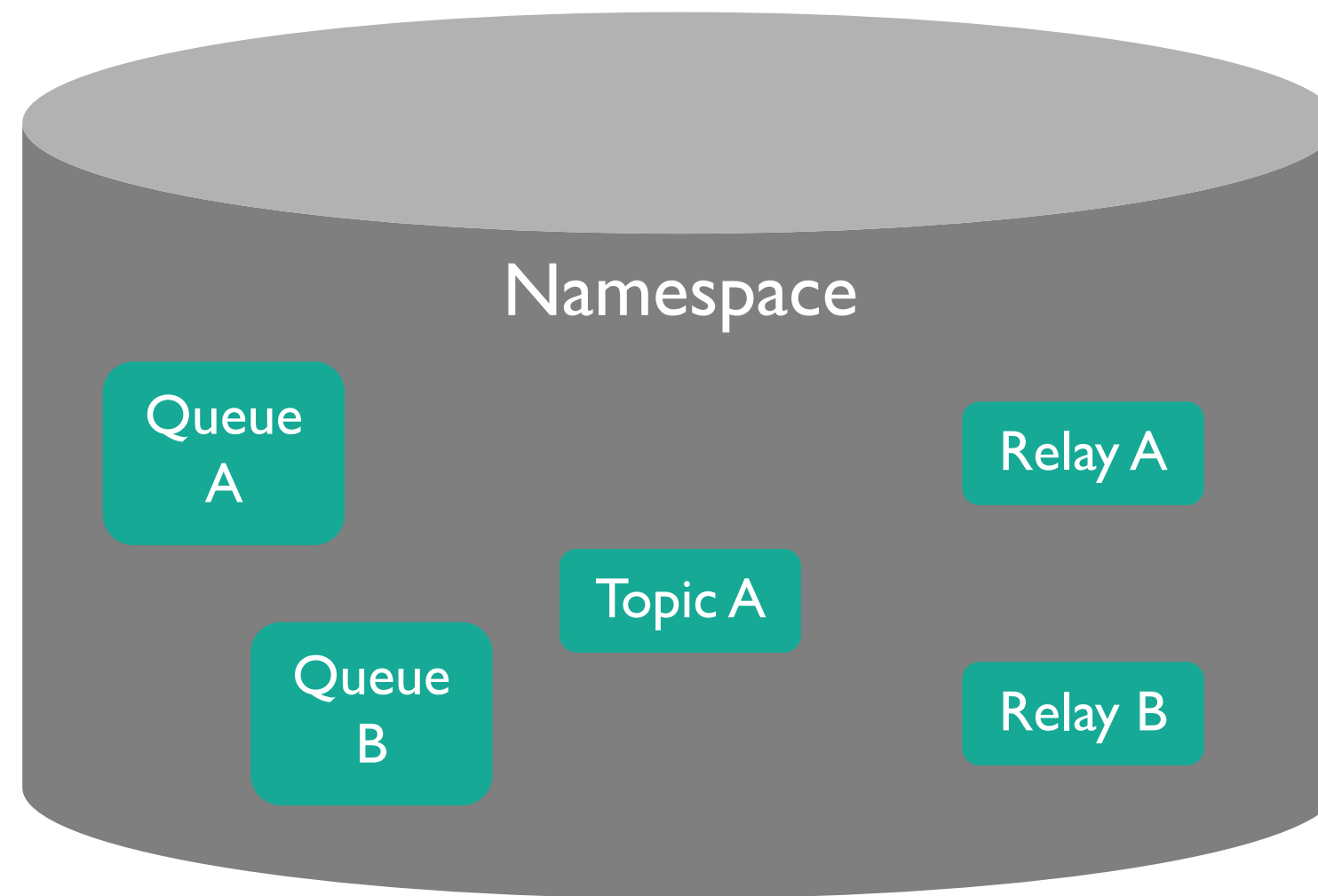
Data is transferred between different applications and services using *messages*. A message is in binary format, which can contain JSON, XML, or just text.

1. Messaging : Transfer business data, such as sales or purchase orders, journals, or inventory movements
2. Decouple applications : Improve reliability and scalability of applications and services
3. Topics and subscriptions : Enable 1:*n* relationships between publishers and subscribers
4. Message sessions : Implement workflows that require message ordering or message deferral

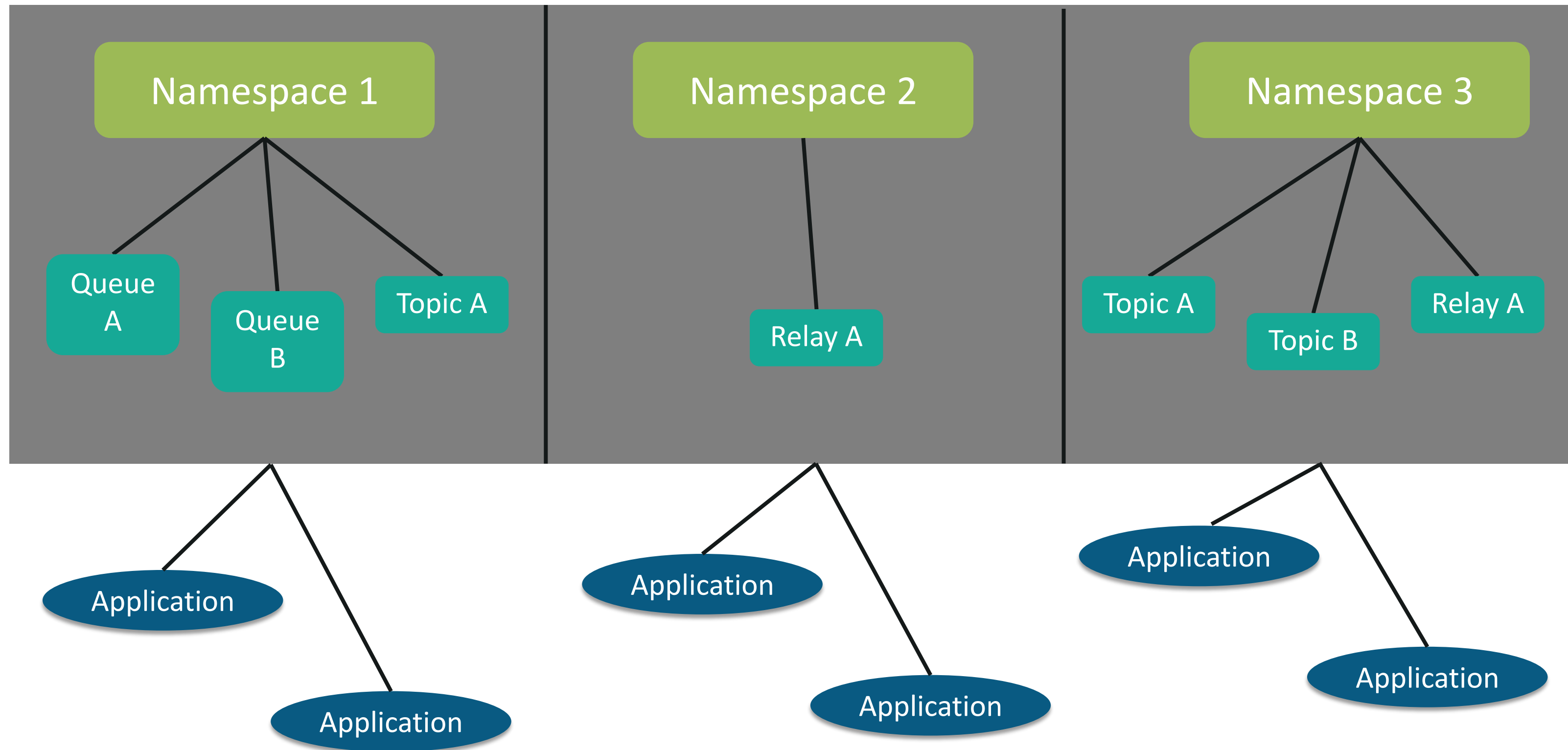


Azure Service Bus Namespace

A namespace is a scoping container for all messaging components. Multiple queues and topics can reside within a single namespace, and namespaces often serve as application containers.



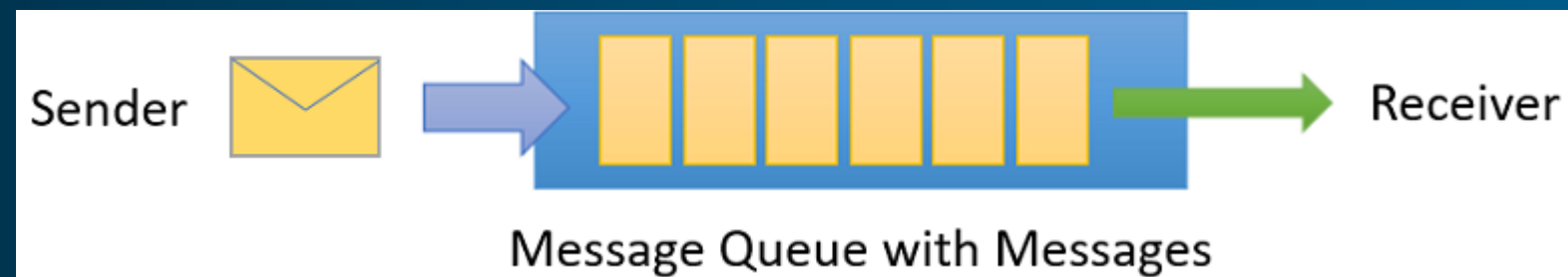
Azure Service Bus Architecture



Azure Service Bus Queue and Topic

Queues

1. Messages are sent to and received from *queues*. Queues enable you to store messages until the receiving application is available to receive and process them.

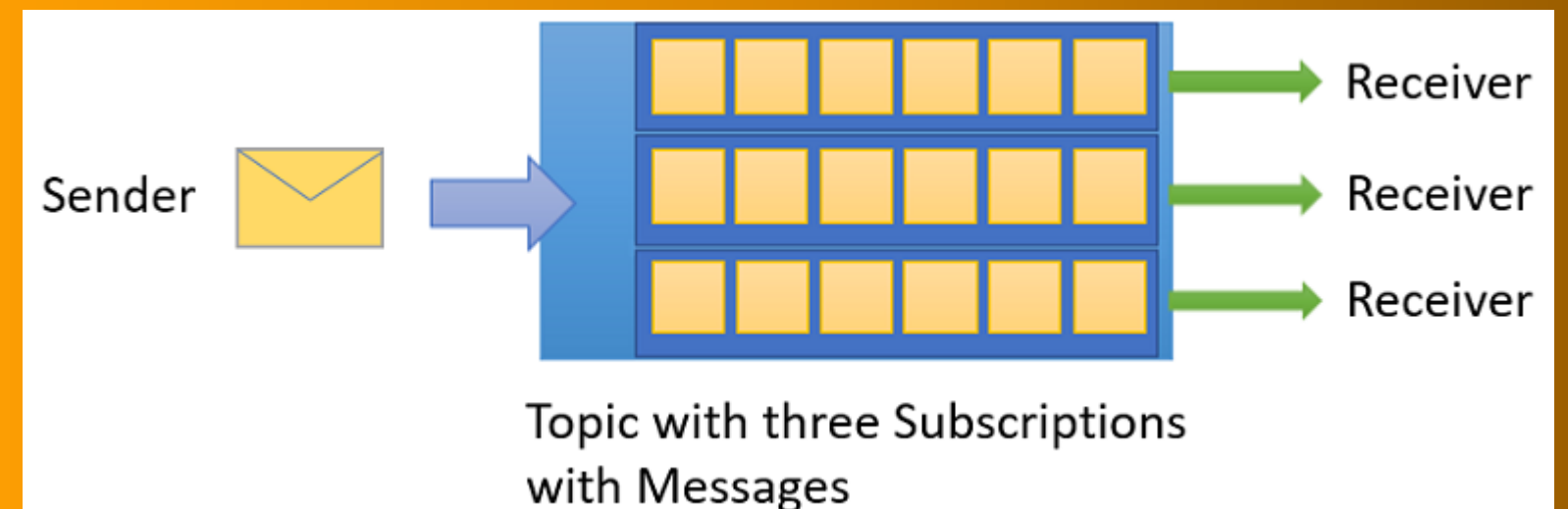


2. Messages in queues are ordered and timestamped on arrival

3. Once accepted, the message is held safely in redundant storage. Messages are delivered in *pull* mode, which delivers messages on request.

Topics

1. You can also use *topics* to send and receive messages. While a queue is often used for point-to-point communication, topics are useful in publish/subscribe scenarios.



2. Topics can have multiple, independent subscriptions
3. A subscriber to a topic can receive a copy of each message sent to that topic

Azure Service Bus Features

1

Message Sessions
and Auto
Forwarding

2

Dead Lettering

3

Scheduled Delivery

4

Message Deferral's

5

Batching and
Transactions

6

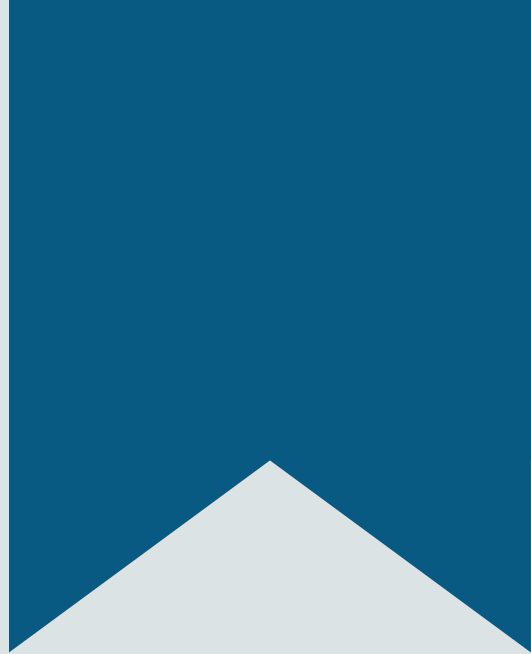
Filtering and
Actions

7

Auto Delete On Idle

8

Geo Disaster
Recovery



Demo 1 – Create and Integrate Service Bus Queue



Demo 2 – Deploy Service Bus Topics and Subscriptions

Demo 2 – Details

This tutorial shows how to use Service Bus topics and subscriptions in a retail inventory scenario, with publish/subscribe channels using the Azure portal and .NET.

In this Demo, you will:

- Create a Service Bus topic and one or more subscriptions to that topic using the Azure portal
- Add topic filters using .NET code
- Create two messages with different content
- Send the messages and verify they arrived in the expected subscriptions
- Receive messages from the subscriptions

Azure Event Hubs

Introduction to Azure Event Hubs

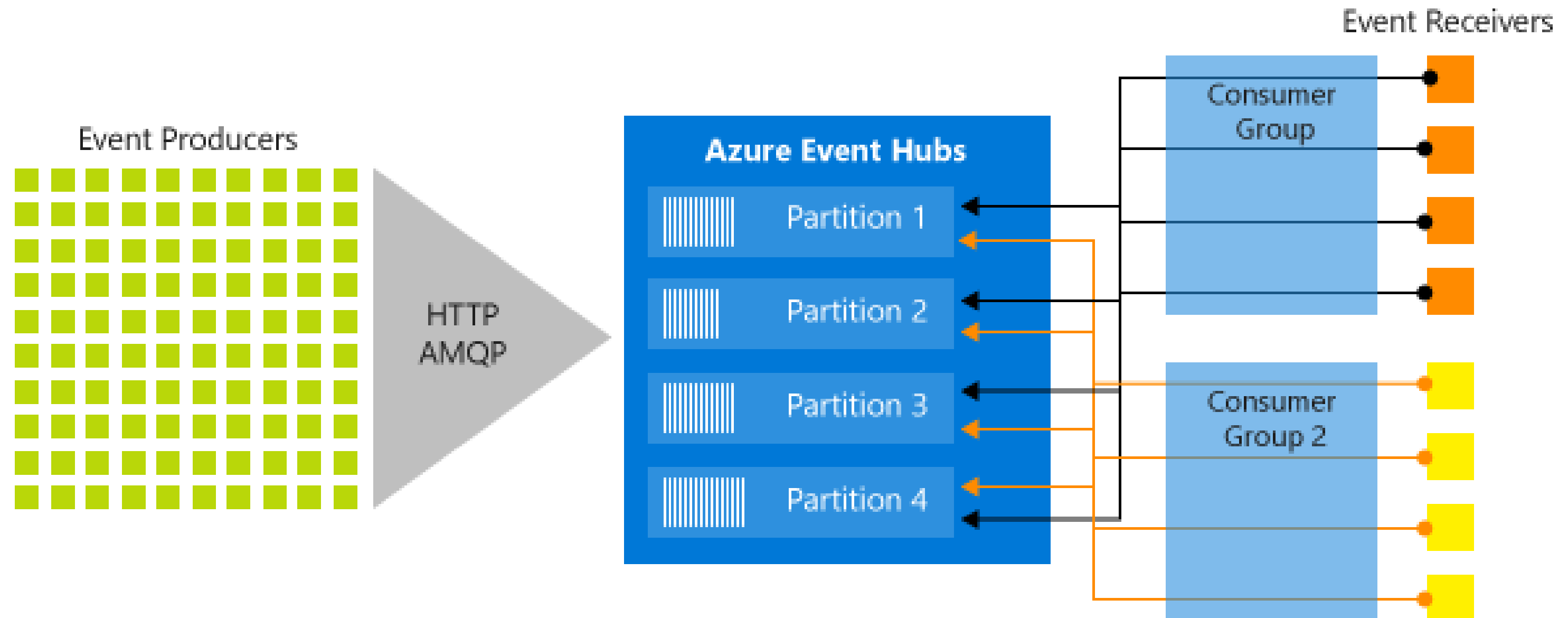
Azure Event Hubs is a big data streaming platform and event ingestion service. It can receive and process millions of events per second. Data sent to an event hub can be transformed and stored by using any real-time analytics provider or batching/storage adapters. Event Hubs is a fully managed Platform-as-a-Service (PaaS) with little configuration or management overhead.

Following are some of the scenarios where you can use Event Hubs:

1. Anomaly detection (fraud/outliers)
2. Application logging
3. Analytics pipelines, such as clickstreams
4. Live dashboarding
5. Archiving data
6. Transaction processing
7. User telemetry processing
8. Device telemetry streaming



Azure Event Hub Architecture



Azure Event Hub Components

1. Events Producer : Any entity that sends data to an event hub.

2. Partitions : Each consumer only reads a specific subset, or partition, of the message stream.

3. Consumer groups : A view (state, position, or offset) of an entire event hub.

4. Throughput units : Pre-purchased units of capacity that control throughput capacity.

5. Event receivers : Any entity that reads event data from an event hub.



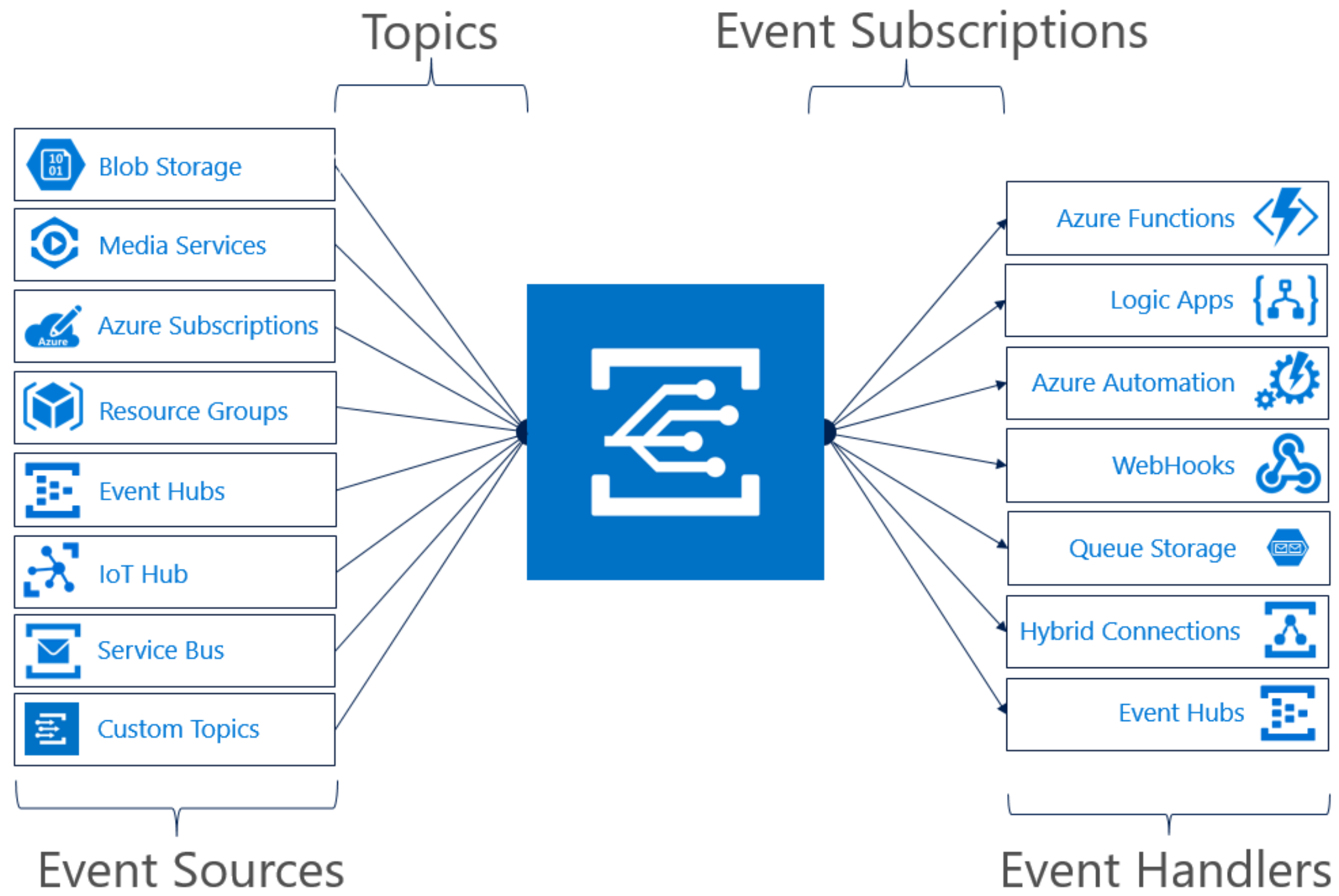
Demo 3 – Implement Azure Event Hubs

Azure Event Grid

Introduction to Azure Event Grid

Azure Event Grid allows you to easily build applications with event-based architectures. Event Grid has built-in support for events coming from Azure services, like storage blobs and resource groups. Event Grid also has support for your own events, using custom topics. You can use filters to route specific events to different endpoints, multicast to multiple endpoints, and make sure your events are reliably delivered. Currently, Azure Event Grid is available in all public regions. It isn't yet available in the Azure Germany, Azure China, or Azure Government clouds.

Azure Event Grid Architecture



Azure Event Grid Components

1. Events - What happened ?

2. Event sources - Where the event took place ?

3. Topics - The endpoint where publishers send events.

4. Event subscriptions - The endpoint or built-in mechanism to route events.

5. Event handlers - The app or service reacting to the event.

Azure Event Grid Features

1

Simplicity

2

Advanced
filtering

3

Fan-out

4

Reliability

5

Pay-per-event

6

High throughput

7

Built-in Events

8

Custom Events

NOTE : Azure Event Grid uses a pay-per-event pricing model, so you only pay for what you use. The first 100,000 operations per month are free. Operations are defined as event ingress, subscription delivery attempts, management calls, and filtering by subject suffix.

Azure Notification Hub

Introduction to Azure Notification Hub



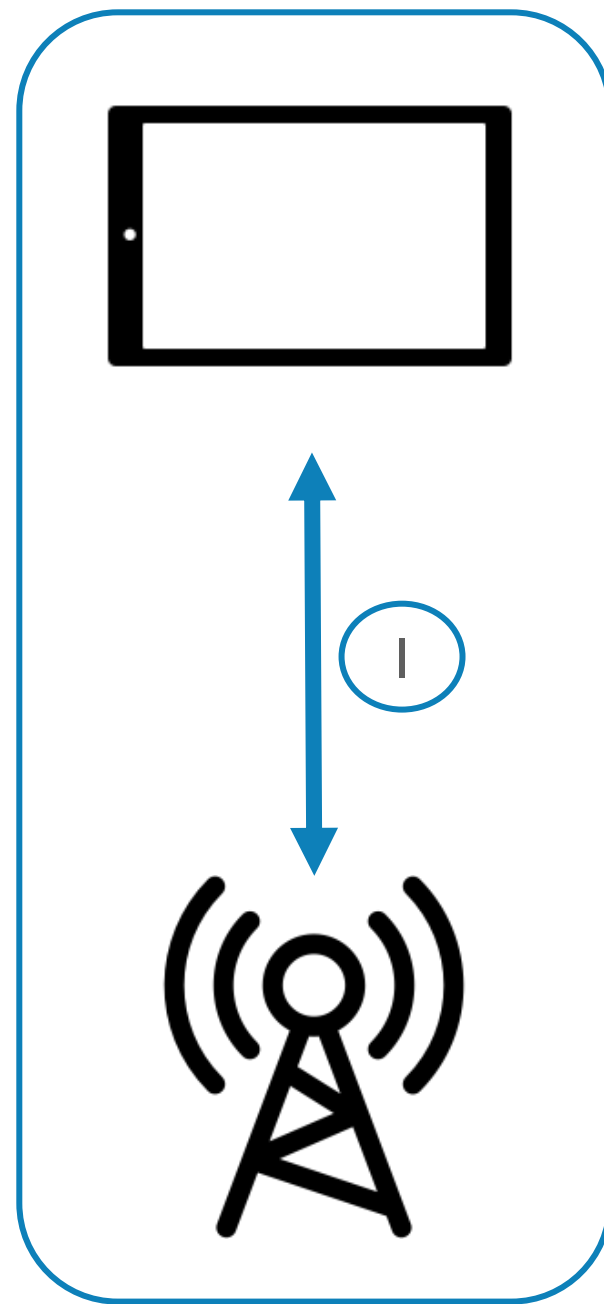
Azure Notification Hub

Azure Notification Hubs are push notification software engines designed to alert users about new content for a given site, service or app and are part of Microsoft Azure's public cloud service offerings.

According to Microsoft, Azure Notification Hubs currently offers a free push service for up to a million notifications a month and with some conditions, up to 100 namespaces and up to 500 active devices per namespace.

Azure Notification Hubs allow developers to take advantage of push notifications without having to deal with the back-end complexities of enabling push on their own

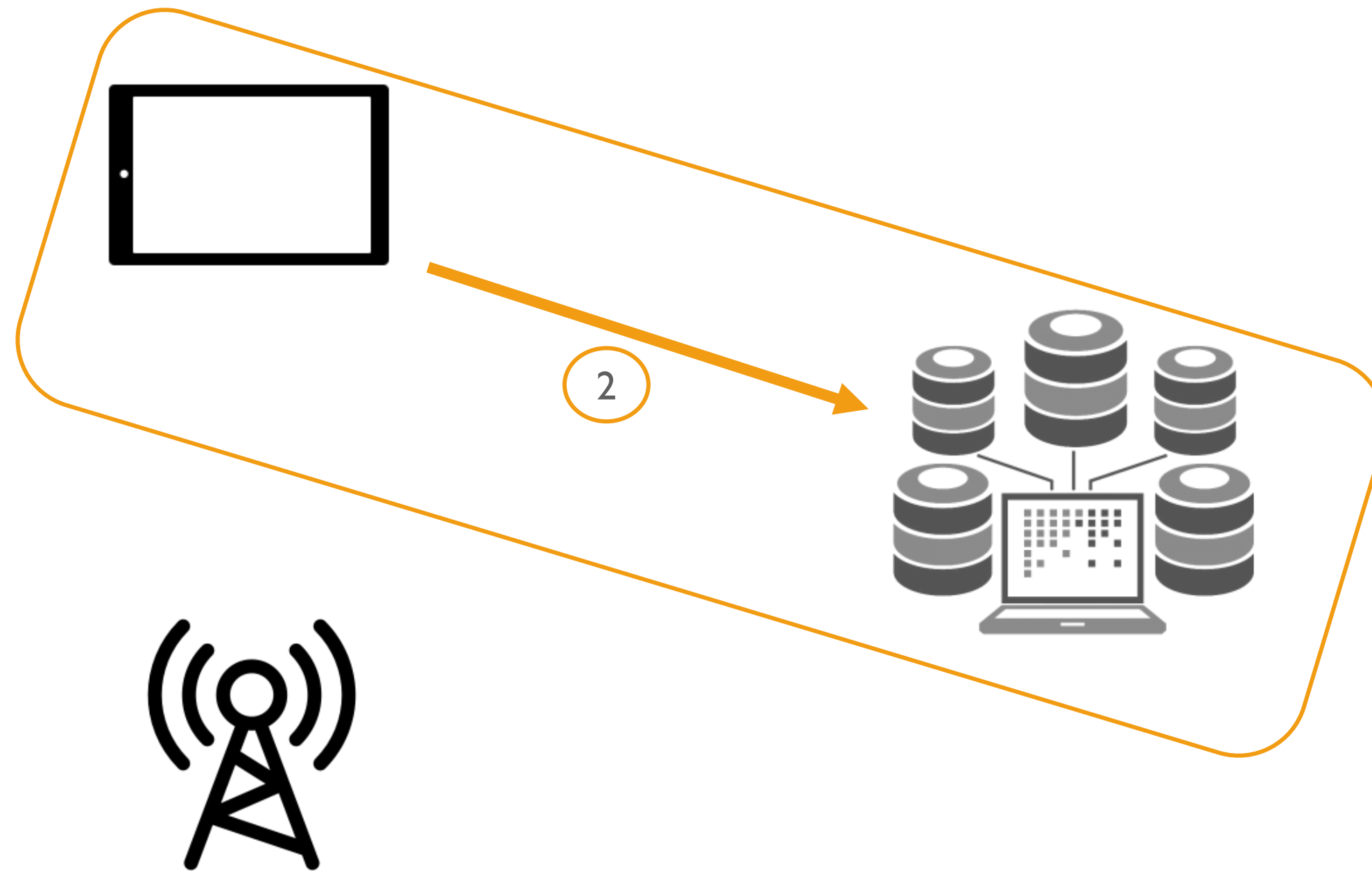
Working of Azure Notification Hub



The client app decides it wants to receive notification. Hence, it contacts the corresponding PNS to retrieve its unique and temporary push handle.

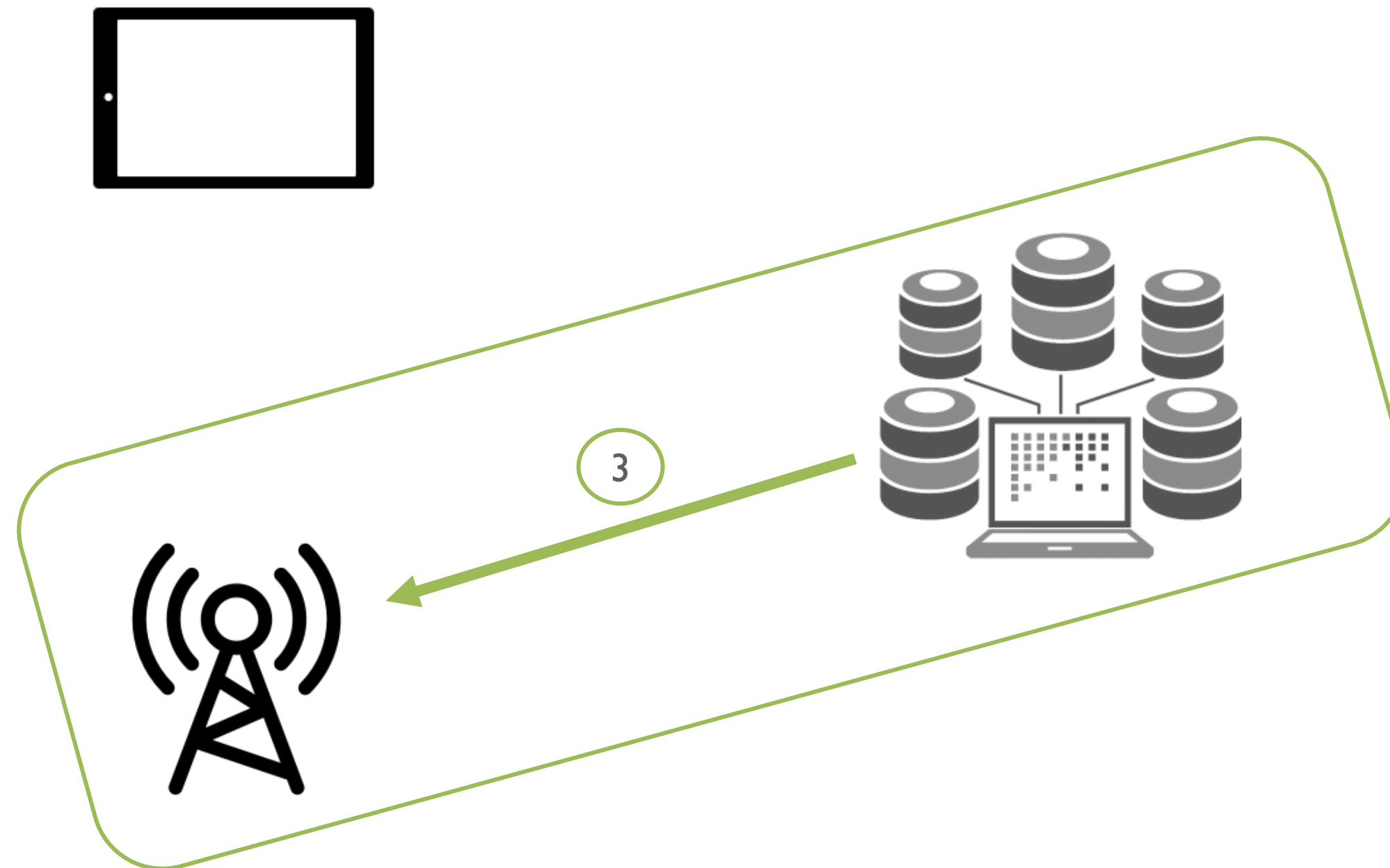
The handle type depends on the system (for example, WNS has URIs while APNS has tokens).

Working of Azure Notification Hub (Cont.)



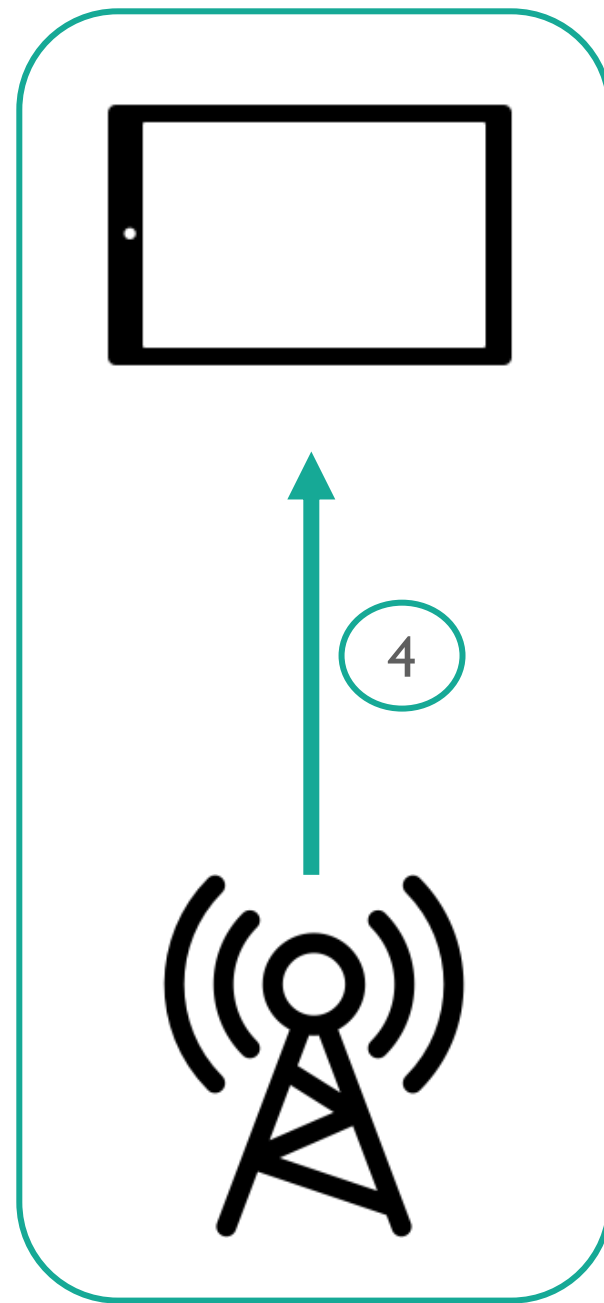
The client app stores this handle in the app back-end or provider.

Working Of Azure Notification Hub (Cont.)



To send a push notification, the app back-end contacts the PNS using the handle to target a specific client app.

Working Of Azure Notification Hub (Cont.)



The PNS forwards the notification to the device specified by the handle.



Microsoft Azure Service Bus Messaging

Microsoft Azure Service Bus Messaging

When two or more parties want to exchange information, they need a communication facilitator

Service Bus is a brokered, or third-party communication mechanism, which is similar to a postal service in the physical world

Microsoft Azure Service Bus is a reliable information delivery service to make communication easier



Azure Service Bus

The messaging service ensures that the information is delivered even if the two parties are not online at the same time

Service Bus Uses

Build reliable and elastic
cloud apps with
messaging

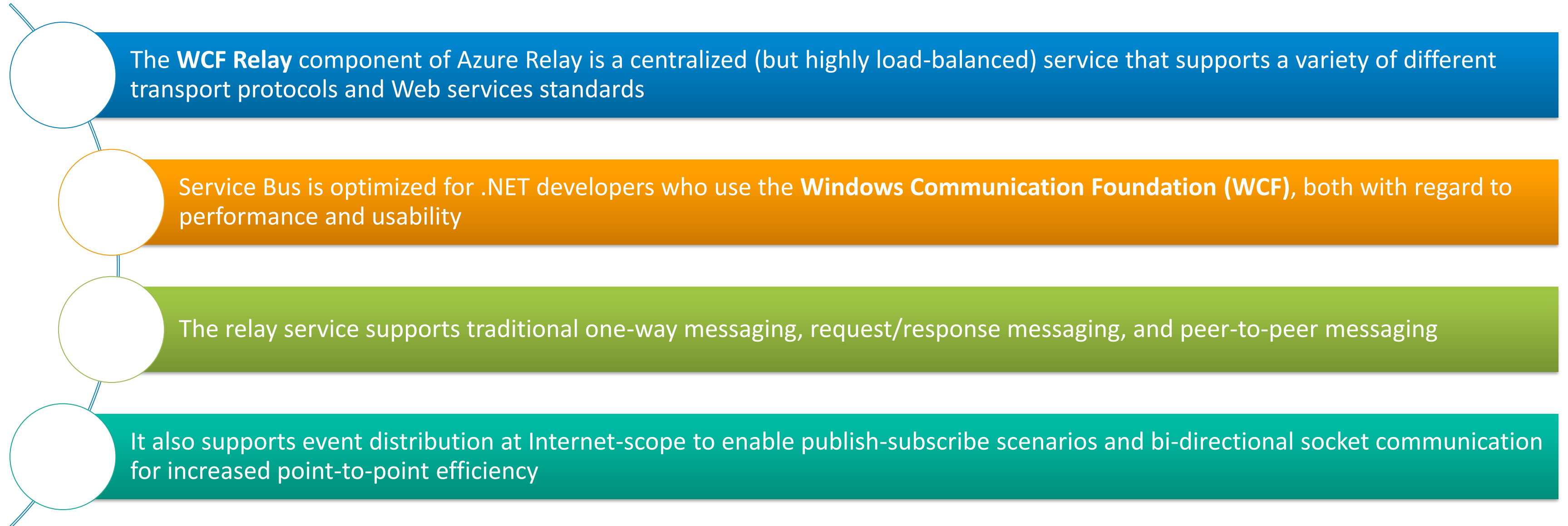
Protect your application
from temporary peaks

Distribute messages to
multiple independent
backend systems

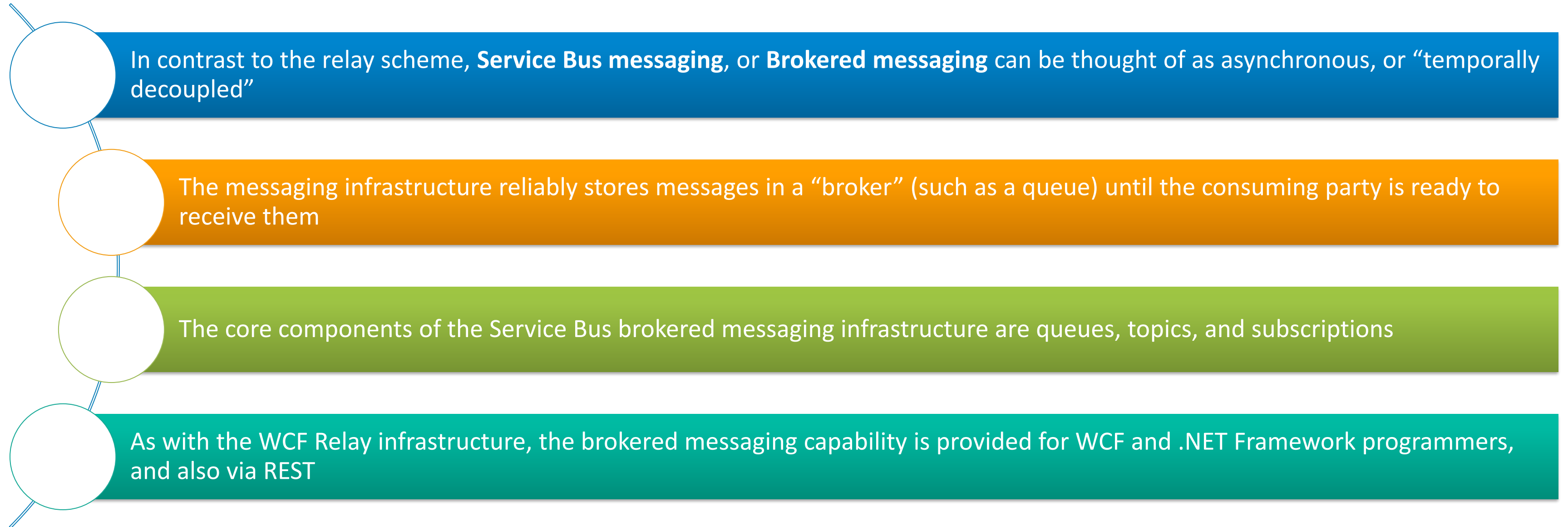
Decouple your
applications from each
other

Ordered messaging
scaled out to multiple
readers

Azure Service Bus Messaging Patterns – Azure Relay




Azure Service Bus Messaging Patterns – Brokered






Service Bus Fundamentals


Communication Scenarios



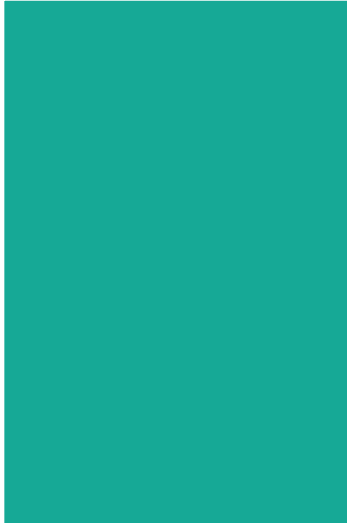
Sometimes, letting applications send and receive messages through a simple queue is the best solution



In other situations, an ordinary queue isn't enough; a queue with a publish-and-subscribe mechanism is better

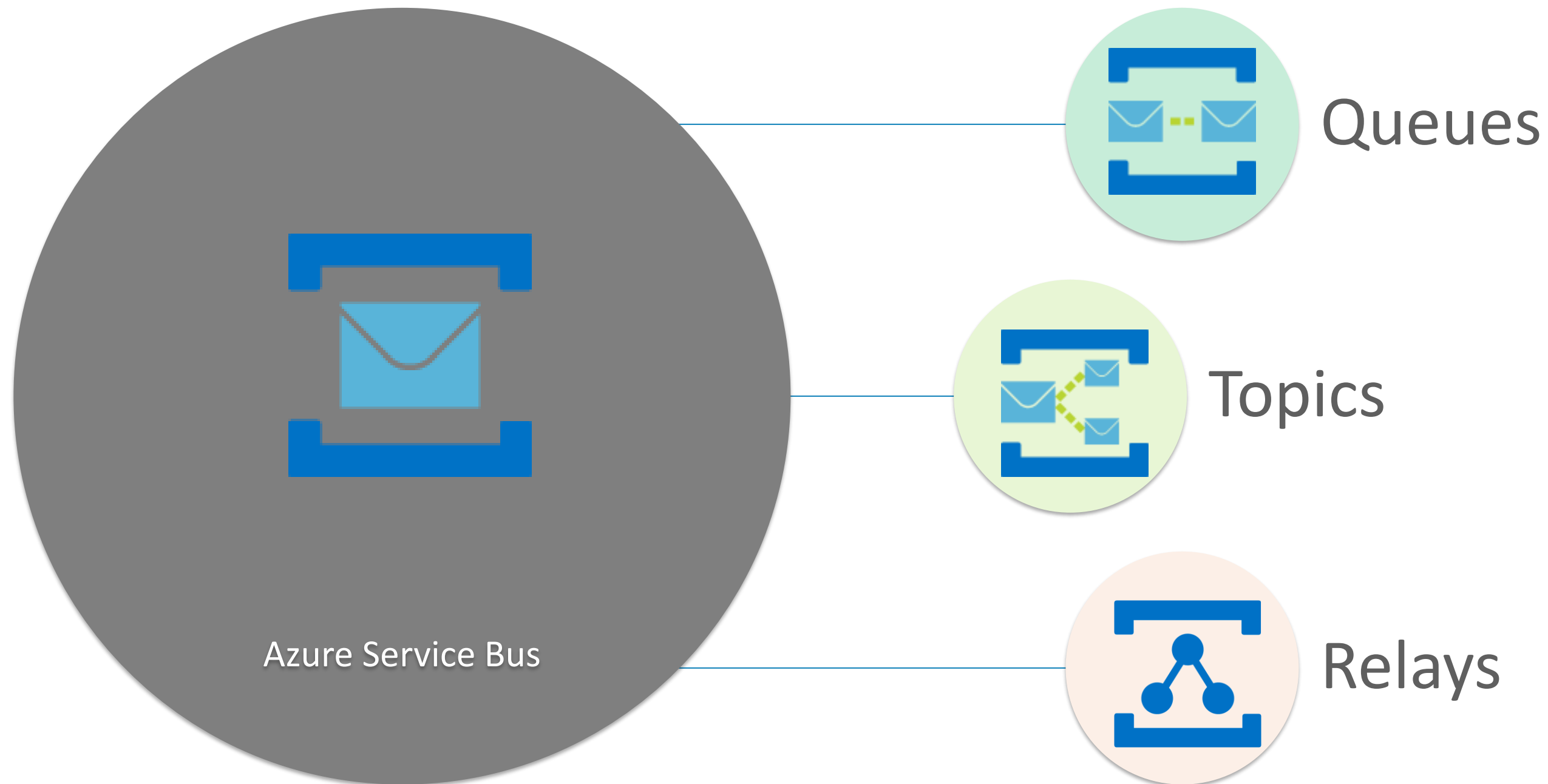


In some cases, all that's needed is a connection between applications, and queues are not required



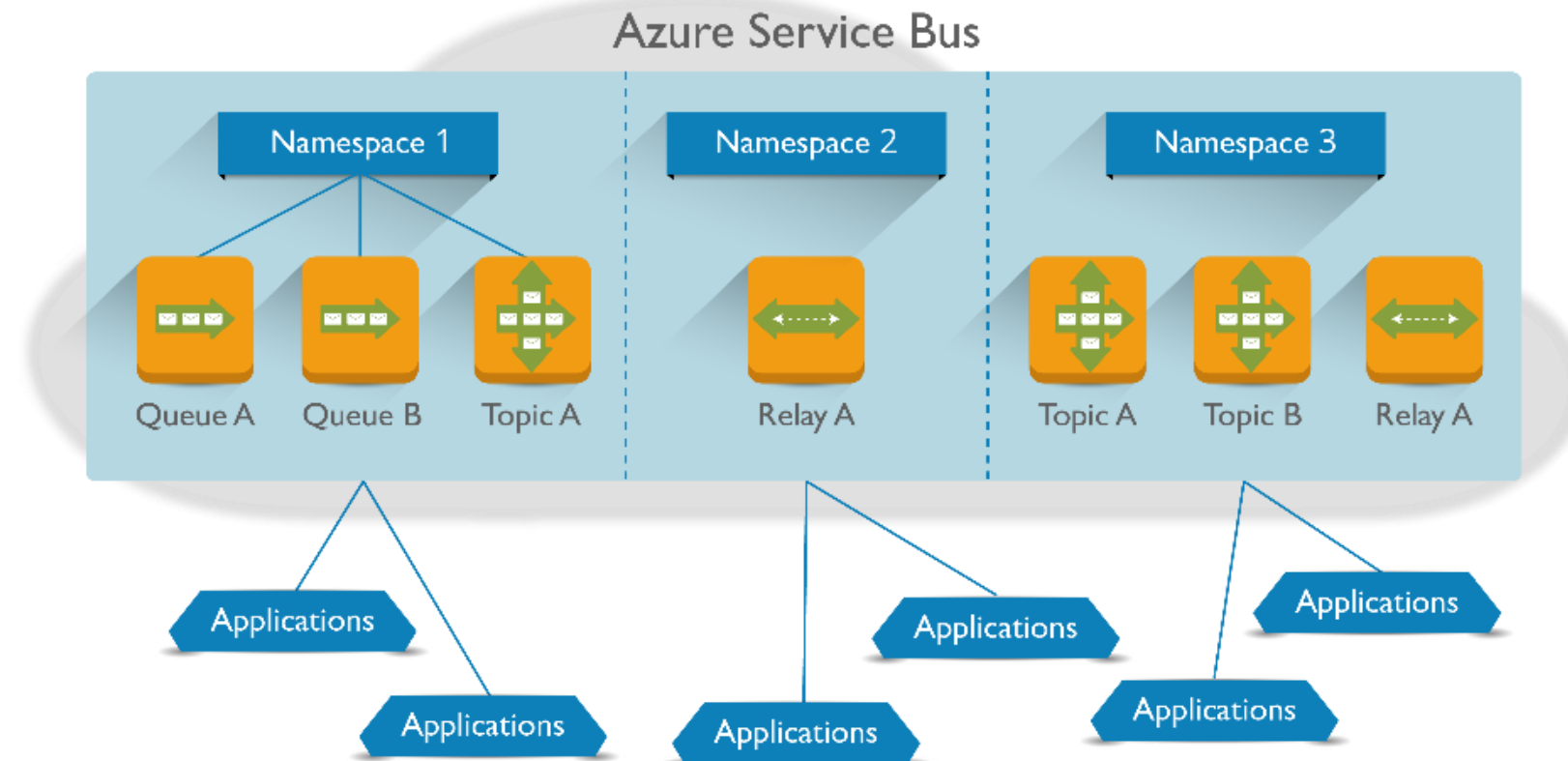
Service Bus provides all three options, enabling your applications to interact in different ways

Service Bus Fundamentals



Service Bus – A Multi-tenant Cloud Service

- Service Bus is a multi-tenant cloud service, which means that the service is shared by multiple users
- Each user, such as an application developer, creates a *namespace*, then defines the communication mechanisms needed within that namespace
- Below is how the Service Bus Architecture looks:



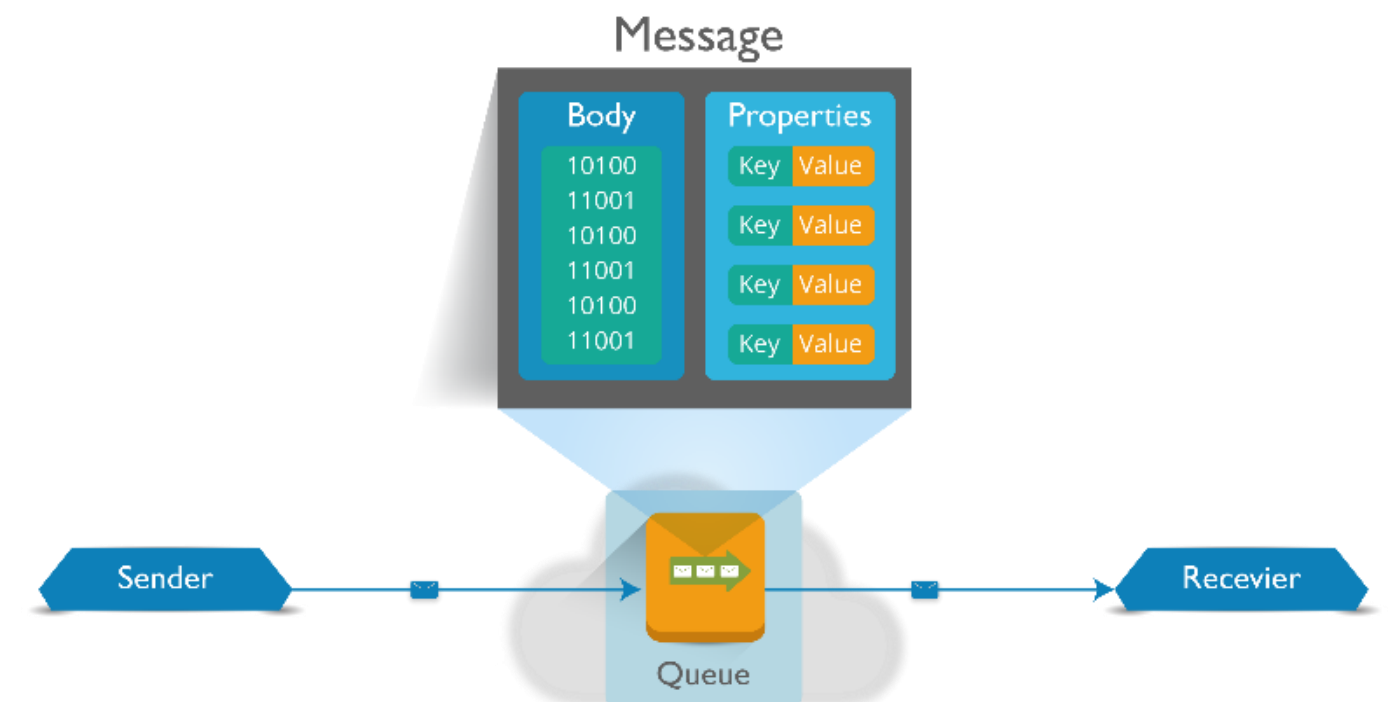
Service Bus provides a multi-tenant service for connecting applications through the cloud

Service Bus – Different Communication Mechanisms

- Within a namespace, you can use one or more instances of three different communication mechanisms, each of which connect applications in a different way. The choices are:
 1. **Queues:** Allows one-directional communication, where each queue acts as an intermediary (*broker*) that stores sent messages until they are received. Each message is received by a single recipient
 2. **Topics:** Provides one-directional communication using *subscriptions*-a single topic can have multiple subscriptions. Like a queue, a topic acts as a broker, but each subscription can optionally use a filter to receive only messages that match specific criteria
 3. **Relays:** Provides bi-directional communication. Unlike queues and topics, a relay doesn't store in-flight messages; it's not a broker. Instead, it just passes them on to the destination application

Service Bus Communication Mechanisms – Queues

- Suppose you decide to connect two applications using a Service Bus queue, the process is simple:
 - A sender sends a message to a Service Bus queue, and a receiver picks up that message after some time
 - A queue can just have a single receiver, as shown in the figure below or multiple applications can read from the same queue
 - In the latter situation, each message is read just by one receiver. For a multi-cast service, you should use a topic instead



Service Bus Communication Mechanisms – Topics

Service Bus Topics are better than Queues and the difference is that topics enable each receiving application to create its own *subscription* by defining a *filter*



A subscriber then sees only the messages that match that filter. For example, Figure 3 shows a sender and a topic with three subscribers, each with its own filter:

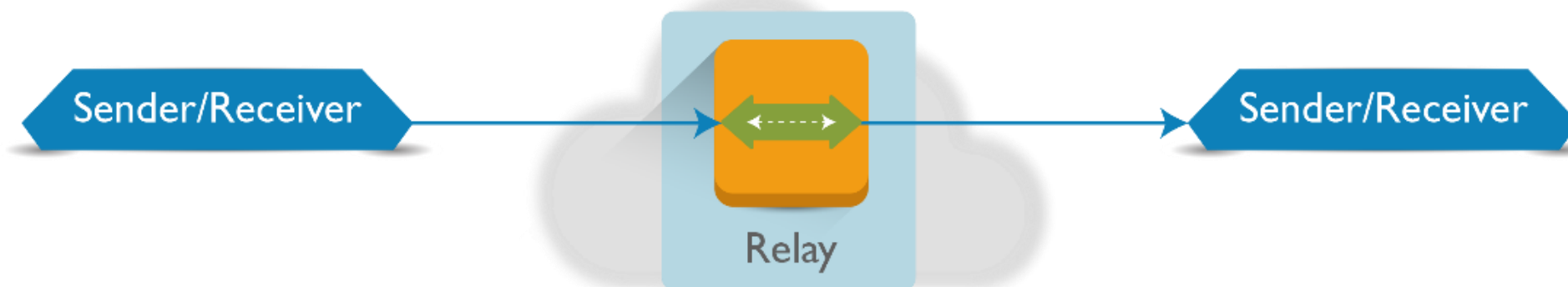
Subscriber 1 receives only messages that contain the property *Seller="Ava"*

Subscriber 2 receives messages that contain the property *Seller="Ruby"* and/or contain an *Amount* property whose value is greater than 100,000, perhaps Ruby is the sales manager

Subscriber 3 has set its filter to *True*, which means that it receives all messages (Example: For Audit Trail)

Service Bus Communication Mechanisms – Relays

- Both queues and topics provide one-way asynchronous communication through a broker
- Traffic flows in just one direction, and there's no direct connection between senders and receivers
- In case you don't want this connection and suppose your applications need to both send and receive the messages, or perhaps you want a direct link between them and you don't need a broker to store messages. Hence to address scenarios like this Service Bus provides *relays*





Demo 4 – Service Bus Queue Implementation

Summary

Introduction To Cloud Based Messaging Service

There are many different scenarios where applications would use Messaging. This could be in Microservices, Internet of Things (IoT), Inter-Application communications, and many other scenarios where messaging makes sense to use. As a result of it there are many different use cases for messaging, there are a number of messaging services that can be used within the Microsoft Azure cloud platform.



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

Azure Service Bus Queue and Topics

- 01
- 02 Azure Service Bus Queue and Topics
- 03
- 04
- 05
- 06

The Azure Service Bus is a set of features in Microsoft Azure that are centred around inter & intra-application messaging. Within Azure Service Bus are 2 message queue services: Queues and Topics. The Azure Service Bus Queues are simply a more robust message queue service than what is provided by Azure Storage Queues. Azure Service Bus Topics take the same general features and scalability of Azure Service Bus Queues, & adds on a few more advanced features.



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

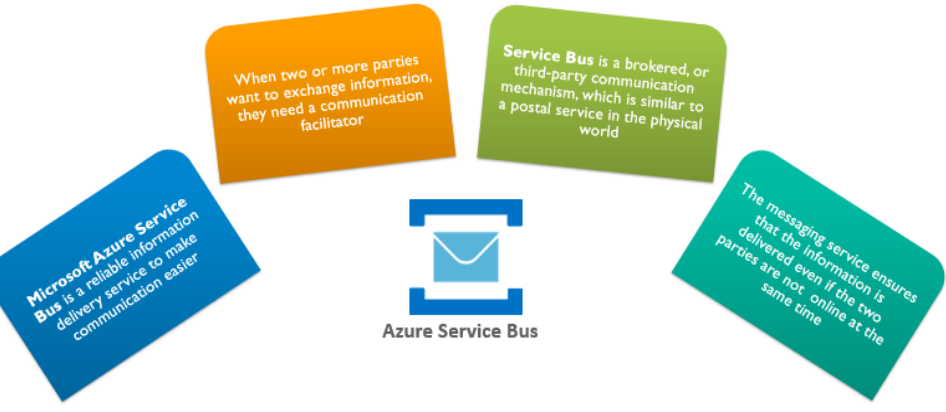
Azure Service Bus Features



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

Microsoft Azure Service Bus Messaging

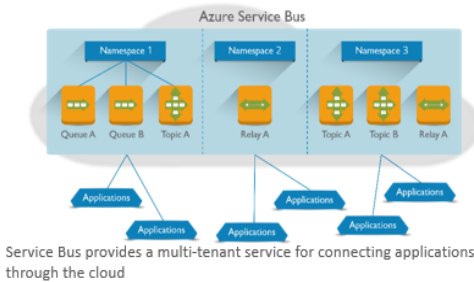


edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

Service Bus - A Multi-Tenant Cloud Service

- Service Bus is a multi-tenant cloud service, which means that the service is shared by multiple users
- Each user, such as an application developer, creates a *namespace*, then defines the communication mechanisms needed within that namespace
- Below is how the Service Bus Architecture looks:



Service Bus provides a multi-tenant service for connecting applications through the cloud

edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

Service Bus Communication Mechanisms – Relays

- Both queues and topics provide one-way asynchronous communication through a broker
- Traffic flows in just one direction, and there's no direct connection between senders and receivers
- In case you don't want this connection and suppose your applications need to both send and receive the messages, or perhaps you want a direct link between them and you don't need a broker to store messages. Hence to address scenarios like this Service Bus provides *relays*



edureka!

Copyright © edureka and/or its affiliates. All rights reserved.

Questions



FEEDBACK





Thank You

For more information please visit our website
www.edureka.co