azenix

Azure Service Bus

Brendan Richards

About Me

Brendan Richards
Lead Engineer @ Azenix

UK -> Australia
Java -> .NET

Linux, Open Source,
Event-Driven Code

azenix

- Core Components
- .NET Usage
- Azure Function Apps
- Message Based Architectures
- Infrastructure As Code

Queues

Queues offer **First In, First Out** (FIFO) message delivery to one or more competing consumers. That is, receivers typically receive and process messages in the order in which they were added to the queue. And, only one message consumer receives and processes each message.

azenix

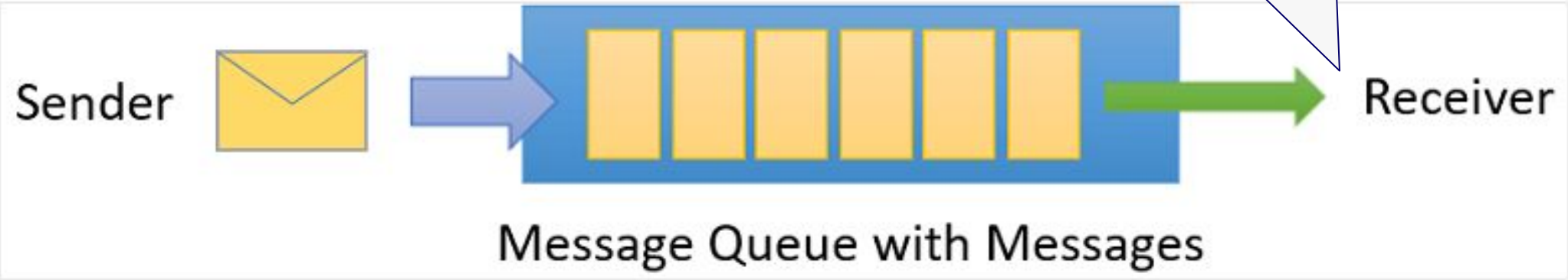
Constructors

ServiceBusMessage()	Creates a new message.
ServiceBusMessage(BinaryData)	Creates a new message from the specified BinaryData instance.
ServiceBusMessage(ReadOnlyMemory<Byte>)	Creates a new message from the specified payload.
ServiceBusMessage(ServiceBusReceivedMessage)	Creates a new message from the specified received message by copying the properties.
ServiceBusMessage(String)	Creates a new message from the specified string, using UTF-8 encoding.

Queues

Queues offer **First In, First Out** (FIFO) message delivery to one or more compe ly receive and process messages in the order in which they were added to the qu receives and processes each message.

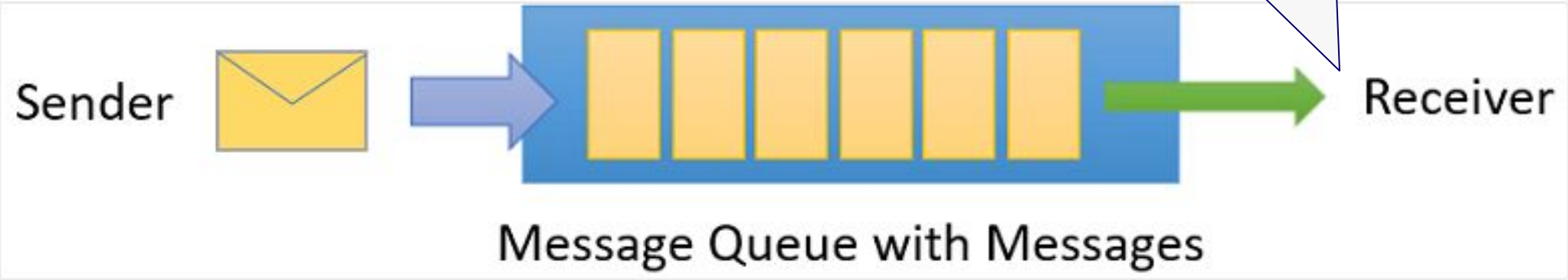
Each message goes to only one receiver - 2 read modes…



Queues

Queues offer **First In, First Out** (FIFO) message delivery to one or more compe ly receive and process messages in the order in which they were added to the qu receives and processes each message.

Receive and Delete
As soon as you get the message it's already gone from the queue

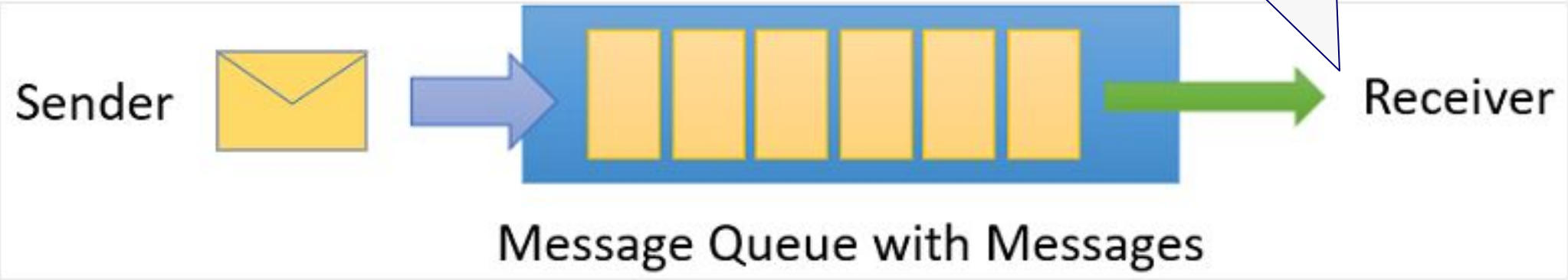


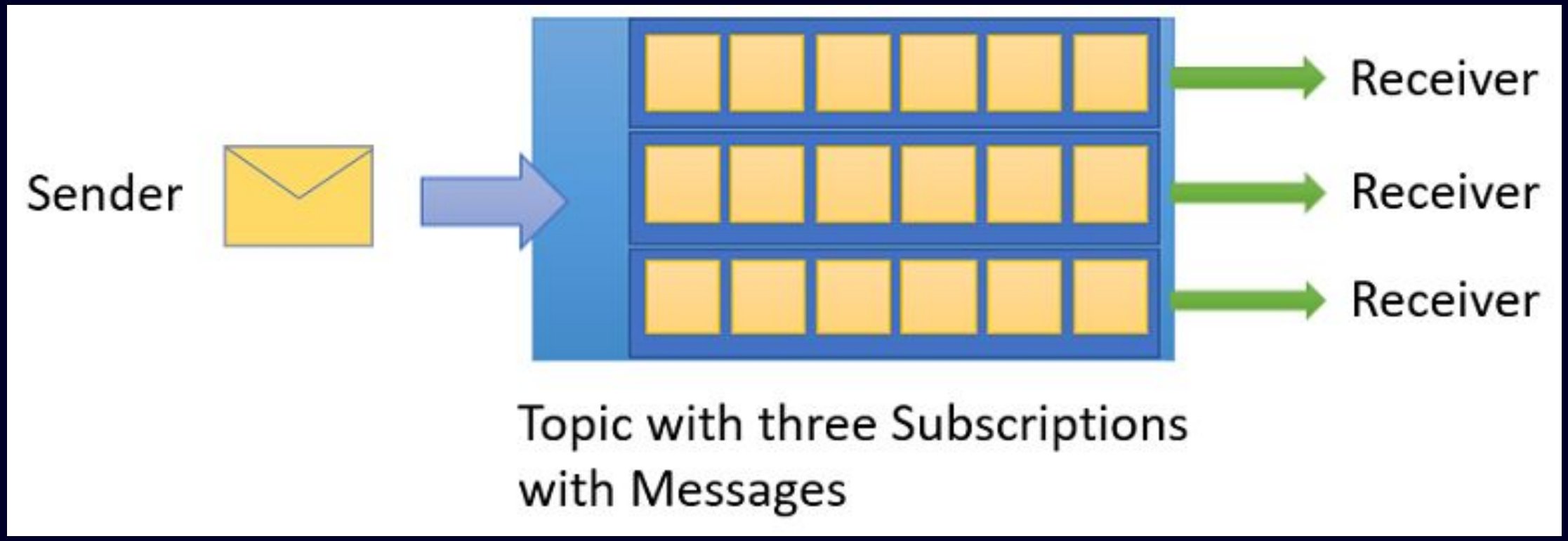
azenix

Queues

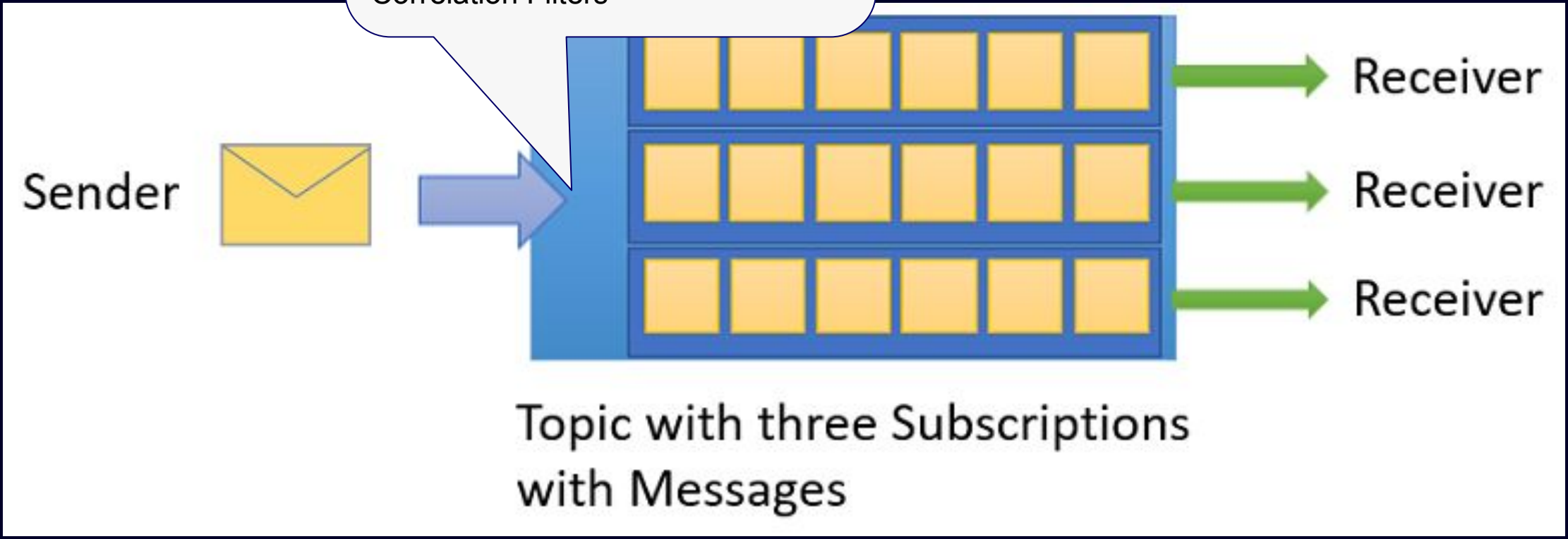
Queues offer **First In, First Out** (FIFO) message delivery to one or more compe ly receive and process messages in the order in which they were added to the qu receives and processes each message.

Peek Lock
A lock is acquired on the message. Receiver can then Complete or Abandon the processing attempt. At-least-once pattern.



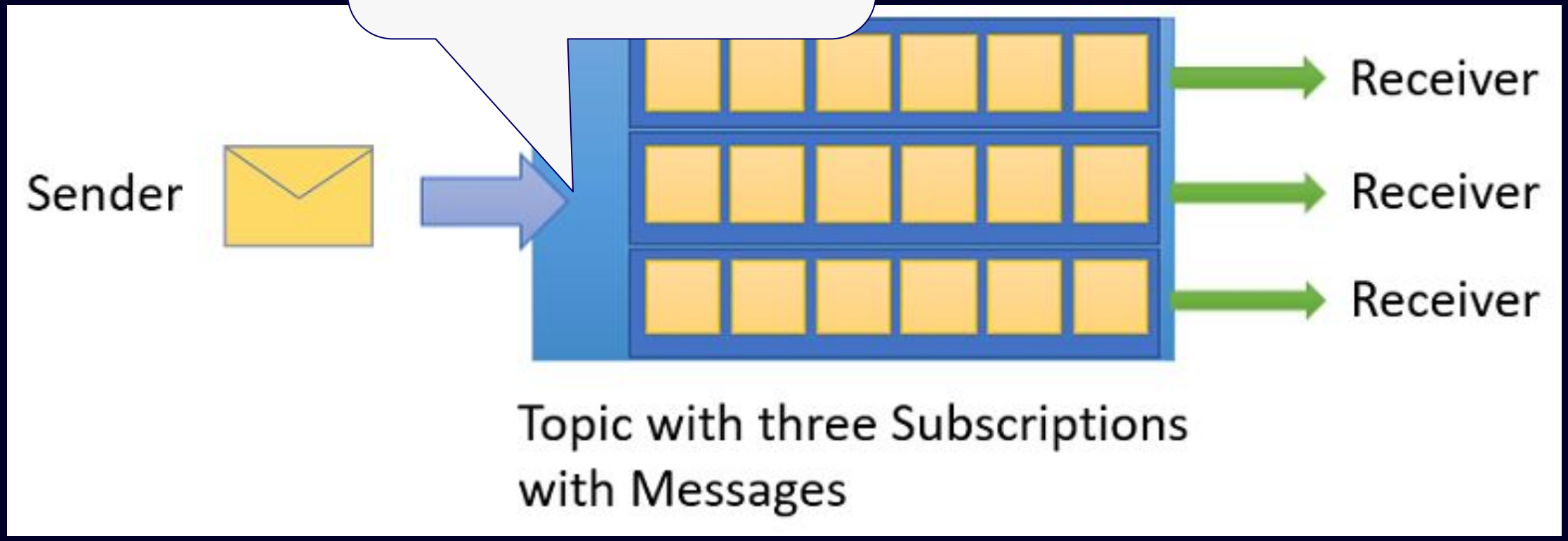

Sender
Receiver
Receiver
Receiver
Topic with three Subscriptions
with Messages

Filters
Can control which messages go to which subscription.
SQL Filters,
Boolean Filters,
Correlation Filters



azenix

Actions
Can modify message properties



Dead Letter Queue

Shadows every queue and subscription

Dead-letter reason	Dead-letter error description
HeaderSizeExceeded	The size quota for this stream has been exceeded.
TTLExpiredException	The message expired and was dead lettered. See the Time to live section for details.
Session ID is null.	Session enabled entity doesn't allow a message whose session identifier is null.
MaxTransferHopCountExceeded	The maximum number of allowed hops when forwarding between queues has been exceeded. This value is set to 4.
MaxDeliveryCountExceeded	Message couldn't be consumed after maximum delivery attempts. See the Maximum delivery count section for details.

Service Bus comes in Basic, standard, and premium tiers. Here's how they compare:

Feature	Basic	Standard	Premium
Queues	✔	✔	✔
Scheduled messages	✔	✔	✔
Topics		✔	✔
Transactions		✔	✔
De-duplication		✔	✔
Sessions		✔	✔
ForwardTo/SendVia		✔	✔
Message Size	256 KB	256 KB	100 MB
Resource isolation			✔
Geo-Disaster Recovery (Geo-DR)			✔ *Requires additional Service Bus Premium namespaces in another region.
Java Messaging Service (JMS) 2.0 Support			✔
Availability Zones (AZ) support			✔

Demo: Azure Portal

azenix

nuget Packages Upload Statistics Documentation Downloads Blog

Search for packages...

WindowsAzure.ServiceBus 6.2.2

Prefix Reserved

.NET Framework 4.6.2

.NET CLI | Package Manager | PackageReference | Paket CLI | Script & Interactive | Cake

```
> dotnet add package WindowsAzure.ServiceBus --version 6.2.2
```

README | Frameworks | Dependencies | Used By | Versions | Release Notes

azenix

nuget Packages Upload Statistics Documentation Downloads Blog

Search for packages...

WindowsAzure.ServiceBus 6.2.2

Prefix Reserved

.NET Framework 4.6.2

.NET CLI | Package Manager | PackageReference | Paket CLI | Script & Interactive | Cake

```
> dotnet add package WindowsAzure.ServiceBus --version 6.2.2
```

README | Frameworks | Dependencies | Used By | Versions | Release Notes

azenix

Microsoft.Azure.ServiceBus 5.2.0

Prefix Reserved

.NET Standard 2.0

⚠ This package has been deprecated.

Suggested Alternatives

Azure.Messaging.ServiceBus

Additional Details

Please note, a newer package is available at https://nuget.org/packages/Azure.Messaging.ServiceBus as of 11/2020.
While this package will continue to receive critical bug fixes, we strongly encourage you to upgrade.
See the Migration Guide at https://aka.ms/azsdk/net/migrate/sb for more details.

.NET CLI | Package Manager | PackageReference | Paket CLI | Script & Interactive | Cake

```
> dotnet add package Microsoft.Azure.ServiceBus --version 5.2.0
```

azenix

Microsoft.Azure.ServiceBus 5.2.0

Prefix Reserved

.NET Standard 2.0

⚠ This package has been deprecated.

Suggested Alternatives

Azure.Messaging.ServiceBus

Additional Details

Please note, a newer package is available at https://nuget.org/packages/Azure.Messaging.ServiceB as of 11/2020.
While this package will continue to receive critical bug fixes, we strongly encourage you to upgrade.
See the Migration Guide at https://aka.ms/azsdk/net/migrate/sb for more details.

.NET CLI | Package Manager | PackageReference | Paket CLI | Script & Interactive | Cake

```
> dotnet add package Microsoft.Azure.ServiceBus --version 5.2.0
```

azenix

nuget Packages Upload Statistics Documentation Downloads Blog

Search for packages...

Azure.Messaging.ServiceBus 7.13.1

Prefix Reserved

.NET Standard 2.0

.NET CLI | Package Manager | PackageReference | Paket CLI | Script & Interactive | Cake

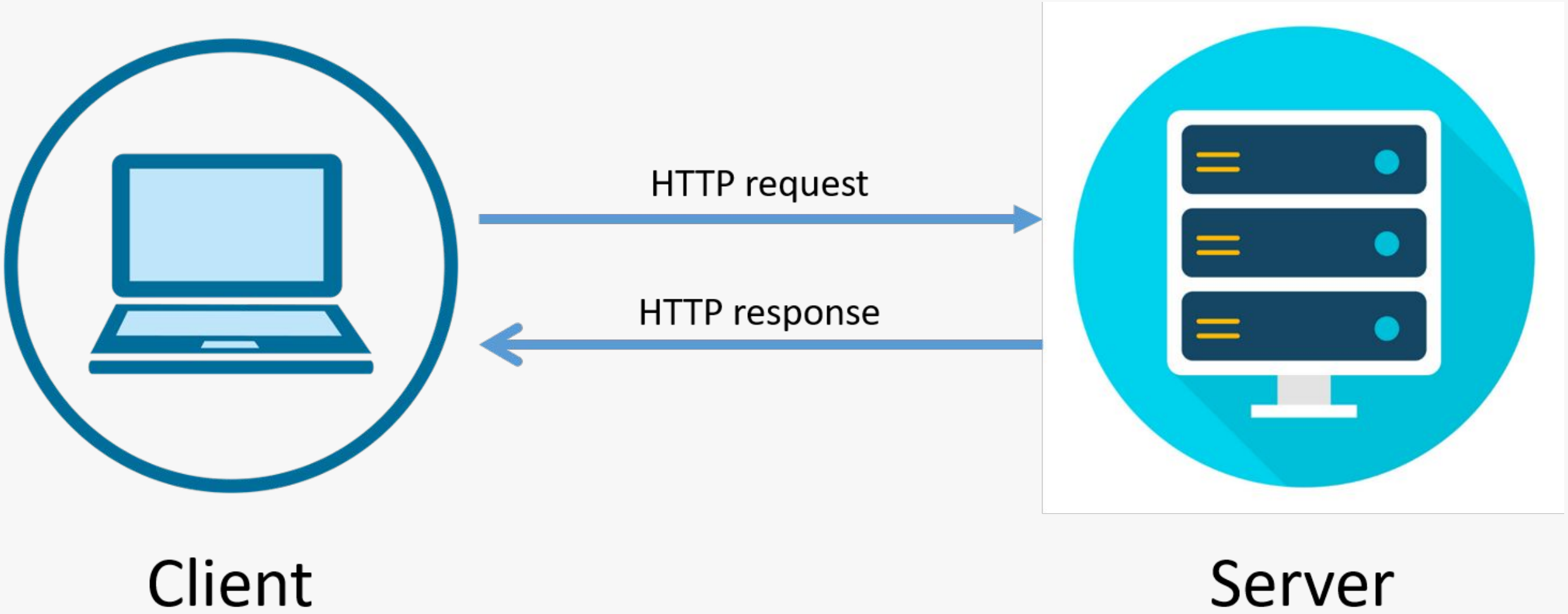
```
> dotnet add package Azure.Messaging.ServiceBus --version 7.13.1
```

Demo: .NET

Demo: Function Apps

azenix

- Core Components
- .NET Usage
- Azure Function Apps
- Message Based Architectures
- Infrastructure As Code


HTTP request
HTTP response
Client
Server

Commands vs Events

Command: ProcessOrder

Tell one other service to do something

Core path

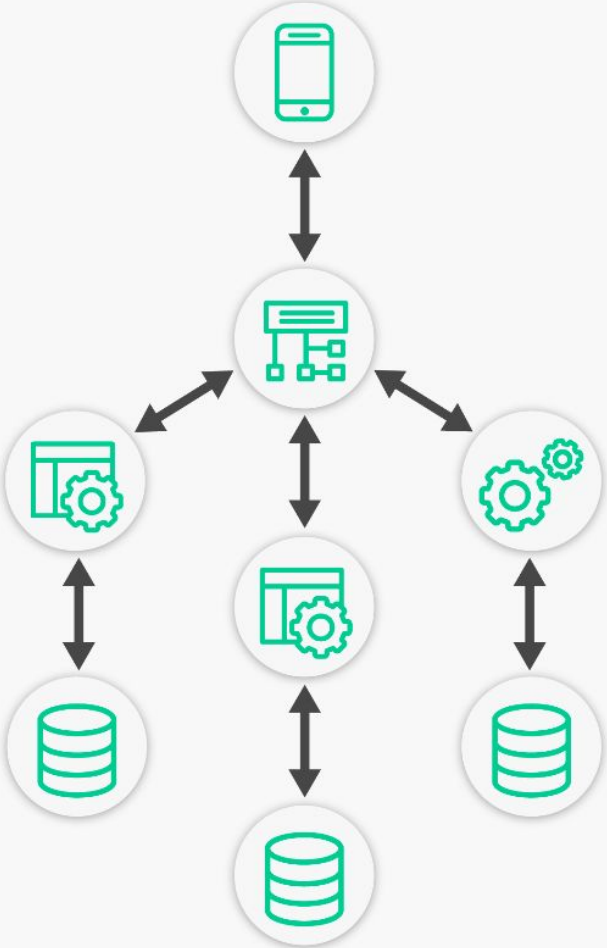
Event: OrderReceived

Notify 0-many services

Side Effects

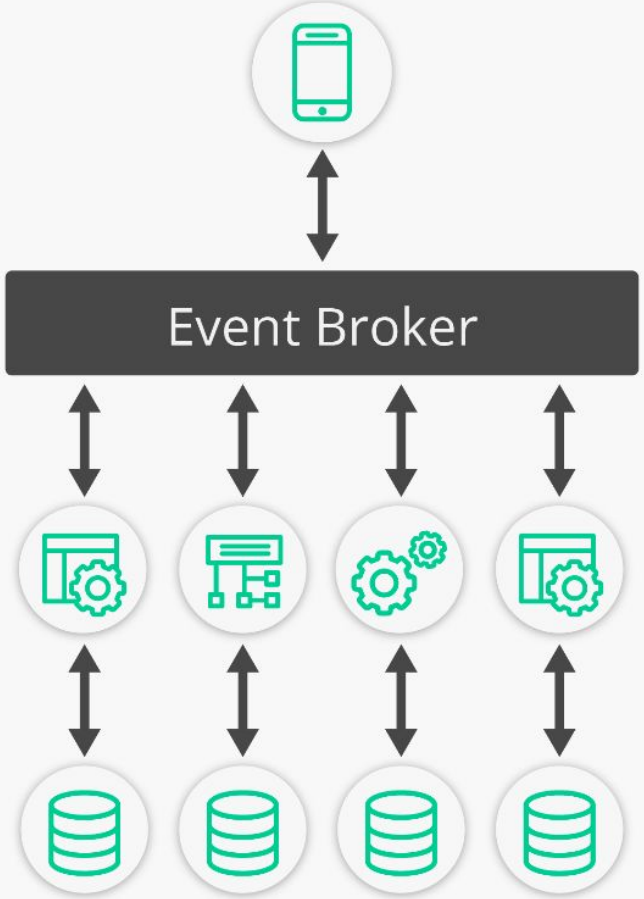
azenix

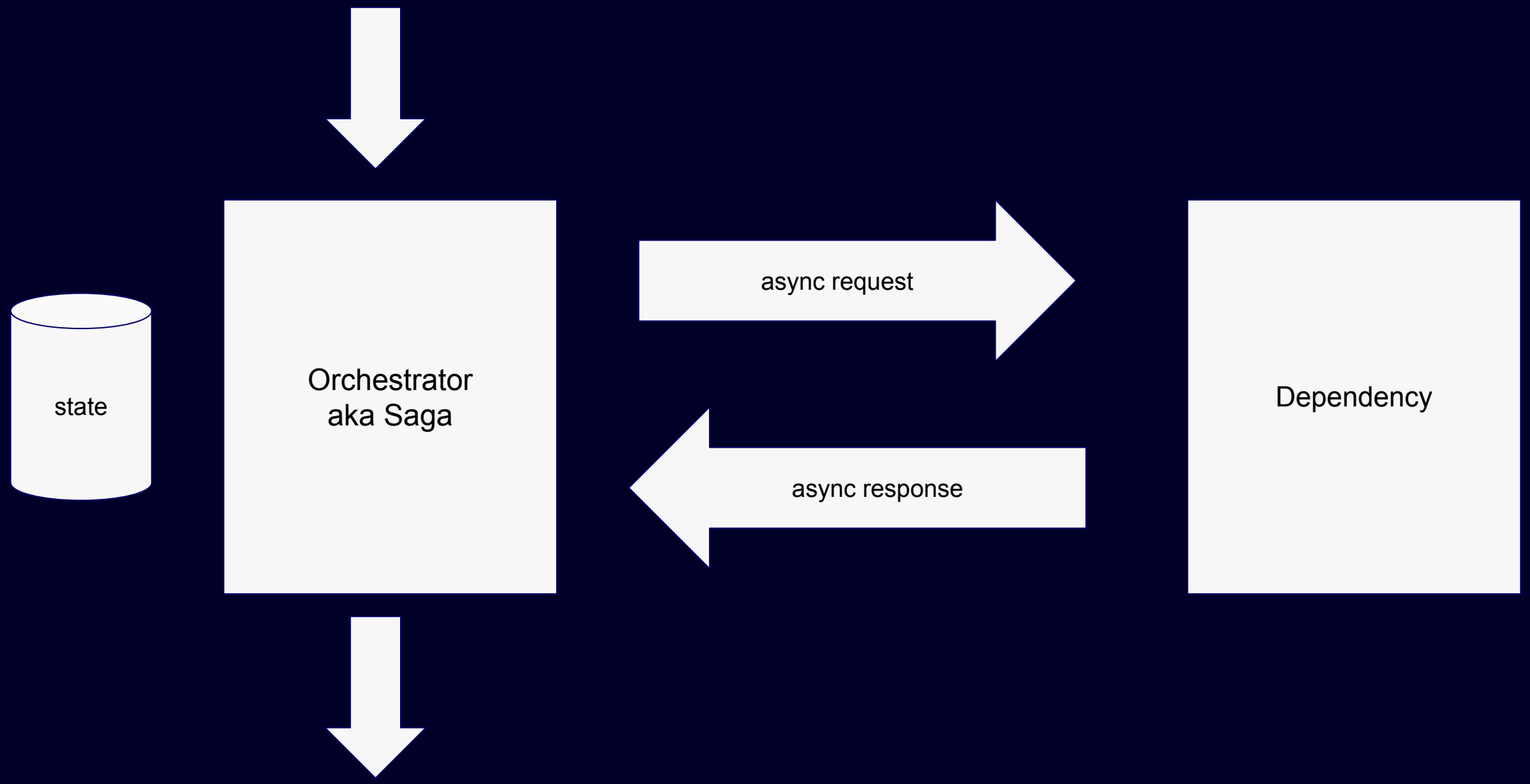
Orchestration

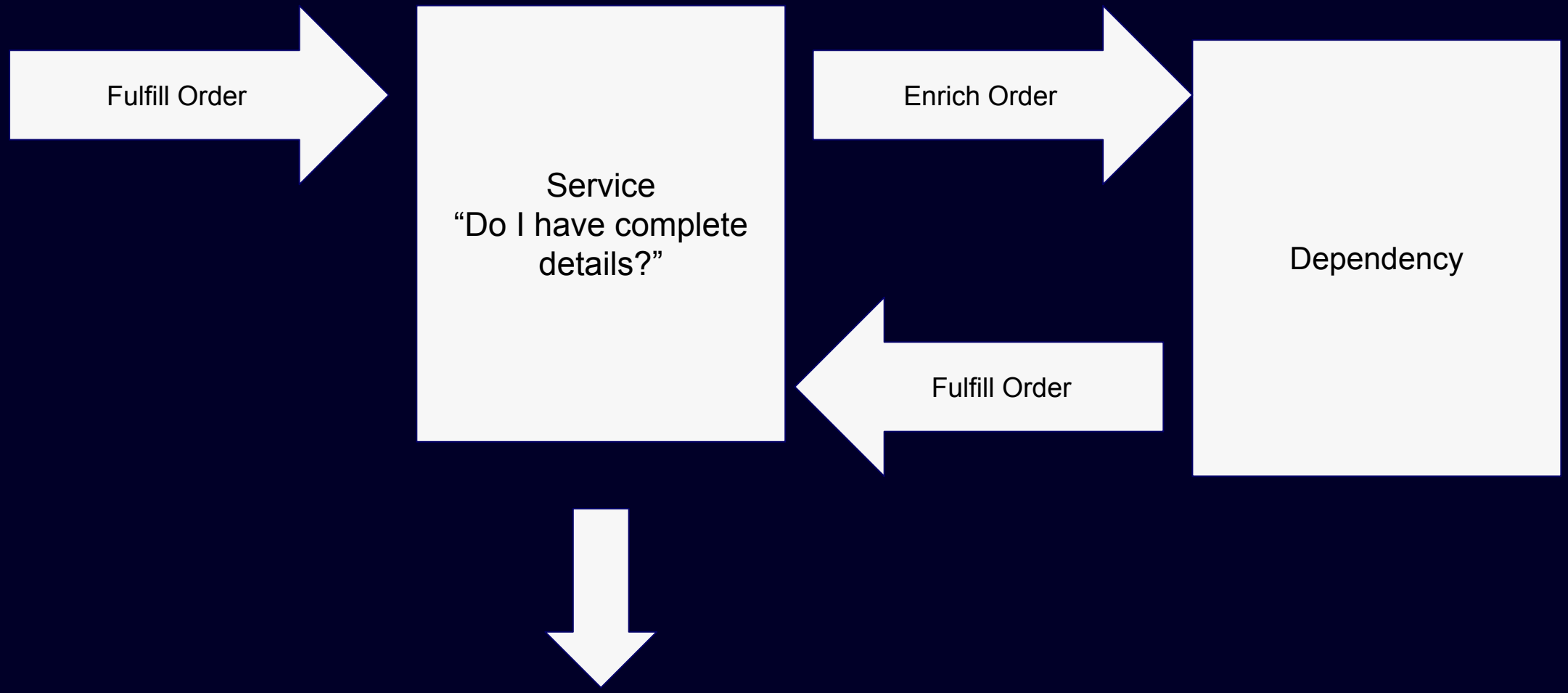


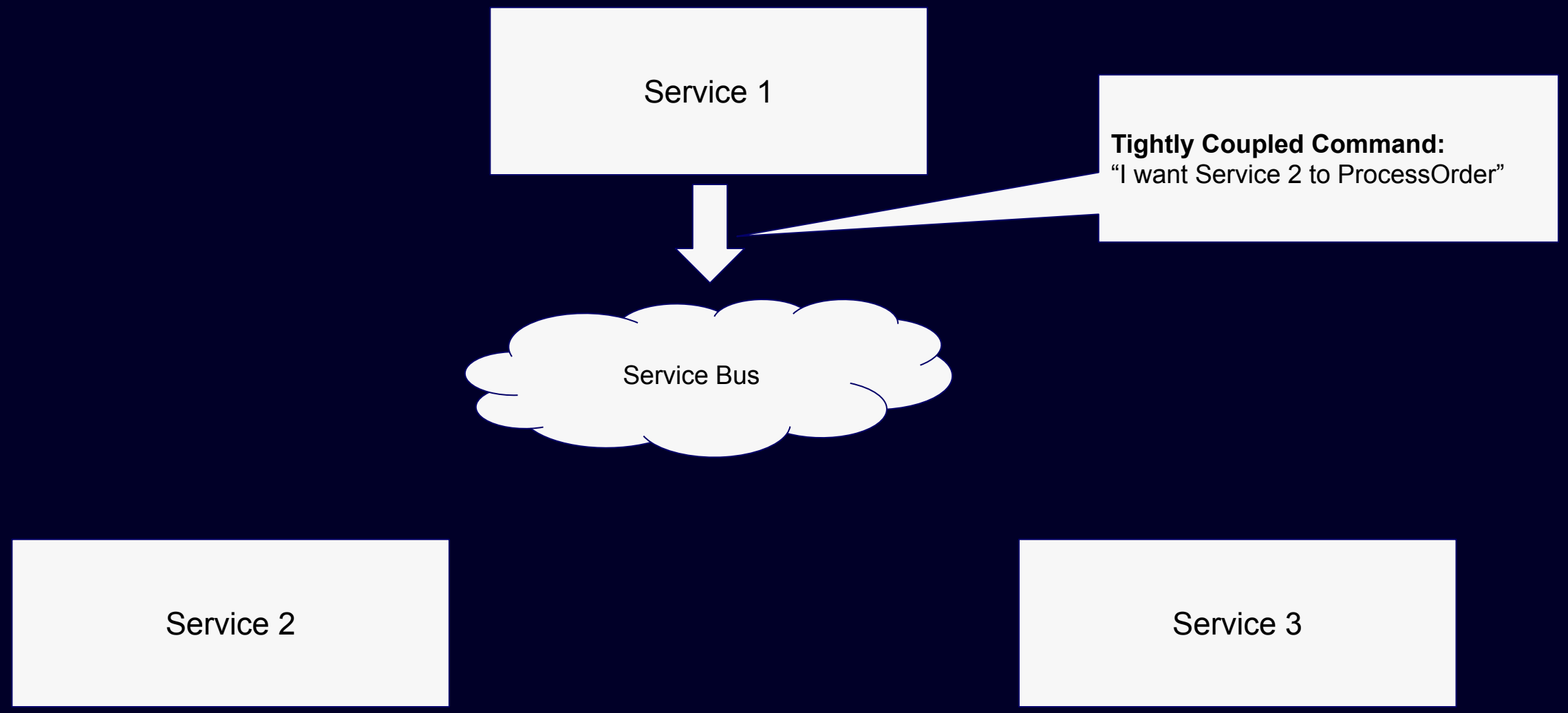
VS

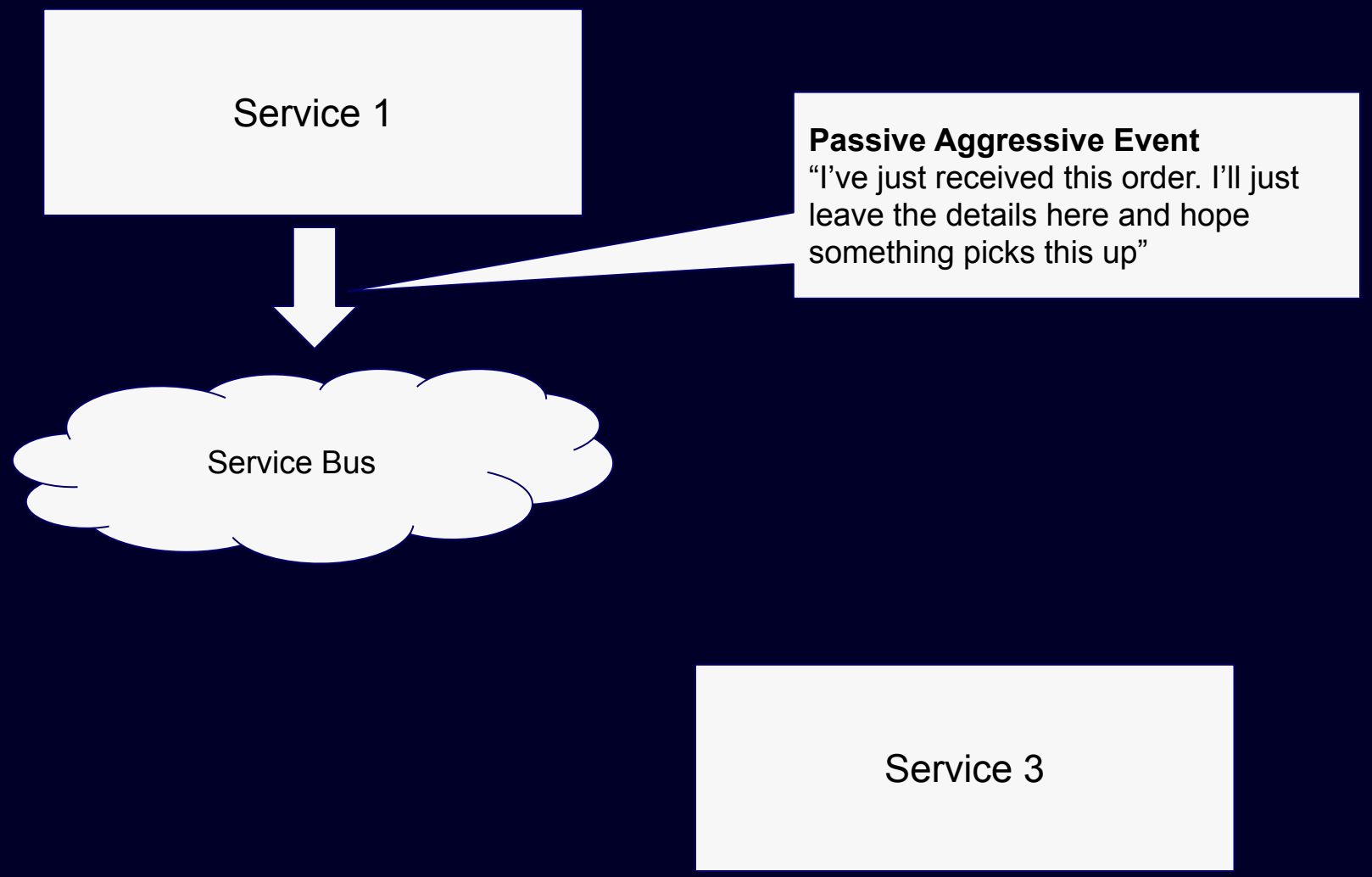
Choreography

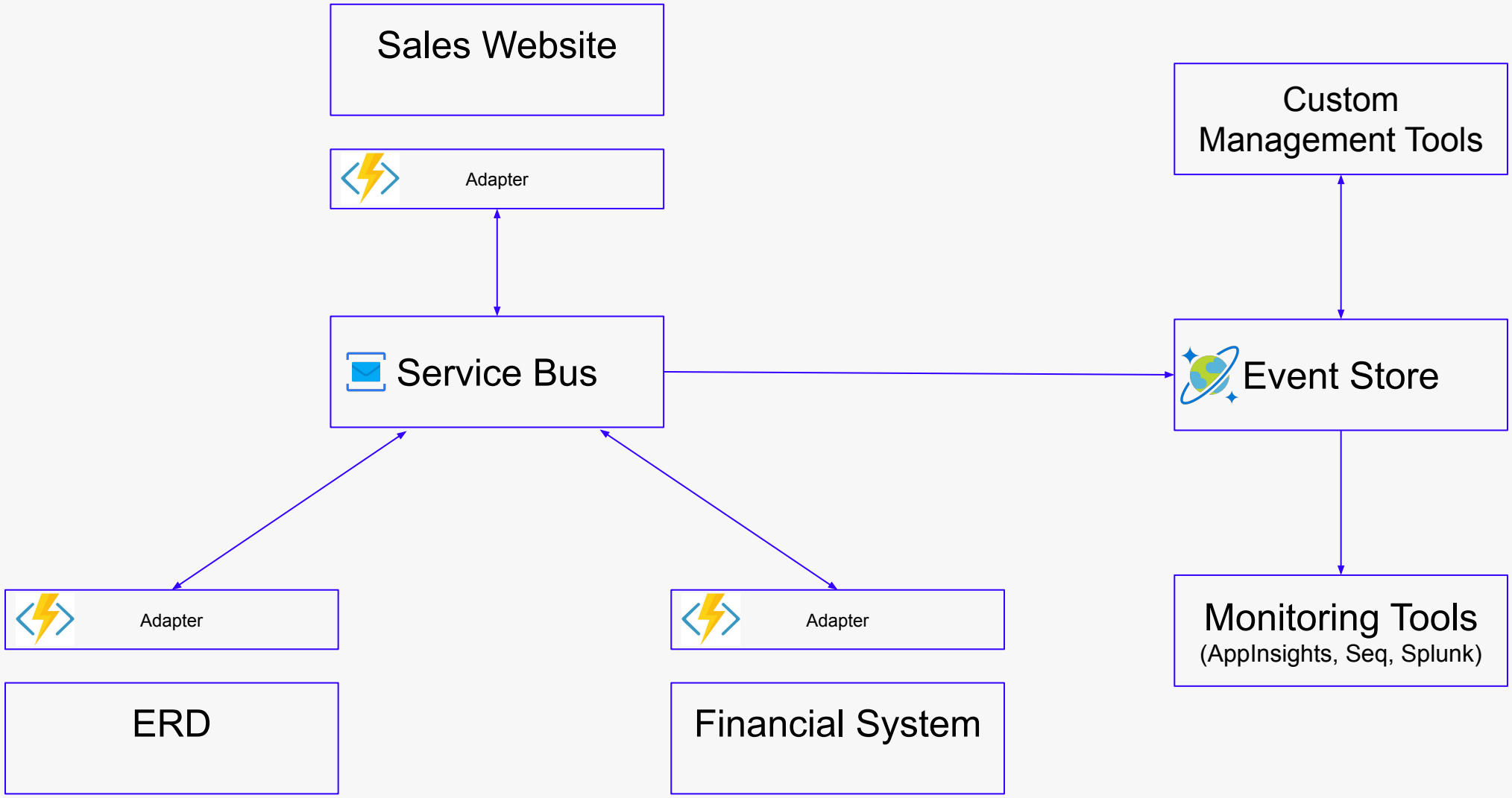



state
Orchestrator
aka Saga
async request
async response
Dependency


Fulfill Order
Service
“Do I have complete details?”
Enrich Order
Dependency
Fulfill Order


Service 1
Tightly Coupled Command:
“I want Service 2 to ProcessOrder”
Service Bus
Service 2
Service 3


Service 1
Passive Aggressive Event
“I’ve just received this order. I’ll just leave the details here and hope something picks this up”
Service Bus
Service 2
Service 3


Sales Website
Adapter
Service Bus
Adapter
ERD
Adapter
Financial System
Custom Management Tools
Event Store
Monitoring Tools
(AppInsights, Seq, Splunk)

Service Bus Deployment

Tools and deployment options

- Azure Portal
  - No local dev options
  - Service Bus Explorer
- Azure CLI
- Arm Templates
- Bicep
- Hashicorp Terraform
- Pulumi

```
using Microsoft.Azure.WebJobs;
using Microsoft.Extensions.Logging;

namespace ServiceBusDemo.FunctionApp;

public class MessageReceiver
{
    [FunctionName("MessageReceiver")]
    public static void Run(
        [ServiceBusTrigger("my-queue", Connection = "SERVICE_BUS_CONN_STR")]
        string myQueueItem,
        Int32 deliveryCount,
        DateTime enqueuedTimeUtc,
        string messageId,
        ILogger log)
    {
        log.LogInformation($"C# ServiceBus queue trigger function processed message: {myQueueItem}");
        log.LogInformation($"EnqueuedTimeUtc={enqueuedTimeUtc}");
        log.LogInformation($"DeliveryCount={deliveryCount}");
        log.LogInformation($"MessageId={messageId}");
    }
}
```

azenix

```
// get existing resource group
var resourceGroupName = config.Require("resource-group-name");
var resourceGroup = new ResourceGroup(resourceGroupName,
    new ResourceGroupArgs()
    {
        ResourceGroupName = resourceGroupName
    },
    new CustomResourceOptions() { ImportId = $"/subscriptions/{azureConfig.SubscriptionId}/resourceGroups/{resourceGroupName}" } )

// get existing service bus namespace
var serviceBusNamespaceName = config.Require("service-bus-namespace-name");
var serviceBusNamespace = new Namespace(serviceBusNamespaceName, new NamespaceArgs()
{
    NamespaceName = serviceBusNamespaceName,
    ResourceGroupName = resourceGroup.Name,
    Location = "Australia East",
    Sku = new SBSkuArgs() { Tier = SkuTier.Standard, Name = SkuName.Standard }
}, new CustomResourceOptions() { ImportId = $"/subscriptions/{azureConfig.SubscriptionId}/resourceGroups/{resourceGroupName}/provi


// add BookieBusTopic
var topic = new Topic("bookie-bus", new TopicArgs()
{
    TopicName = "bookie-bus",
    NamespaceName = serviceBusNamespace.Name,
    ResourceGroupName = resourceGroup.Name
});
```

azenix

```
3 references
public static IList<MethodInfo> FindFunctionMethods(Assembly assembly)
{
    return assembly.GetTypes().SelectMany(t => t.GetMethods()
            .Where(m => m.GetCustomAttributes(typeof(FunctionNameAttribute), false).Any()))
        .ToList();
}

2 references
public static IList<ServiceBusTriggerAttribute> FindServiceBusTriggers(MethodInfo method)
{
    return method.GetParameters().SelectMany(p => p.GetCustomAttributes(typeof(ServiceBusTriggerAttribute)))
        .OfType<ServiceBusTriggerAttribute>()
        .ToList();
}


1 reference
public static IList<ServiceBusAttribute> FindServiceBusAttributes(MethodInfo method)
{
    return method.GetParameters().SelectMany(p => p.GetCustomAttributes(typeof(ServiceBusAttribute)))
        .OfType<ServiceBusAttribute>()
        .ToList();
}
```

azenix

```
10 references
public class ServiceBusDetails
{
    6 references
    public IList<string> Queues { get; set; } = new List<string>();

    6 references
    public IList<string> Topics { get; set; } = new List<string>();

    4 references
    public IList<SubscriptionDetails> Subscriptions { get; set; } = new List<SubscriptionDetails>();

    3 references
    public void AddTopic(string name)
    {
        if (!Topics.Contains(name)) Topics.Add(name);
    }

    7 references
    public void AddQueue(string name)
    {
        if (!Queues.Contains(name)) Queues.Add(name);
    }
```

```
1 reference
private static void AddTopicsAndSubscriptions(ServiceBusDetails serviceBusDetails,
    Namespace serviceBusNamespace, ResourceGroup resourceGroup)
{
    foreach (var topicName in serviceBusDetails.Topics)
    {

        var topic = new Topic(topicName, new TopicArgs()
        {
            Name = topicName,
            ResourceGroupName = resourceGroup.Name,
            NamespaceName = serviceBusNamespace.Name
        });
        foreach (var subDetail in serviceBusDetails.Subscriptions
            .Where(s => s.TopicName == topicName))
        {
            // logical name must be unique as per https://github.com/pulumi/pulumi/issues/5814
            // name from SubscriptionArgs sets the actual name. we have duplicate sub names across different topics.
            var sub = new Subscription($"{topicName}-{subDetail.SubscriptionName}", new SubscriptionArgs()
            {
                Name = subDetail.SubscriptionName,
                ResourceGroupName = resourceGroup.Name,
                TopicName = topic.Name,
                NamespaceName = serviceBusNamespace.Name,
                MaxDeliveryCount = 10
            });
        }
    }
```

azenix

- Core Components
- .NET Usage
- Azure Function Apps
- Message Based Architectures
- Infrastructure As Code

azenix

What we're about

Automate

Automated continuous delivery allows for incremental benefits to be delivered immediately, so that from day one your cloud platform is providing benefits

Innovate

We delve deeply to understand your business needs; we're adventurous and collaborative in our quest for truly fit-for-purpose solutions - enabling business outcomes

Transform

The best part of our job is delivering a platform that meets objectives and delivers value, scale and opportunity to our customers

Azure

.NET

azenix

Links

https://github.com/
brendan-nobadthing/
AzureServiceBus-for-dotnet

www.azenix.com.au

brendan-nobadthing

brendan richards