Versioning Messages

Mark Heath
CLOUD ARCHITECT

@mark_heath www.markheath.net

Overview

Benefits of messages

- Commands and events

Avoiding breaking changes

Creating a new version of a message

Messaging Benefits

Improved availability

Messages can be retried

Responsive front-end

- E.g. Post a new “order” message
- Handle it in the background
- Send the user a confirmation later

Scaling to process messages in parallel

Two Types of Message

Event

Something happened

e.g. OrderShipped

Publish and subscribe

Outgoing

Command

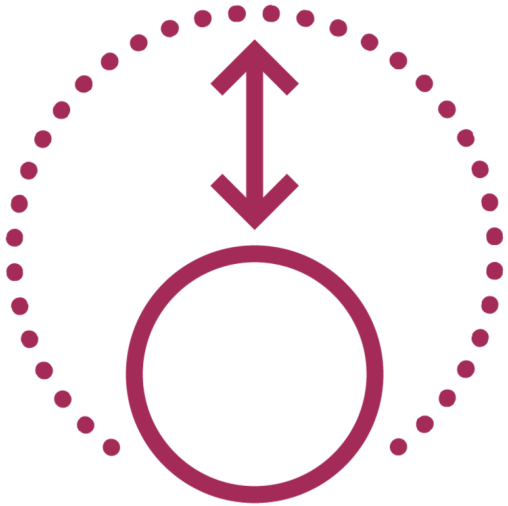
Request an action

Only handled once

e.g. ProcessPayment

Incoming

Changing Messages



Messages are serialized (e.g. JSON)

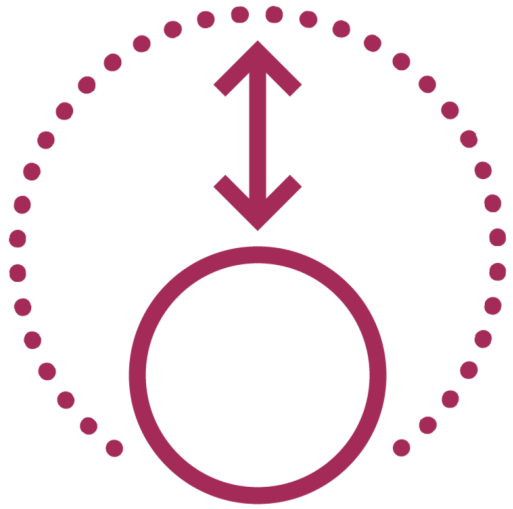
The recipient deserializes the message

New code may still receive old messages

Additive changes are safe (e.g. new property)

- Command messages may be missing the new property
- Event handlers may ignore the new property

Breaking Changes



Renaming a property

Changing the type of a property

- e.g. string -> string[]

Introduce a new version of the message

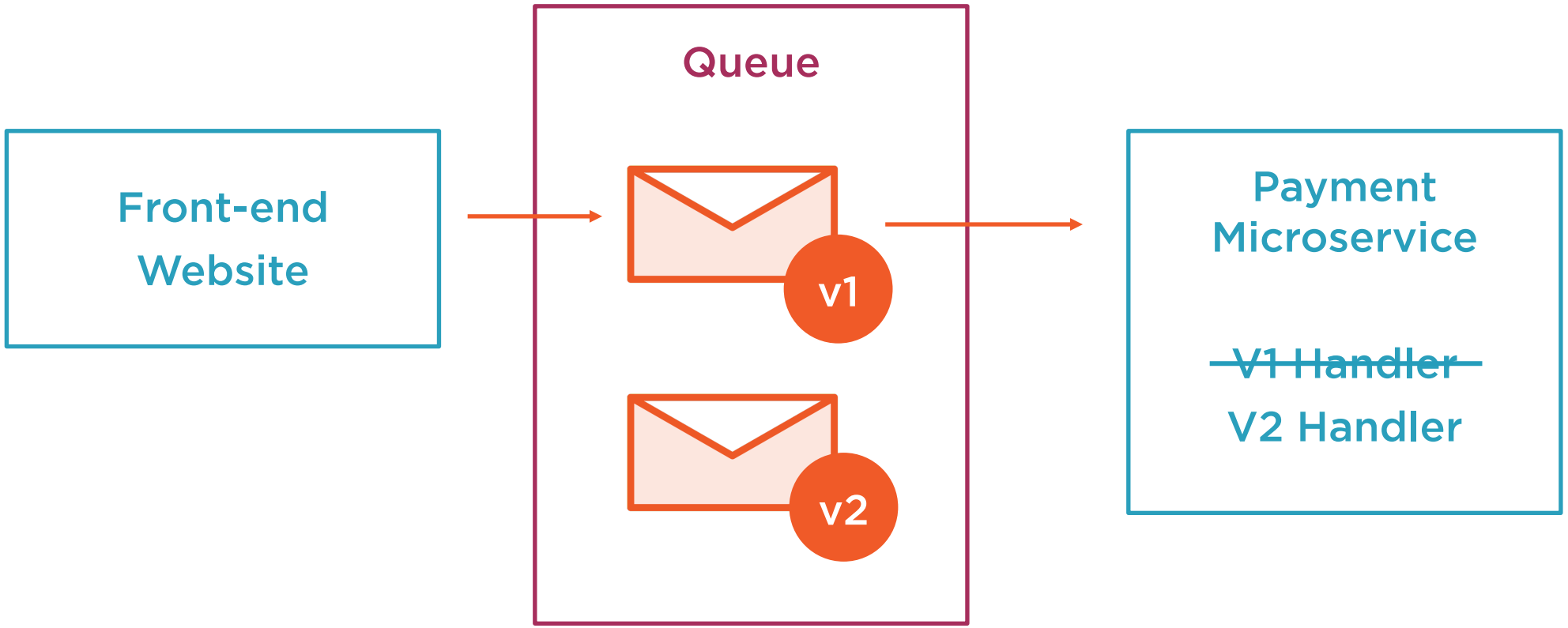
- Metadata can include message version

Recipient can deserialize to the correct type

- Or ignore unknown versions

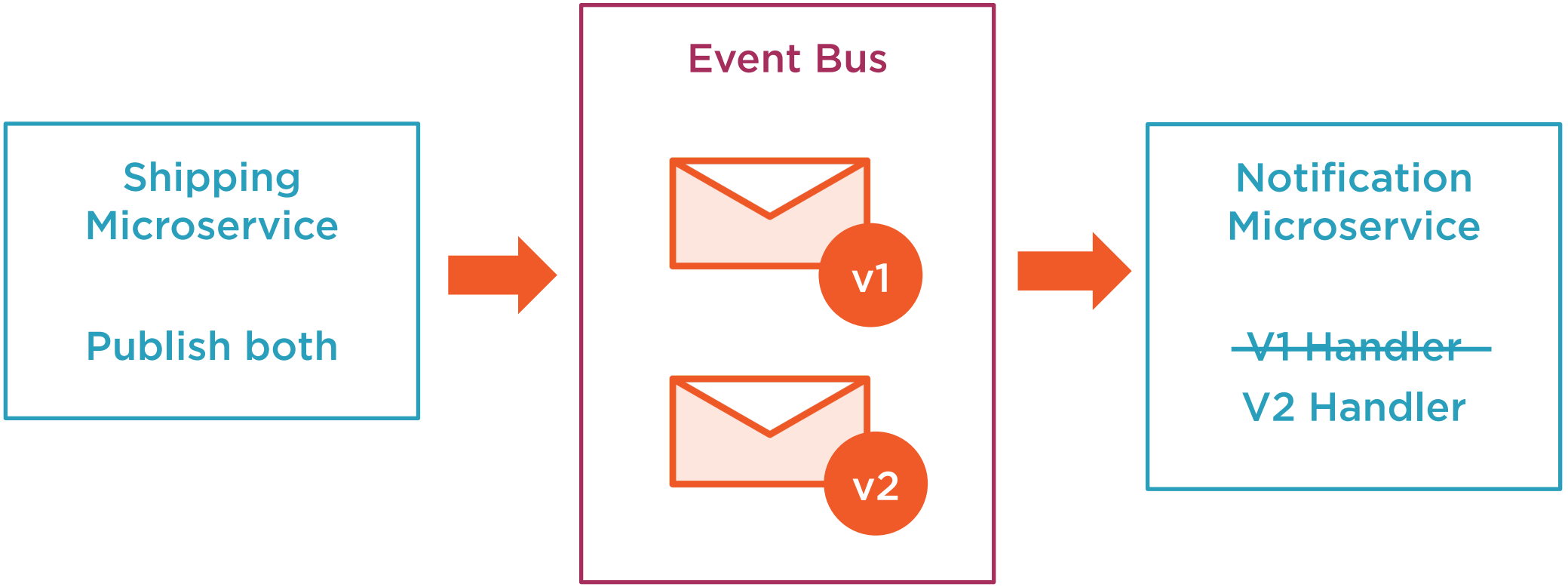
Updating a Command Message

e.g. ProcessPayment command message



Updating an Event Message

e.g. OrderShipped event message



Demo

Introduce a new message version

Strictly control all changes to messages that pass between microservices

Demo

Testing Backwards Compatibility

- Publish an old message
- Publish a new message
- Start the Payment microservice
- Check we can handle both messages

Summary

Maintain backwards compatibility for messages

Make additive changes wherever possible

Command and event messages

Temporarily handle both old and new message versions

- Legacy handlers can be retired later

Key Takeaways

Avoid breaking changes by making additive changes

Don’t assume microservices will be upgraded at the same instance

Write integration tests to verify backwards compatibility

Building Microservices with ASP.NET Core

This course is part of a learning path on Pluralsight

Microservices: The Big Picture	Getting Started	Microservices Communication
Data Management	Securing Microservices	**Versioning**
Deploying Microservices	Cross-cutting concerns	Scalability and availability