© The Ministry of Software Design Ltd.

Every Event Everywhere All at Once



Jacqui Read

Chief Architect
@ The Ministry of Software Design
Author of *Communication Patterns*

JacquiRead.com
linkedin.com/in/jacquelineread
fosstodon.org/@tekiegirl
tekiegirl.bsky.social

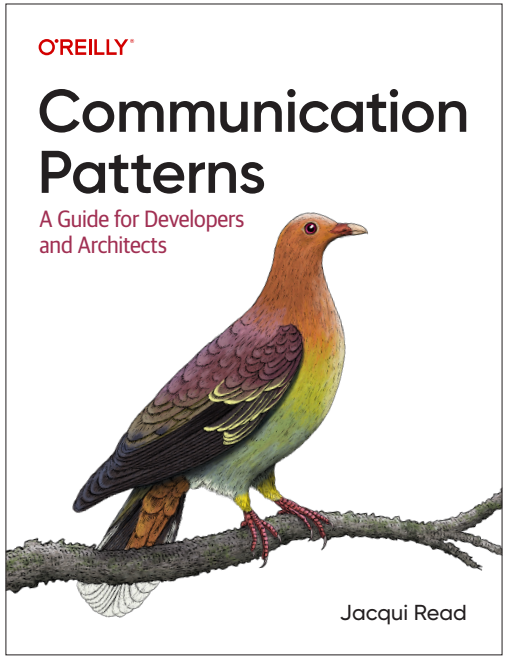
© The Ministry of Software Design Ltd.

~~discombobulating~~

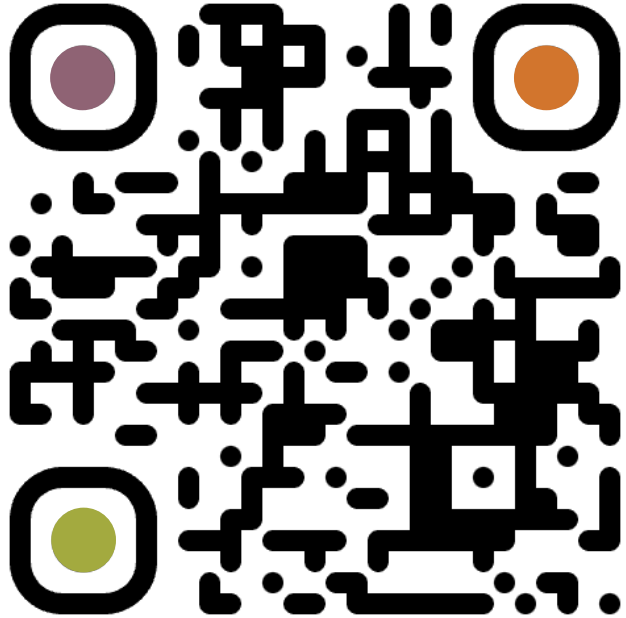
disambiguation



© The Ministry of Software Design Ltd.









Jacqui Read

Chief Architect
@ The Ministry of Software Design
Author of *Communication Patterns*

JacquiRead.com
linkedin.com/in/jacquelineread
fosstodon.org/@tekiegirl
tekiegirl.bsky.social

© The Ministry of Software Design Ltd.

All-the-Things Event

Calendar Event

Clickbait Event

Command

Domain Event

Event

Event-Carried State Transfer

Event-Driven

Event Granularity

Event Handler

Event Listener

Event Sourcing

Event Store

Event Storming

Event Streaming

Integration Event

Pivotal Event

Private Event

Public Event

Tombstone Event

© The Ministry of Software Design Ltd.

Event	Calendar Event	Command	Event Listener	Event Handler
Event Streaming	Event-Driven	Event Storming	Pivotal Event	Event Sourcing
Event Store	Tombstone Event	Domain Event	Integration Event	Private Event
Public Event	Event Granularity	Event-Carried State Transfer	Clickbait Event	All-the-Things Event

© The Ministry of Software Design Ltd.

What is an Event?



© The Ministry of Software Design Ltd.

event

noun [ih-vent]

1. an occurrence;
   something that happens

from https://en.wiktionary.org/wiki/event#Noun

© The Ministry of Software Design Ltd.

calendar
event

noun [ih-vent]

2. a prearranged social activity



from https://en.wiktionary.org/wiki/event#Noun

© The Ministry of Software Design Ltd.

event

noun [ih-vent]

4. an end result; an outcome

from https://en.wiktionary.org/wiki/event#Noun

© The Ministry of Software Design Ltd.

computing event:
something that has happened

© The Ministry of Software Design Ltd.

compact1, compact2, compact3
java.util

Class EventObject

java.lang.Object
    java.util.EventObject

All Implemented Interfaces:

Serializable

Direct Known Subclasses:

AWTEvent, BeanContextEvent, CaretEvent, ChangeEvent, ConnectionEvent, DragGestureEvent, DragSourceEvent, DropTargetEvent, FlavorEvent, HandshakeCompletedEvent, HyperlinkEvent, LineEvent, ListDataEvent, ListSelectionEvent, MenuEvent, NamingEvent, NamingExceptionEvent, NodeChangeEvent, Notification, PopupMenuEvent, PreferenceChangeEvent, PrintEvent, PropertyChangeEvent, RowSetEvent, RowSorterEvent, SSLSessionBindingEvent, StatementEvent, TableColumnModelEvent, TableModelEvent, TreeExpansionEvent, TreeModelEvent, TreeSelectionEvent, UndoableEditEvent, UnsolicitedNotificationEvent

---

```
public class EventObject
extends Object
implements Serializable
```

The root class from which all event state objects shall be derived.

All Events are constructed with a reference to the object, the "source", that is logically deemed to be the object upon which the Event in question initially occurred upon.

Since:

JDK1.1

See Also:

Serialized Form

Class EventObject

java.lang.Object
    java.util.EventObject

All Implemented Interfaces:

Serializable

Direct Known Subclasses:

AWTEvent, BeanContextEvent, CaretEvent, ChangeEvent, ConnectionEvent, DragGestureEvent, DragSourc
HyperlinkEvent, LineEvent, ListDataEvent, ListSelectionEvent, MenuEvent, NamingEvent, NamingExcep
PreferenceChangeEvent, PrintEvent, PropertyChangeEvent, RowSetEvent, RowSorterEvent, SSLSessionBi
TreeExpansionEvent, TreeModelEvent, TreeSelectionEvent, UndoableEditEvent, UnsolicitedNotificatio



© The Ministry of Software Design Ltd.

```
public class EventObject
extends Object
implements Serializable
```

The root class from which all event state objects shall be derived.

All Events are constructed with a reference to the object, the "source", that is logically deemed to be the object up

Since:

JDK1.1

java.util.prefs

Class PreferenceChangeEvent

java.lang.Object
    java.util.EventObject
        java.util.prefs.PreferenceChangeEvent

All Implemented Interfaces:

Serializable

```
public class PreferenceChangeEvent
extends EventObject
```

An event emitted by a `Preferences` node to indicate that a preference has been added, removed or has had its value changed.

Note, that although PreferenceChangeEvent inherits Serializable interface from EventObject, it is not intended to be Serializable. Appropriate serialization methods are implemented to throw NotSerializableException.

Since:

1.4

See Also:

Preferences, PreferenceChangeListener, NodeChangeEvent

© The Ministry of Software Design Ltd.

© The Ministry of Software Design Ltd.

Constructor Detail

PreferenceChangeEvent

```
public PreferenceChangeEvent(Preferences node,
                             String key,
                             String newValue)
```

Constructs a new PreferenceChangeEvent instance.

Parameters:

node - The Preferences node that emitted the event.

key - The key of the preference that was changed.

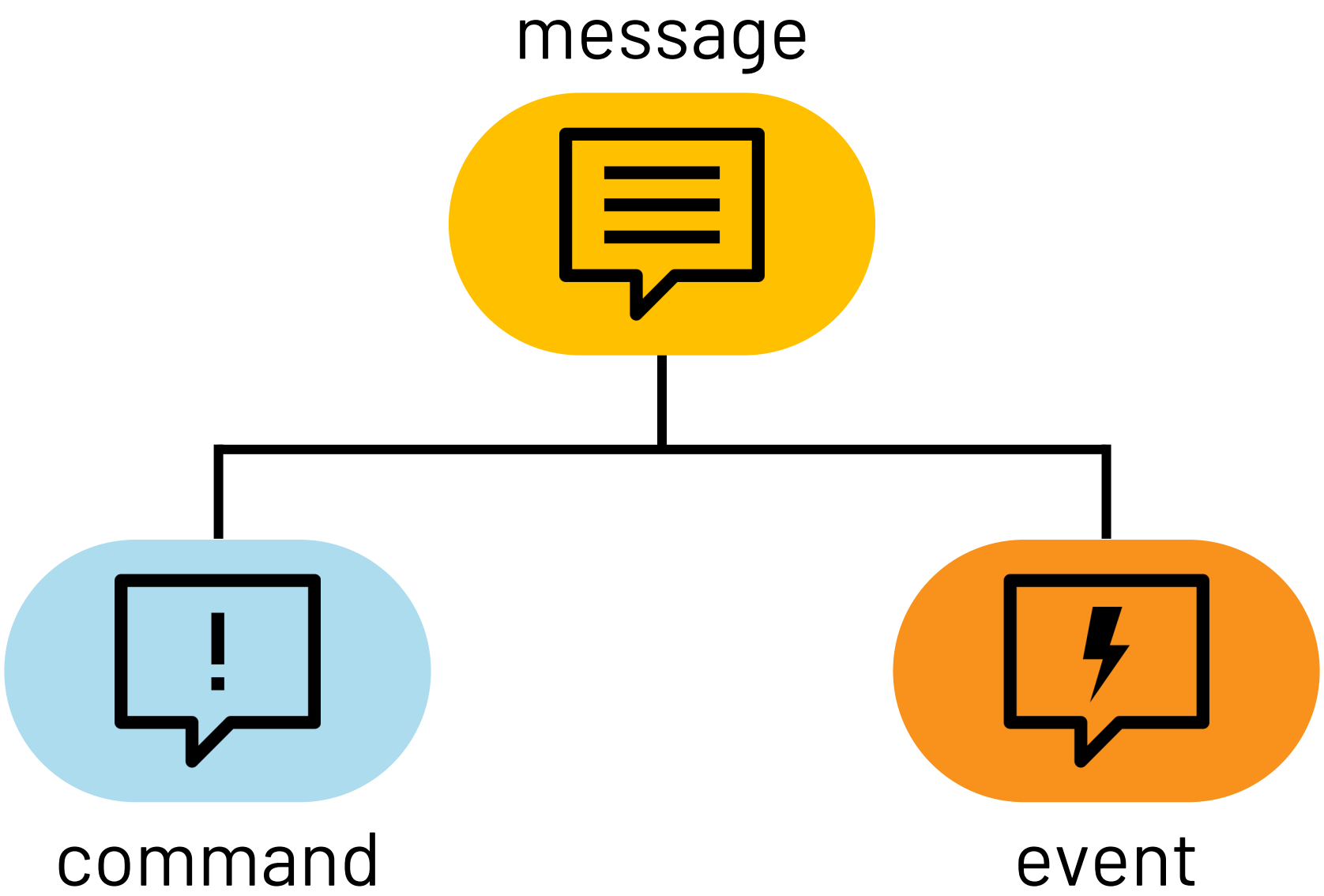
newValue - The new value of the preference, or null if the preference is being removed

© The Ministry of Software Design Ltd.

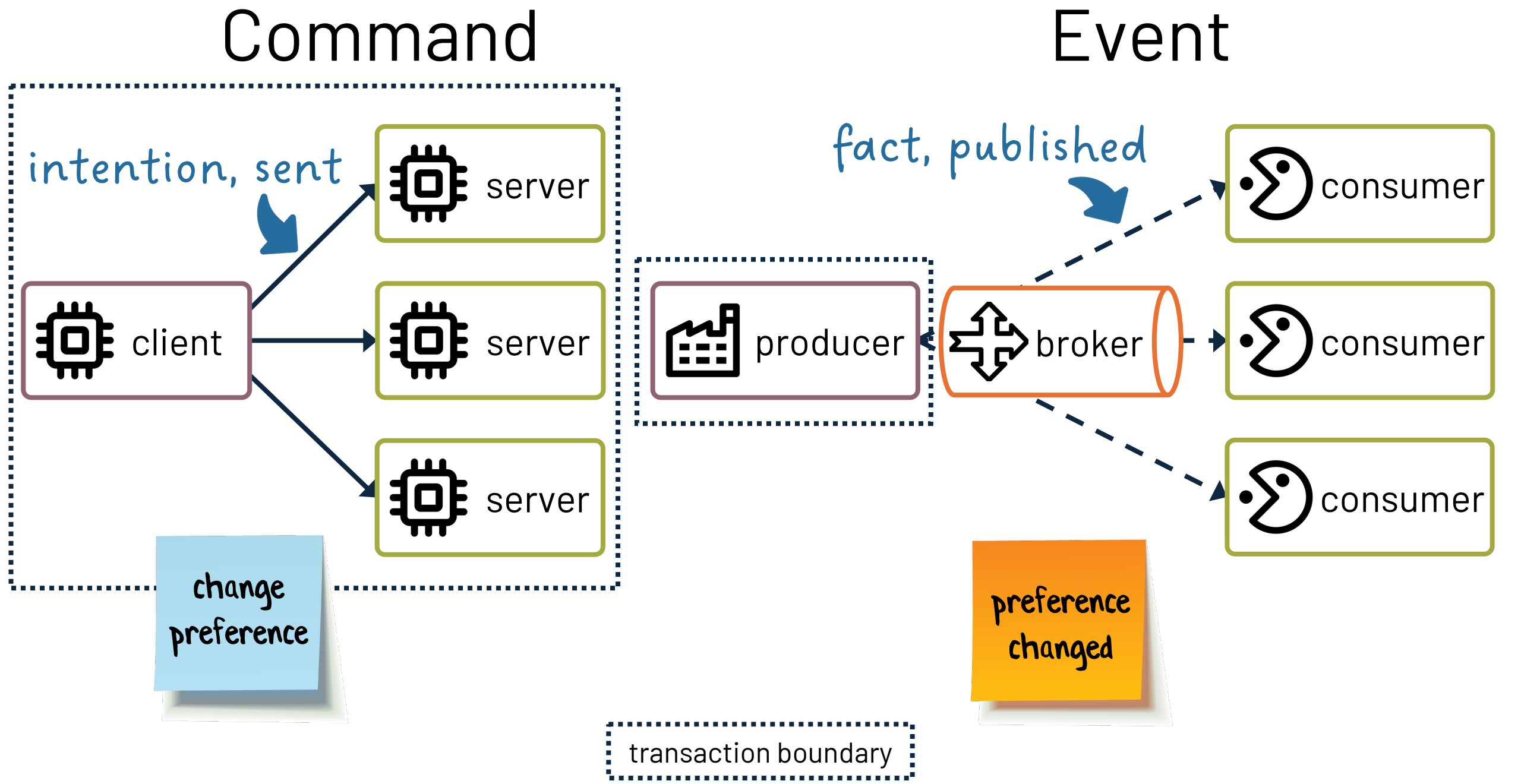
Event vs. Command



© The Ministry of Software Design Ltd.



© The Ministry of Software Design Ltd.



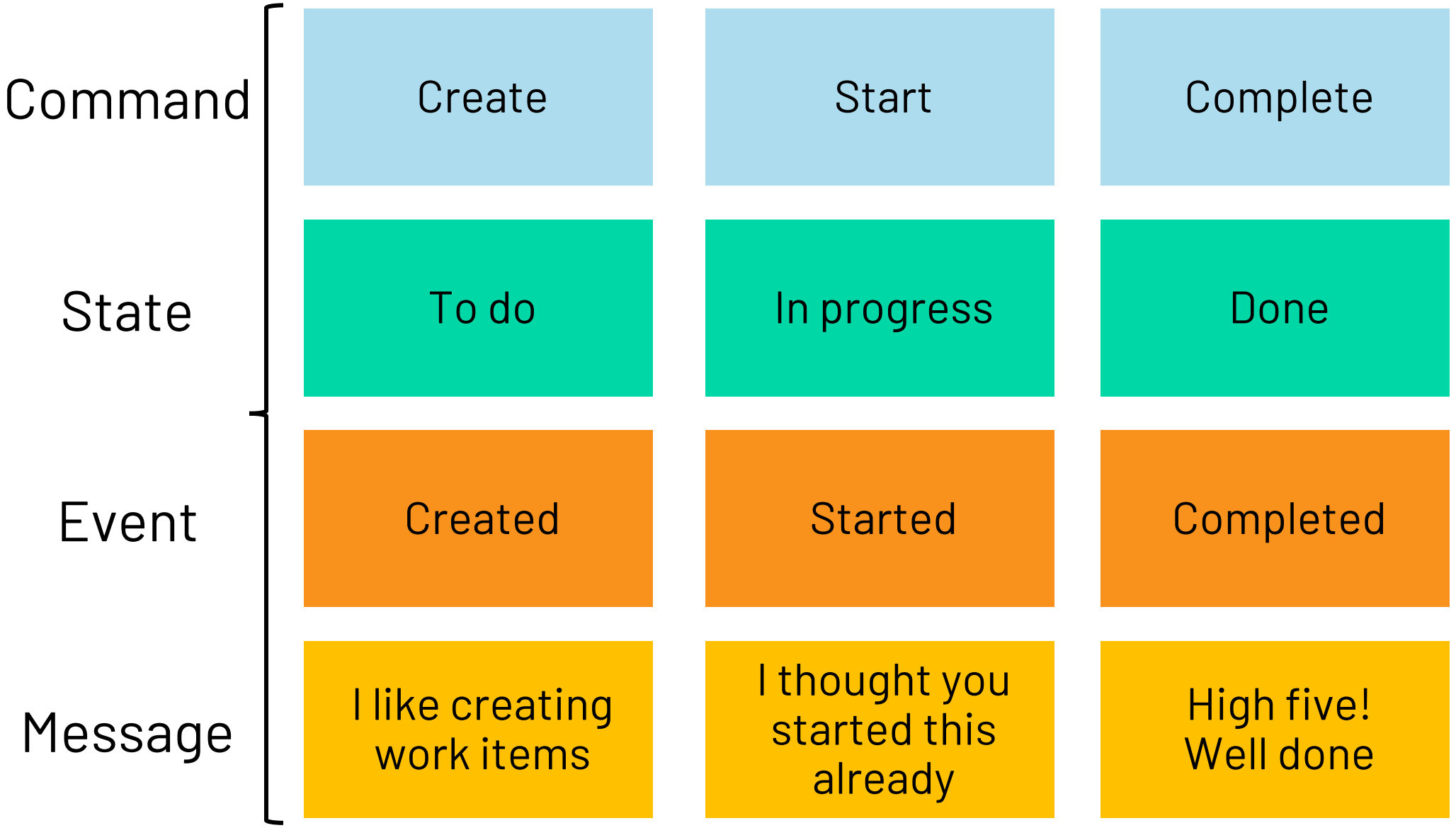
© The Ministry of Software Design Ltd.



https://event-driven.io/en/whats_the_difference_between_event_and_command/

© The Ministry of Software Design Ltd.

Work Item



© The Ministry of Software Design Ltd.

Event Listener & Event Handler



© The Ministry of Software Design Ltd.

Handler



triggers



Listener

Callback

© The Ministry of Software Design Ltd.

Event Streaming

continuous flows of data

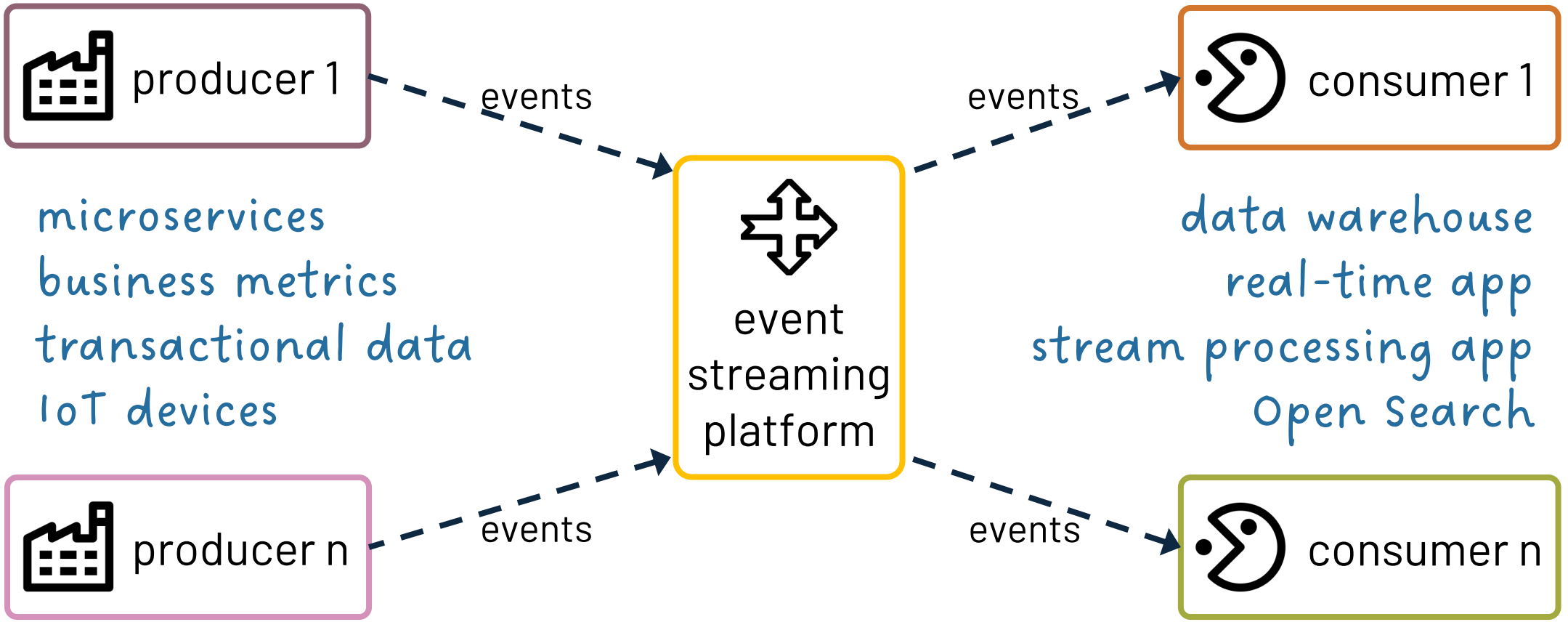




© The Ministry of Software Design Ltd.

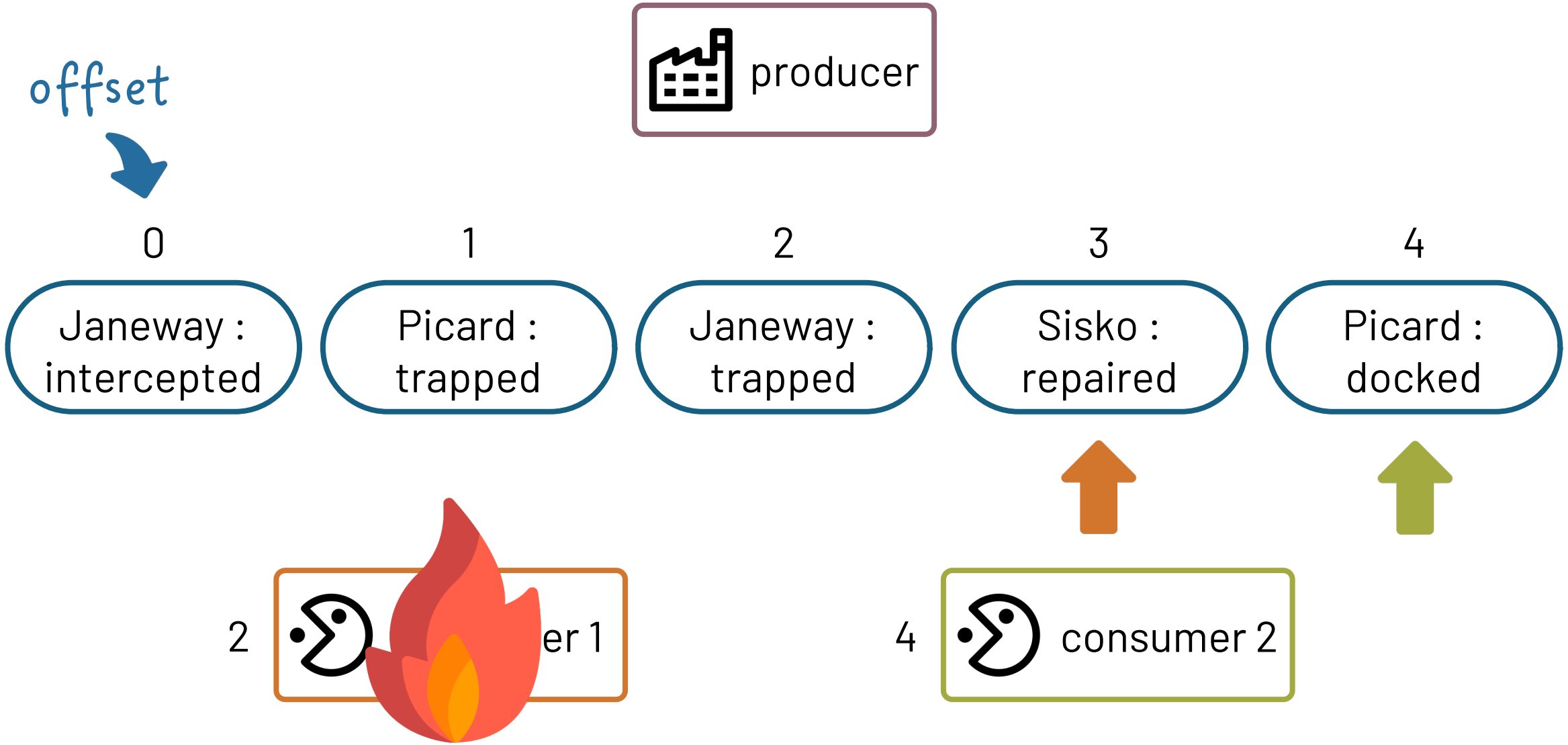
analytical

Event Streaming



© The Ministry of Software Design Ltd.

Message Log



© The Ministry of Software Design Ltd.

Message Queue



producer

consumer 1

Sisko : repaired

Janeway : trapped

Picard : trapped

Janeway : intercepted

consumer 2

© The Ministry of Software Design Ltd.

Message Queue



producer

consumer 1

Sisko : repaired

Janeway : trapped

Picard : trapped

consumer 2

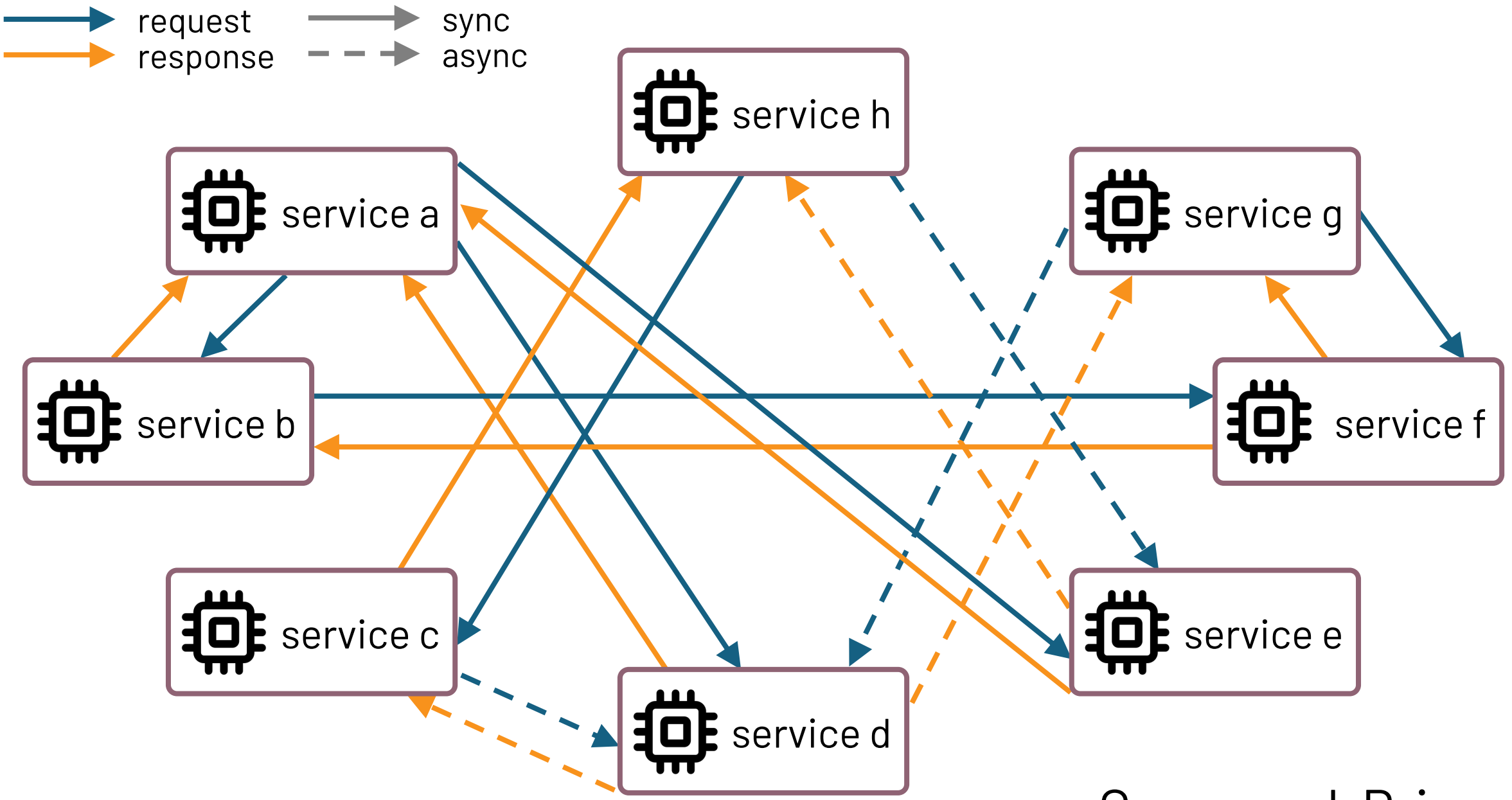
© The Ministry of Software Design Ltd.



Event-Driven Architecture

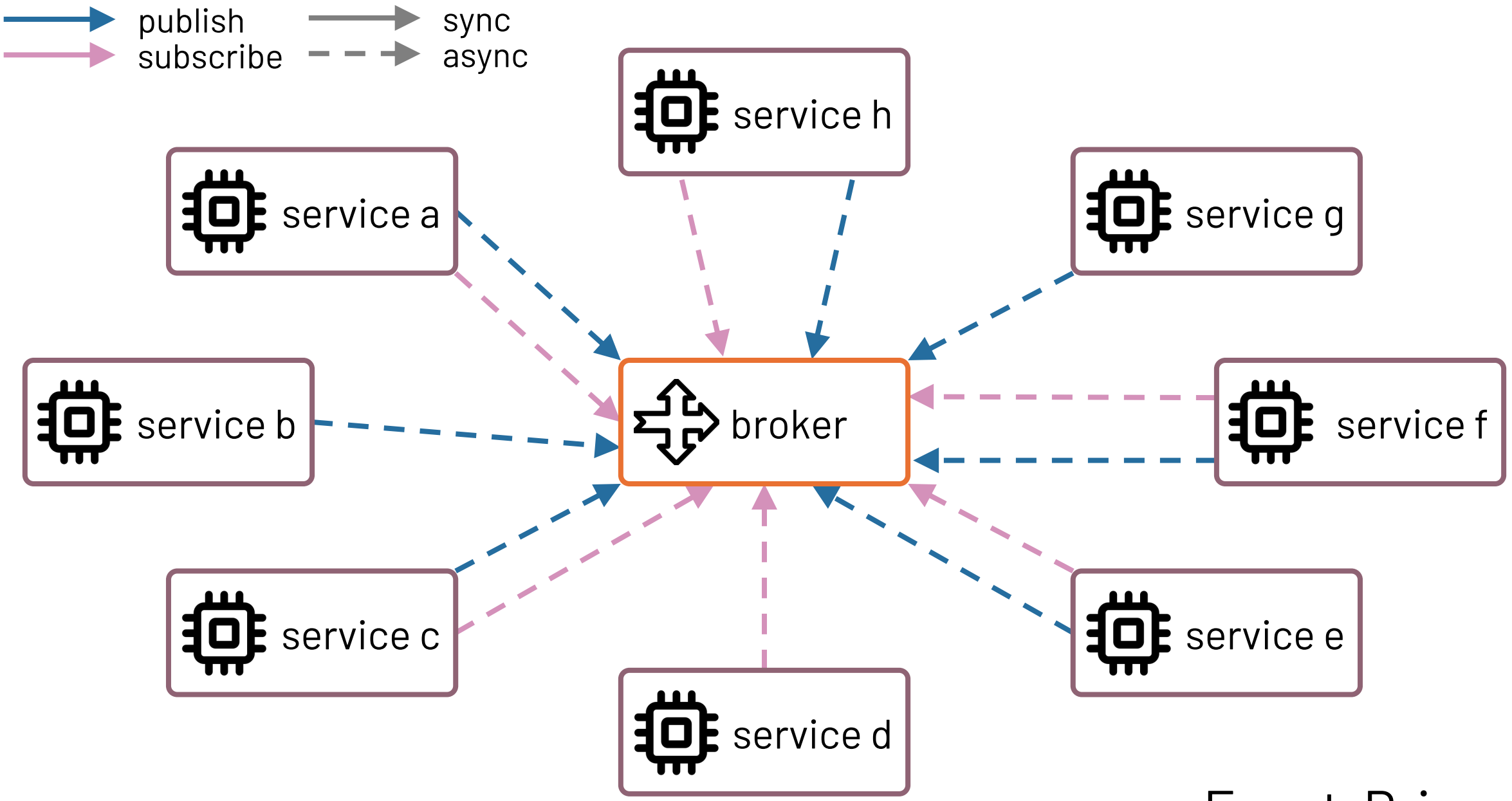
EDA

© The Ministry of Software Design Ltd.



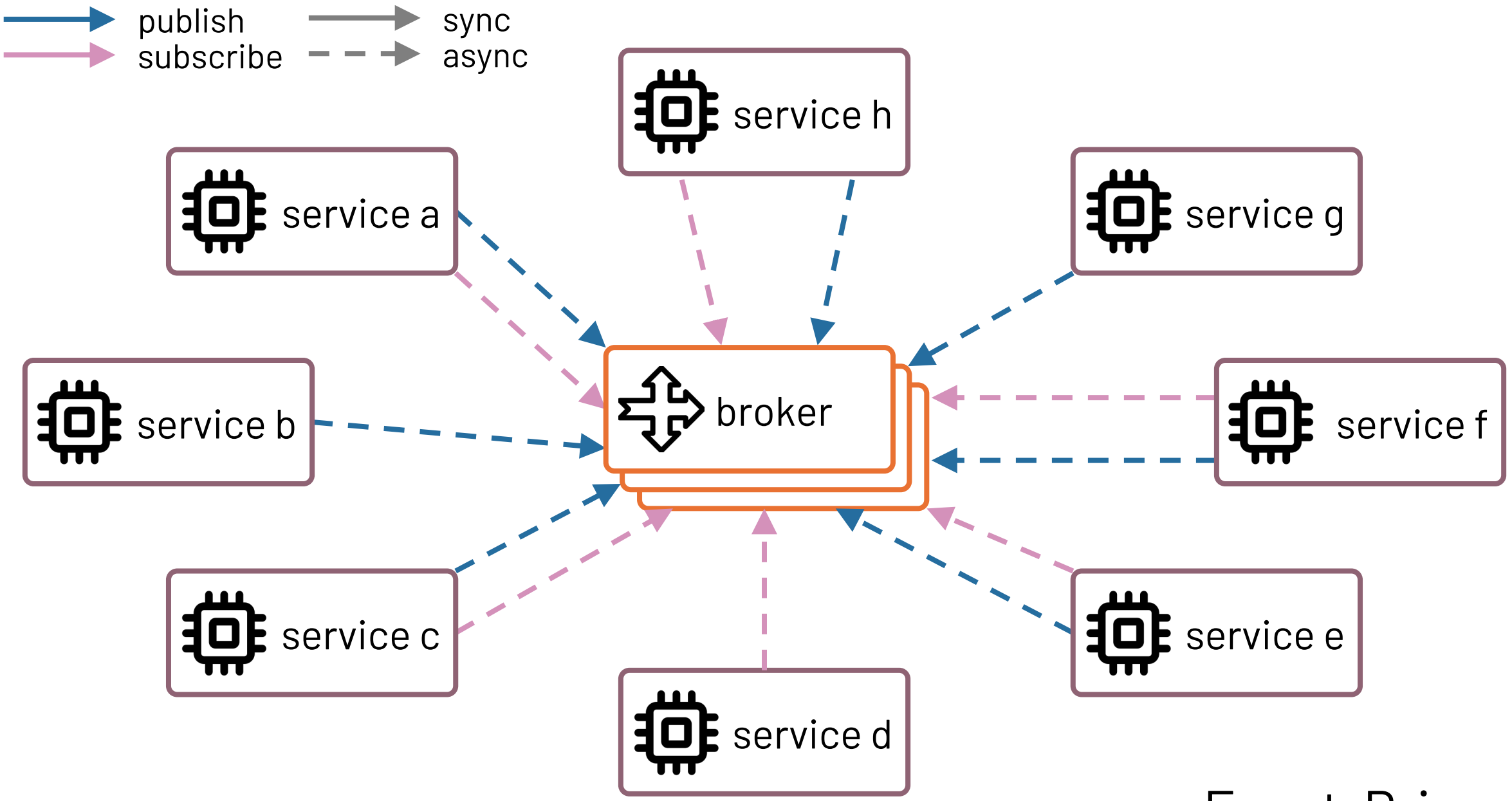
Command-Driven

© The Ministry of Software Design Ltd.



Event-Driven

© The Ministry of Software Design Ltd.



Event-Driven



© Read the Architecture Ltd.

© The Ministry of Software Design Ltd.

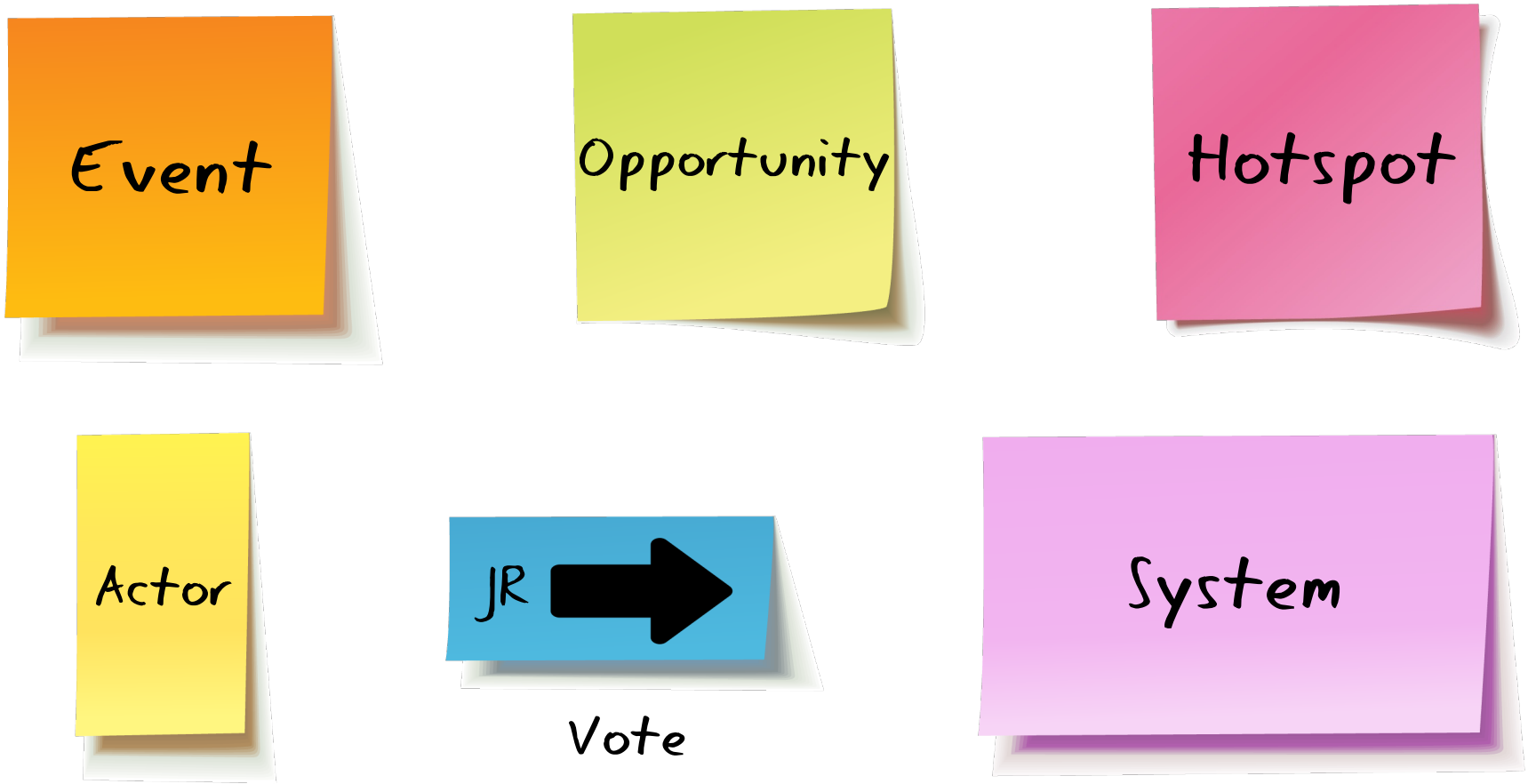


EventStorming

the one with sticky notes

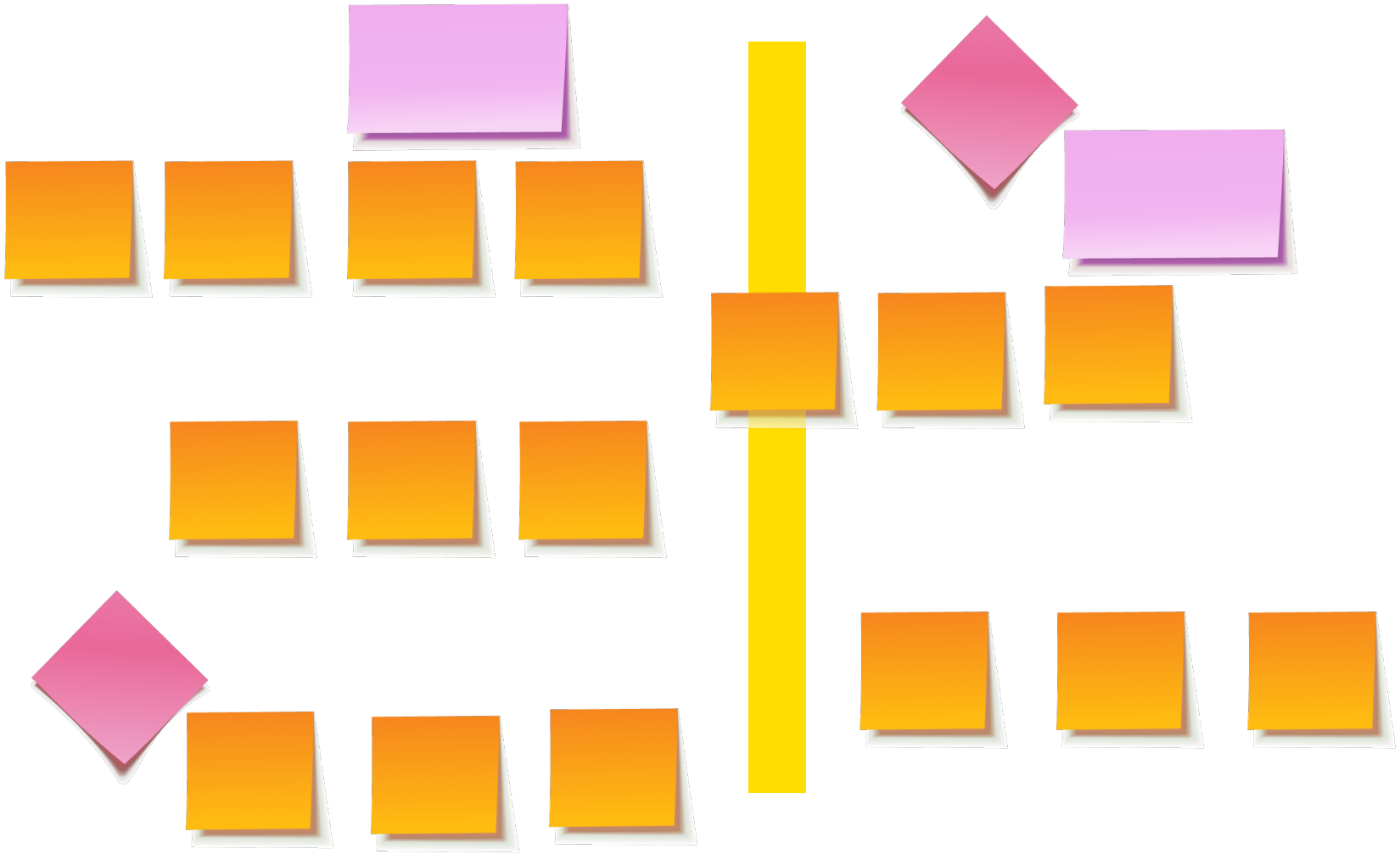
© The Ministry of Software Design Ltd.

Big-Picture EventStorming Legend



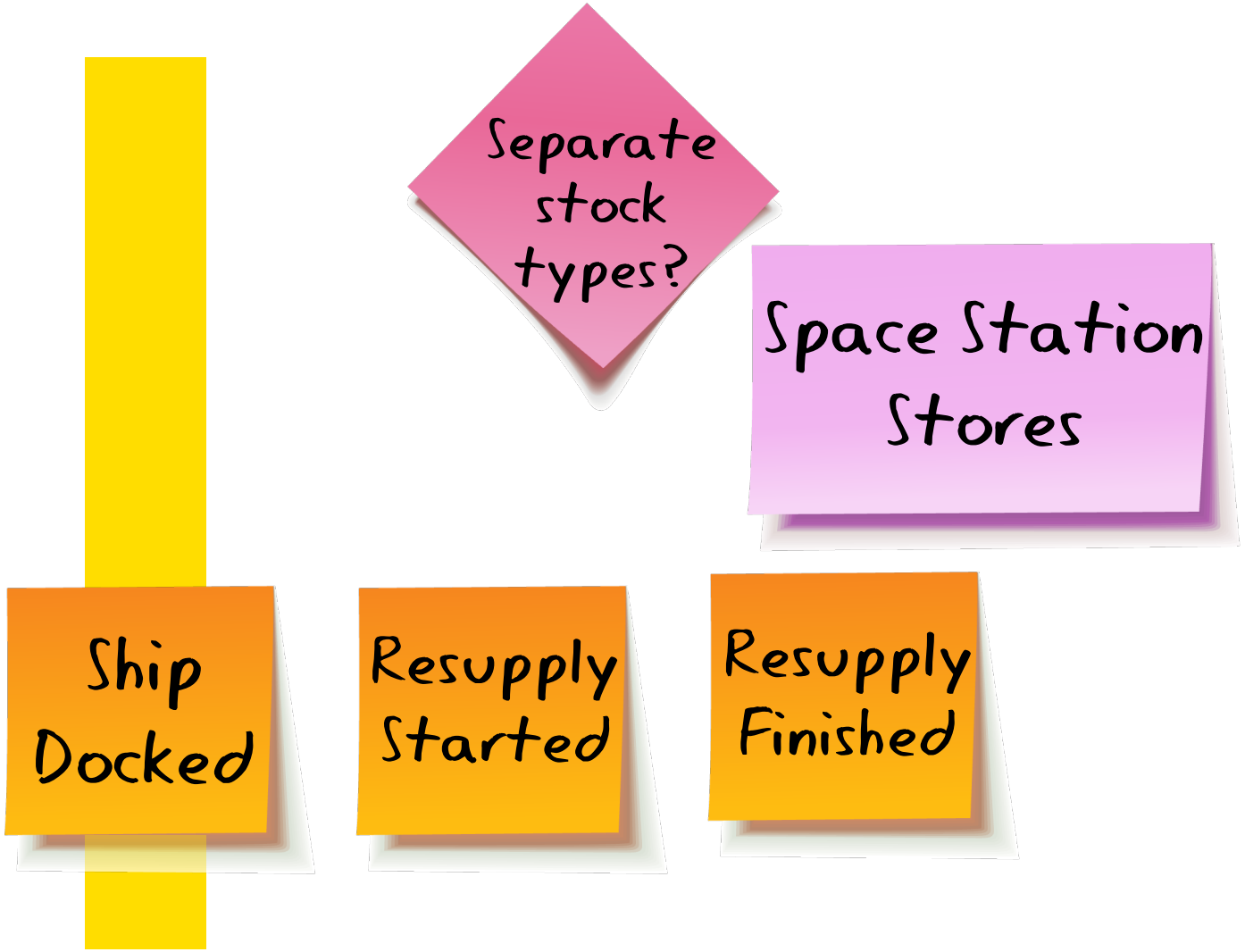
© The Ministry of Software Design Ltd.

Big-Picture EventStorming



© The Ministry of Software Design Ltd.

Big-Picture EventStorming



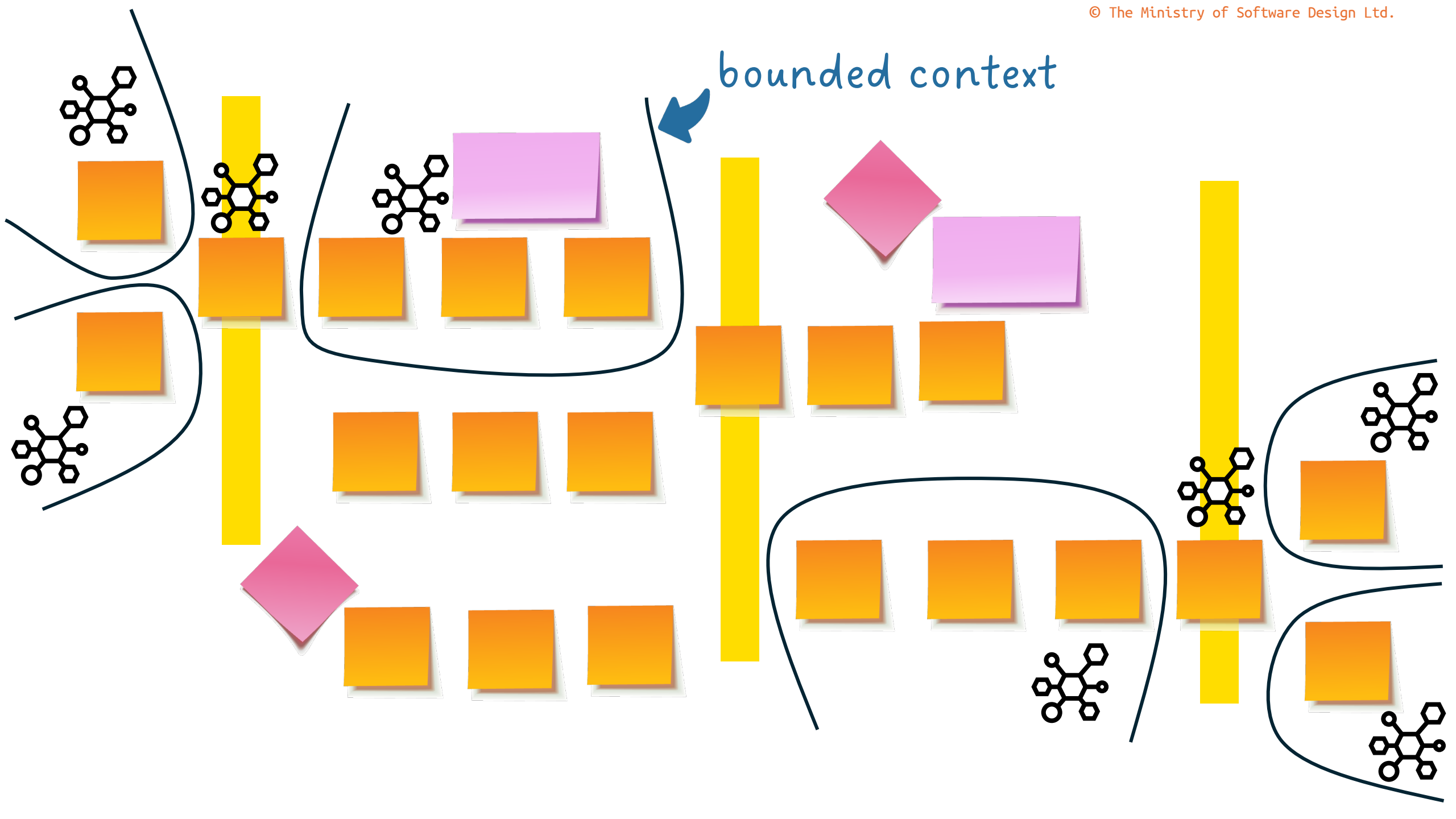
© The Ministry of Software Design Ltd.



Pivotal

Event

© The Ministry of Software Design Ltd.



© The Ministry of Software Design Ltd.

EventStorming Flavours

Big-Picture

- discovery-oriented
- many people (business/tech)

Process Modelling

- design-oriented
- smaller group

Software Design

- tech/UX design-oriented
- smaller group (tech/UX)

© The Ministry of Software Design Ltd.



Event Sourcing

© The Ministry of Software Design Ltd.

Stateful vs. Sourced Entities

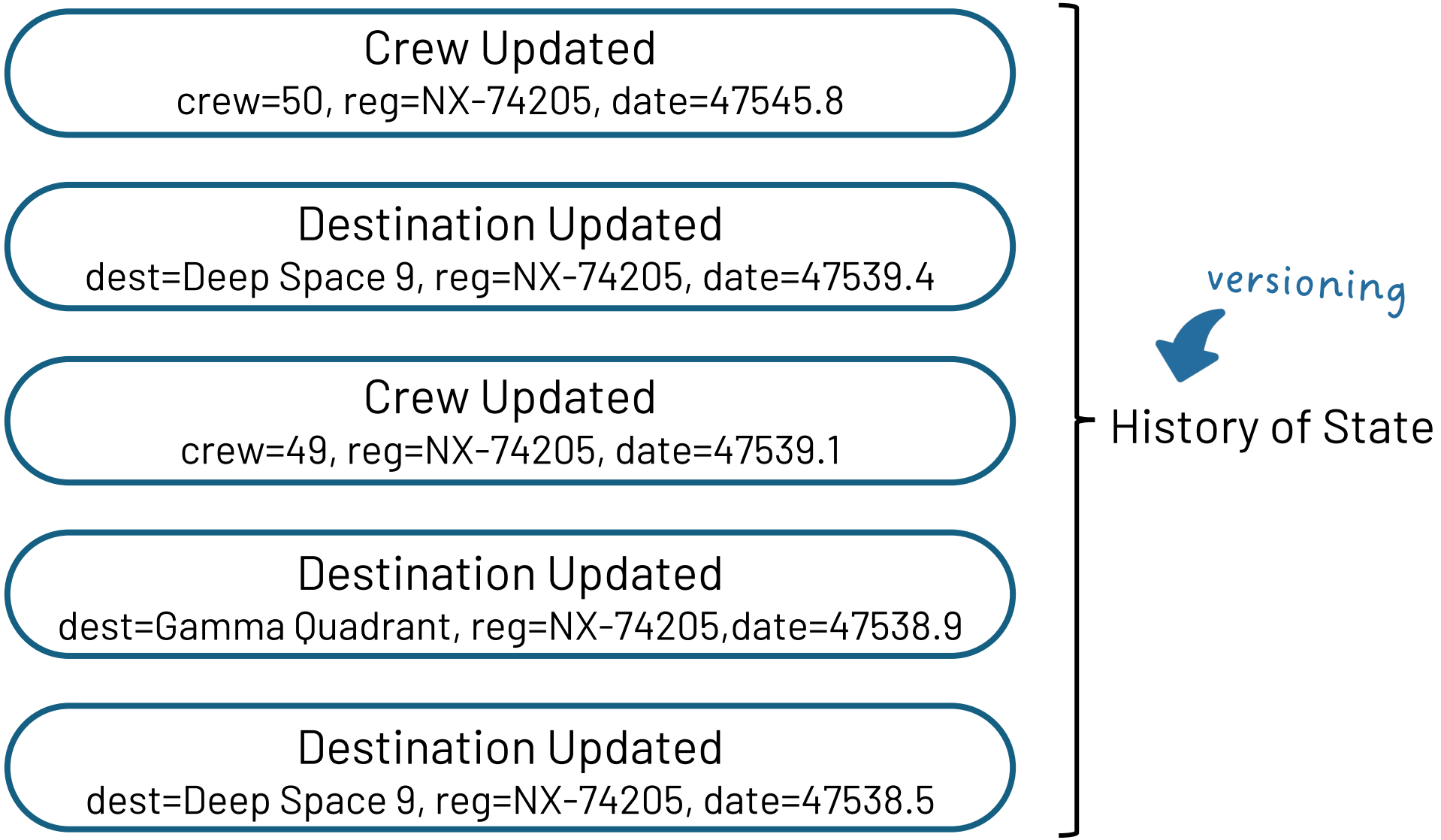
Registry	ShipName	Destination	Crew	...	LastUpdated
NCC-1701-D	USS Enterprise-D	Vulcan	1014	...	47925.4
NX-74205	USS Defiant	Gamma Quadrant	50	...	48379.6
NCC-74656	USS Voyager	The Badlands	153	...	48307.5



Current State

© The Ministry of Software Design Ltd.

Stateful vs. Sourced Entities



© The Ministry of Software Design Ltd.

Event Sourcing Streams



© The Ministry of Software Design Ltd.

Reading Sourced Events

Ship

Registry :	NX-74205
ShipName :	USS Defiant
Destination :	Deep Space 9
Crew :	50
LastUpdated :	47545.8



NX-74205

NCC-1701-D

© The Ministry of Software Design Ltd.

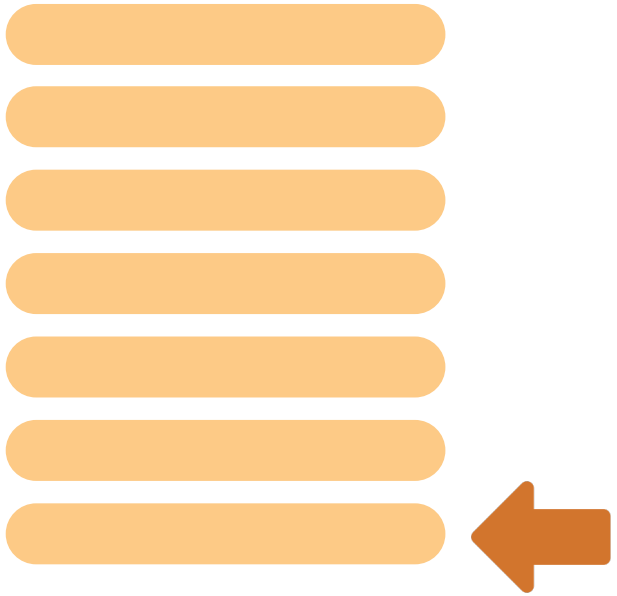
Reading Sourced Events

Ship

Registry :	NCC-1701-D
ShipName :	USS Enterprise
Destination :	Starbase 2
Crew :	1015
LastUpdated :	47938.5



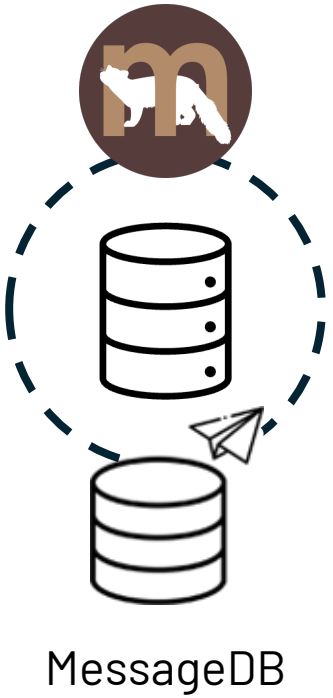
NX-74205



NCC-1701-D

© The Ministry of Software Design Ltd.

Marten



MessageDB

generic data stores

Axon Server

EVENT STORE

Event Store

disambiguation

write once, read many



© The Ministry of Software Design Ltd.

Tombstone

Event



© The Ministry of Software Design Ltd.

“this data is no longer there”

https://en.wikipedia.org/wiki/Tombstone_(programming)

© The Ministry of Software Design Ltd.

Tombstone events are context-specific

© The Ministry of Software Design Ltd.

Event Sourcing Tombstones

5. Tombstone
4.
3.
2.
1.



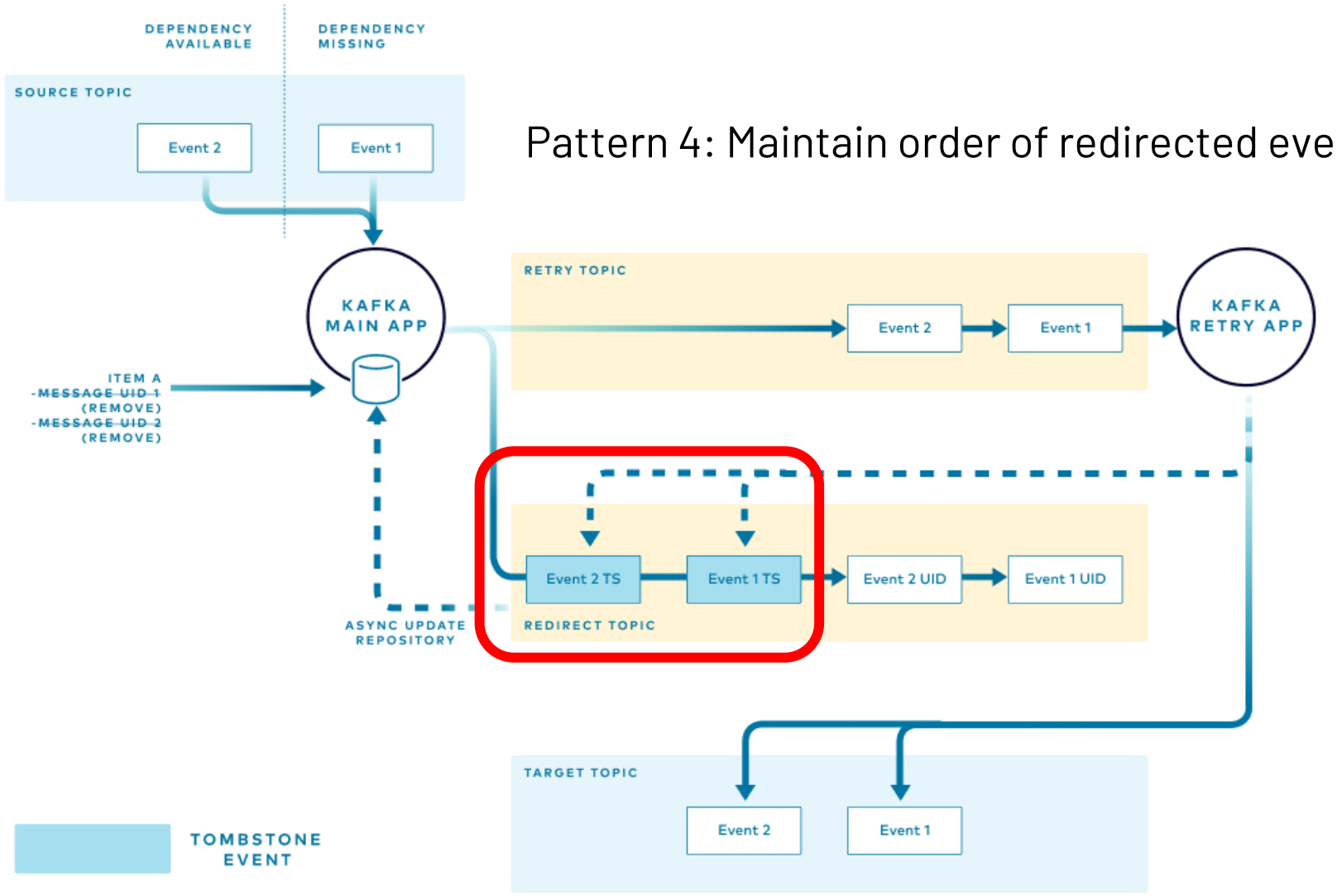
3.
4.
3. Tombstone
2.
1.



© The Ministry of Software Design Ltd.

streaming

Kafka Tombstones



Pattern 4: Maintain order of redirected events

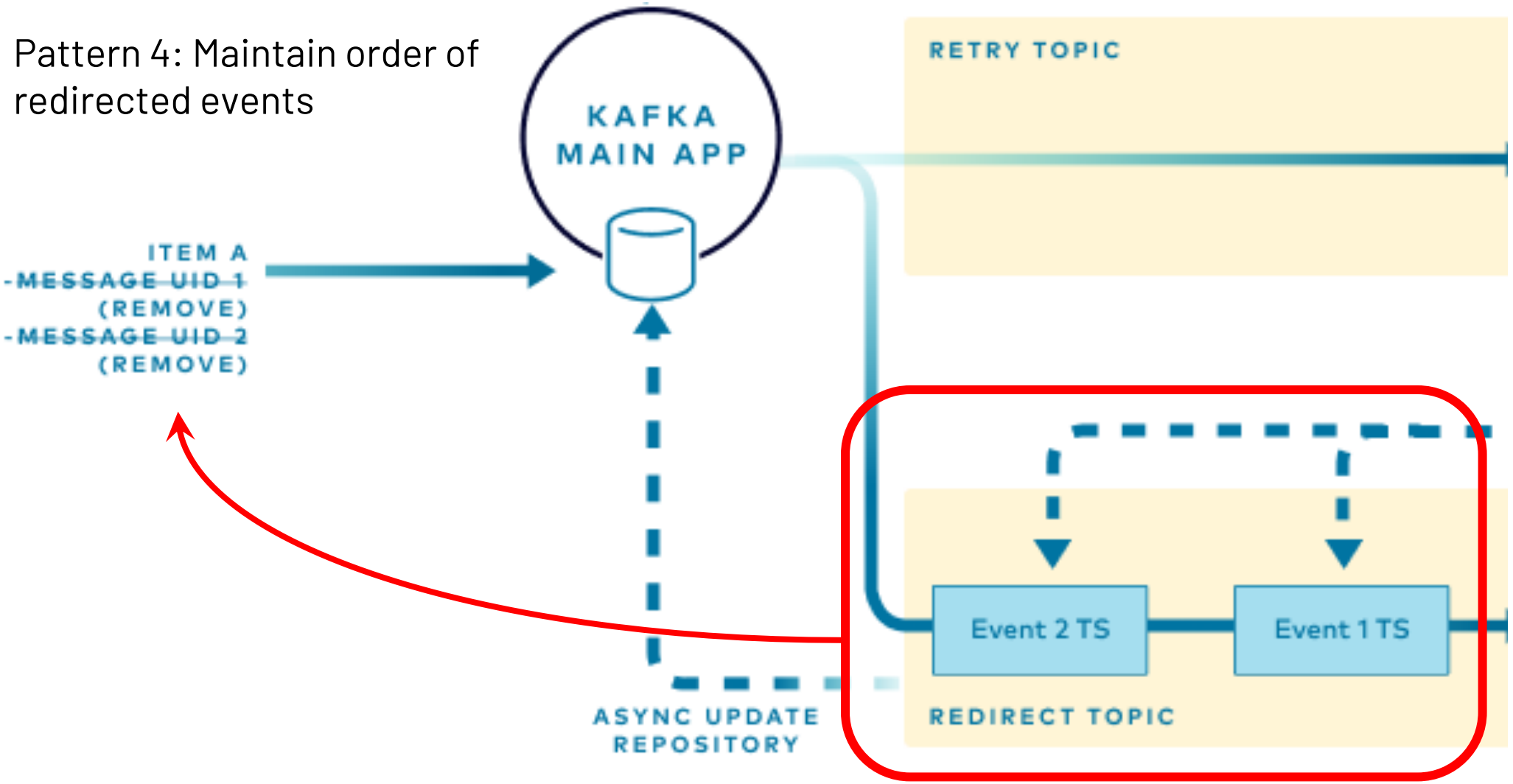
https://www.confluent.io/blog/error-handling-patterns-in-kafka/#pattern-4

© The Ministry of Software Design Ltd.

streaming

Kafka Tombstones

Pattern 4: Maintain order of redirected events



https://www.confluent.io/blog/error-handling-patterns-in-kafka/#pattern-4

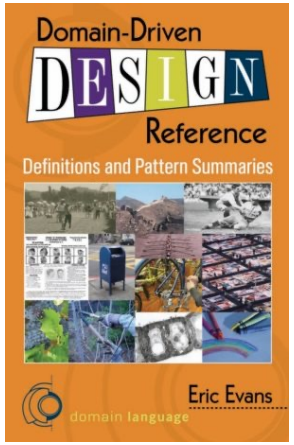
© The Ministry of Software Design Ltd.

Domain & Integration

Events



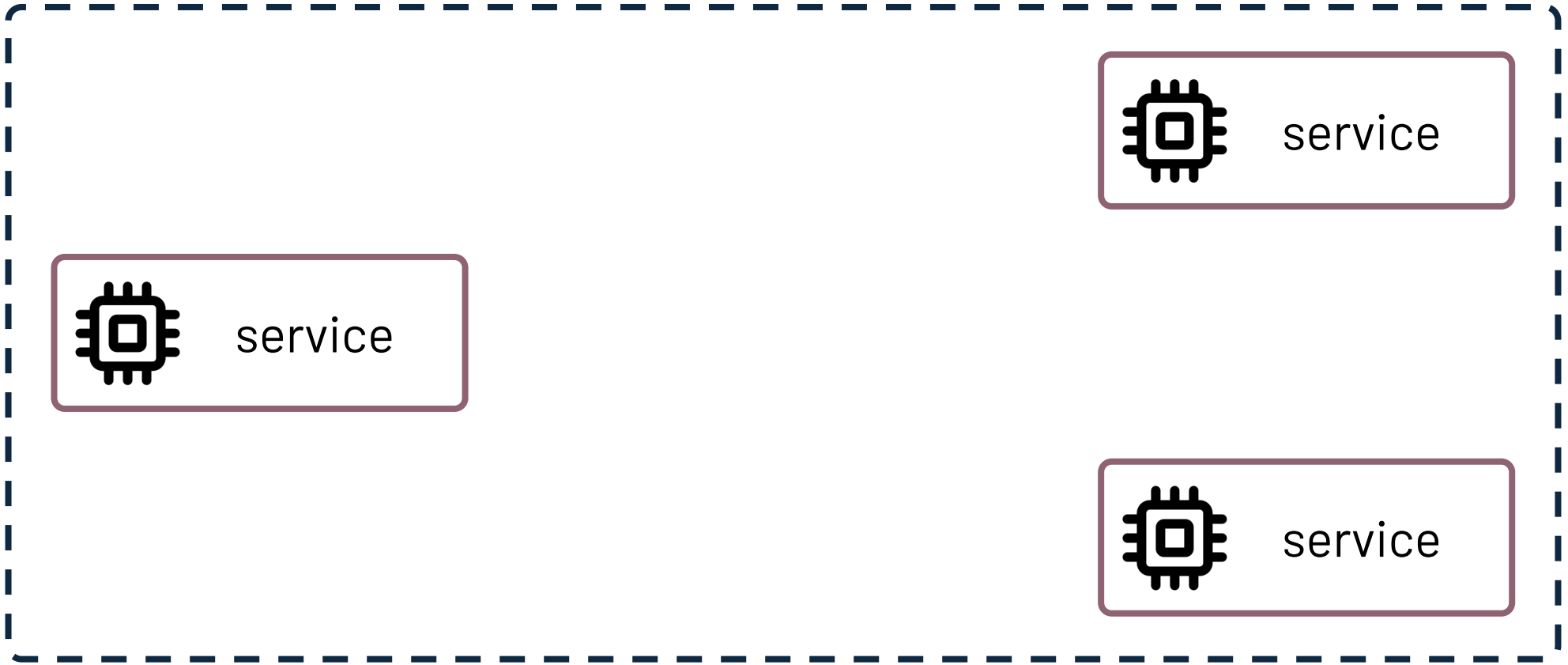
© The Ministry of Software Design Ltd.



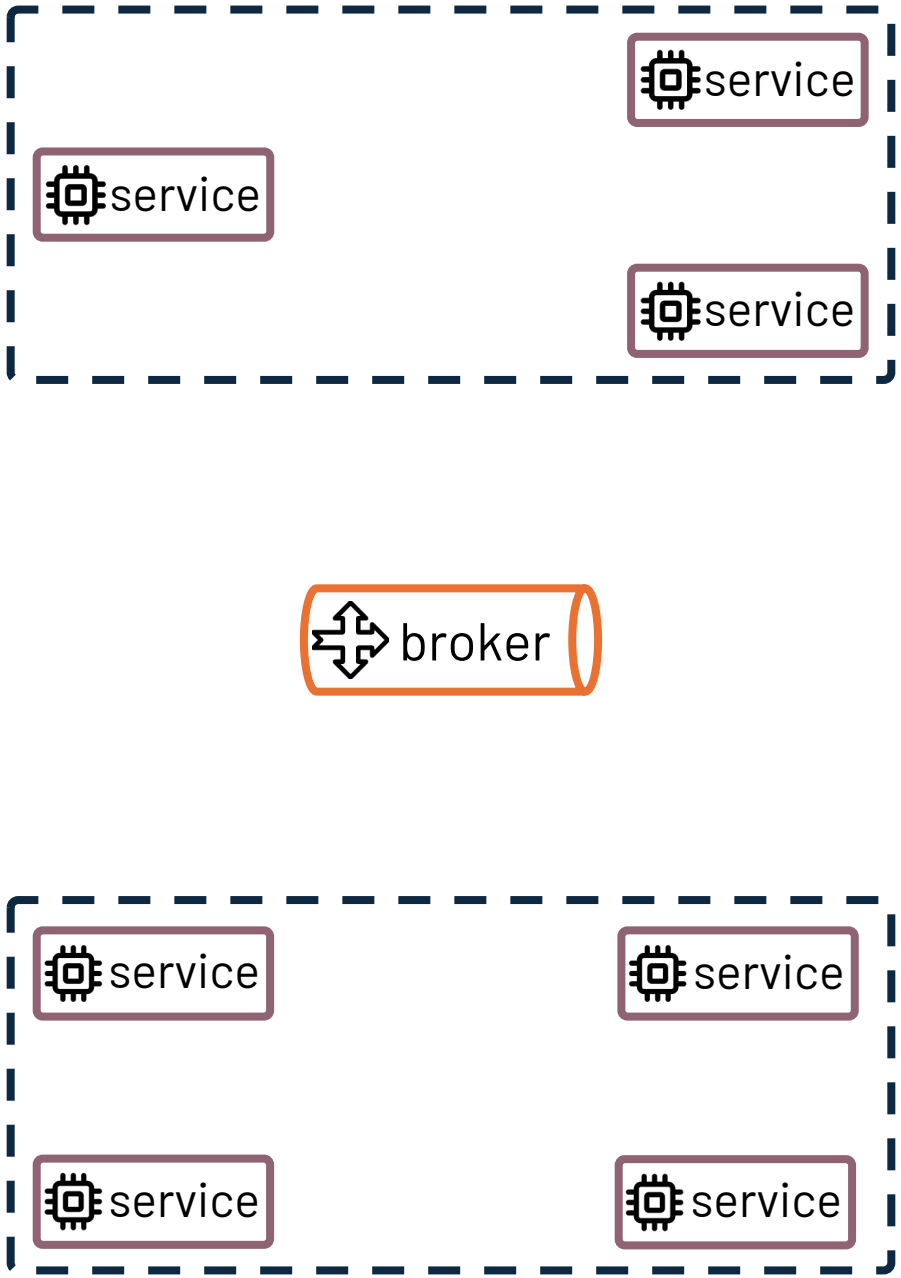
Something happened that domain experts care about … a representation of something that happened in the domain.

Eric Evans

© The Ministry of Software Design Ltd.



© The Ministry of Software Design Ltd.



© The Ministry of Software Design Ltd.

Events should be designed like APIs



Oskar Dudycz • 1st

Event Sourcerer / Helping others build Event-Driven systems

7mo •

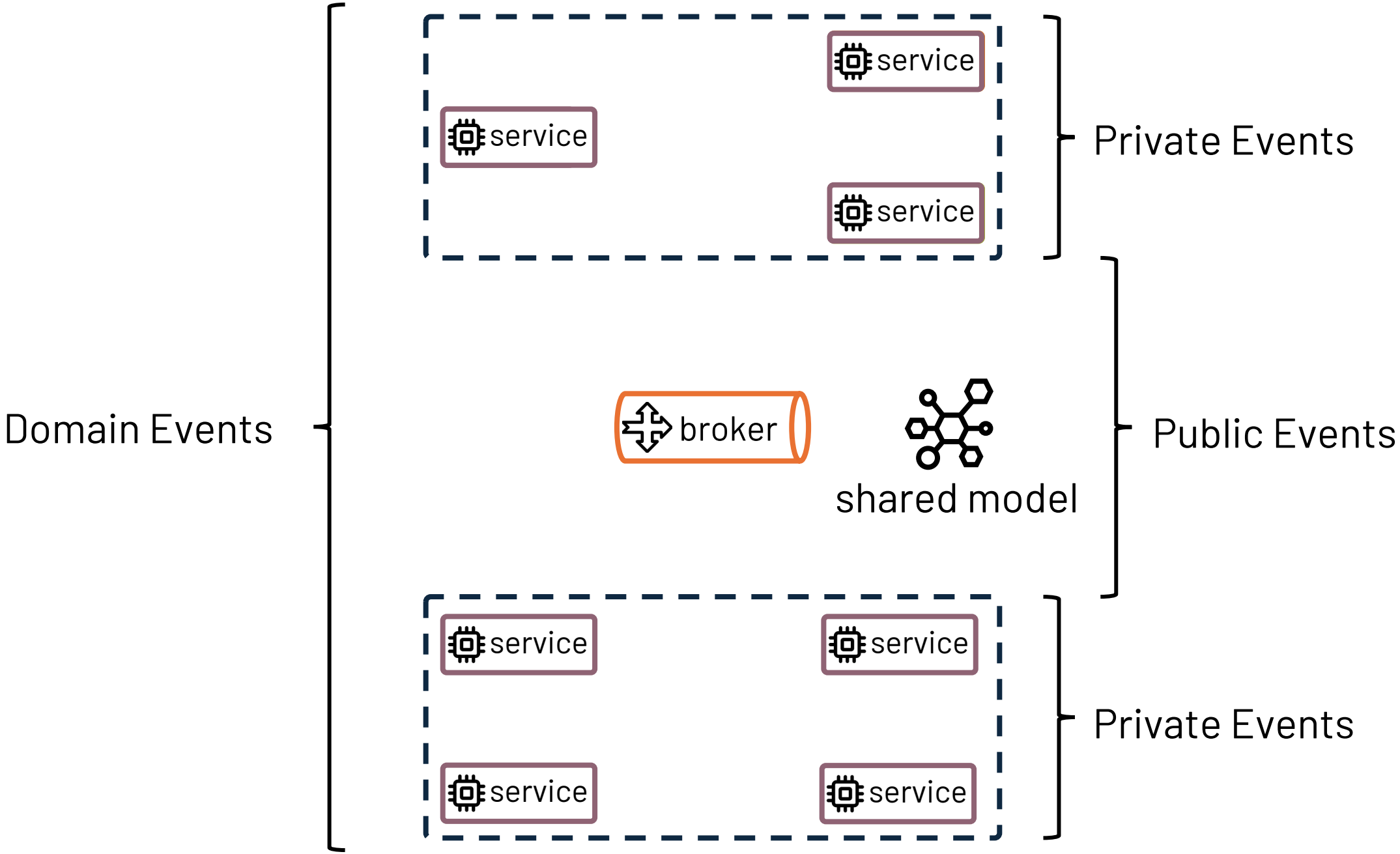
I extremely dislike the split of "domain" and "integration" events. The split is highly misleading. All of them should be domain events and represent business facts.

I prefer to split them into internal and external (or, as **Nick Tune** 🇺🇦, into private and public). That helps to understand that those events are just understandable in different contexts:
- Internal/Private in the inner module context,
- External/Public in the whole system context.

Understanding this split is essential in defining the event-driven API. Yes, API. Events should be treated as such.

https://tinyurl.com/oskdom

© The Ministry of Software Design Ltd.



© The Ministry of Software Design Ltd.



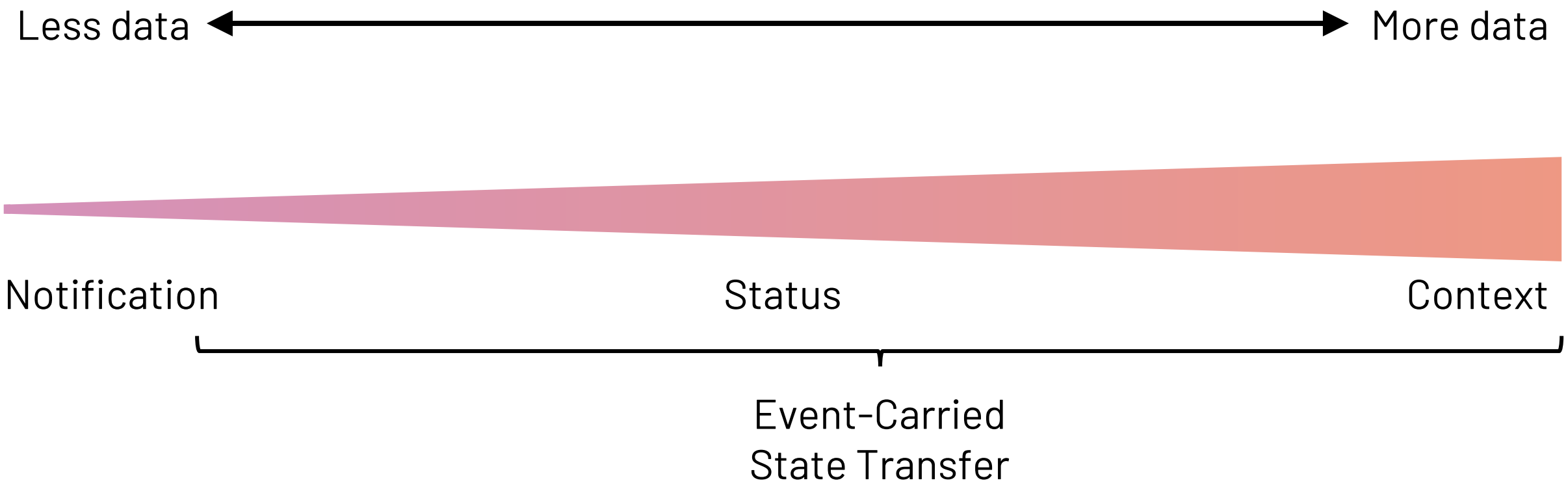
Event Granularity

© The Ministry of Software Design Ltd.

it depends

© The Ministry of Software Design Ltd.

Event Granularity Continuum



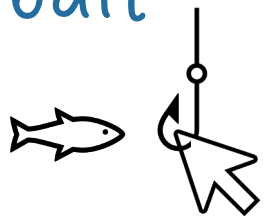
© The Ministry of Software Design Ltd.

Trade-offs

Destination Updated
Registration: NCC-1701-D, ShipName: USS Enterprise-D, Destination: Vulcan, PreviousDestination: Wolf 359, Crew: 1014, LastUpdated: 47925.4, UpdatedBy: W. T. Riker, BulkUpdate: true, KitchenSinks: 19, ...

Ship Updated
Registration: NCC-1701-D

"clickbait"



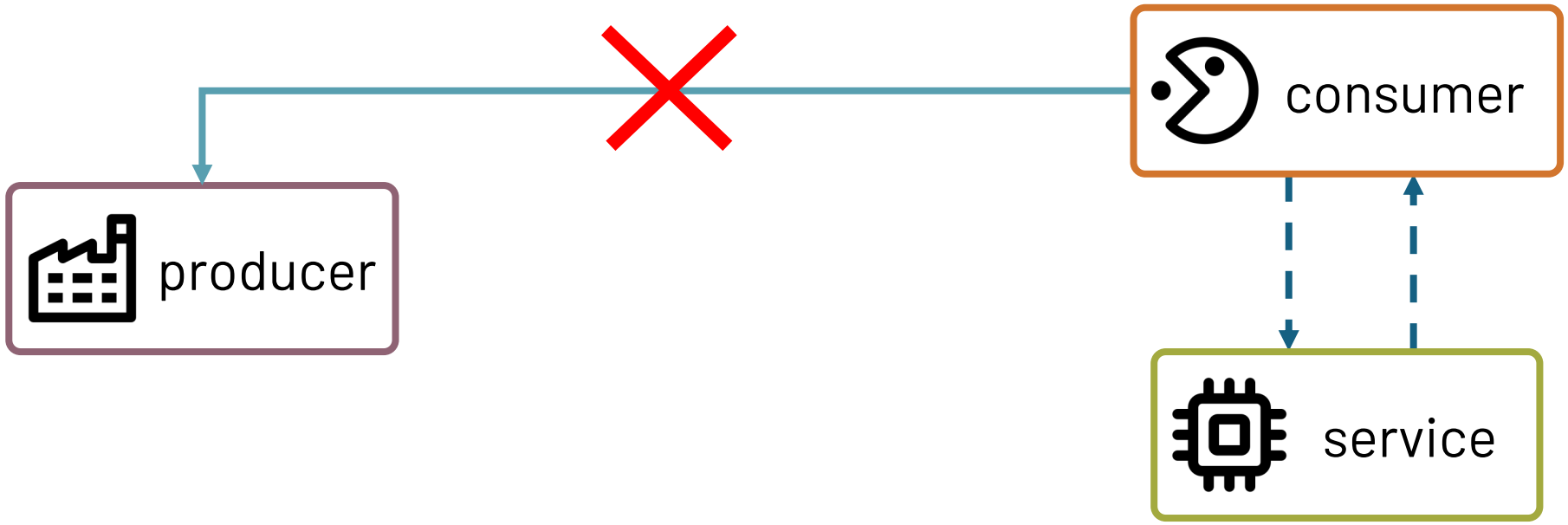
"all the things"



Notification — Status — Context

© The Ministry of Software Design Ltd.

As small as possible and as large as necessary



Event-Carried State Transfer

© The Ministry of Software Design Ltd.



BONUS

© The Ministry of Software Design Ltd.

Non

Event

© The Ministry of Software Design Ltd.

© The Ministry of Software Design Ltd.

Event	Calendar Event	Command	Event Listener	Event Handler
Event Streaming	Event-Driven	Event Storming	Pivotal Event	Event Sourcing
Event Store	Tombstone Event	Domain Event	Integration Event	Private Event
Public Event	Event Granularity	Event-Carried State Transfer	Clickbait Event	All-the-Things Event

© The Ministry of Software Design Ltd.

MOSTLY



ALL THE EVENTS

© The Ministry of Software Design Ltd.



Diagrams-as-Code with AI

Create diagrams alongside your code that are useable with LLMs.

More and more teams are starting to use coding to tackle their diagramming and documentation challenges. Using plain text notations to create diagrams turns a traditionally artistic process into a systematic and data-driven approach.

Using diagrams-as-code and models instead of traditional diagrams gives you many benefits, including storing your diagrams alongside your code, automatic consistency, and the ability to engage AI in the process.

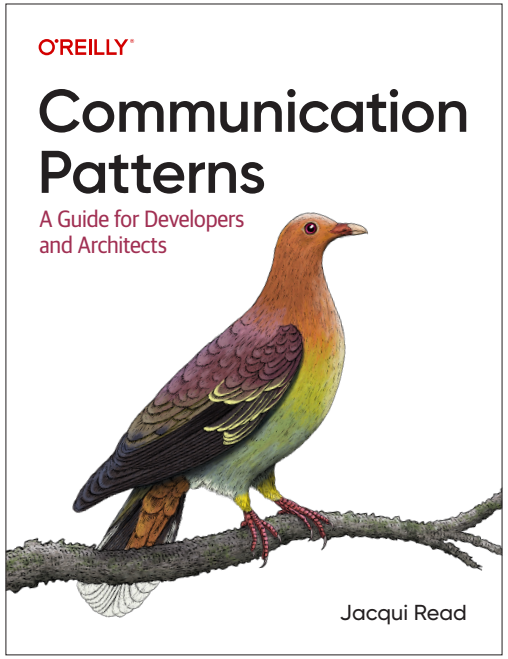
Software Design Accelerator

In a world where 75%+ software projects still fail, what is going wrong? Failure means the system doesn't meet customer and/or business needs. This workshop allows you to super-charge your design learning, combining technical skills, team-working, conflict resolution, and complexity-management techniques.

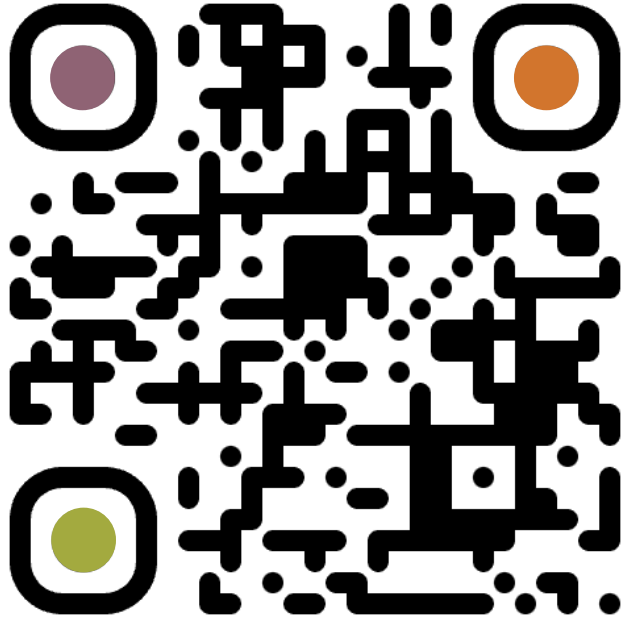
Let's break this cycle of failure; take these skills and apply them straight away to your work. The Software Design Accelerator is about increasing your velocity towards successful outcomes.

© The Ministry of Software Design Ltd.

Thank you









Jacqui Read

Chief Architect
@ The Ministry of Software Design
Author of *Communication Patterns*

acedmodel.com

JacquiRead.com

linkedin.com/in/jacquelineread

fosstodon.org/@tekiegirl

tekiegirl.bsky.social

© The Ministry of Software Design Ltd.

Further Reading pt 1

- https://www.gentlydownthe.stream/
- https://event-driven.io/en/whats_the_difference_between_event_and_command/
- https://leanpub.com/introducing_eventstorming
- https://github.com/ddd-crew/eventstorming-glossary-cheat-sheet
- https://event-driven.io/en/category/
- https://udidahan.com/2009/12/09/clarified-cqrs/
- https://event-driven.io/en/cqrs_facts_and_myths_explained/
- https://event-driven.io/en/should_you_always_keep_streams_short/
- https://en.wikipedia.org/wiki/Event_store

© The Ministry of Software Design Ltd.

Further Reading pt 2

- https://www.youtube.com/watch?v=20zvAJAhqS0 (Let's build the worst Event Sourcing system!)
- https://en.wikipedia.org/wiki/Tombstone_(programming)
- https://codeopinion.com/should-you-publish-domain-events-or-integration-events/
- https://devblogs.microsoft.com/cesardelatorre/domain-events-vs-integration-events-in-domain-driven-design-and-microservices-architectures/
- https://verraes.net/2019/05/eventsourcing-patterns-forgettable-payloads/