Event Deduplication Scenarios

# Duplicates don’t matter

In this scenario, duplicate entries do not matter. Examples of this include archival of event message, informational telemetry that may not be processed (just represents a point in time)

# Idempotent Events

In this scenario, each event is indempotent and as such processing duplicates of the event results in the same output. So there’s no need to handle it. Examples of this scenario include IOT applications where the events are used to represent the “current” state of a physical device, when using mechanisms such as ‘etags’ to try and prevent out of date transactions from being applied

# Duplicates do matter

When duplicates do matter, you have 3 choices to make about how to handle them.

1. Prevent them from occurring
2. Prevent duplicates from entering your channel
3. Prevent the event consumer from processing duplicates

Preventing them from occurring requires handling all duplicate detection at the publisher level. This is difficult, especially in scenarios where you want to handle HA/DR failover and there may be multiple publishers.

To prevent duplicates from entering your channel, you need to place something in front of it that helps pre-processes the events before placing them into the channel. Some channels (such as Service Bus Queues) offer optional deduplication on ingestion such as a feature. Otherwise, you will have to build your own façade, which means that at best you’re helping mitigate the risk of duplicates, not eliminating it because your façade could unintentionally introduce duplicates into the pipeline. Only the channel itself could eliminate it entirely because it has the ability to know what’s already in the channel and will avoid accepting anything it already has.

This leaves the 3rd option of preventing duplicates upon consumption. This usually involves having the consumer read the message, extra the meta data that’s used for identifying the duplicate, comparing that to a data store, and using that to decide if the message should be processed or discarded.

There are several considerations when determining how you want to implement deduplication:

* What is the time window over which you’ll check for duplicates?
* What is the meta data you need to use?
* What resource cost will be paid when checking for duplicates incur?
* What latency does checking for duplicates introduce?

The time window is important because that helps determine amount of data that must be in your duplicate detection data store. If you’re receiving 1,000 messages a second, and try to store 1 hour of data, that means you need a duplicate data store 3.6 million messages. And each time you are checking for a duplicate, you need to check against them to see if its in the data store.

The meta data you use is also important. If you have to get to the message payload, that creates a processing cost. So perhaps you can just use header information that is on the record. Perhaps you have a composite key that requires multiple pieces of data? This means you need to have them all together and that you need to compare them. You could negate this by creating a hash of the values, but that comes as a resource cost.

This leads into the next consideration…. What is the cost of checking? As we mentioned in the first bullet point, the larger a dataset you need to compare to, the more resource costly the check will be. If you have to extra the message payload.

Time is also a resource, so we need to be sensitive that when we check for duplicates, we’re potentially slowing the consumption of messages. This constriction of throughput can affect the rate at which messages are processed which depending on your situation creates new challenges.

Reference links:

<https://weblogs.asp.net/sfeldman/bend-message-deduplication-on-azure-service-bus-to-your-will>

<https://docs.microsoft.com/en-us/azure/service-bus-messaging/duplicate-detection>

<https://docs.microsoft.com/en-us/dotnet/architecture/microservices/architect-microservice-container-applications/asynchronous-message-based-communication>

<https://stackoverflow.com/questions/29647656/effective-strategy-to-avoid-duplicate-messages-in-apache-kafka-consumer>

<https://lostechies.com/jimmybogard/2013/06/03/un-reliability-in-messaging-idempotency-and-de-duplication/>

<https://queue.acm.org/detail.cfm?id=2187821>

<https://stackoverflow.com/questions/33220685/does-azure-event-hub-guarantees-at-least-once-delivery>

<https://domaincentric.net/blog/event-sourcing-projection-patterns-deduplication-strategies>