How to Identify Your Event-Driven Architecture Use Cases to Select the Best-Fit Event Broker

Published 27 July 2021 - ID G00744367 - 11 min read
By Analyst(s): Keith Guttridge, W. Roy Schulte

Initiatives: Software Engineering Technologies; Software Engineering Strategies

Treating EDA as a single paradigm, selecting a single technology, can lead to scalability and performance issues or disappointing business outcomes. Software engineering leaders must select the right combination of technologies and instances to fulfill each of the four typical high-level use cases.

Overview

Key Findings

- The use of event-driven architecture (EDA) is on the rise, as software engineering leaders seek to connect data and share behavior between applications, within applications, and across devices and data centers.

- Event brokers are not interchangeable. They offer different features and capabilities that are designed to support different use cases.

- Finding skills for implementing EDA can be challenging, as these skills are far less common than skills for request-oriented architectures.

- Software engineers often try to use their preferred event broker to solve all EDA use cases. This frequently results in stretching the event broker beyond its functional design.

Recommendations

Software engineering leaders responsible for EDA strategies should:

- Identify your EDA requirements by assessing your needs against four typical use cases.
Introduction

Interest in EDA and event broker platforms is on the rise, as software engineering leaders seek to connect data and share behavior between applications, components, services and repositories, as well as between devices and data centers.

Gartner client inquiries on event-driven architecture increased by 90% between 2019 and mid-2021.¹

The market for event broker and messaging infrastructure grew by about 5% in 2020 (with $2.2 billion in estimated global revenue).² While this moderate growth may not look like an explosion in implementing EDA, lower-cost open-source software (OSS) event brokers and managed cloud services are seeing double-digit growth in both adoption and revenue.

Software engineering leaders that are successful in implementing their first EDA initiative use an event broker platform that was selected to align with the needs of that project. They often attempt to replicate this success by using the same event broker platform for future EDA initiatives. For example, it is common for teams to choose the popular Apache Kafka broker for microservices architecture communication and then attempt to use it for other use cases. However, different EDA use cases may require different technologies. When software engineers try to stretch an event broker platform beyond its functional design, the EDA initiative often fails to deliver the desired flexibility, scalability and performance improvements.

To realize the benefits of EDA, software engineering leaders must select event broker platforms that provide the capabilities needed to fulfill all their use cases. They must:

- Identify the EDA use case for each initiative
Select technologies suitable for your initiative by evaluating your existing event broker and associated skills, but do not be afraid to evaluate new technology.

Deploy dedicated instances of each event broker by selecting platforms and developing skill sets according to the reliability and stability needs of your use case.

**Analysis**

**Identify Your Required EDA Use Case**

Before implementing a new event broker platform, software engineering leaders must understand the four EDA use cases and classify their initiative. By first identifying the use case of each EDA initiative, they will be better able to select technologies that are designed to achieve their goals.

When planning to adopt EDA, software engineering leaders must evaluate each initiative independently to ensure they identify the right use cases and, in turn, select the best event broker for the task.

While there is a degree of overlap between the four EDA use cases, each use case is defined by key characteristics such as its scope, technology type, ownership model and maturity. Table 1 outlines and compares the EDA use cases — enterprise messaging, application messaging, event streaming and edge messaging.
Table 1: Comparing the Four Event-Driven Architecture Use Cases

<table>
<thead>
<tr>
<th>Scope of Capabilities</th>
<th>Enterprise Messaging</th>
<th>Application Messaging</th>
<th>Event Streaming</th>
<th>Edge Messaging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distribute events between application domains</td>
<td>Distribute events within a single application domain</td>
<td>Ingest events from multiple endpoints for subsequent distribution</td>
<td>Distribute events between devices and data centers</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Main Technology Type</th>
<th>Proprietary, multiprotocol brokers</th>
<th>Open-source, multiprotocol brokers</th>
<th>Open-source, standards-based brokers</th>
<th>Proprietary, standards-based brokers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ownership Model</td>
<td>Centralized delivery</td>
<td>Federated delivery</td>
<td>Centralized delivery</td>
<td>Federated delivery</td>
</tr>
<tr>
<td>Market Share and Growth Trajectory</td>
<td>50% (steadily declining)</td>
<td>25% (steadily growing)</td>
<td>20% (rapidly growing)</td>
<td>5% (stagnant)</td>
</tr>
</tbody>
</table>

Source: Gartner (July 2021)

Use Case 1 — Enterprise Messaging

The enterprise messaging use case aims to provide reliable event distribution between application domains, both within a data center, between the organization's data centers and branch locations. Most enterprise messaging today is implemented using technologies that are proprietary and multiprotocol, though lower-cost technologies are beginning to target this space. These are commonly centrally delivered as a shared service by a team of specialists who operate and administer the platform. The team controls topics, queue names and data formats to ensure interoperability between different application and data domains.
As the longest-established EDA use case, event broker products targeting this use case continue to command a 50% share of the market for event broker and messaging infrastructure. However, its share of the market is shrinking at a rate of about 4% annually, as newer use cases and technologies become more popular. 2 Examples of commercial event broker platforms that are widely used for enterprise messaging include IBM’s MQ series, Oracle’s Advanced Queuing (AQ), Solace PubSub+ Event Broker and TIBCO Software’s Enterprise Message Service. These are often selected for their high qualities of service and proven track records in high-volume processing.

Use Case 2 — Application Messaging

Application messaging aims to provide reliable event distribution within a single application domain (for example, creating a distributed processing application or building out an application based on microservices). Open-source application messaging brokers (such as Apache ActiveMQ, Apache Kafka and RabbitMQ) and cloud messaging services (such as Azure Service Bus and Azure Event Grid; Amazon MQ, Amazon Simple Notification Service (SNS), and Amazon Simple Queue Service (SQS); and Google Cloud Pub/Sub and Eventarc) are exploding in popularity. These brokers enable software engineering teams to achieve greater autonomy and federated ownership of the technology — from deployment and delivery to operations and administration. This localized delivery simplifies the process of implementing new topics, queues or changes to data formats.

Vendors delivering the application messaging use case account for an approximately 25% share of the market for event broker and messaging infrastructure, and this use case’s market share continues to steadily grow. Examples of commercial event broker platforms that are widely used for application messaging include Amazon SQS, Google Pub/Sub, Microsoft Azure Service Bus, Red Hat’s AMQ and VMware Tanzu RabbitMQ.

Use Case 3 — Event Streaming

Event streaming aims to provide reliable event ingestion and distribution using numerous data sources, including web browsers, desktop clients and Internet of Things (IoT) devices and providing that data to subscribers for processing. As a result, the technology is centrally delivered as a shared service to enable greater control and to gather insights across a larger data store. Event streaming platforms are mostly open-source, standards-based brokers (although some platforms are offered as managed services or as proprietary software built on an open-source foundation).
Vendors delivering the event streaming use case account for an approximately 20% share of the market for event broker and messaging infrastructure, but its market share is growing rapidly. This growth is largely due to the strong enterprise interest in Kafka, although organizations are also adopting Amazon Kinesis, Microsoft Event Hubs, Pulsar and other alternatives. Of all Gartner client inquiries about EDA, about half of these discussions focused on Kafka and event streaming. ¹

Use Case 4 — Edge Messaging

Edge messaging aims to provide reliable event distribution between devices and data centers — both from the device to the data center and from the data center to the device. It is delivered as a localized service within an application domain, lending itself to federated ownership. The technology is mostly proprietary, standards-based brokers to provide last-mile connectivity beyond the data center.

Vendors delivering the edge messaging use case account for approximately 5% of the market for event broker and messaging infrastructure and have remained relatively stagnant since 2019 (though we expect this to grow). Examples of commercial event broker platforms that are widely used for edge messaging include Ably Realtime, Push Technology, Pusher and PubNub.

Select an Event Broker Suitable for Your Initiative

After identifying the use case for each EDA initiative, software engineering leaders must determine whether their existing event broker platforms are suitable. They should ask:

- Do our event broker platforms provide the functionality required to fulfill our use cases?
- Do our architects, developers and operations personnel have the skills to use the platform in a way that fully leverages its features?
- Based on our use cases and skill set, are our current platforms appropriate? Do we need to evaluate new event broker platforms?

Software engineering leaders should determine the right combination of event broker platforms and dedicated instances to meet their EDA objectives.
Several EDA vendors offer event broker platforms that provide capabilities across multiple use cases. For example, some platforms are designed to distribute data both within and between applications to support enterprise messaging and application messaging use cases. Other platforms provide both data ingestion and distribution capabilities to support event streaming and edge use cases.

However, few event broker platforms are designed to fulfill every use case. Analysis of Gartner client inquiry shows that most large organizations have at least one event broker product for enterprise and application use cases, and at least one event broker product for the event streaming use case.¹ Most global organizations use several event broker products to meet their EDA needs. Software engineering leaders should evaluate their event broker technology each and every time they have a new initiative, and they should not be afraid to evaluate new event broker technology when their existing technology is reaching its limits.

Clients have often expressed the desire to standardize onto a single event broker platform that fulfills all use cases. By using a single platform, clients hope to simplify procurement by reducing vendor sprawl and to simplify technology use requirements due to skill shortages. However, the reality is that the best-fit platform for enterprise use cases will most likely not be right for edge use cases. Focusing on a single event broker for all use cases means that an organization will often ignore technology innovations in the market. As a result, it will continue to use an event broker for use cases outside of its sweet spot, leading to poor EDA implementations.

Figure 1 highlights an example scenario where an organization adopted three event broker platforms to fulfill each of their four use cases. Each platform provides the appropriate capabilities to support two use cases, and the technologies are not stretched beyond their functional design.
Software engineering leaders with multiple event brokers, and especially those with a single event broker product, should investigate multiple deployments of each event broker rather than trying to fit all workloads through a single deployment. In most cases, we recommend deploying dedicated instances for separate workloads. This approach improves stability and accelerates delivery by:

- **Reducing dependency.** As more systems, services and endpoints are involved in the EDA initiative, more people are required to make updates to data models and topic structures. Dedicated instances reduce the number of producers and consumers needed to deliver improved operations.

- **Improving maintainability.** Dedicated instances enable teams to manage their own outages, upgrades, maintenance and other changes within the scope of the EDA use case.

- **Contributing to better security.** Keeping a dedicated instance means that it is much easier to control security with fewer identities to manage.

- **Reducing performance demands.** Dedicated instances distribute the performance requests more evenly across the network.

- **Localizing ownership.** Dedicated instances provide teams with greater autonomy and freedom to build an EDA solution that accomplishes their objectives (without the need for a quorum of stakeholder approvals).
Figure 2 extends the earlier example by showing how the organization deployed multiple instances across its three event broker platforms.

**Figure 2: Deploy Dedicated Instances for Separate EDA Initiatives**

In this example, the organization deployed two dedicated instances on the first platform — one that works with the EDA for the enterprise messaging (Instance 1) and one that publishes events within the application (Instance 2). They used a similar approach for the second platform, deploying two dedicated instances for application workloads and a third instance for event streaming.

Note that the organization did not use dedicated instances for its third platform. Dedicated instances are not always required, especially for platforms spanning streaming and edge use cases (which often provide the capability to support workloads using a single instance).

By evaluating EDA initiatives independently and determining the right mix of platforms and dedicated instances, software engineering leaders will provide their teams with the capabilities required to deliver on all their EDA use cases.
Evidence

1 Based on Gartner client inquiry data.

2 Market Share: All Software Markets, Worldwide, 2020

Recommended by the Authors

Some documents may not be available as part of your current Gartner subscription.

Essential Patterns for Event-Driven and Streaming Architectures

© 2021 Gartner, Inc. and/or its affiliates. All rights reserved. Gartner is a registered trademark of Gartner, Inc. and its affiliates. This publication may not be reproduced or distributed in any form without Gartner’s prior written permission. It consists of the opinions of Gartner’s research organization, which should not be construed as statements of fact. While the information contained in this publication has been obtained from sources believed to be reliable, Gartner disclaims all warranties as to the accuracy, completeness or adequacy of such information. Although Gartner research may address legal and financial issues, Gartner does not provide legal or investment advice and its research should not be construed or used as such. Your access and use of this publication are governed by Gartner’s Usage Policy. Gartner prides itself on its reputation for independence and objectivity. Its research is produced independently by its research organization without input or influence from any third party. For further information, see "Guiding Principles on Independence and Objectivity."

Gartner's Usage Policy. Gartner prides itself on its reputation for independence and objectivity. Its research is produced independently by its research organization without input or influence from any third party. For further information, see "Guiding Principles on Independence and Objectivity."
Table 1: Comparing the Four Event-Driven Architecture Use Cases

<table>
<thead>
<tr>
<th></th>
<th>Enterprise Messaging</th>
<th>Application Messaging</th>
<th>Event Streaming</th>
<th>Edge Messaging</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scope of Capabilities</strong></td>
<td>Distribute events between application domains</td>
<td>Distribute events within a single application domain</td>
<td>Ingest events from multiple endpoints for subsequent distribution</td>
<td>Distribute events between devices and data centers</td>
</tr>
<tr>
<td><strong>Main Technology Type</strong></td>
<td>Proprietary, multiprotocol brokers</td>
<td>Open-source, multiprotocol brokers</td>
<td>Open-source, standards-based brokers</td>
<td>Proprietary, standards-based brokers</td>
</tr>
<tr>
<td><strong>Ownership Model</strong></td>
<td>Centralized delivery</td>
<td>Federated delivery</td>
<td>Centralized delivery</td>
<td>Federated delivery</td>
</tr>
<tr>
<td><strong>Market Share and Growth Trajectory</strong></td>
<td>50% (steadily declining)</td>
<td>25% (steadily growing)</td>
<td>20% (rapidly growing)</td>
<td>5% (stagnant)</td>
</tr>
</tbody>
</table>

Source: Gartner (July 2021)