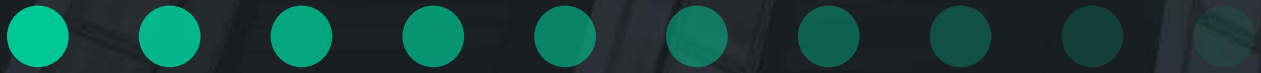



The Architect's Guide to Implementing Event-Driven Integration with iPaaS



Jesse Menning
CTO Group Architect

solace.



Traditional iPaaS implementations rely on polling to detect changes, leaving significant changes unnoticed for seconds, minutes, or even hours. As consumers demand more from enterprises, those seconds negatively affect a user's experience.”

Jesse Menning

Architect in the Office of the CTO at Solace

© Solace

All rights reserved. No part of this work may be reproduced, or stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without permission from Solace. For inquiries about permissions, contact: legal@solace.com

Table of Contents

- Introduction 4
- iPaaS and Event Broker: A Real-Time Partnership .. 5
- 1. Pick an Event Broker 7
 - Special Considerations..... 8
 - What about the iPaaS’s own messaging? 8
 - What about Kafka?..... 9
- 2. Start Generating Events..... 9
- 3. Select a Pattern 10
 - iPaaS-Centric 11
 - Event Broker-Centric 12
 - Hybrid Approach 14
- 4. Document and Design your Implementation ... 14
- Conclusion 15



Introduction

As discussed previously in the paper Event-Driven Integration with iPaaS: The Power of Real-Time Connectivity, event-driven integration allows your enterprise to better serve your customers by accelerating the flow of information and helping you incorporate cutting-edge technologies like Internet of Things (IoT) sensors, mobile applications, hybrid multi-cloud adoption, microservices, and machine learning into your operations.

In addition, event-driven integration with iPaaS makes it easier to bring innovative ideas to market by making it easier to overcome challenges that occur during integration.

If you're reading this paper, there's a good chance you're sold on the fact that event-driven integration with iPaaS is the next logical step for your enterprise. In this paper, I will explain the four technical steps you should take to make your enterprise event-driven:

1. Pick an Event Broker
2. Start Generating Events
3. Select a Pattern
4. Document and Design Your Implementation

First though, I want to explain the relationship between an iPaaS and an event broker, and help you understand why you need both to accelerate your enterprise.



iPaaS and Event Broker: A Real-Time Partnership

Together, an iPaaS and an event broker form the basis of event-driven integration, the benefits of which include:

- **Real-time, global responsiveness:** event-driven integration gets data moving the instant an event occurs, without the delay inherent in polling and batch processing;
- **Increased robustness:** event-driven integration can better handle application and network faults, absorb sudden floods of data, and simplify error handling;
- **Agility:** event-driven integration can distribute events to many consumers, and makes it easier to add more information consumers as your business changes;
- **Scalability:** event-driven integration grows with your business, and insulates legacy systems from increased traffic;
- **Design simplicity:** event-driven integration componentizes your integration process for simpler choreography;
- **Operational simplicity:** event-driven adjusts to different topologies such as on-premises and cloud-based deployments, while offering built-in high-availability and load-balancing capabilities; and,
- **Next generation innovations:** integration that allows for easier adoption of technologies like microservices, cloud, artificial intelligence, and machine learning.

iPaaS and event brokers both bring plenty of capabilities to the table, including some that overlap, so it's important to understand what each does, how they are similar, and how they differ.

iPaaS solutions provide a simplified and standardized toolset for common integration tasks and excel at interacting with the content of messages. This makes them a great pick for tasks such as graphical data mapping, content-based routing, pre-built connectivity to legacy applications, and more.

In addition, most iPaaS solutions include source management and deployment capabilities for their processes, along with integrated logging facilities and centralized repositories for exposing and managing synchronous APIs.

Event brokers, on the other hand, excel at the routing of messages based on topics, not concerning themselves with their content. At the message-level, event brokers offer intelligent, resilient, high-speed, highly available connectivity and event distribution.

These capabilities allow event brokers to distribute events to many interested consumers simultaneously and independently. That independence (or decoupling) lets consumers process information at their own speed while the event broker preserves messages if they're not available, can't keep up, or if the iPaaS or datacenter goes down.

It's important to note that those consumers can be running on-premises or in the cloud—the event broker handles all of the details of how to deliver the message to each consumer no matter where they're running. It is because of this relationship that our first step in implementing event-driven integration is selecting the best event broker for the job.



Together, an #iPaaS and an #EventBroker form the basis of event-driven integration. Benefits include: real-time global responsiveness, next-gen innovations, and design simplicity.



- What should I consider when selecting an event broker?
- Should I use the messaging technology provided by the iPaaS solution itself?
- Where does Kafka fit into the equation?

- To begin, there are the table stakes for an event broker. Given that the event broker forms the backbone of event-driven integration, it should dynamically and gracefully handle the details of delivering the message to a consumer.

From there, questions arise about the integration of the iPaaS and the event broker:

- ## 7 The Architect's Guide to Implementing Event-Driven Integration with iPaaS

- How integrated are the toolsets for the event broker and the iPaaS? Will developers and administrators enjoy them, or gnash their teeth in frustration switching between browser tabs?
- Do the companies involved have established relationships and joint processes, or will you be in the middle if there is an issue?
- Can the joint performance of the event broker/iPaaS handle your data volume at the speed that your business needs?
- How difficult is it to implement the high-availability and disaster recovery options of the event broker?
- Do they supply a standard way of designing and governing your events?
- When something goes wrong, how easy is it for you to identify and fix the problem?

Assuming you've already picked your iPaaS, be sure to pick an event broker that answers all of these questions to your satisfaction.

Special Considerations

What about the iPaaS's own messaging?

With event-driven architecture increasing in popularity, several iPaaS providers have bundled lightweight messaging solutions into their platform. There are some obvious benefits to selecting iPaaS-supplied messaging including cost, integration of the administration into the iPaaS dashboard, and the proverbial “one throat to choke” so you know whose fault it is if something goes wrong.

The capabilities vary widely between vendors, but generally speaking they don't offer very good performance, lack disaster recovery support, don't let you create an event mesh that spans environments, and offer limited accessibility outside the iPaaS itself.

What about Kafka?

Kafka is almost always in the mix when architects examine event-based architecture, and deservedly so, because Kafka is great for many tasks, particularly high-speed analysis of data-at-rest.

Kafka struggles at other things because of its inability to both filter messages and deliver them in order, its coarse, rigid topics, and inconsistent security. My colleague Ken Barr does a great job of covering these topics in more depth in his blog series.

In many cases, an iPaaS, an event broker and Kafka make a nice three-legged stool, with the event broker handling real-time events in motion, Kafka streaming data for analytics, and the iPaaS solution providing the specialized integration toolset.

2. Start Generating Events

With the major infrastructure pieces in place, it's time to generate events. Starting from a more traditional synchronous model, how does an enterprise move to a real-time iPaaS?

The first step is to liberate your data, which can take many forms depending on what your legacy IT infrastructure looks like:

- “Event-driven” applications are designed from the start (or converted) to emit events in response to changes within the application. In turn, they natively react to events from other parts of the enterprise. This requires a change in both coding style and mindset from development teams as they move from systems synchronously chained together to a decoupled architecture. If your enterprise uses micro-

service-based applications, you can find more information about [the process of designing event-native here](#). For a detailed breakdown of the type of sources you may encounter and getting them to work with your event broker, check out [How to Tap into the 3 Kinds of Event Sources and Sinks](#).

- IoT sensors that emit events at set time intervals or in response to changing conditions. This might be crucial data flowing from an automated factory or an IoT-connected vehicle.
- Using a proxy to convert commercial applications using a traditional REST API into events that downstream applications can consume.
- Change data capture mechanisms, either custom solutions such as database triggers or commercially available products like Oracle Golden Gate.

3. Select a Pattern

As with every project, implementation details vary by the needs of the business and the existing technical landscape, but in broad strokes enterprises pairing an iPaaS with an event broker either take an iPaaS-centric view of the world, an event broker-centric view of the world, or a hybrid of the two.

Those views can evolve over time, as your enterprise matures its event-driven integration strategy.

Given that the event broker forms the backbone of event-driven integration, it should dynamically and gracefully form an event mesh that spans clouds and datacenters, and it should support common protocols (including those for IoT), high availability, and disaster recovery.”

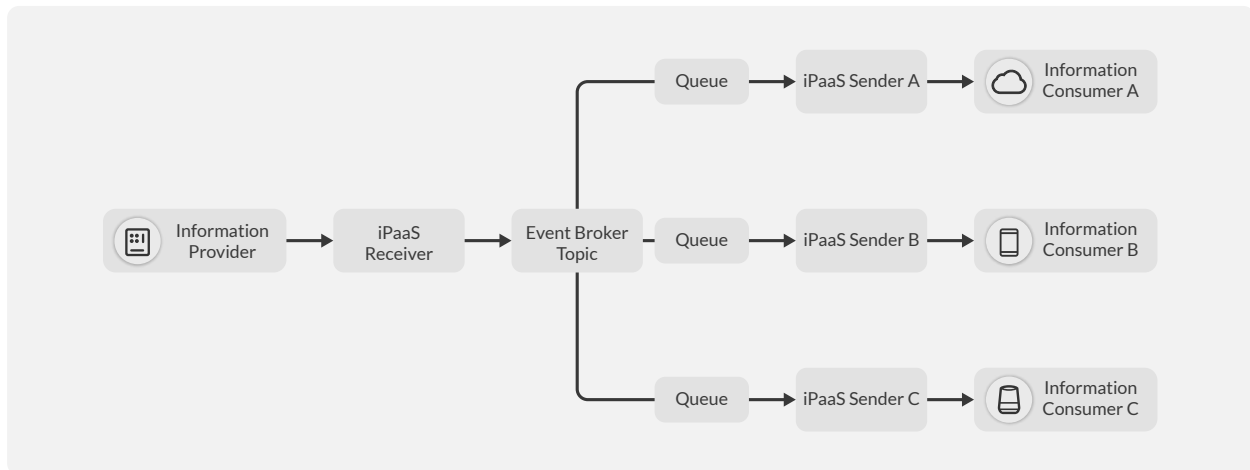


Jesse Menning

Architect in the Office of the CTO,
Solace

iPaaS-Centric

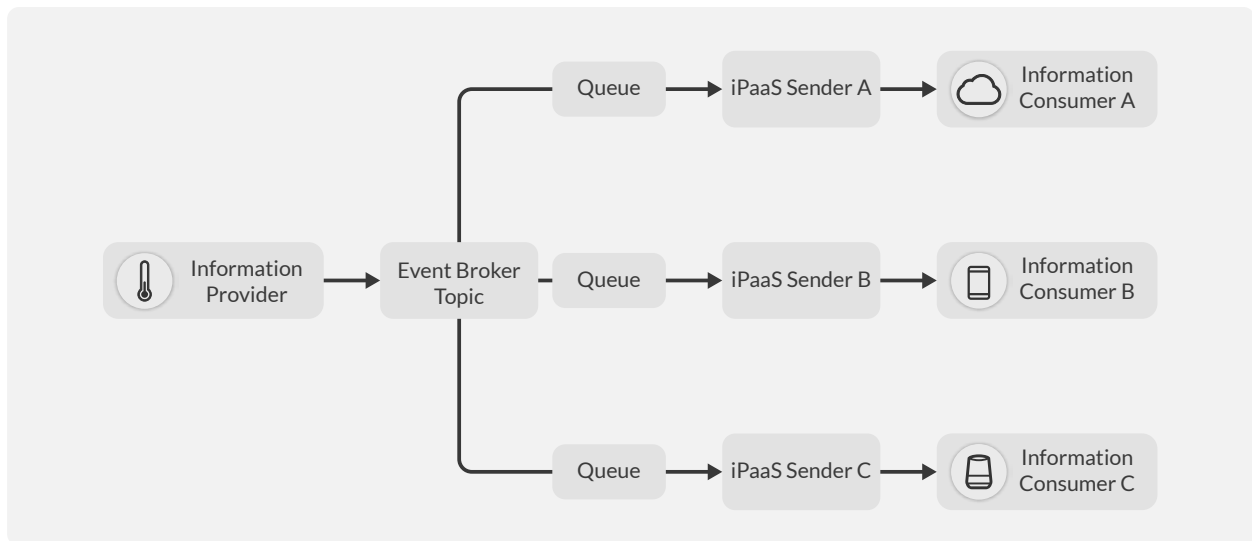
One common integration strategy for small- or mid-size enterprises is to focus on synchronous APIs such as REST or SOAP. The following generic implementation shows how an architect could introduce event-driven architecture:



1. An information provider (in this case a legacy application) detects a change in the system and connects to an iPaaS receiver using any protocol available from the iPaaS.
2. The iPaaS receives the change data from the information provider and creates an event message with the payload and adds a topic that classifies its content.
3. The iPaaS receiver publishes the message to the event broker. The event broker uses the event's topic to determine which information consumers are interested in, then places a copy of the event in three queues, one for each information consumer.
4. Each of the iPaaS senders independently processes its copy of the event. Each sender ensures that the message meets the requirement of its information consumer by transforming and enriching the data, routing or filtering based on content and other operations.
5. Each iPaaS sender then connects via the chosen protocol to its information consumer.

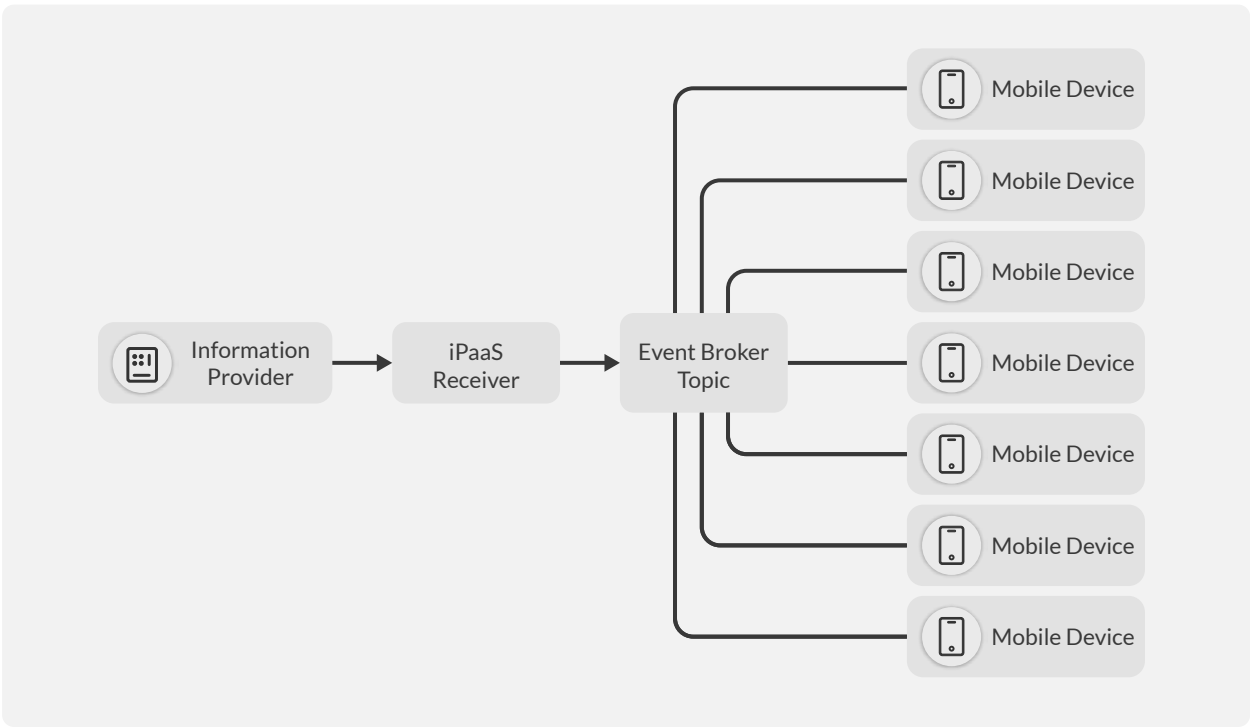
Event Broker-Centric

Other enterprises' existing integration strategy includes information providers that either are currently producing events or can produce events. These tend to be larger enterprises. The following generic implementation introduces event-driven integration into this scenario:

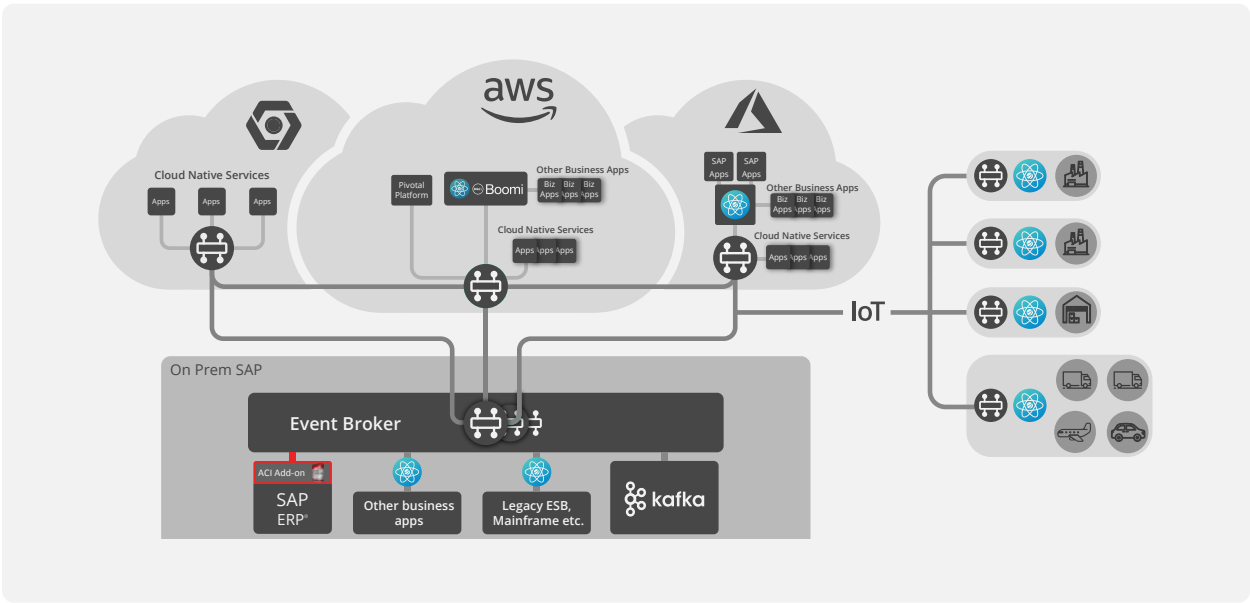


1. An information provider (in this case an IoT sensor) emits data to an event broker topic.
2. The event broker uses the event's topic to determine which information consumers are interested in the event, then places a copy of the event in three queues, one for each information consumer.
3. Each of the iPaaS senders independently process their copy of the event. Each sender ensures that the message meets the requirement of its information consumer by transforming and enriching the data, routing or filtering based on content and other operations.
4. Each iPaaS sender then connects via the chosen protocol to its information consumer.

The event broker-centric architecture also excels when you need to push information updates to multiple IoT devices at the same time. Typically, event brokers can scale to handle massive numbers of concurrent connections.



Here is how that may look from a technology integration standpoint:



Hybrid Approach

It is of course possible—and potentially preferable—to blend these two styles of event-driven integration in a single enterprise. For instance, this could allow for a mid-size enterprise to leverage IoT for a new capability, or a large enterprise to expose legacy data in new and exciting ways.

4. Document and Design your Implementation

No matter what infrastructure you use, how you liberate your data and whether you're iPaaS-centric or event broker-centric, you'll need to clearly communicate your designs to developers, stakeholders and support personnel.

For all its benefits and charms, the decoupling of an event-based architecture [can make it difficult](#) to understand how information flows through your enterprise.

In a tightly coupled system, each application knows exactly which other applications it is communicating with, and how/where to reach them. While a loosely coupled system is more flexible, it can also be confusing when an application publishes information without any idea who is using it.

That makes it important to establish a single authoritative map of your system.

Fortunately for iPaaS architects (and integration architects in general), an event management ecosystem has emerged that parallels the API gateway concept found in a synchronous world. Event portals apply many of the API gateway concepts to the event-driven world, while adding features that capture the unique aspects of an asynchronous architecture.

An event portal plays a crucial role in event-driven integration and should be in your toolset from the beginning. Among the benefits:


- Define and model events;
- Visualize what applications and iPaaS processes consume events and the impact of a change;
- Share and discover events inside and outside the enterprise;
- Govern APIs evolution over time; and,
- Secure APIs by supplying tokens to authorized users.



Conclusion

Implementing event-driven integration with iPaaS won't be an easy task, but pairing your existing (or new) solution with an event broker will create a robust and scalable backbone for your enterprise.

The event broker dynamically handles events and delivers them to where they need to be, while an event mesh ensures the data seamlessly moves across environments. This seamless global communication will occur without any applications or the iPaaS needing to know the details.



About Jesse Menning

Jesse Menning is an Architect in the Office of the CTO at Solace. He focuses on pushing innovation faster and further by combining Solace with integration technologies like iPaaS. Prior to joining Solace, he created event-driven integration solutions at clients as varied as major medical institutions, big box retailers, government agencies and medium-sized businesses.



Jesse Menning is an Architect in the Office of the CTO at Solace. He focuses on pushing innovation faster and further by combining Solace with integration technologies like iPaaS. Prior to joining Solace, he created event-driven integration solutions at clients as varied as major medical institutions, big box retailers, government agencies and medium-sized businesses.



Ottawa | Toronto | New York | Chicago | Atlanta | Silicon Valley | London | Paris | Zurich
Tokyo | Seoul | Hong Kong | Shanghai | Singapore | Mumbai | New Delhi | Melbourne | Sydney

About Solace ● ● ● ● ● ● ● ●

Solace helps large enterprises become modern and real-time by giving them everything they need to make their business operations and customer interactions event-driven. With PubSub+, the market's first and only event management platform, the company provides a comprehensive way to create, document, discover and stream events from where they are produced to where they need to be consumed – securely, reliably, quickly, and guaranteed. Learn more at solace.com.

Follow us on



A Few of Our Customers



Our Featured Partners



VMware Tanzu