



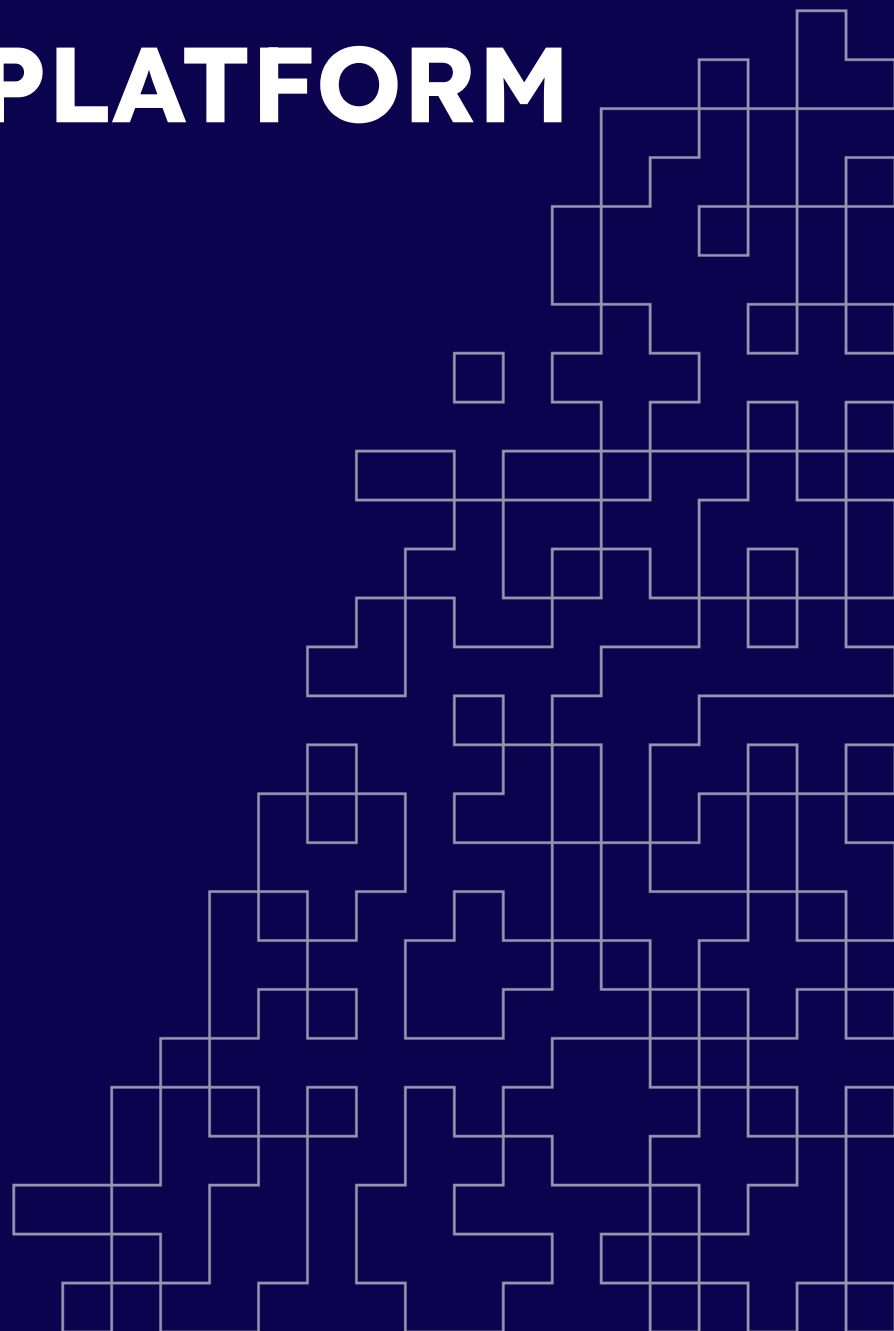
WHITE PAPER

# THE OSO GUIDE FOR BUILDING KAFKA AS A SERVICE PLATFORM

PREPARED BY

**Sion Smith**

PUBLISHED  
JUNE 2023



# 1. Introduction

Self hosted Apache Kafka deployments which include a self service wrapper is the great way to enable an outstanding developer experience. With the rise of the Kubernetes operator pattern it is now relatively straightforward to create this Kafka as a Service offering inside your company. GitOps is the best way to enable this self service platform approach - these two sentences are a concise summary of the key ideas of this white paper.

In the last few years, the role of a developer has changed dramatically. The “benevolent dictator for life” of any given codebase is being replaced with a team of developers who have to collaborate on a codebase, integrate countless datasources in real-time to move it forward. Instead of the traditional single developer who owns the ETL codebase making all the decisions, creating a bottleneck on data decisions, now a group of developers works together to make a real-time data product.

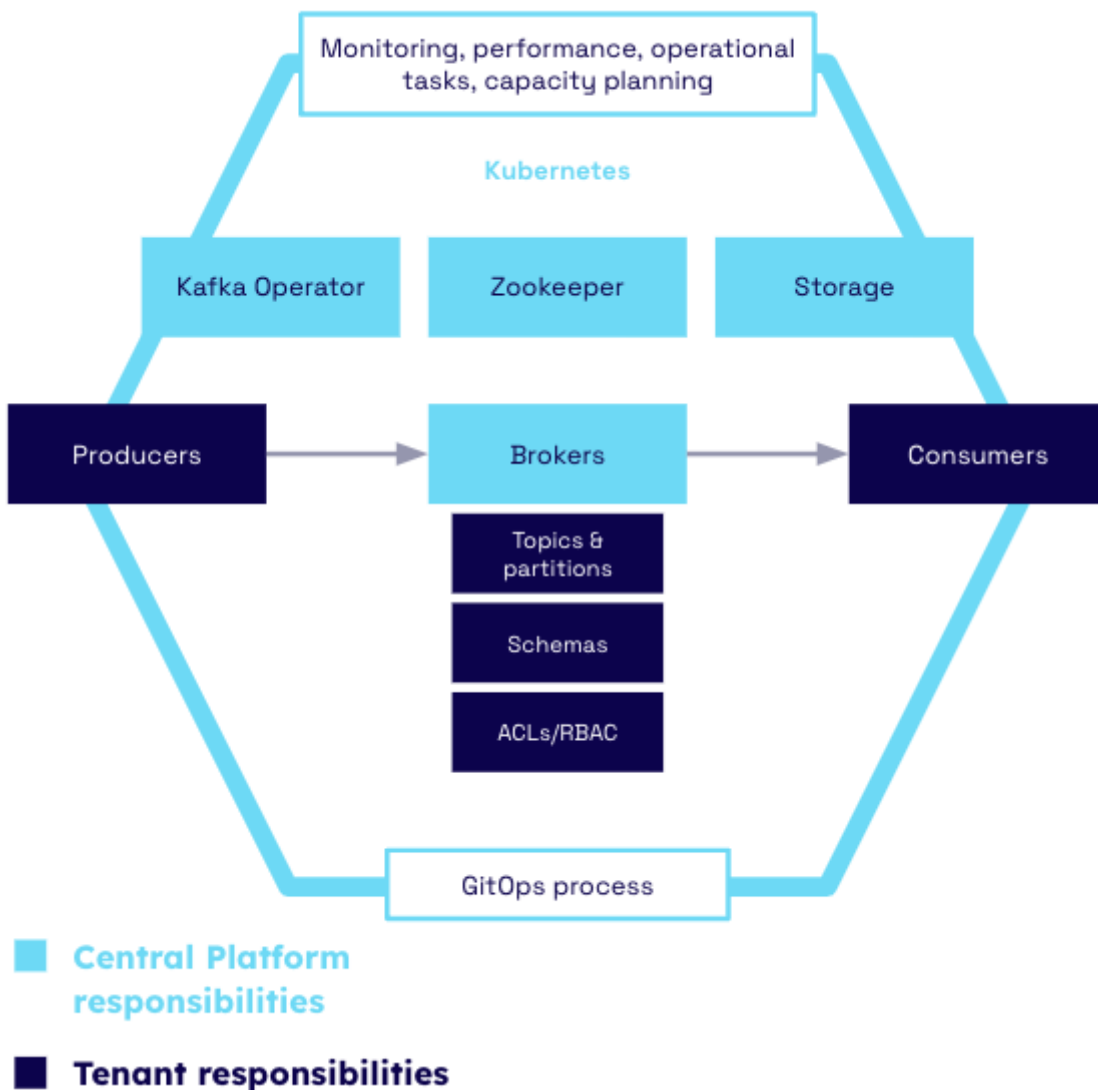
In this situation, developer experience is of utmost importance as developers collaborate around data hungry interconnected platforms. The way to foster an outstanding developer experience of Kafka is to leverage the internal platform approach. An internal platform helps to get event driven best practices to developers on-demand and in a secure way.

Adopting the platform approach alone is not enough. Organisations need to think strategically about how they would go about building and maintaining the Kafka deployments. Further, how they would build a new development culture around Kafka that makes software delivery seamless.

In this paper, we look at how a self service data platform serves to deliver an outstanding developer experience. Further, we look at how GitOps is the way to build and deploy Kafka at scale. With these key strategic initiatives in place, software delivery teams can release software continuously, access data in real time and with full confidence in the reliability of the system.

## 2. Self-service developer experience (DX)

Historically the way a developer could get access to a Kafka broker, topics or data was to ask a system administrator. In the modern era, however, developers require on-demand access to a variety of tools and data sources that enable them to be more productive and deliver increased value from company data in real-time. Having the ability to spin up Kafka clusters that developers can build, test and ship applications quickly are what make up the developer experience. An example of the target operating model is shown below:



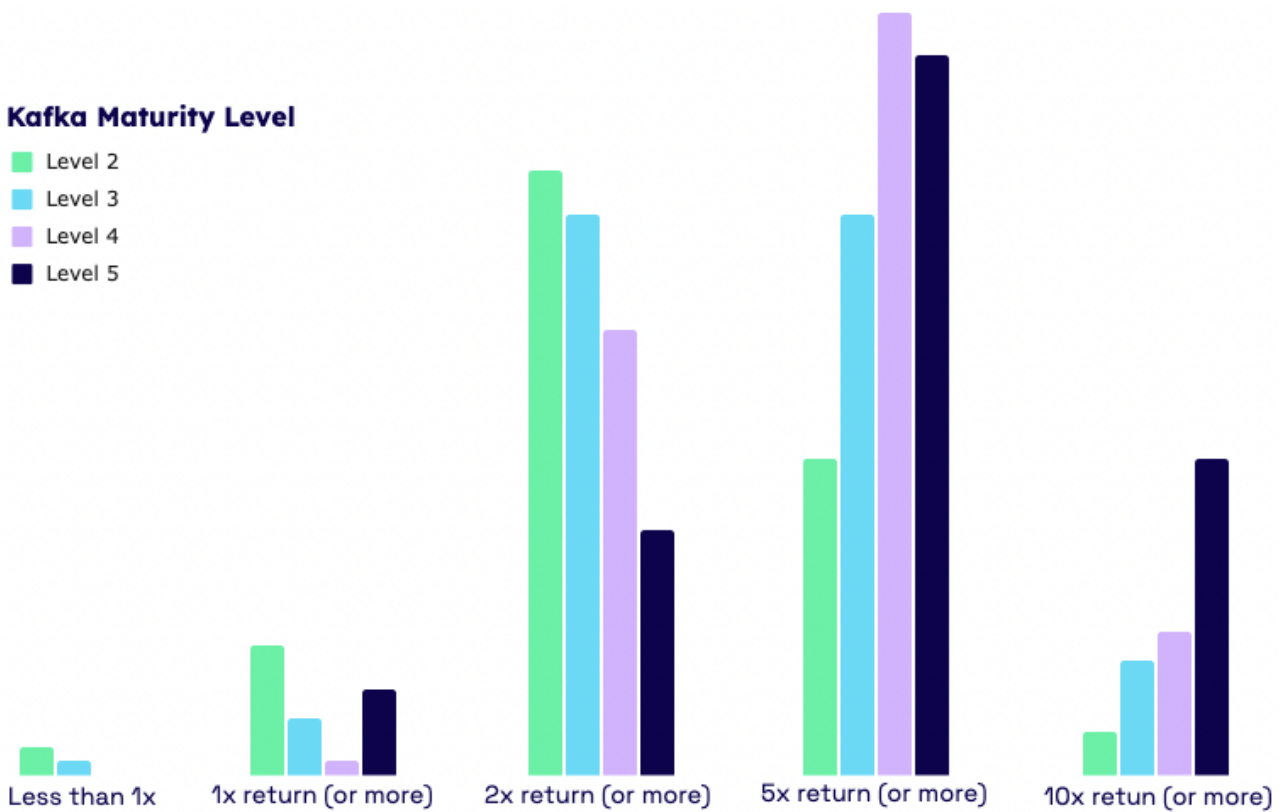
A lot of companies are still struggling to provide a good developer experience around Kafka that will improve developer productivity and wider adoption of event driven applications. But it's not just about providing access to bigger clusters. It's about providing a better way to access these clusters and resources that eliminate obstacles and allow self services. Developers should not have to file a list of support tickets and wait forever for a new Kafka cluster to be deployed.

It's a good idea to have a consistent process in place, but it's even better to have a consistent experience for your developers. Especially if your organisation has numerous different teams, each with its own set of use cases being built on top of Apache Kafka.

## 2023 Data Streaming Report

The 2023 Data Streaming Report released by Confluent has surveyed many organisations and discovered Data streaming's return on investment grows as adoption and breadth of Kafka use-cases becomes more widespread and integrated.

### Overall ROI achieved or anticipated



### The Key Findings

**1 Data Streaming Drives Business Value and Strong Return on Investment**

**64%** of organisations in Level 2 are achieving or anticipating 2x-5x return, and this increases to 78% with companies at Level 3

**2 Data Streaming is a Top Priority for IT Investments**

**89%** of respondents say investments in data streaming are important, with 44% citing it as a top strategic priority.

**3 Data Streaming's Biggest Challenges: Silos and Skill Gaps**

**74%** of IT leaders cite fragmented projects, uncoordinated teams, and budgets as major hurdles or challenges.

### 3. Application Development Teams

The application development teams deliver event driven applications and data products using Apache Kafka at the core. These teams are aligned according to the organisation's data strategy. A common approach is for organisations to organise their development teams by-product, although in some cases it can be by the data set owners. Application development teams care about moving faster and deploying new features and updates with little or no friction.

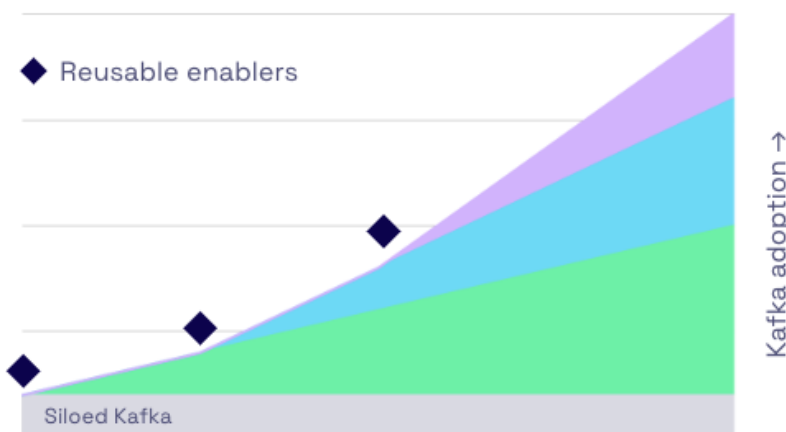
### 4. The Kafka as a Service (KaaS) approach

While there are many ways to deliver the Apache Kafka platform and its tooling to application developers, the favourite approach of high-performing DevOps teams is to deploy an Self Service, prescriptive set of YAML files on Kubernetes that developers use to create and access the Kafka resources they need on their own.

The platform team has a critical mission: to provide the foundational capability to create these Kafka resources and the guard rails that enable application development teams to work effectively when building on top of Apache Kafka. A platform team delivers self-service automation to developers to repeatedly create, update and manage Kafka clusters at scale. The Kafka as a Service (KaaS) offering they build is a collection of self-service automations, tools, knowledge, and support enabling Kafka cluster deployment at scale throughout the whole organisation.

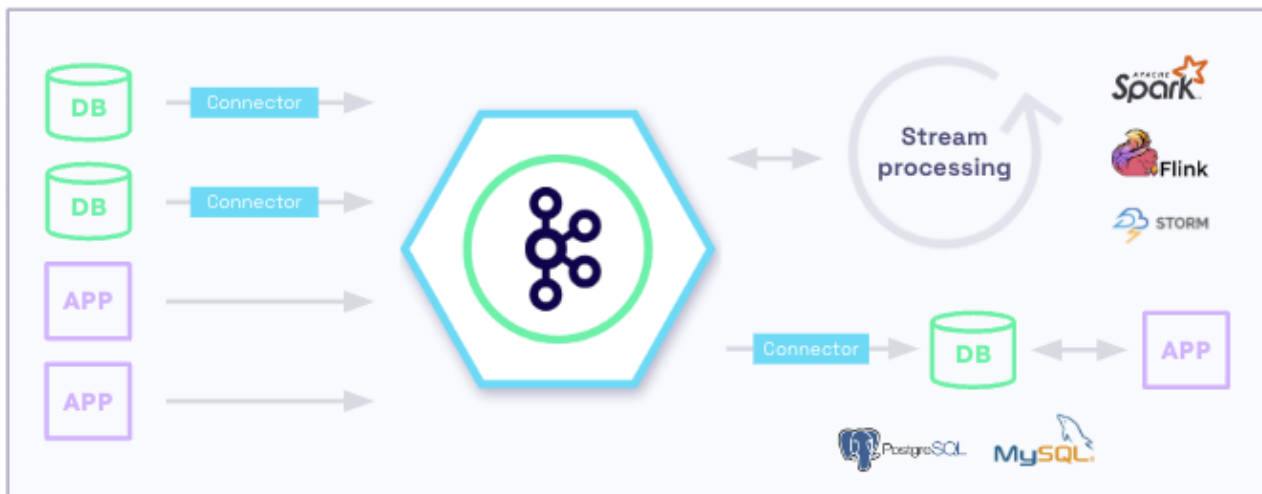
A development team should have a high degree of autonomy, but they also rely on the central Kafka platform team to deliver the resources they need to get their work done. In this way, application development teams are internal customers known as tenants of the platform team. In order to properly fulfil its role as an internal service provider, the Kafka platform team must take a strategic approach to resource management.

#### Increasing your Kafka adoption curve with KaaS



## 5. Why a KaaS approach

The KaaS approach is being adopted by every kind of organisation, from every industry, across every part of the world. Having a combination of a well defined onboarding process matched with a documented governance process on the self service wrapper enables every product team to deploy changes to Kafka in a simple, compliant, and repeatable manner.



The job of the KaaS platform team in this case is to enforce compliance while giving more power to developers. There are numerous reasons for adopting a KaaS approach to Apache Kafka delivery. Here are some of the key reasons:

- A. Manage Kafka deployments from on-premise to cloud to edge
- B. Scale Kafka clusters to meet demands
- C. Enable built-in compliance using templated ACLs
- D. Enforce zero-trust security using Open Policy Agent

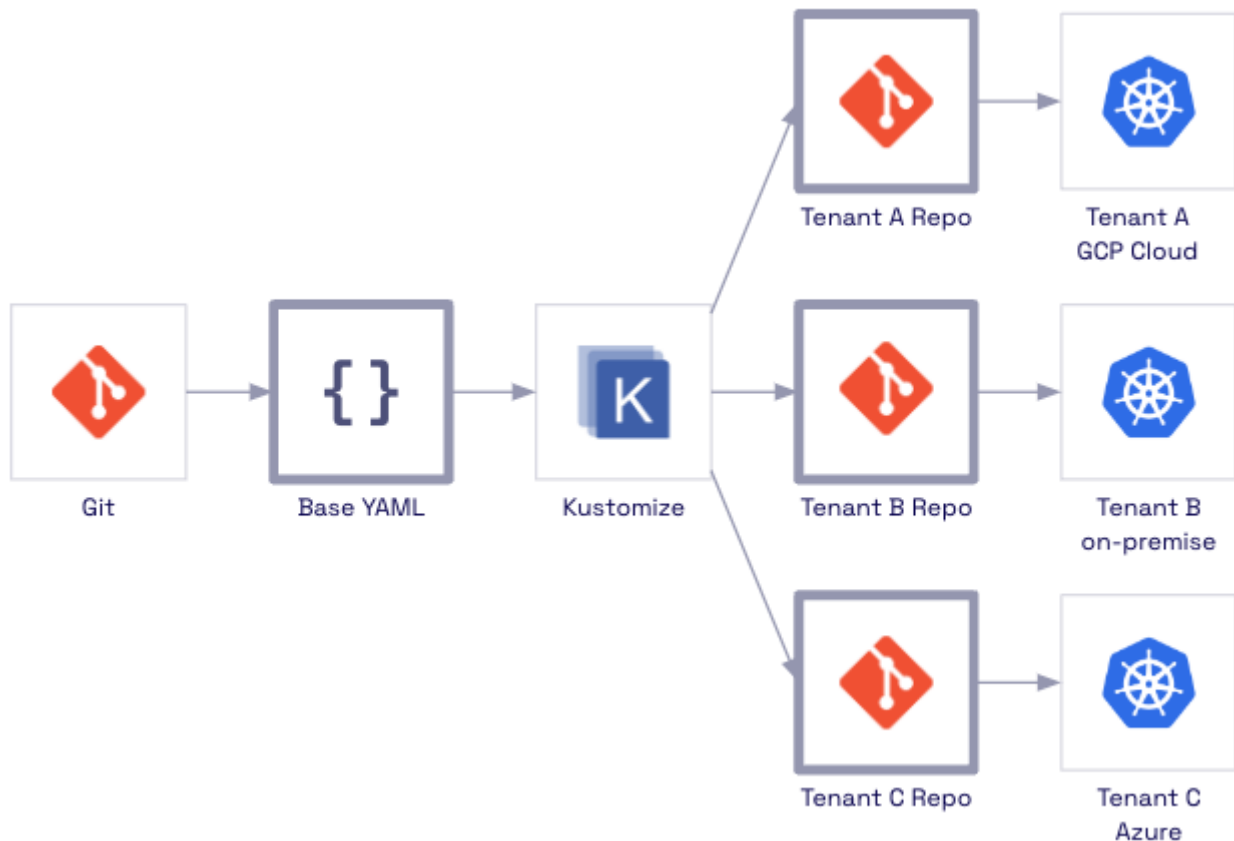
**Let's look at each of these reasons in detail.**

## 6. Benefits of KasS approach

### A. Manage Kafka deployments from on-premise to cloud to edge

As the infrastructure stack becomes more complex, what organisations need is a consistent way to build and deliver Kafka deployments from on-prem to edge. With the use of the Kafka Connect framework, organisations that need to support legacy applications can leverage Change Data Capture (CDC) in a federated model to capture changes from one deployment to another.

The KaaS platform approach at Lloyds Banking Group in the UK enables thousands of application owners and vendors to securely adopt Kafka and deploy their clusters on-premise, across multiple sites, and even across public clouds with the standard GitOps operating model.



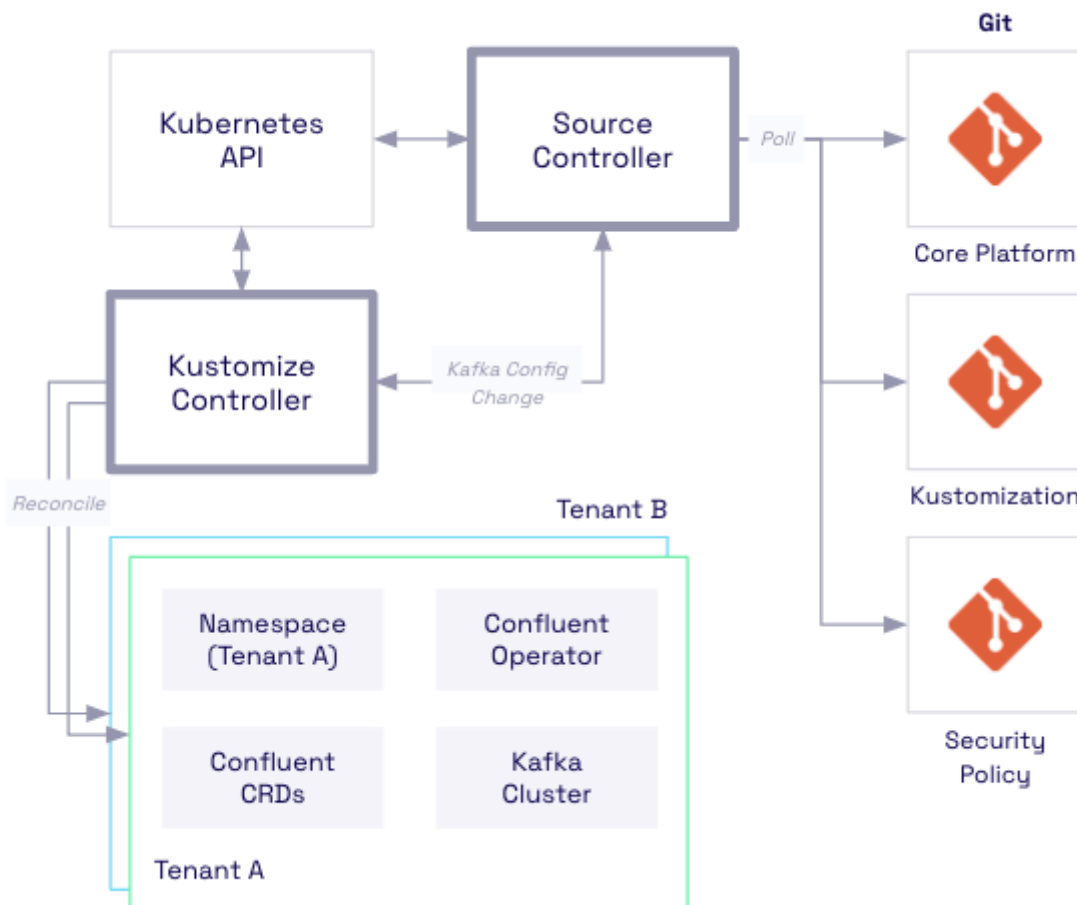
With this tremendous infrastructure and data complexity, Lloyds Banking Group relies heavily on the consistency that the KaaS platform approach brings.

## B. Scale Kafka clusters to meet demands

Platforms are exceptionally good at making components repeatable and reusable using the standard Kubernetes operators. This trait is especially useful when an Apache Kafka cluster needs to be scaled to increase throughput.

Being in the retail space, Dufry which has 5,500 outlets in 75 countries together with online has many days of unusually high traffic spikes. For example, to prepare for a Cyber Monday sale, the platform team had to start provisioning additional Kafka cluster resources (brokers) two to three weeks in advance, and always over provision. Very short internal deadlines for big product launches, sometimes 30 minutes or less, had the teams struggling to scale up to support a 10 fold increase in web traffic. Some of their applications that usually experience around 300 OPS (operations per second) can shoot up to 3000 OPS in a matter of minutes.

By leveraging the Kubernetes Kafka Operator approach, Dufry was able to adopt a cloud-first approach. This gave them the power of cloud scalability as they could easily configure new Kafka cluster resources in minutes to meet the surge in demand.

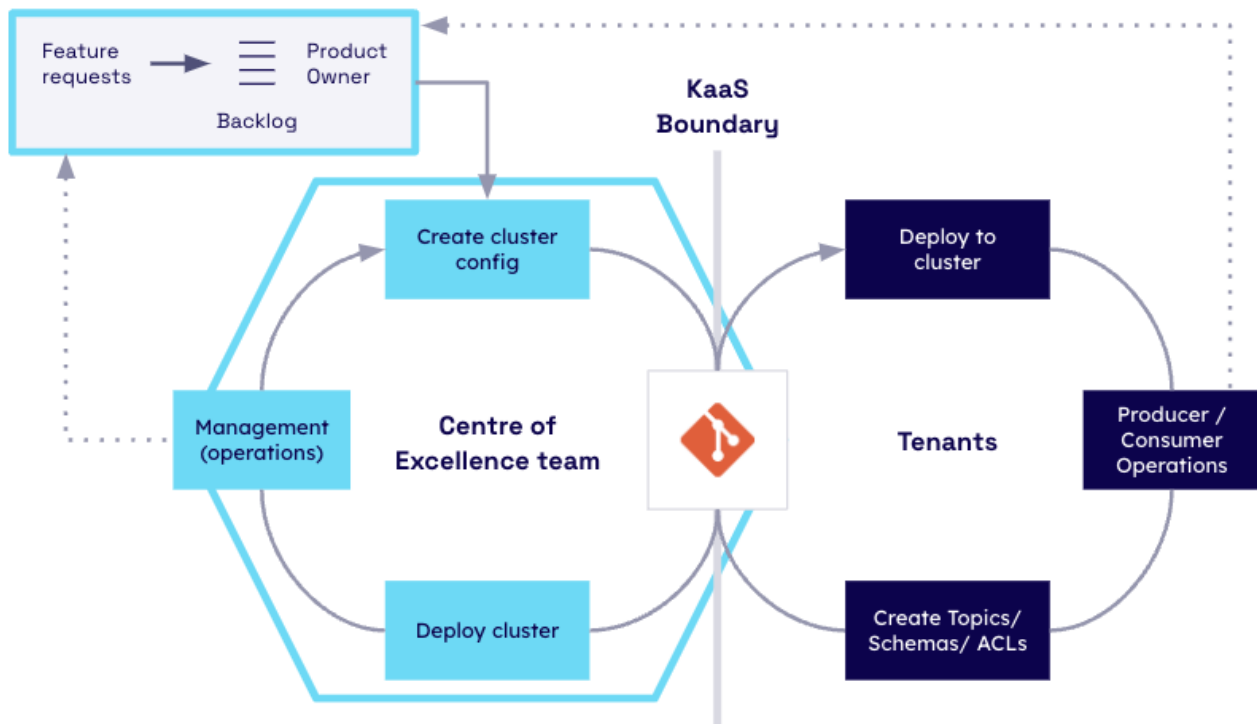


During the POC stage, the evaluating product team discovered that they were able to operate a bunch of Kafka clusters with only 3 developers. This instilled confidence in the team to move away from traditional VM architecture, and to build consistent end-to-end workflows that simultaneously increase consistency and introduce standardisation. Today, Dufry is confident that their Kafka clusters meet any spike in traffic without a hassle.

### C. Enable built-in compliance using templated ACLs

Security comes first for any organisation, particularly an organisation like the Department for Work and Pension (DWP) in the UK. With the whole of the UK government pension system based on the information the department processes, there is no room for error with security. The DWP has been transforming itself in recent years from a traditional organisation with siloed departments and legacy systems to one that is founded on open source software and modern cloud-native practices. Along the way, they have kept DevSecOps as their top priority. To make this transformation possible, they leveraged Kafka to connect and integrate modern data pipelines with legacy data.

Security is baked into the KaaS platform at DWP using a templated ACL approach so that developers only need to approve read permissions requests to certain topics to have capability of sharing data with other services in the organisation. Application developers are able to consume data from these topics without ever reaching out to the cybersecurity people outside of their team. This bakes in security from the get-go and allows a security-first mindset even when developers are working with test data.

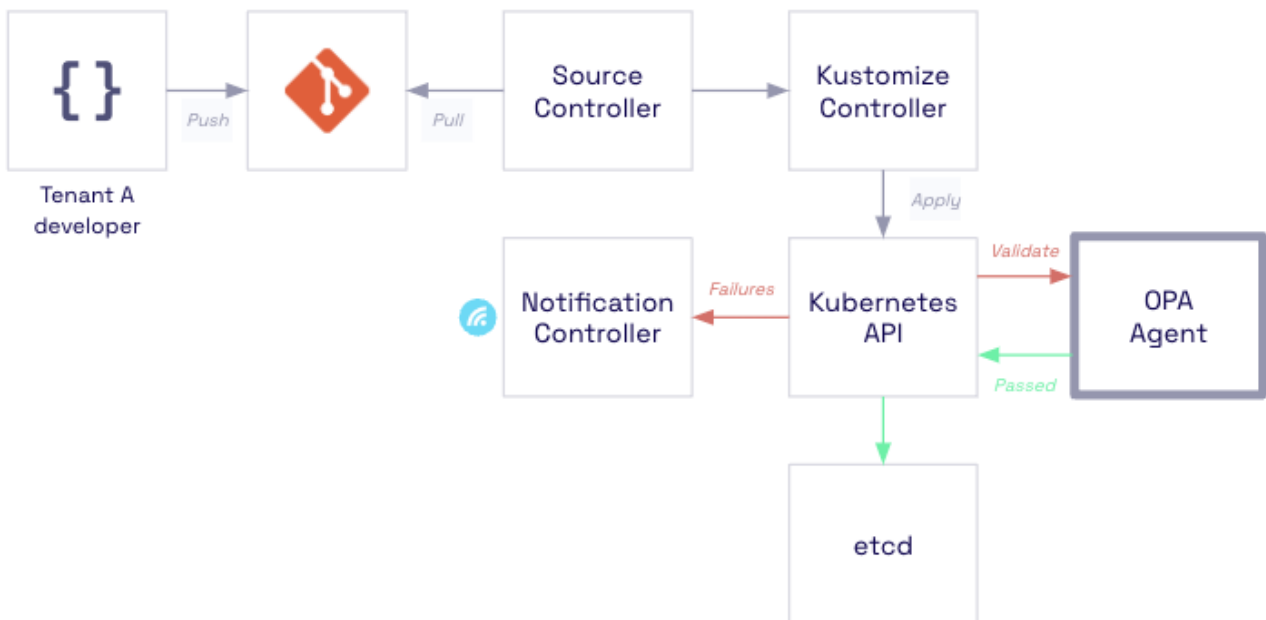


### D. Enforce zero-trust security using Open Policy Agent (OPA)

Implementing a policy engine to block objects that could harm or affect the cluster if they don't meet the central Kafka requirements is essential in self service platforms. Using policies enables users to build complex configurations that other tools, such as Terraform or Ansible, cannot achieve.

Using policy as code to establish Kafka guardrails, enforcing built-in compliance, Open Policy Agent is a tool that allows the use of a high-level declarative language called [Rego](#) to specify these policies as code, such as validating tenant configuration before its applied to Kafka clusters. This greatly improves security as bad actors cannot easily apply / update new configuration from their repositories.

The first benefit of using OPA is you decouple your policies from your infrastructure and Kafka clusters so that people responsible for policy management can control the policy separately from the tenant requests for Kafka based resources. The second benefit is localising violations and conflicts, and reducing the risk of impact on other tenants by human error or misconfigurations. Policy decisions are not restricted to yes/no or allow/deny answers, this flexibility can be used to validate tenant throughput and topic partition count. Like query inputs, your policies should create structured data as an output which can be re-used across tenants. This is what baked-in security looks like in practice.



## 7. An internal KaaS platform is built not bought

A KaaS platform is a complex system made of technical and business processes custom to your organisation and it cannot be bought ready made from any vendor. It's something that needs to integrate into the operations of your teams and tailored to the specific needs of each organisation. No two organisations will have the same kind of KaaS platform. However, there is one common trait that all platforms have. They need to be built using a set of modular 'building blocks' that work across any type of infrastructure, and are flexible enough to meet all the varied needs of the business.

One option that has proved to be an elegant and effective solution to platform building is Git. Git is widely used in software delivery, and its success is the reason why platforms like GitHub and GitLab are household names in every software delivery team today. Git has led to the rise of a modern approach to software delivery called GitOps.

## 8. The way to build a KaaS platform through GitOps

GitOps views Git as the single source of truth. The entire system needs to be declared in Git and any changes are initiated using a simple pull request. With the introduction of [CFK](#) and the [Strimzi](#) operators it is now possible to declaratively define your entire Kafka cluster and all its resources in YAML.

GitOps allows you to create and manage your entire Kafka platform through Git and treat your Kafka resources as you would your source code. Every change and broker configuration is done through a pull request and everyone has access to the repository, which can be rolled back if needed. Just like with your code, you should trust the platform you are building on. With GitOps, you can make sure that each configuration change is verified by your peers and automatically tested before it is deployed.

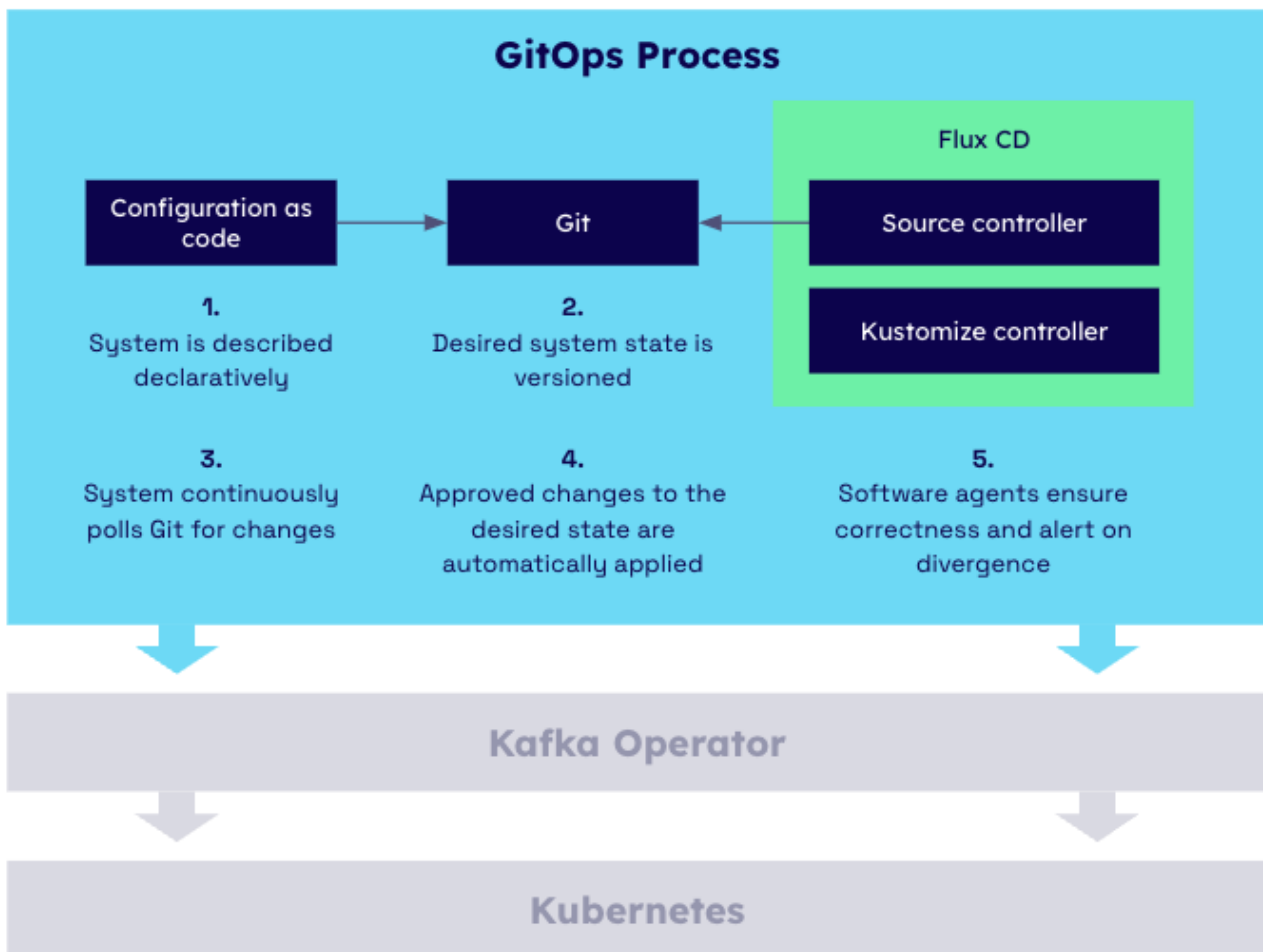
Your KaaS platform should not be managed by one single person. It's a bad idea to have one person who deploys code without anyone else knowing about it. GitOps allows you to have a team of people who are responsible for each change being made on the platform as a whole. Beyond this, it helps to build a platform that is modular, flexible, and importantly, secure.

## 9. GitOps for KaaS platform building

GitOps is a way to manage Kafka configuration, topics and ACLs from Git repositories. It requires that every part of the Kafka platform be defined in Git. With Git as the single source of truth, and all platform configuration stored in.

Using Git, Kafka operations can be automated with ease. Whenever configuration drift occurs, it is immediately spotted and corrected in a controlled manner.

GitOps is great for building internal platforms because it has a set of strong defaults for versioning using Git repositories. The GitOps framework can greatly simplify and speed up the process of building and maintaining internal platforms. It is much easier to manage internal applications when using a strategy like GitOps because you can share and replicate resources, tools, and workflows across various teams from a single platform.



With GitOps, everything about your Kafka cluster (Brokers, Topics, Users, ACLs, Schemas) must be declarative, which means deployments are reliable and that any action can be replayed or rolled back at any time.

## 10. Benefits of GitOps for KaaS platform building

Here are the top benefits of using GitOps to build and manage your Kafka clusters:

- A. Make self contained risk-aware decisions
- B. Detect broker configuration and topology drift
- C. Enable built-in compliance
- D. Implement on-demand Kafka clusters

**Let's look at each of these benefits in detail.**

### A. Make self contained risk-aware decisions

Risk awareness in deployments is about being aware of what lines of configuration have changed, and having the ability to rollback specific changes if necessary. This kind of precise control over Kafka clusters and change management is a hallmark of the platform approach to software delivery.

Thanks to GitOps, in conjunction with the Kubernetes Kafka Operators, the central KaaS team can now experience almost no downtime rolling out changes to multiple clusters at once. Before GitOps, it was the norm for teams to experience outages during a broker upgrade. Now, instead of outages, we have the ability of automatic rollbacks with the operator doing the hard work. In fact, trusting the operator to order the resource creation process is now common practice, something which was purely a manual sequence a couple of years ago.

### B. Detect broker configuration and topology drift

Drift of any sort plagues distributed teams. Especially as the kafka cluster becomes larger, drift leads to incompatibility, performance issues and even outages in parts of the application stack. GitOps counters this challenge by embracing a declarative approach to building and configuring Kafka clusters. This ensures that the clusters, topics, Users and ACLs always stays faithful to the original declared state as in Git.

GitOps effectively reduces your attack surface by not allowing people to make manual changes to production - changes can only be applied through a reviewed change in Git. In the cluster, in the infrastructure, in your data governance rules, you do everything in Git. Drift detection doesn't just make Kafka clusters easier and more predictable, it greatly tightens security controls.

## C. Enable built-in compliance

For organisations in highly regulated industries like financial services and healthcare, compliance is a high priority. They have specific needs to have data stored only on-premises and ensure a complete audit trail of all actions performed within the system. With these high expectations, “ClickOps” or any hoc changes will not hold up. Instead, only a battle tested platform approach with full delivery pipeline automation can deliver the peace of mind required.

Lloyds Banking Group, the UKs largest bank by customer accounts, faced the challenge that every Kafka cluster they deployed must meet a unique mix of regulatory, security and data governance requirements. The central KaaS platform wanted the scalability and reliability of adopting Kubernetes, and also sought to leverage the ecosystem of cloud-native open source projects like [ArgoCD](#) and [FLuxCD](#) in order to improve their time to value.

Lloyds Banking Group, defining a Kafka resource naming standard which was enforced with a combination of OPA policy and Argo events. Automating these governance rules and managing it with GitOPps provided a way for the SREs to validate new topics through a pull request. The result is a repeatable, scalable platform that spans environments with security guardrails specific to the underlying Kafka cluster. A good example of this, topic replication factor in a dev environment will be different to that of production due to resilience requirements.

## D. Implement on-demand Kafka clusters

With the introduction of [ApplicationSet](#) in the Argo CD project, specifically the [Git Generator](#) Kafka clusters can now be provisioned intuitively from a new branch / directory or Pull Request using a GitOps workflow. These are essentially ephemeral environments, which will be blown away after a testing cycle or feature branch has been tested and merged. In traditional testing workflows, you may have deployed your changes to a shared development Kafka cluster, waited for the QA to validate your changes and received feedback. At this point, time was wasted between various teams with environment coordination or test data to validate the new changes.

With the combination of the ApplicationSet templating and the concept ephemeral environments, you can quickly spin up a test Kafka cluster based on the changes of your feature branch. This means the QA or UX team can suggest improvements or changes during the code review process without wasting cycles in isolation of affecting any other team consuming Kafka.

## 11. The path to Kafka as a Service adoption

The time it takes for your organisation to adopt Kafka as a Service greatly depends on where you are in your Kafka maturity journey. If you've already been leveraging Git for collaboration, have a CI/CD pipeline in place, and use open source tooling across your pipeline, you may see quicker adoption time. However, for many organisations, it can take more time.

OSO recommends a 3 phase approach when adopting Kafka as a Service through GitOps:

1. Identify a developer advocate / team who is happy to act as the early adopters.
2. Build a MVP with a clearly defined set of functional and operational requirements.
3. Gather feedback from your tenants and iterate on feature requests.
4. Onboard more tenants as platform maturity and trust in the stack increases.

This phased-out approach ensured adequate time for the various teams to get on board at their own pace. It also provided the KaaS platform team time to adapt to issues as they arose, ensuring a smoother Kafka adoption.

### Describe everything as code

Even for organisations that have been deploying Kafka clusters over multiple environments for a while, the concept of 'everything as code' can be a far stretch. Often many changes are made adhocly, yet it is essential for GitOps and to enable a platform model. To build declarative systems, every part of the system first needs to be defined before it can be declared. This is why it is important to take the time to transpose the entire system from various disparate pieces to Git.

### Leverage tools like CFK Operator or Strimzi, Flux & Argo as a POC

KaaS is powered by a combination of tools that are backed by large developer communities. Flux and Argo are two key open source tools that make up the GitOps toolchain. The Kafka operators available each have their own benefits and drawbacks, an evaluation activity outside the scope of this guide should be undertaken before using one over the other.

## Key Takeaways

To summarise, here are the key points to take away to enable you to start building a Kafka as a Service capability in your organisation:

- **Developer experience matters:** Development teams today require a self-service developer experience (DX) to enable them to consume Kafka for their specific use-case.
- **Leverage the Kafka as a Service platform:** Top performing data driven organisations find the KaaS platform approach to be the most effective way to share data products across functional business units . Here are the top benefits of the KaaS approach:
  - ◆ Manage deployments from on-premise to cloud to edge
  - ◆ Scale infrastructure to meet demands
  - ◆ Enable built-in compliance using templated ACLs
  - ◆ Enforce zero-trust security using Open Policy Agent
- **Use GitOps for KaaS platform building:** Building and managing platforms can be challenging and requires an approach that is flexible, yet secure. GitOps fits the bill as the ideal solution for platform building. Here are the top benefits of using GitOps to build an internal platform:
  - ◆ Make self contained risk-aware decisions
  - ◆ Detect broker configuration and topology drift
  - ◆ Enable built-in compliance
  - ◆ Implement on-demand Kafka clusters
- **KaaS adoption best practices:** As you look to develop and adopt Kafka as a Service in your organisation, here's how to go about it:
  - ◆ Take a phased approach
  - ◆ Describe everything as code
  - ◆ Leverage tools like CFK Operator or Strimzi, Flux & Argo as a POC
- **Engage experts:** Leverage OSO enterprise experience to greatly simplify and increase the adoption of Kafka inside your organisation.

## OSO - The Kafka Experts

OSO, is on the journey to becoming the number one Kafka consulting company in Europe.

Our mission is to help teams build event driven applications on Kafka to enable real-time access to your data and introduce meaningful innovations that drive business growth.

We offer a variety of tailored services and bring industry-leading expertise to support Apache Kafka in your organisation. Our developer-first culture, combined with our cross-industry experience and battle-tested delivery methods allow us to deliver the most impactful solutions for your business.

The OSO team has worked with digital native businesses globally to build best-in-class sector-specific solutions and partnered with the world's leading technology companies to help businesses implement Kafka at scale.

Since being formed, we have worked with a number of well-established clients on these initiatives including Lloyds Banking Group, HSBC, Fortescue Metals Group, HS2, M1 Finance, and Dufry.

To learn more about how OSO can help your organisation adopt Apache Kafka visit: [www.oso.sh](http://www.oso.sh)



Contact Us for a Free Kafka Health Check

## Further reading and sources:

1. **Customer Story:** [Dufry creates value from data with Kafka architecture](#)
2. **Video:** [Enterprise guide to building a DataMesh](#)
3. **GitHub:** [Kafka GitOps examples](#)
4. **Customer Story:** [Automating Kafka deployments through GitOps](#)
5. **Customer Story:** [How the Department for Education became a data-synchronous organisation](#)