FEDERATED OWNERSHIP

Accelerate Revenue, Scale With the Same Team, and Cut Operational Costs

Automate guardrails, define ownership, and enable discovery. Developers get autonomy, platform teams maintain control, and every team creates data products they can trust.

Kafka scales. Manual processes don't.

The scripts, the Slack requests, the 'just ask the platform team'? They worked at 5 teams, but scaling them is a full-time job. We see this pattern repeatedly across organizations growing from a handful of Kafka consumers to dozens of teams with hundreds of applications: provisioning bottlenecks, zero accountability for resources, and teams rebuilding what already exists because they can't find it. Federated ownership gives developers autonomy and platform teams control.

01

Misconfigurations cause downtime and erode reliability

A single misconfigured topic can take down a production service, costing hours of developer time to diagnose and fix.

Scripts provision resources but can't validate against standards, so edge cases multiply across teams and broken configs slip through undetected.

Every hour of unplanned downtime directly impacts revenue and customer trust, while 10+ hrs/week of manual remediation drains platform capacity.

With Conduktor
CEL-based policies validate every resource request automatically. No manual review, no broken configs in production.

02

Unclear ownership slows delivery and inflates platform costs

Developers wait days for access and provisioning because every request routes through the platform team.

At 50+ teams, there's no system of record for ownership, just institutional memory that walks out the door when people leave.

Kafka adoption slows across the organization as teams hit friction at every step, reducing the return on your streaming investment.

With Conduktor
Application Catalog maps every resource to an accountable team. Ownership is defined, not guessed.

03

Duplication inflates infrastructure costs and wastes engineering capacity

Teams rebuild topics, schemas, and pipelines that already exist because there's no way to discover what's available.

Across dozens of teams, duplication compounds with redundant infrastructure, engineering, and no visibility into what's already been built.

$100k+ per year in wasted spend, with sensitive data shared through Slack threads instead of governed access controls.

With Conduktor
Topic catalog enables discovery across clusters with visibility controls. Teams reuse instead of rebuilding.

IMPACT WITH CONDUKTOR

UP TO
75%
Fewer Provisioning Tickets
From developers self-serving instead of filing requests

OVER
$200k
Annual Savings
From reduced duplication, faster provisioning, and clear ownership

OVER
3,500hrs
Saved Per Year
Across provisioning, access reviews, and ownership resolution

UP TO
4x
Faster Provisioning
From 16 hours down to 4 hours per request

FLIX

TRANSPORTATION & LOGISTICS

Federated Ownership at Scale

FlixBus adopted a data mesh architecture to give domain teams full ownership of their Kafka resources. Each team defines their resources and permissions in YAML files, validated through CI/CD pipelines. Conduktor enforces centralized policies while teams maintain autonomy over their own domains.

Their internal motto: **"You build it, you run it, you own it."** Cross-team access requests go through pull requests where resource owners approve, not a central platform team.

50+	10	2,300+	Zero
Teams operating independently	Domains with autonomous ownership	Topics governed	Centralized bottlenecks

> ***"Conduktor easily integrated with our CI/CD pipelines to enhance data governance. It's given us the tools to centralize compliance standards across the business, while allowing team-specific autonomy to speed up our processes."***
>
> *Taras Slipets, Staff Data Engineer, FlixBus*

How We Do It

Conduktor Console gives developers autonomy within guardrails and gives platform teams control through policies, without becoming a bottleneck.

Topic Catalog

- Discover data across clusters. Control visibility to protect sensitive resources.
- Reuse existing topics instead of rebuilding. Reduce duplication across teams.

Self-Service Provisioning with Automated Guardrails

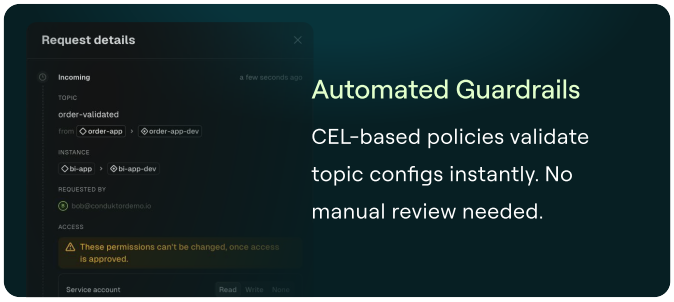
- Developers create topics, schemas, and connectors themselves. CEL-based policies validate instantly.
- No ticket queue, no human reviewer, no broken configs in production.

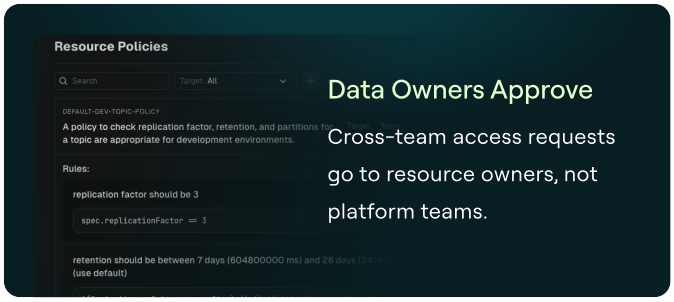
Application Catalog

- Define applications with ownership patterns so every resource maps to an accountable team.
- No more guessing who owns what when incidents happen.

Approval Workflows

- Request access to other teams' resources for cross-team collaboration. Data owners approve or reject, not platform teams.
- Works with GitOps or direct approval.





Ready to automate Kafka governance?

See how federated ownership works in your environment.

See it in action now