STATE OF CLOUD NATIVE DEVELOPMENT Q1 2026

/DATA

CLOUD NATIVE COMPUTING FOUNDATION

MARCH 2026

Can I share data from this report?

1. License Grant

This report is licensed under the Creative Commons Attribution-NoDerivatives Licence 4.0 (International) . Put simply, subject to the terms and conditions of this license, you are free to:

Share — You can reproduce the report or incorporate parts of the report into one or more documents or publications, for commercial and non-commercial purposes.

Under the following conditions:

Attribution — You must give appropriate credit to SlashData™ , and to the Continuous Delivery Foundation as sponsors of this report, and indicate if changes were made. In that case, you may do so in any reasonable manner, but not in any way that suggests that SlashData™ endorses you or your use.

NoDerivatives — you cannot remix or transform the content of the report. You may not distribute modified content.

2. Limitation of Liability

SlashData™, believes the statements contained in this publication to be based upon information that we consider reliable, but we do not represent that it is accurate or complete and it should not be relied upon as such. Opinions expressed are current opinions as of the date appearing in this publication only and the information, including the opinions contained herein, are subject to change without notice. Use of this publication by any third party for whatever purpose should not and does not absolve such third party from using due diligence in verifying the publication's contents. SlashData™ disclaims all implied warranties, including, without limitation, warranties of merchantability or fitness for a particular purpose.

SlashData™, its affiliates, and representatives shall have no liability for any direct, incidental, special, or consequential damages or lost profits, if any, suffered by any third party as a result of decisions made, or not made, or actions taken, or not taken, based on this publication.

The analyst of the developer economy | formerly known as VisionMobile

SlashData © Copyright 2026 | Some rights reserved

TABLE OF CONTENTS

KEY INSIGHTS 4

INTRODUCTION 5

TRENDS IN CLOUD NATIVE DEVELOPMENT 9

THE PATH TO INCREASED CLOUD NATIVE MATURITY 30

APPENDIX 46

METHODOLOGY 49

REPORT HIGHLIGHTS

- 19.9M cloud native developers globally in Q1 2026. →
- 71% of backend developers are using at least one technology or practice associated with cloud native computing. →
- Hybrid cloud reached an all-time high, with 34% of developers deploying to it, likely driven by data sovereignty requirements and regulatory compliance. →
- 88% of backend developers work with some form of DevOps standardization. →
- As developers increase their cloud native maturity, feature flagging acts a crucial gateway technology between mainstream infrastructure practices and more advanced practices. →
- The advanced practices (chaos engineering, service meshes, immutable infrastructure practices, multicluster management) form a tight "elite cluster" with higher levels of cross usage than expected by chance. →
- AI developers follow a different maturity pathway, with immutable infrastructure practices being a gateway to more advanced practices rather than a destination. →

INTRODUCTION

01

1. Introduction

Cloud deployment has become an integral part of software development, with the vast majority of developers now relying on cloud infrastructure in some capacity. While cloud deployment has become nearly universal, not all cloud usage is equally sophisticated. Despite widespread adoption of cloud infrastructure, many developers and organizations have yet to fully leverage cloud native technologies and architectural patterns that unlock the cloud's full potential.

This report, produced in partnership with the Cloud Native Computing Foundation, presents the latest insights into the state of cloud native developers in Q1 2026. It provides an analysis of key trends shaping the cloud native ecosystem, drawing on data from the 31st edition of SlashData's Developer Nation survey, fielded between December 2025 and January 2026, which reached more than 12,500 respondents from 100 countries around the world.

1. Introduction

As the term “cloud native” means different things to different developers, for this report, we define developers as cloud native based on the use of two or more specific technologies, which include containers, container orchestration and management tools, service meshes, Kubernetes, cloud functions or serverless computing, event-driven architecture, observability tools, immutable infrastructure practices, chaos engineering, multicluster management, and streaming and messaging services.

This is the second report in the series that includes the “expanded” cloud native audience, which covers the community that exists outside of the traditional backend services space. As such, in this report, we continue to provide estimates for the proportion of cloud native developers across a wide range of development spaces and how this may impact the future of the cloud native community and the development of technologies to serve these new developers. In addition, we look at the popularity of different cloud technologies across these divergent development spaces. Throughout the analysis of the expanded cloud native community, we include additional attention to the machine learning and artificial intelligence community, which is an important group to understand as cloud native technologies seek to meet this moment.

[1] Those involved in backend development are shown more cloud native technologies during the survey. As such, those in backend development must select at least three cloud native technologies, while all other respondents must select at least two.

A new section of this report examines technology adoption pathways and maturity journeys in the cloud native space. Using association analysis, we identify which cloud native technologies are commonly adopted together, revealing distinct progression stages from foundational tools through mainstream practices to advanced reliability and scale management capabilities. This analysis provides technology creators and platform teams with insights into natural adoption sequences and architectural affinities.

Regional Spotlight: Throughout this report, we highlight European cloud native adoption patterns and trends in collaboration with KubeCon + CloudNativeCon Europe 2026 in Amsterdam.

TRENDS IN CLOUD NATIVE DEVELOPMENT

02

The cloud native developer community continues to expand in absolute terms, growing from 15.6M developers in Q3 2025 to approximately 19.9M in Q1 2026. This represents roughly 39% of the entire global developer population and reflects cloud native practices spreading beyond their backend infrastructure origins into mainstream software development across domains.

19.9M

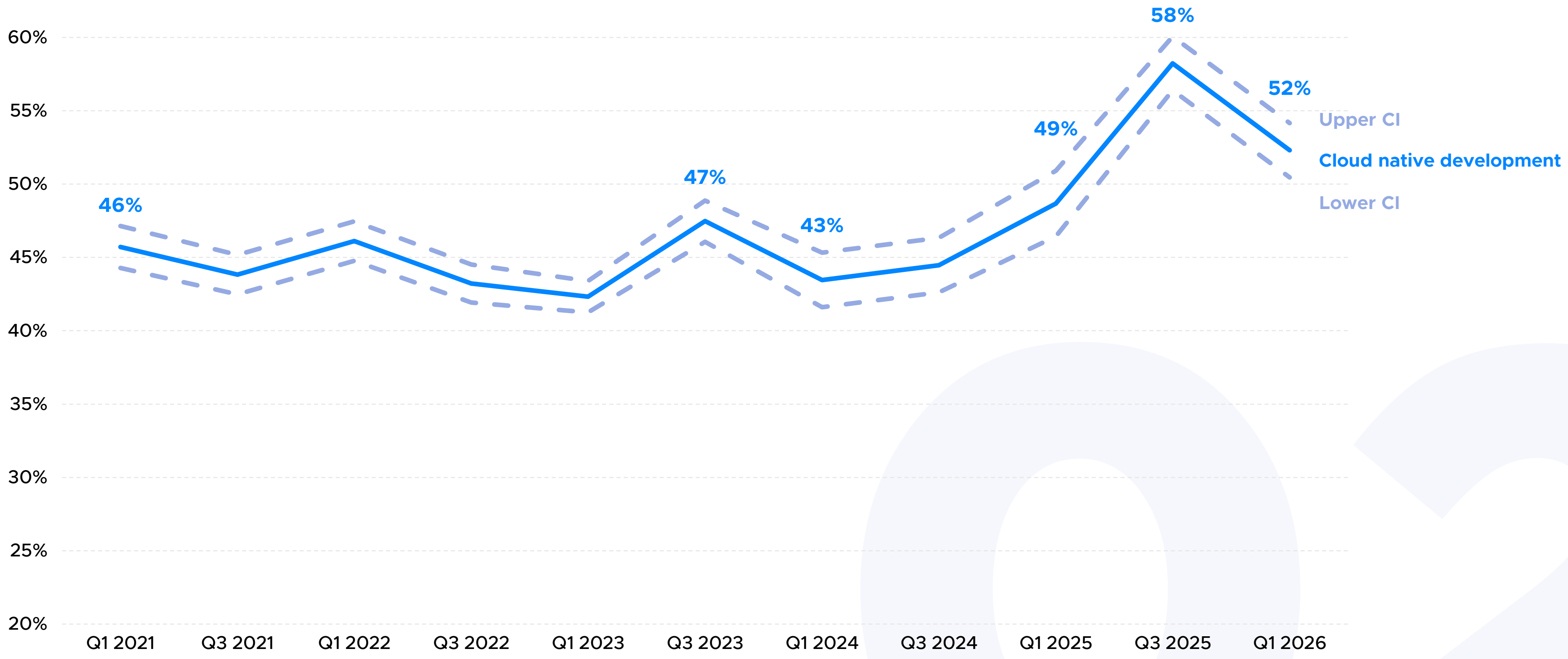
CLOUD NATIVE DEVELOPERS IN Q1 2026

Among backend developers specifically — historically the core cloud native community — 52% are classified as cloud native in Q1 2026, down from 58% in Q3 2025 but still above the 49% recorded in Q1 2025. While the quarter-on-quarter decline is notable, it may reflect normal variation or indicate that Q3 2025's 58% was elevated. The Q3 2026 measurement will clarify whether the typical baseline is closer to 52%, 58%, or somewhere between. Year-over-year growth remains positive.

It is also worth noting that these shifts in values are happening in a period where the developer job market is undergoing pressures — in particular, the restructuring that was seen during the 2024–2025 period. While it is not possible to establish direct causation between the market forces and cloud native adoption, developers educated under the normalization of cloud native technologies entering the market at restricted or impacted rates may be a driver of this six-month decline.

Proportion of Cloud Native Backend Developers

% of backend developers (Q1 2021 (n=4,668) | Q3 2021 (n=5,260) | Q1 2022 (n=5,261) | Q3 2022 (n=5,619) | Q1 2023 (n=8,182) | Q3 2023 (n=4,847) | Q1 2024 (n=2,743) | Q3 2024 (n=2,754) | Q1 2025 (n=1,903) | Q3 2025 (n=2,791) | Q1 2026 (n=2,793)



Question wording: Which of the following technologies have you used as part of your backend services development in the last 12 months?

The expansion of the cloud native population was broad-based rather than concentrated in any single area. Professional games developers, for example, increased from 30% cloud native in Q3 2025 to 39% in Q1 2026, while industrial IoT developers rose from 38% to 42% over the same period. These increases reflect cloud native technologies becoming standard tools across diverse development contexts, not just those in specialized infrastructure work.

In Q1 2026, we estimate that there are 4.5M cloud native developers in Western Europe and 1.2M in Eastern Europe.

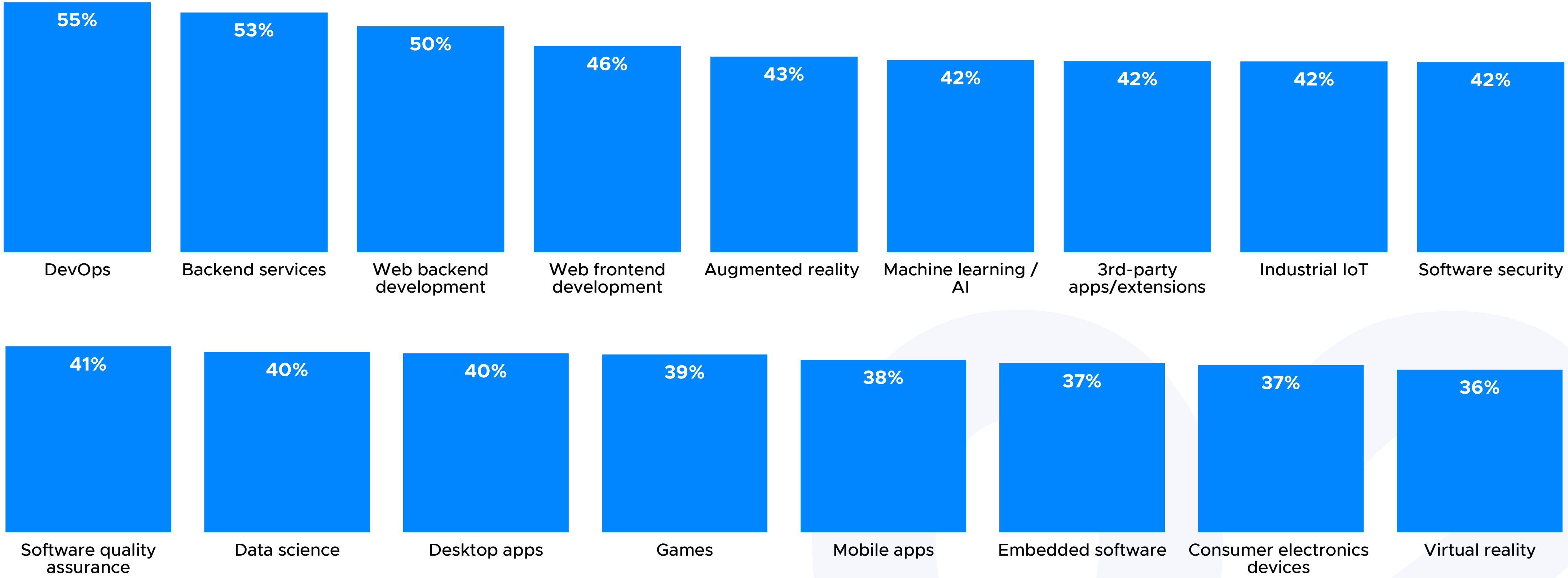
The machine learning and AI developer community showed similar, but more modest, growth, with approximately 7.3M cloud native AI developers in Q1 2026, up from 7.1M in Q3 2025. This segment remains strategically important as cloud native technologies increasingly enable scalable AI operations, distributed training, and production model deployment.

7.3M

AI DEVELOPERS ARE CLOUD NATIVE

Proportion of professional developers in each development area that are cloud native

% of professional cloud native developers in each development area (Q1 2026 n=9,082)



Question wording: How are you involved in the following types of development or projects? | Which of the following technologies have you used as part of your backend or cloud services development in the last 12 months?

Trend in cloud deployment

Beyond the trends in cloud native developers, we also examine how cloud deployment choices have evolved over time. When looking at the entire developer community, we can see that hybrid cloud has shown a strong increase in adoption over the last five years. 22% of developers were deploying to hybrid cloud in Q1 2021, compared to 34% in Q1 2026. Hybrid cloud has emerged as a very popular choice for medium-sized enterprises and those in heavily regulated industries such as finance and healthcare, as it offers additional security and compliance benefits that are increasingly attractive or necessary for many. Evolving data sovereignty requirements and privacy regulations, such as GDPR in Europe and the US Cloud Act, make hybrid deployments attractive for organizations needing to balance cloud scalability with on-premises data residency.

34%

OF ALL DEVELOPERS ARE DEPLOYING TO HYBRID CLOUD IN Q1 2026, THE HIGHEST PROPORTION RECORDED TO DATE

Among backend developers, the usage of hybrid cloud also achieved a new high of 26%, after being stable at around 23% since Q3 2023. It was previously suggested that this stability was likely a consequence of those in backend development having already made the shift towards hybrid cloud several years ago, and only now is the impact beginning to trickle out to the wider developer community that relies on them. The recent increase among backend developers may instead indicate that the remaining non-hybrid backend teams are now adopting hybrid approaches, potentially driven by the same regulatory and security factors that motivated earlier adopters. Alternatively, organizations that previously deployed hybrid infrastructure for specific workloads may be expanding hybrid adoption across additional services.

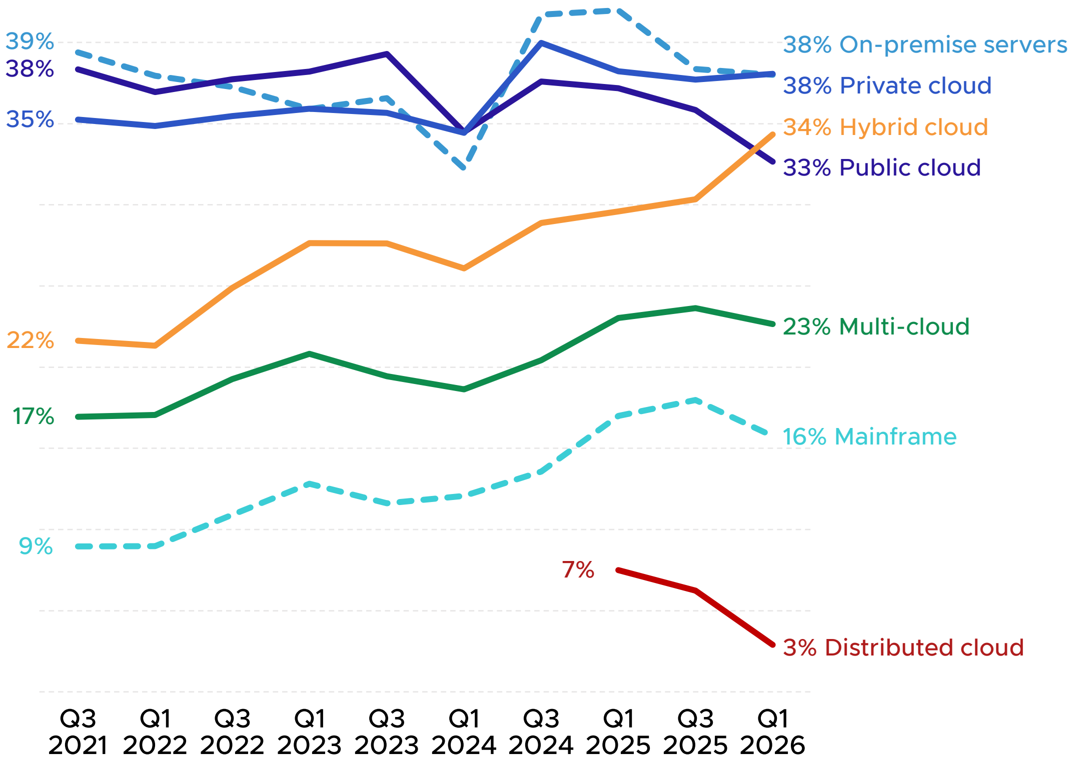
Among European developers, private cloud remains the leading deployment choice, done by 39% of developers, slightly ahead of the rest of the world with an average of 37%, with hybrid cloud at 32% the next most popular cloud deployment approach, but trailing other regions that average 35%. Despite the compliance benefits of hybrid, many European organizations are cautiously remaining with private cloud and on-premise servers.

Public cloud deployment declined by 3 percentage points across both populations (36% to 33% among the general population and 33% to 30% among backend developers), coinciding with hybrid cloud's growth. While we cannot definitively establish causation from cross-sectional data, the pattern suggests many organizations are supplementing or partially replacing public-only deployments with hybrid architectures, likely motivated by cost-optimization or the previously mentioned regulatory and security concerns.

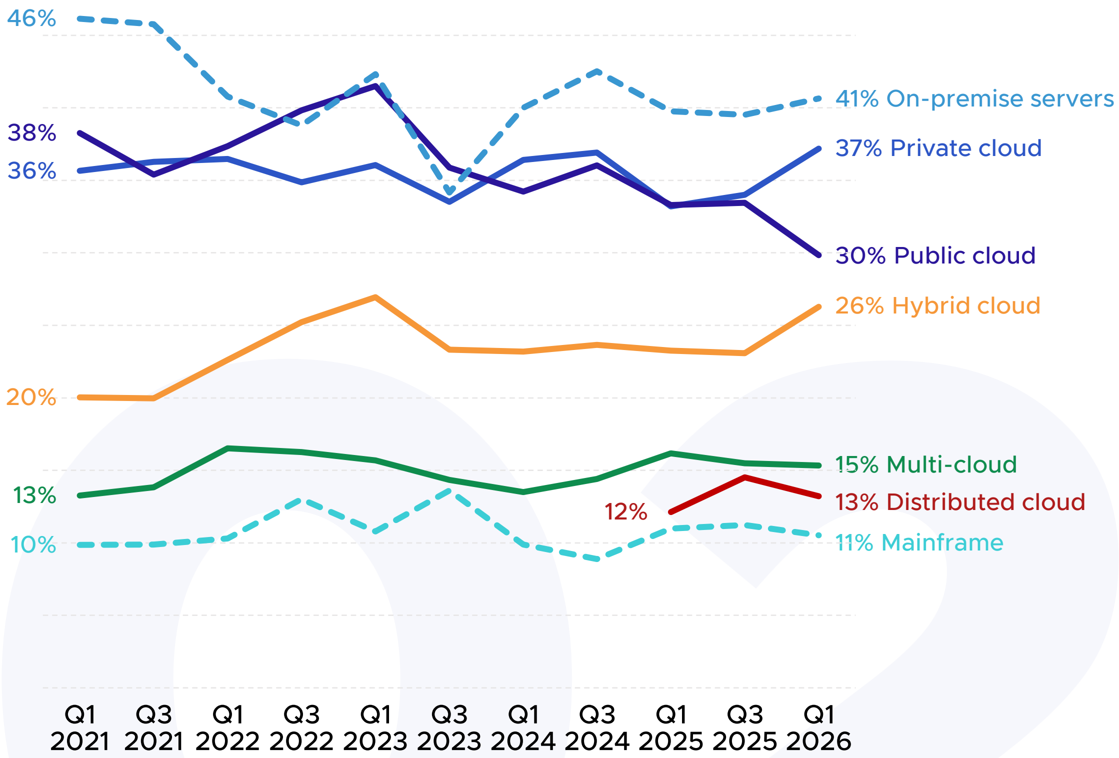
Trends in cloud deployments

% of developers | % of backend developers (21Q1 n=11,177 | 21Q3 n=12,132 | 22Q1 n=13,344 | 22Q3 n=17,820 | 23Q1 n=20,879 | 23Q3 n=10,521 | 24Q1 n=10,521 | 24Q1 n=9,886 | 25Q1 n=10,731 | 25Q3 n=12,021 | 26Q1 n=11,937)

Cloud deployment trend among all developers



Trends in Deployment of Backend Service Developers



Question wording: Which of the following technologies have you used as part of your backend services development in the last 12 months?

Cloud technologies

Cloud native development extends beyond the technologies developers use; however, tracking the use of technologies associated with cloud native development provides important context for how the space is evolving. In this section, we look at the adoption of these technologies among backend developers.

The most immediate change is the sharp decline in container usage (40%) after being consistently above 60% since Q3 2020. While the decrease might look like a dramatic and unexpected falloff of containers, experience with the development world makes it clear that this isn't the case. As discussed in the previous edition of this report, the decline reflects a methodological change in how we define the backend developer community. The current definition includes web backend developers, who typically work with higher-level services that abstract underlying infrastructure, alongside traditional backend infrastructure developers. This compositional shift mechanically reduces reported container usage, even if adoption rates within each subgroup remain more stable.

[1] *A modeling of the historic backend developers has a 58% adoption of containers.*

For other technologies, we see a small decline in most, with an exception for service meshes, which have increased over the last six months, from 8% to 11%, but are still below the previous high before the population change was undertaken. This growth in service mesh adoption, despite the compositional changes affecting other technologies, suggests a substantial increase in adoption, potentially driven by organizations reaching the scale and complexity where service mesh benefits justify implementation costs.

The continued technology self-reported adoption declines, occurring after the compositional shift from expanded backend definitions, suggest a behavioral factor may be at work. As platform engineering practices expand, developers increasingly interact with infrastructure through abstraction layers rather than directly configuring technologies like containers, Kubernetes, or service meshes. This allows developers to focus on application logic and business problems while platform teams manage the underlying infrastructure.

> **The proportion of backend developers with no formalized platform or DevOps practices declined sharply from 20% to 12%, indicating growing infrastructure standardization and abstraction**

While individual platform engineering approaches remain minority practices, the combination has reached critical mass: 88% of backend developers now work with at least one form of infrastructure standardization, up from 80% six months ago. Specifically, those with developer experience teams have increased from 38% to 42%, and unified DevOps systems have grown from 36% to 41%. The 8-percentage-point decline in developers working without any infrastructure standardization coincides with a 6-percentage-point drop in container usage and similar modest declines across other cloud native technologies.

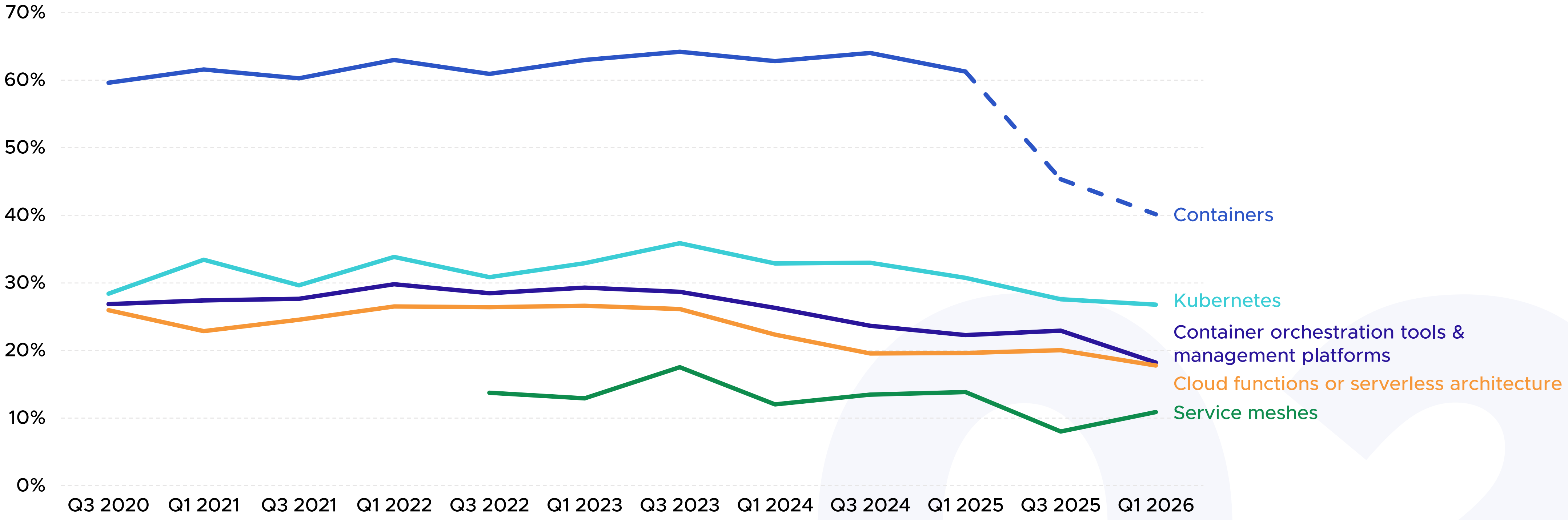
As discussed previously in our research on the "DevOps divide", successful platform engineering creates a separation between infrastructure operators (who manage technologies) and application developers (who consume them through simplified interfaces). This raises a measurement question: When developers deploy containerized applications through an IDP without writing Dockerfiles or configuring registries, should they report themselves as “using containers?”

The simultaneous rise of standardized environments and decline in reported technology usage suggest that infrastructure abstraction is affecting how developers perceive and report their technology use, even if the underlying infrastructure hasn't changed.

Container usage is higher among backend developers in Europe, with 45% of those in Western Europe and 55% of those in Eastern Europe reporting they used them in the last 12 months. Eastern Europe's particularly high container usage may reflect the region's strong representation in infrastructure-focused development roles and cloud service providers, where containerization is standard practice.

Trends in the adoption of technologies associated with cloud native development

% of backend developers (Q1 2021 n=4,668 | Q3 2021 n=5,260 | Q1 2022 n=5,261 | Q3 2022 n=5,619 | Q1 2023 n=8,182 | Q3 2023 n=4,847 | Q1 2024 n=2,743 | Q3 2024 n=2,754 | Q1 2025 n=1,903 | Q3 2025 n=2,666 | Q1 2026 n=2,792)



Note: Due to the changing nature of backend service development and its methodological capture within our survey, the large decrease in containers is reflective of a change in this developer composition. Please see the State of Cloud Native from Q3 2025 for estimates of the trends for the historical group.

Question wording: Which of the following technologies have you used as part of your backend or cloud services development in the last 12 months?

Server technologies and approaches

For the last part of this section, focusing on backend development, we dig into greater granularity on the technologies and approaches used for their infrastructure. This data also enables us, for the first time, to have a clearer understanding of their relative popularity. Further, it allows us to see how widespread the adoption of cloud native technologies and approaches is, even for developers who don't meet the threshold to be classified as cloud native, many of whom still use individual cloud native technologies. 71% of backend developers use at least one cloud native technology, even though only 52% meet the threshold for cloud native classification (3+ technologies). This 19-percentage-point gap reveals many developers are in early stages of cloud native adoption, using one or two technologies but not yet employing the multi-technology stacks characteristic of mature cloud native practice.

> While 39% use microservices, only 6–7% have adopted advanced practices like chaos engineering or immutable infrastructure

API gateways are the most popular technology, used by 47% of backend developers, with microservices being the leading approach, adopted by 39% of backend developers. This reflects the foundational nature of both API gateways and microservices, as they are often either required for other technologies or essential to the functionality of modern service stacks. Notably, while API gateways and microservices are architectural patterns that typically run on Kubernetes infrastructure, only 27% explicitly state that they are using Kubernetes. This suggests that developers may be deploying microservices on non-Kubernetes deployments or, as previously discussed, platform abstraction may make Kubernetes invisible to developers even when it underlies their deployments.

Eastern European developers reported the highest rate of microservices in their projects, 57%, as well as being the most likely to use feature flagging. This pattern may reflect the region's concentration of infrastructure-focused engineering roles and strong representation in cloud service provider teams, where these practices are standard.

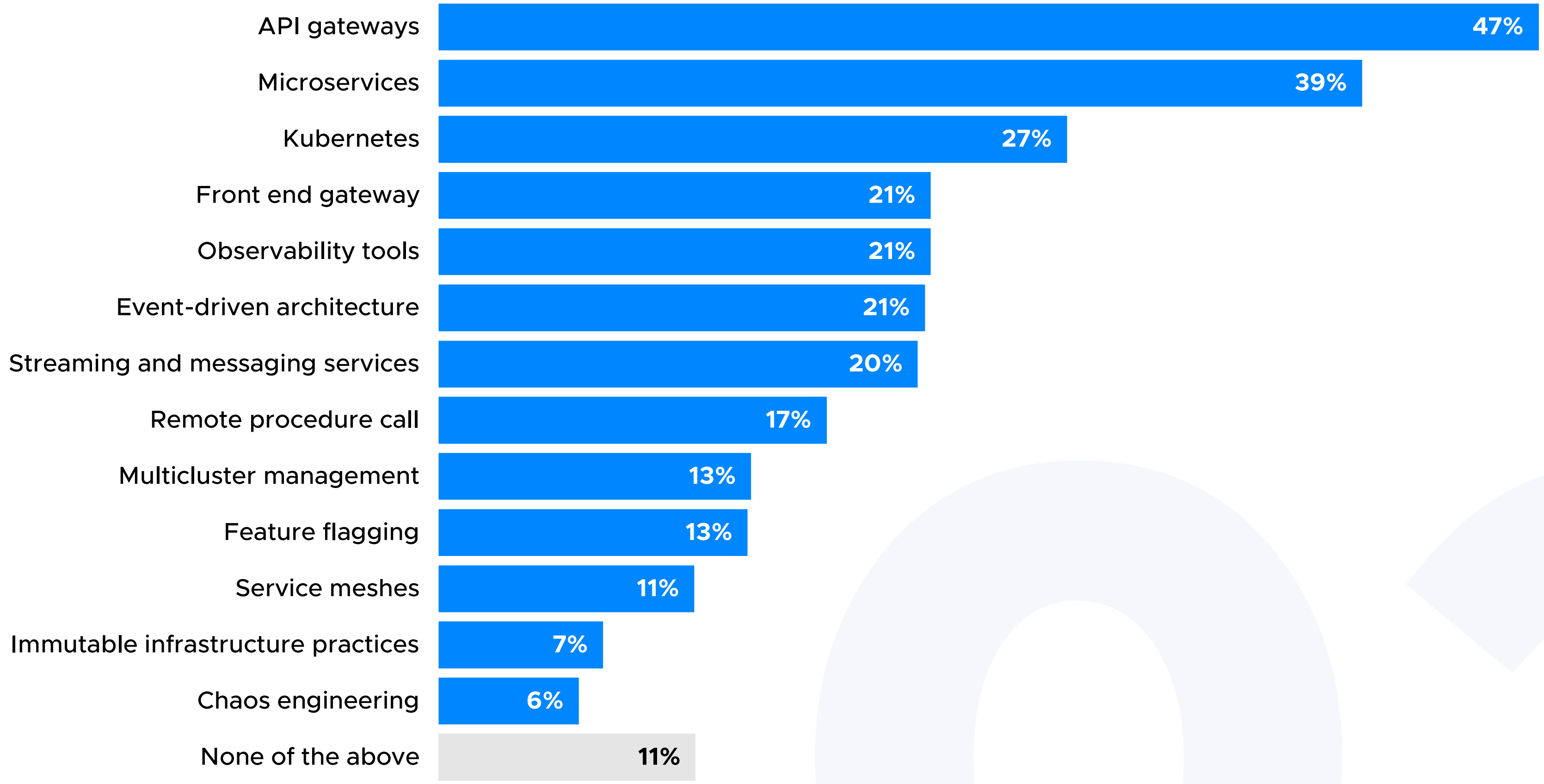
As discussed previously, the rise of IDPs and DevOps process centralization is likely contributing to a reduction in self-reporting of technology usage. In this case, we may have developers who do not consider themselves users of Kubernetes, despite using technology frequently dependent on it. While increased usage of IDPs and centralized DevOps processes does reflect a stark and impressive move towards greater maturity of an organization’s platform engineering, organizations should still ensure infrastructure expertise exists within their teams to manage, troubleshoot, and optimize the underlying platforms that enable developer productivity.

After Kubernetes (27%), the next most common technologies and approaches are frontend gateways (21%), observability tools (21%), event-driven architecture (21%), and streaming and messaging services (20%). While these are less popular than API gateways and microservices, they constitute more advanced architectural approaches. This statistic indicates that a sizable portion of the backend community is working with more mature infrastructure approaches.

While Kubernetes and containers provide immutable infrastructure by default, we ask developers if they are engaged in practices and behaviours that reflect a philosophical approach to development around immutable infrastructure practices. We see a low adoption rate in those identifying as engaged in immutable infrastructure practices (7%), as well as chaos engineering (6%). This is likely a reflection of multiple factors: many organizations have not yet reached the scale or complexity where these practices provide clear ROI; implementation requires significant organizational and cultural maturity beyond technical capability; and current tooling may still present adoption friction relative to their benefits for typical team sizes. As platform engineering continues maturing and tooling improves, these practices may become more accessible to mainstream development teams.

Adoption of backend technologies or infrastructure approaches

% of backend developers (Q1 2026 n=2,792)



Question wording: Have you used any of the following infrastructure approaches or technologies in your backend services in the last 12 months? If so, which ones?

THE PATH TO INCREASED CLOUD NATIVE MATURITY

03

3. The Path to Increased Cloud Native Maturity

Cloud native technologies continue expanding throughout the developer community, with the cloud native developer population reaching almost 20 million in early 2026. However, understanding cloud native's proliferation requires examining two dimensions: breadth (how many developers have adopted cloud native practices) and depth (how sophisticated their implementations are). While breadth is more straightforward to measure, counting developers using cloud native technologies, depth is more complex, as we are not only interested in usage of more advanced technologies but signals of sophisticated architectural patterns.

Not all developers need advanced cloud native practices; some work on projects where foundational tools suffice. However, for those managing complex, scaled systems, understanding the pathway to cloud native maturity can inform strategic technology decisions.

To explore this problem, in this edition of the report, we attempt to map the path that developers go down in terms of technology adoption, as they move from mainstream cloud native practices like Kubernetes and microservices to advanced practices like immutable infrastructure and chaos engineering. To do this, we examine the relative cross-usage of technologies and, more importantly, the “lift” between technologies. Lift measures whether two technologies are used together more frequently than would be expected by chance (see Methodology). In other words, a lift of more than 1 indicates a positive association and a value of less than 1 indicates a negative association.

This analysis produces two complementary outputs:

Lift matrix: A comprehensive table showing association strength between all technology pairs, revealing which combinations are commonly adopted together

Developer journey map: A visual representation of the maturity pathway, identifying the sequence in which technologies are typically adopted as organizations progress from foundational cloud native practices toward advanced operational patterns.

Understanding these adoption pathways serves multiple stakeholders: **technology creators** can identify which technologies act as gateways to their tools; **platform teams** can design onboarding strategies that align with natural progression routes; and **organizations** can benchmark their maturity and identify logical next steps rather than attempting arbitrary technology adoption.

[1] *The underlying network graph visualization is available in the appendix.*

Developer Journey

For general developers, what emerges is a path to increased technology maturity that passes through two key processes and technologies: feature flagging and observability tools. While API gateways are the most commonly used technology, they show lift values of approximately 1.0 with many technologies, indicating that they are broadly used across maturity levels rather than being associated with specific architectural patterns. A notable exception is their negative association with immutable infrastructure practices (lift: 0.79) and chaos engineering (lift: 0.53). Organizations in the elite practice cluster may favor alternative traffic management approaches that are more sophisticated, suggesting that elite practitioners are architecturally distinct from more mainstream cloud native adopters.

Instead, the first association we see among developers is the co-usage of Kubernetes and microservices. This pairing reflects Kubernetes' original design purpose: orchestrating containerized microservices architectures. Organizations adopting microservices naturally gravitate toward Kubernetes for container orchestration, service discovery, and load balancing, capabilities that become essential as microservice counts grow beyond what manual management can handle.

Among Kubernetes users, we observe elevated adoption of both observability tools (lift: 1.41) and streaming and messaging services (lift: 1.33). This pattern reflects operational necessity: s Kubernetes deployments grow in complexity—managing multiple services, namespaces, and clusters—observability becomes essential for monitoring distributed systems, while streaming and messaging services provide the asynchronous communication infrastructure that microservices architectures require for loose coupling and resilience. However, streaming and messaging services also see a lift for event-driven architectures (lift: 1.45), which is expected. Streaming platforms provide the infrastructure foundation that makes event-driven patterns practical at scale. Event-driven architecture represents the architectural approach, while streaming and messaging services provide the technical implementation.

The next connection is the most important: feature flagging's lift with observability tools (1.53), streaming and messaging services (1.31), and event-driven architectures (1.45). However, in addition, it has a usage lift with immutable infrastructure practices (1.33), multicluster management (1.40), and chaos engineering (1.44). Feature flagging, therefore, acts as a crucial gateway technology approach that may be key for developers who wish to expand the maturity of their cloud native services. Feature flagging bridges mainstream and advanced practices because it introduces two critical operational capabilities: the ability to deploy code changes without immediately exposing them to users (decoupling deployment from release) and the ability to test changes incrementally with subset traffic. These capabilities teach organizations progressive delivery patterns and controlled experimentation—mental models that directly translate to advanced practices like chaos engineering (controlled failure injection) and immutable infrastructure (treating deployments as disposable experiments). Feature flagging serves as a low-risk entry point to sophisticated operational techniques that would otherwise seem daunting.

A secondary pathway to advanced practices runs through observability tools, which show an association with immutable infrastructure (lift: 1.33). This route likely reflects organizations that invest in comprehensive observability and, as such, develop the visibility required to confidently adopt immutable infrastructure. Observability provides the safety net that makes “throw-away and redeploy” strategies viable.

Once developers adopt advanced practices, we observe remarkably high lift values between technologies: immutable infrastructure and chaos engineering (2.22), service meshes and chaos engineering (1.76), and immutable infrastructure and service meshes (1.75). These values, substantially higher than associations among foundational technologies, indicate an “elite cluster” where organizations adopt multiple advanced practices simultaneously rather than incrementally. This extraordinarily high association (2.22—the strongest in the dataset) likely reflects both technical dependency (chaos engineering is safer with immutable infrastructure, as failed experiments can be quickly replaced) and philosophical alignment (both practices treat infrastructure as disposable and embrace failure as a learning tool). Organizations adopting one may naturally gravitate toward the other.

For general developers, the data reveals a three-tier maturity model:

Foundational: Kubernetes + microservices (entry point)

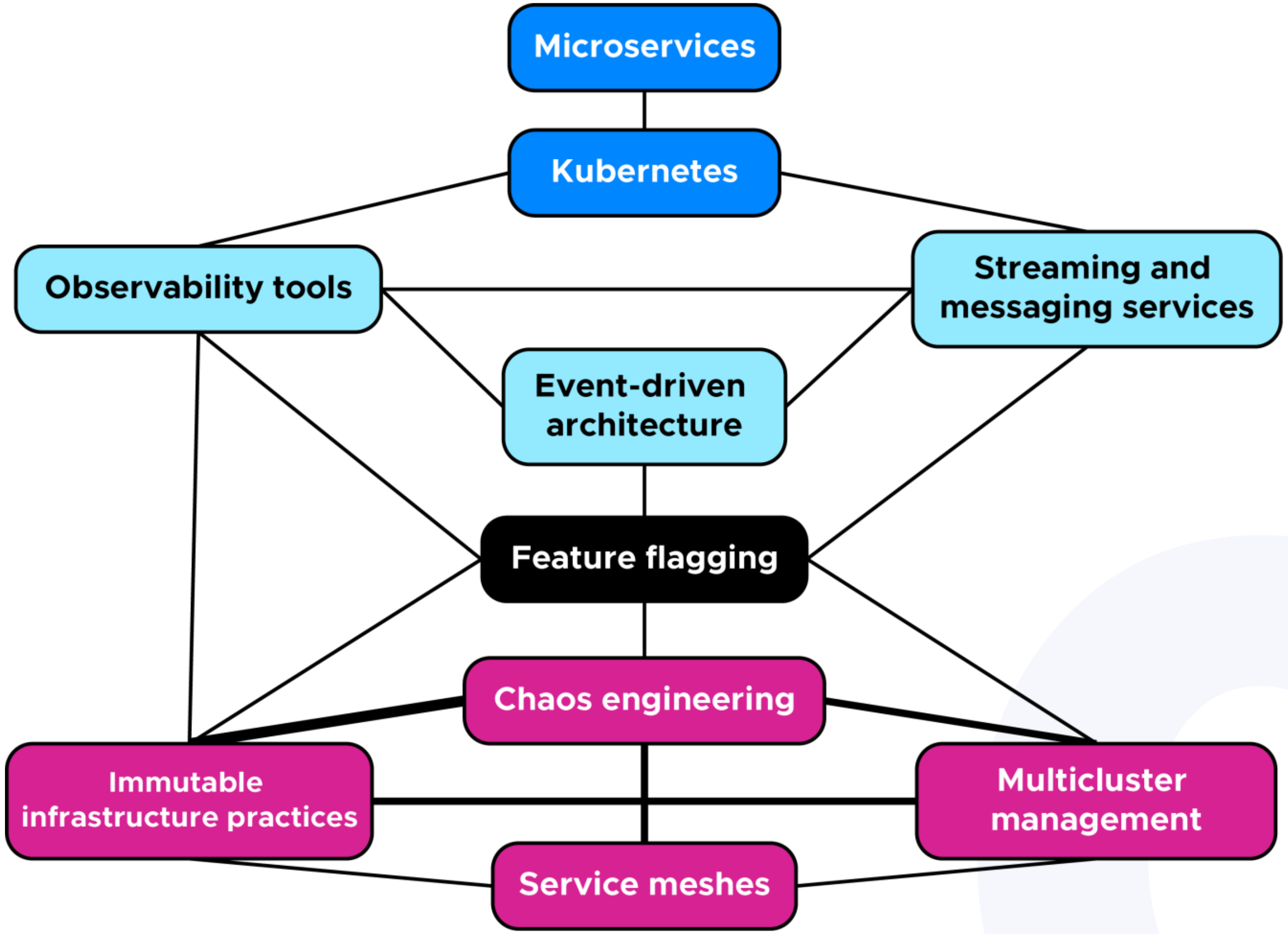
Mainstream cloud native: Observability tools, streaming/messaging, event-driven architecture (operational sophistication)

Advanced practices: Immutable infrastructure, chaos engineering, service meshes (elite operational maturity)

Feature flagging acts as the critical bridge between tiers two and three, with dual pathways available: Organizations can approach advanced practices through either experimentation maturity (feature flagging) or operational visibility (observability tools). Once organizations reach tier three, they typically adopt multiple advanced practices simultaneously rather than incrementally, suggesting these practices form a coherent operational philosophy rather than discrete capabilities.

Developer Cloud Native Journey

Lift association between backend technologies



Microservices

Kubernetes

Observability tools

Streaming and messaging services

Event-driven architecture

Feature flagging

Chaos engineering

Immutable infrastructure practices

Multicluster management

Service meshes

Maturity tiers:

- ● Foundational
- ● Mainstream cloud native
- ● Advanced practices

Lift associations between technologies of greater than 1.3 are shown as lines connecting nodes. The thickness of the line indicates the size of the lift association between technologies or practices.

Question wording: Have you used any of the following infrastructure approaches or technologies in your backend services in the last 12 months? If so, which ones?

Lift association between backend technologies

Lift association between backend technologies (Q1 2026 n=2,792)

	Kubernetes	Microservices	Service meshes	API gateways	Event-driven architecture	Observability tools	Immutable infrastructure practices	Chaos engineering	Feature flagging	Remote procedure call	Front end gateway	Multicluster management
Microservices	1.39											
Service meshes	1.04	0.87										
API gateways	1.08	1.13	0.77									
Event-driven architecture	1.27	1.17	1.24	1.08								
Observability tools	1.41	1.11	1.28	0.98	1.27							
Immutable infrastructure practices	1.12	0.83	1.75	0.79	1.15	1.33						
Chaos engineering	0.66	0.68	1.76	0.53	0.77	0.96	2.22					
Feature flagging	1.19	1.01	1.27	0.97	1.45	1.53	1.33	1.44				
Remote procedure call	0.90	0.87	0.96	1.00	0.95	0.89	0.98	0.88	1.16			
Front end gateway	1.04	0.92	1.09	1.17	0.89	0.97	1.14	0.82	1.19	1.03		
Multicluster management	0.96	0.84	1.42	0.86	1.08	0.91	1.68	1.64	1.40	1.28	1.04	
Streaming and messaging services	1.33	1.15	1.26	0.97	1.45	1.40	1.11	0.95	1.31	1.15	1.06	1.08

Question wording: Have you used any of the following infrastructure approaches or technologies in your backend services in the last 12 months? If so, which ones?

Cloud Native Machine Learning Journey

While the key focus of the developer journey is to understand how the wider developer ecosystem can pursue increased cloud native maturity, we also take time to focus on machine learning developers. With the current intense focus on machine learning and artificial intelligence, and in particular how cloud native can support organizations pursuing these workloads, it is important to highlight this path as organizations focus on expanding their cloud native capabilities.

Overall, the paths and maturities are comparable, but there are specific and notable differences that show a very different ideological framing and technological motivation. For AI developers, the maturity pathway represents AI service lifecycle stages rather than pure cloud native sophistication. While general backend developers progress through foundational to mainstream and elite cloud native tiers, AI teams progress through containerization to data pipeline operations, and then training/experimentation operations, before upskilling for production-serving operations. Feature flagging and immutable infrastructure, adopted during the training/experimentation stage, become dual prerequisites for production serving maturity: a distinctly different pattern from general development, where immutable infrastructure itself represents elite-tier adoption.

The two most striking differences are the movement of immutable infrastructure to a bridging technology and the elevated significance of remote procedure calls among AI developers. Immutable infrastructure's repositioning to a more key requirement for AI developers as they mature their cloud native stack reflects machine learning's reproducibility imperative. In general development, immutable infrastructure associates most strongly with chaos engineering (lift: 2.22), indicating elite reliability practices. For AI, immutable infrastructure instead shows a stronger association with observability tools (1.58 vs. 1.33 general) and a weaker association with chaos engineering (1.28 vs. 2.22).

Among AI developers, immutable infrastructure likely serves a dual purpose. It ensures reproducible training environments and consistent deployment artifacts for production serving. AI teams may adopt immutable infrastructure earlier than general developers in their exploration of cloud native practices, which may explain the weaker chaos engineering association. Many AI teams remain in experimentation stages where reproducibility matters, but chaos testing does not.

Remote procedure call (RPC) patterns, which are largely invisible in general backend development, emerge as uniquely critical for AI workflows. RPC shows the second-strongest association: RPC and feature flagging: 2.07 (vs. 1.16 general). This may be a reflection of RPC's centrality to model inference APIs, distributed training, and feature store access.

The extraordinary association with feature flagging reveals AI teams may be more reliant on feature flags for actions like model version routing. Unlike general backend development, where RPC may be one communication pattern among many, serving AI models places more emphasis on requiring synchronous request-response, making RPC adoption a key part of the technology stack of production AI teams.

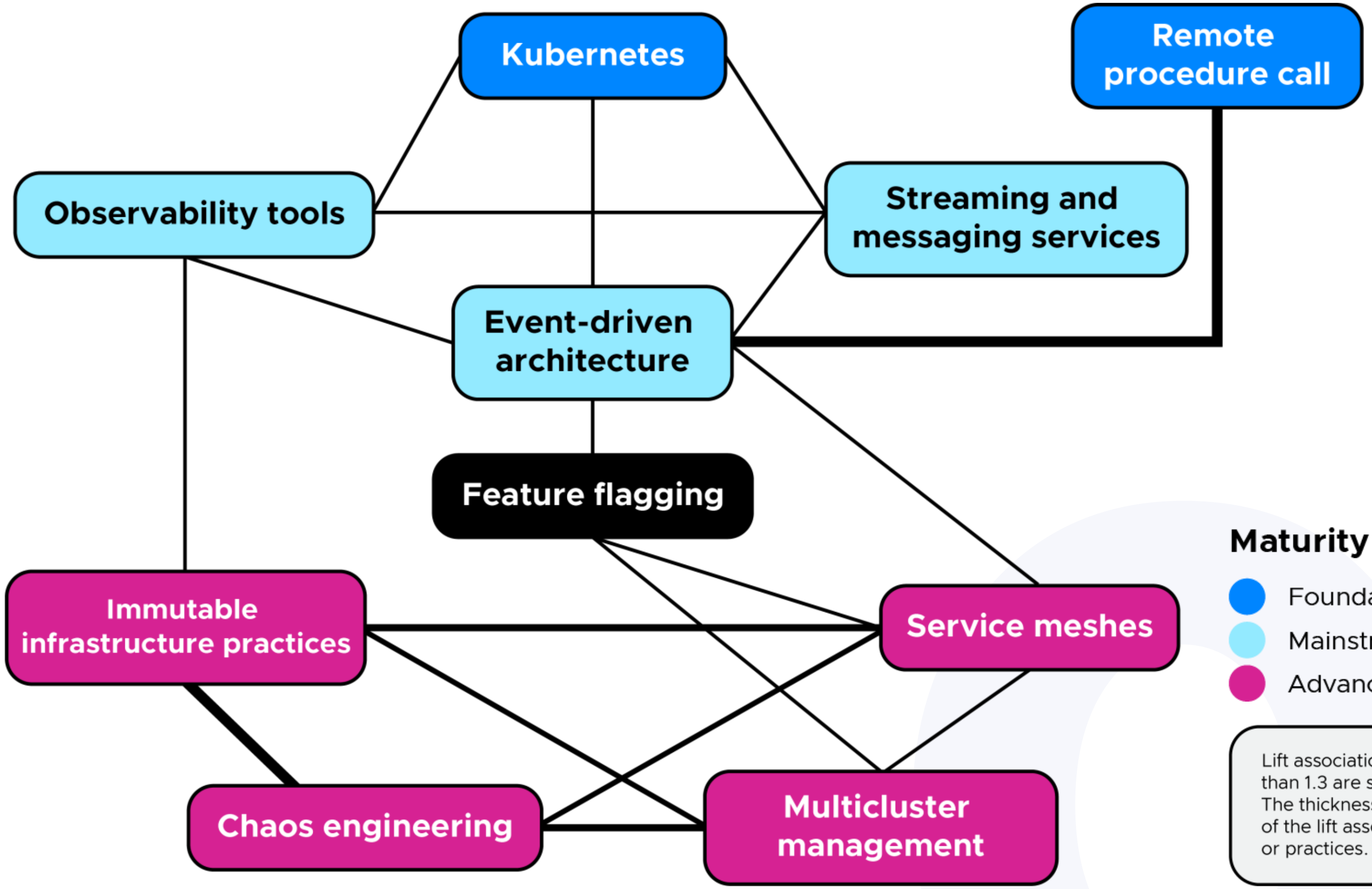
Placed together, we can instead place the maturity journey of an AI developer in a different light. Once entering the cloud native space, i.e. using Kubernetes and microservices, they seek ways to improve their data pipeline operations, which we see with event-driven architecture (1.31 lift with Kubernetes), streaming services (1.50 with event-driven), and observability tools (1.25 and 1.34). These emerge from AI-specific needs: event-driven triggers retraining pipelines, streaming handles continuous feature engineering, and observability monitors data quality and model performance. Notably, observability shows a negative association with chaos engineering (0.50), supporting AI observability focusing on data/model metrics rather than infrastructure resilience.

The next stage focuses on improving training and experimentation operations, and feature flagging, RPC, and immutable infrastructure can represent a natural stopping point for many AI teams. These technologies and approaches support mature AI workflows (reproducible experiments, version-controlled models, basic serving) without requiring a higher level of production sophistication.

The extended journey ends with a tightly coupled elite cluster formed around service meshes, chaos engineering, and multicluster management. Service meshes provide traffic splitting; chaos engineering tests model degradation; multicluster enables distributed inference. Reaching this advanced production cluster requires both feature flagging and immutable infrastructure from the previous stage, suggesting the relative lift between them.

AI Developer Cloud Native Journey

Lift association between backend technologies among AI developers (Q1 2026 n=1,270)



Kubernetes

Remote procedure call

Observability tools

Streaming and messaging services

Event-driven architecture

Feature flagging

Immutable infrastructure practices

Service meshes

Chaos engineering

Multicluster management

Maturity tiers:

● Foundational

● Mainstream cloud native

● Advanced practices

Lift associations between technologies of greater than 1.3 are shown as lines connecting nodes. The thickness of the line indicates the size of the lift association between technologies or practices.

Question wording: Have you used any of the following infrastructure approaches or technologies in your backend services in the last 12 months? If so, which ones?

Lift association between backend technologies among AI developers

Lift association between backend technologies among developers involved in AI (Q1 2026 n=1,270)

	Kubernetes	Microservices	Service meshes	API gateways	Event-driven architecture	Observability tools	Immutable infrastructure practices	Chaos engineering	Feature flagging	Remote procedure call	Front end gateway	Multicluster management
Microservices	1.22											
Service meshes	0.84	0.79										
API gateways	1.09	1.11	0.92									
Event-driven architecture	1.31	1.25	1.38	1.15								
Observability tools	1.25	1.06	1.05	0.99	1.25							
Immutable infrastructure practices	1.15	0.89	2.07	0.92	0.81	1.58						
Chaos engineering	0.78	0.81	2.36	0.50	0.84	0.50	1.28					
Feature flagging	1.26	0.99	1.35	1.02	1.45	1.41	1.10	1.10				
Remote procedure call	1.11	1.13	0.69	1.17	1.01	0.94	1.19	0.59	2.07			
Front end gateway	0.87	1.02	0.83	1.16	0.80	0.79	1.02	0.82	0.87	0.93		
Multicluster management	0.90	0.89	1.62	0.82	1.04	0.81	1.05	2.03	1.48	0.85	0.79	
Streaming and messaging services	1.21	1.23	1.18	0.97	1.50	1.34	1.13	0.97	1.53	1.19	1.07	0.99

Question wording: Have you used any of the following infrastructure approaches or technologies in your backend services in the last 12 months? If so, which ones?

Supporting AI Teams

Based on this model of the maturity journey, for supporting AI developers in the cloud native space, the following strategic priorities emerge for supporting AI cloud native adoption:

- Not all AI teams need production-scale sophistication. Many legitimately stop at experimentation/training and benefit most from improved data pipeline tooling and reproducibility practices rather than advanced serving infrastructure.

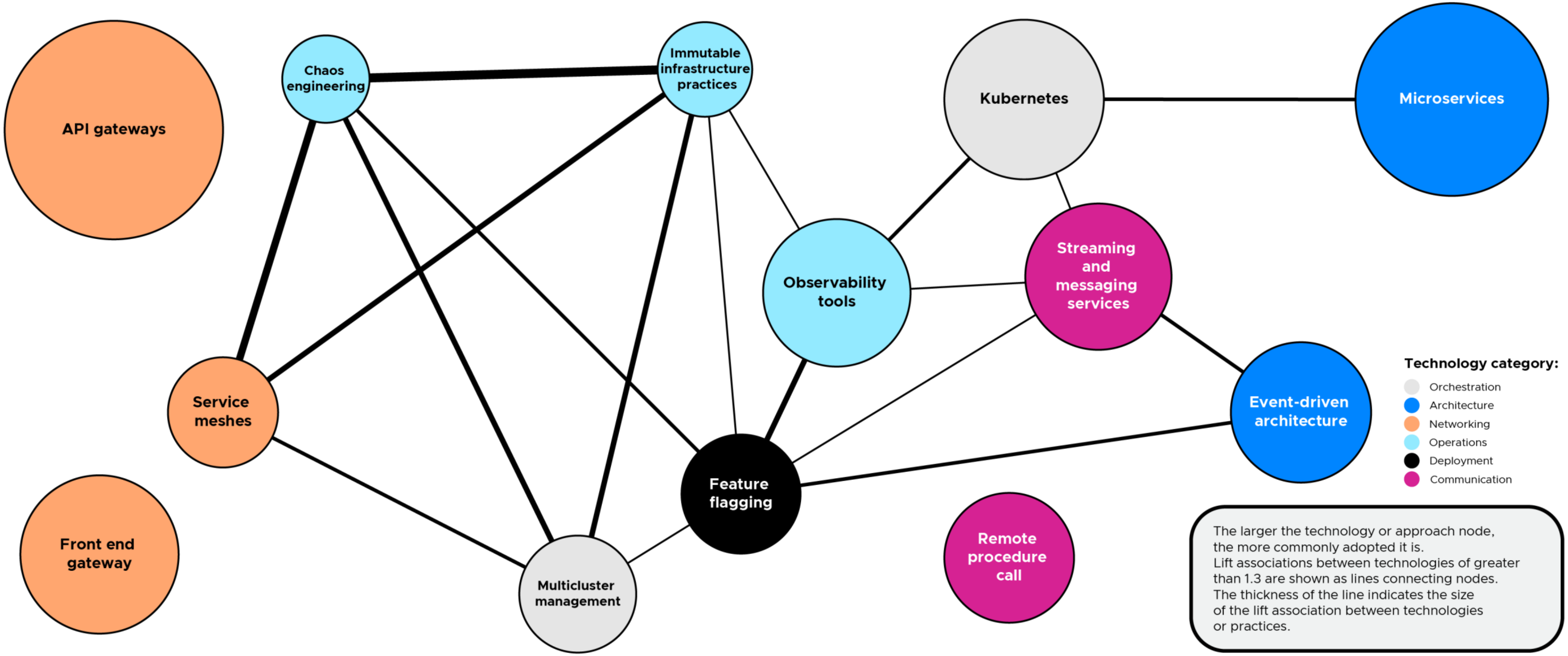
- Feature flagging and immutable infrastructure are dual prerequisites for production AI. Prioritizing education on these intermediate-stage technologies addresses the key barrier preventing AI teams from reaching production maturity.

- Service meshes, chaos engineering, and multicluster management require AI-specific framing: traffic splitting for model A/B testing, resilience testing for model degradation, and geo-distributed inference—not generic infrastructure benefits.

- RPC's centrality to AI serving deserves emphasis in AI cloud native education, as model inference APIs are foundational to production maturity.

APPENDIX

Network graph of technology lift among developers using backend technologies

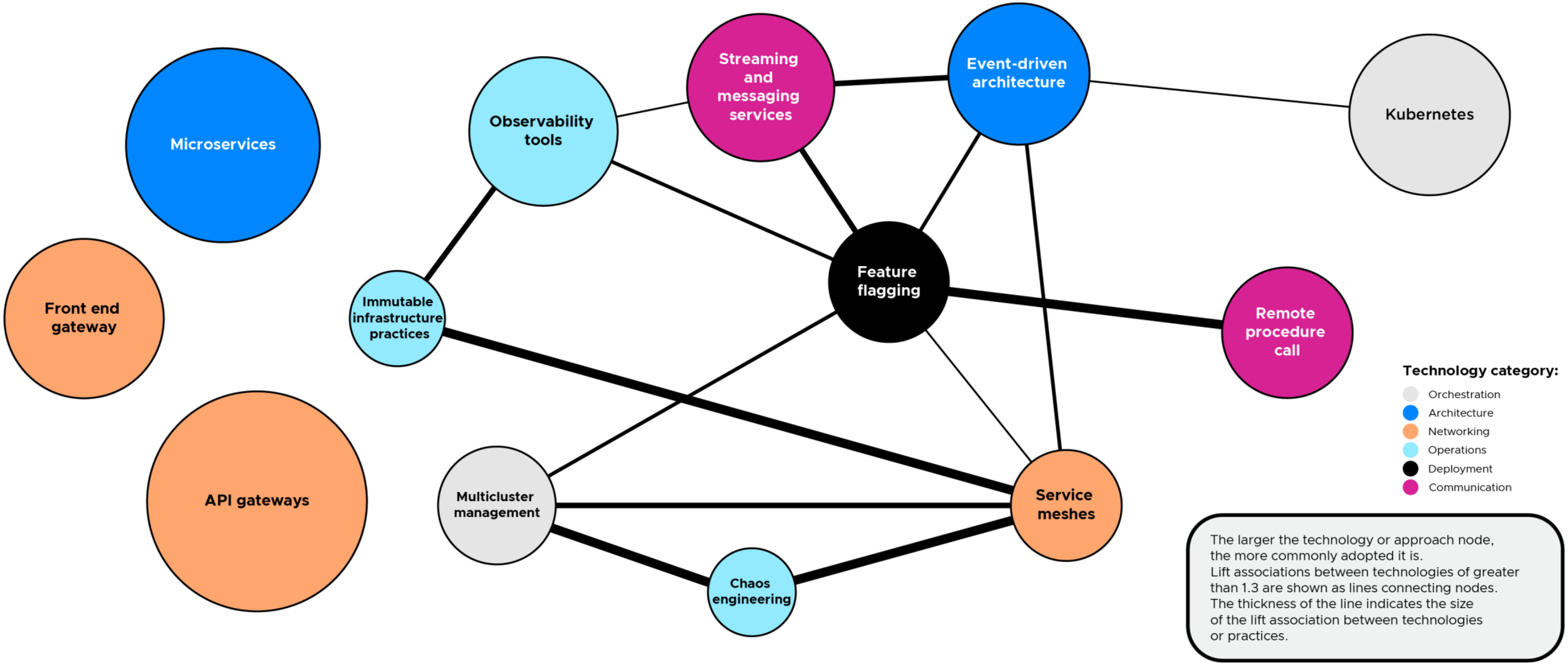


Technology category:

- ● Orchestration
- ● Architecture
- ● Networking
- ● Operations
- ● Deployment
- ● Communication

The larger the technology or approach node, the more commonly adopted it is.
Lift associations between technologies of greater than 1.3 are shown as lines connecting nodes.
The thickness of the line indicates the size of the lift association between technologies or practices.

Network graph of technology lift among AI developers using backend technologies



The larger the technology or approach node, the more commonly adopted it is.
Lift associations between technologies of greater than 1.3 are shown as lines connecting nodes.
The thickness of the line indicates the size of the lift association between technologies or practices.

METHODOLOGY

METHODOLOGY

Lift Calculation

Overview

Lift, also known as the lift ratio or interest factor, is an association rule metric that measures the strength of the relationship between two variables while accounting for their individual prevalence. In the context of technology adoption, lift reveals which technologies are genuinely adopted together as part of coherent architectural patterns, rather than appearing together simply because both are popular.

Mathematical Definition

The lift between technologies A and B is calculated as:

Lift(A,B) = P(A ∩ B) / [P(A) × P(B)]

where:

P(A ∩ B) is the joint probability—the proportion of developers using both technology A and technology B

P(A) is the marginal probability—the proportion of developers using technology A

P(B) is the marginal probability—the proportion of developers using technology B

To account for survey sampling design and ensure results are representative of the broader developer population, we apply survey weights to all probability calculations:

Interpretation

The lift metric has the following interpretation:

Lift = 1.0: Independence. Technologies A and B are used together at exactly the rate expected if adoption decisions were independent. This indicates no relationship between the technologies.

Lift > 1.0: Positive association. Technologies are used together more frequently than expected. The magnitude indicates strength:

- 1.0 < Lift < 1.3: Weak positive association
- 1.3 ≤ Lift < 1.7: Moderate positive association
- Lift ≥ 1.7: Strong positive association

Lift < 1.0: Negative association. Technologies are used together less frequently than would be expected by chance.

METHODOLOGY

The Developer Nation Survey

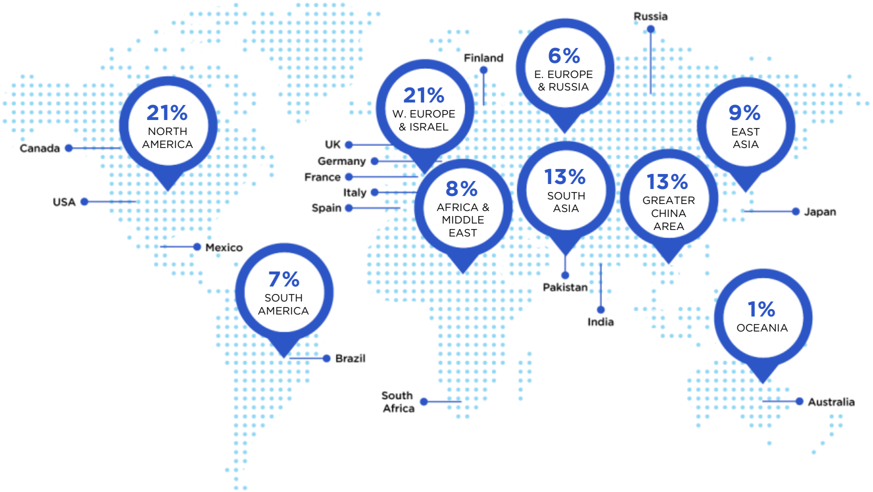
GLOBAL REACH

Geo distribution of respondents of the 31th global Developer Nation Survey (December 2025 to January 2026)

31th
GLOBAL WAVE

12,800+
DEVELOPERS

100+
COUNTRIES REACHED



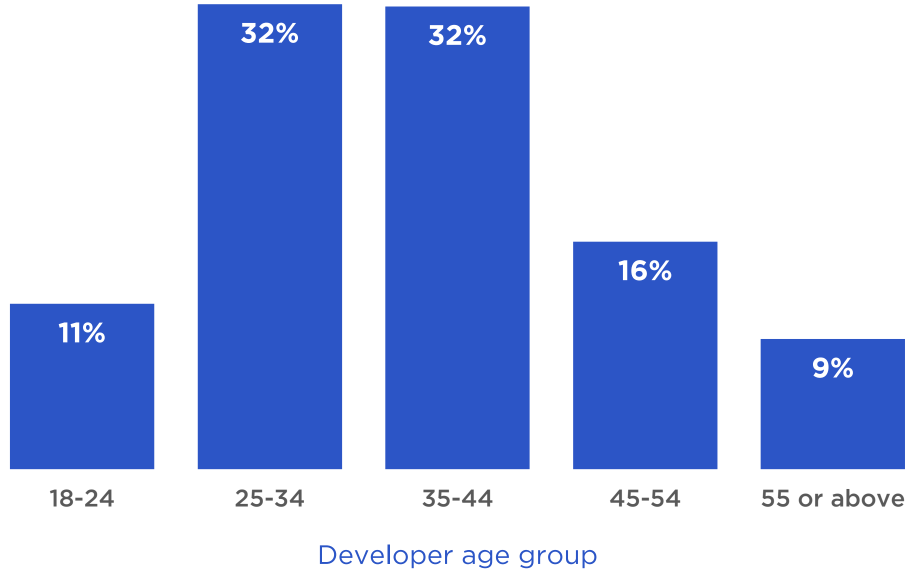
Developer Nation's 31st edition reached 12,800+ respondents around the world. As such, the Developer Nation series continues to be the most comprehensive independent research on mobile, desktop, Industrial IoT, consumer electronics, 3rd party app ecosystems, backend, web, game, AR/VR and artificial intelligence/machine learning developers and data scientists combined ever conducted. The report is based on the large-scale online developer survey designed, produced and carried out by SlashData over a period of seven weeks between December 2025 and January 2026.

Respondents to the online survey came from 100+ countries, including the US, China, India, Japan, Brazil, and the UK. The geographic reach of this survey is reflective of the global scale of the developer economy. The online survey was translated into nine languages in addition to English (Simplified Chinese, Traditional Chinese, French, Spanish, Portuguese, Vietnamese, Russian, Japanese, Korean) and promoted by 14 leading partners and the Developer Nation panel within the software development industry.

Our respondents came from a broad age spectrum, from young coders and creators who are under 18 to the seasoned ones over 55. Excluding those who would rather not answer about their age, the age profile of our respondents is shown below.

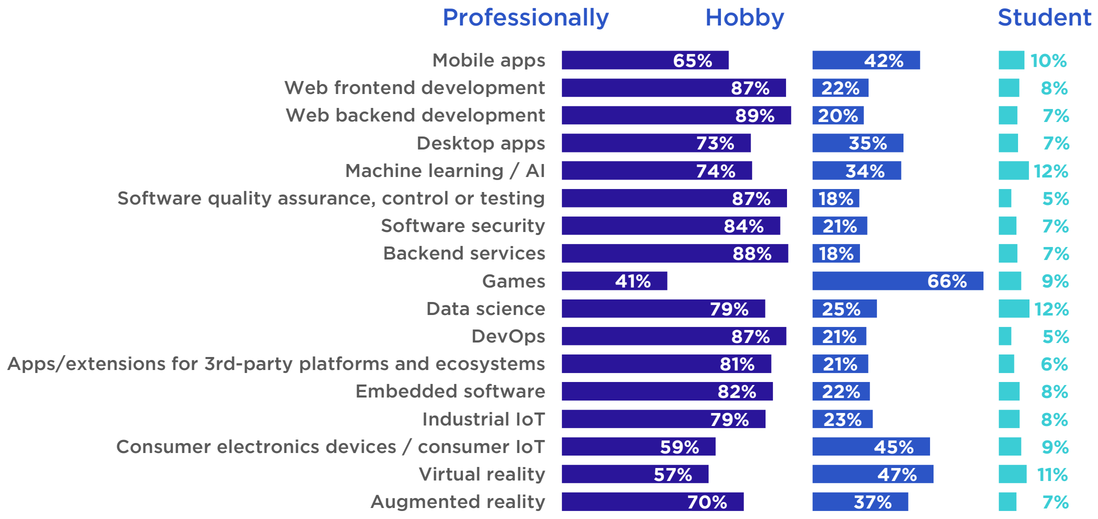
Age distribution of survey respondents

% of respondents (Q1 2026 n=12,818)



How respondents are involved in each software area

% of respondents in each area (Q3 2025 n=12,021)



Respondents were asked which types of projects they are involved in out of the 13 under study, namely web apps / SaaS, mobile apps, desktop apps, backend services, augmented reality, virtual reality, games, data science, machine learning / artificial intelligence, industrial IoT, consumer electronics devices, embedded software, and apps/extensions for third-party app ecosystems. They also told us if they are into their areas of involvement as professionals, hobbyists, or students - or as any combination of these - and how many years of experience they have in each.

METHODOLOGY

To eliminate the effect of regional sampling biases, we weighted the regional distribution across nine regions by a factor that was determined by the regional distribution and growth trends identified in our Developer Nation research. To minimise other important sampling biases across our outreach channels, we weighted the responses to derive a representative distribution for technologies used, and developer segments. Using ensemble modelling methods, we derived a weighted distribution based on data from independent, representative channels, excluding the channels of our research partners to eliminate sampling bias due to respondents recruited via these channels. Each of the separate branches: Industrial IoT, consumer electronics, 3rd party app ecosystems, backend, embedded, augmented and virtual reality were weighted independently and then combined.

For more information on our methodology please visit https://www.slashdata.co/methodology.

Liam Bollmann - Dodd

Principal Market Research Consultant

Liam is a former experimental antimatter physicist, and he obtained a PhD in Physics while working at CERN. He is interested in the changing landscape of cloud development, cybersecurity, and the relationship between technological developments and their impact on society.

✉ liam.dodd@slashdata.co

Álvaro Ruiz Cubero

Research Manager

Álvaro is a market research analyst with a background in strategy and operations consulting. He holds a Master’s in Business Management and believes in the power of data-driven decision-making. Álvaro is passionate about helping businesses tackle complex strategic business challenges and make strategic decisions that are backed by thorough research and analysis.

✉ alvaro.ruiz@slashdata.co

ABOUT THE AUTHORS

Navigate AI technology decisions with confidence & clarity

SlashData is an AI analyst firm which has been working with the top Tech brands to provide clarity and confidence in their decision-making.

For 20 years, we have been tracking software technology trends and helping technology brands make product and marketing investment decisions, challenging assumptions and reframing market trends to empower industry leaders to drive the world towards the future.

Find us at slashdata.co

/DATA

CLOUD NATIVE
COMPUTING FOUNDATION