Improving Developer Experience using Advanced Platform Engineering Techniques

DevOpsDays Cairo 2024
Sep 25, 2024

Ajay Chankramath
Chief Technology Officer & Managing Director
Platforms, Products & Solutions
Brillio, USA

brillio

Agenda

- Understand your DevEx goals
- Basics of applying platform engineering (PE)
- Establishing metrics needed
- 5 Advanced PE techniques
- Case Studies

Understanding your Developer Experience

Experience for the Developers interacting with Tools, Frameworks, Process through SDLC

DevEx has all 3 of People, Process & Technology components which makes it extremely difficult to improve

What's the industry telling us about DevEx?

65% of executives believe improving DevEx is one of the top goals for 2025

88% of tech executives believe PE is critical in achieving their software engineering goals

57% of tech executives believe the most important thing in PE is Platform-As-A-Product

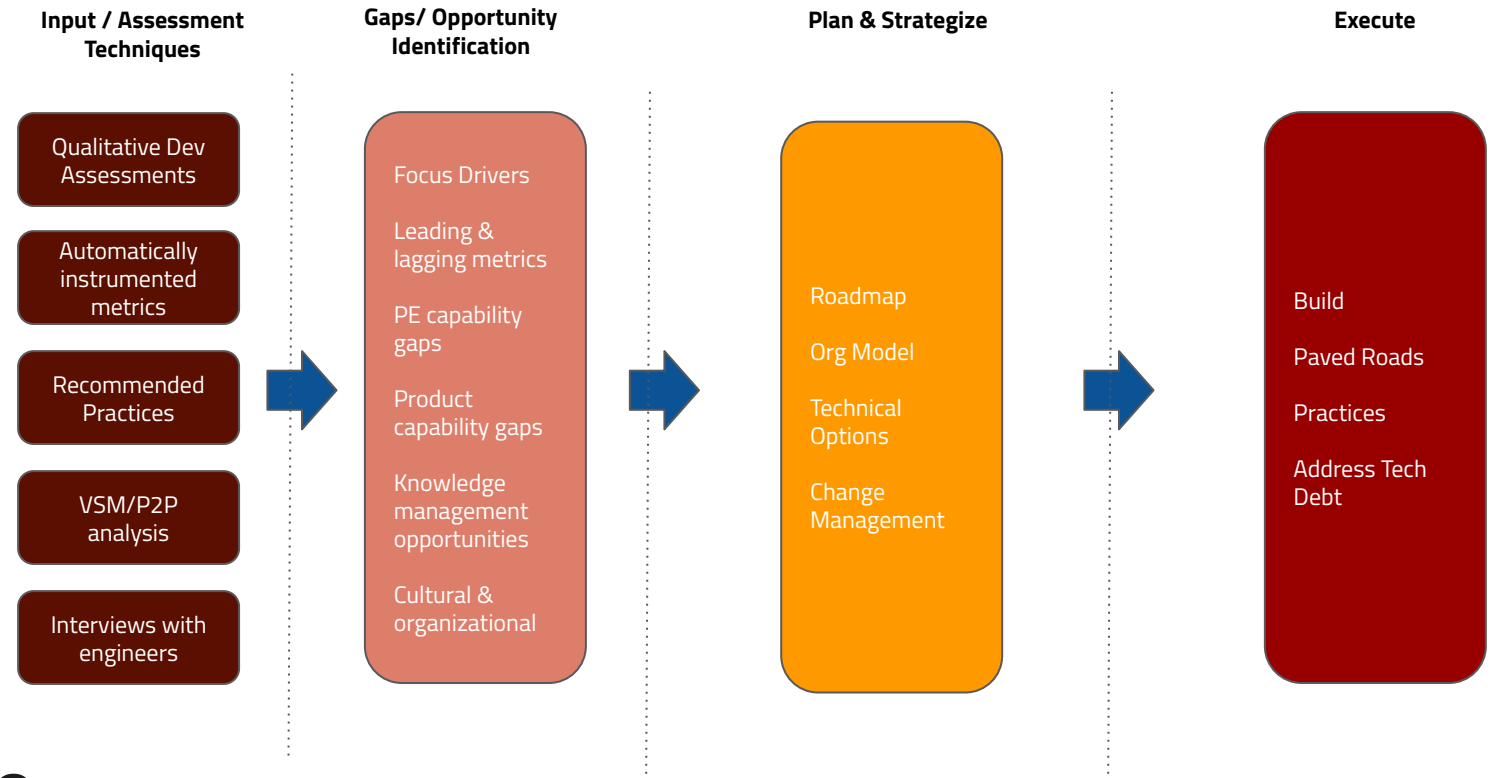
58% of digital organizations have either deployed or deploying a developer portal by late 2023

Dev Portals + Platform Engineering is what organizations do.
By 2026, **75%** of the organizations will have that combination, up from **45%** in 2024

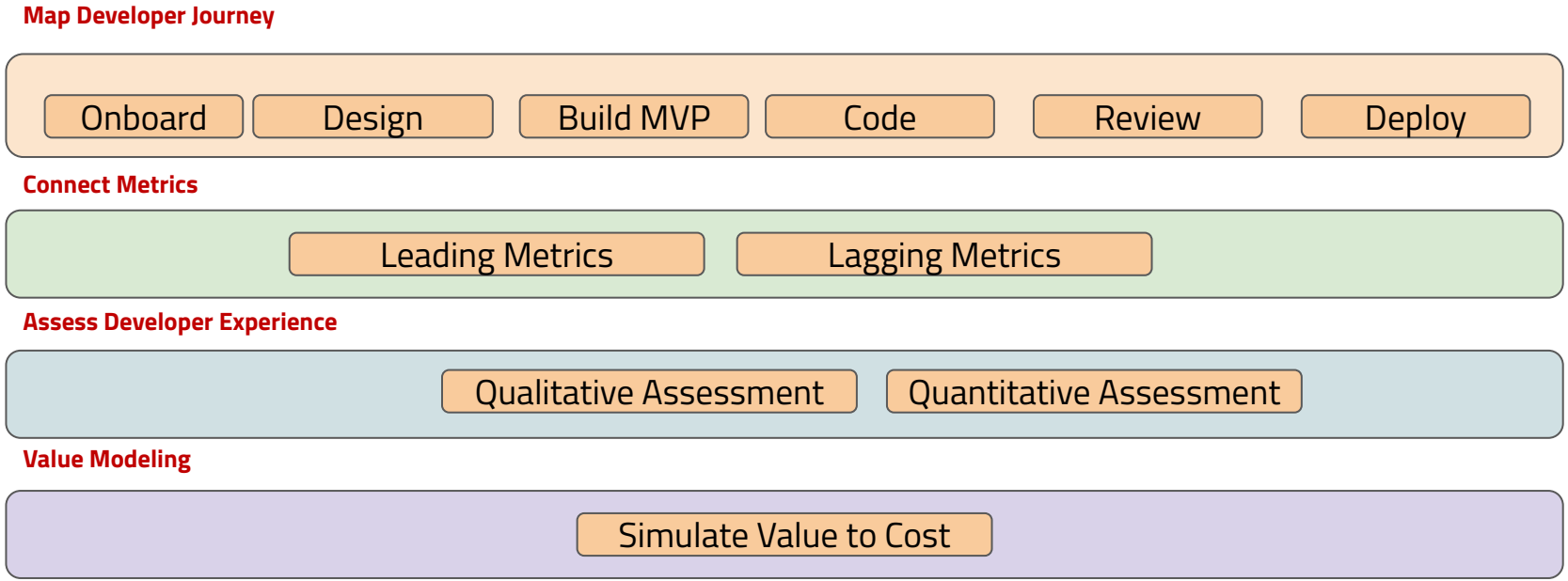
brillio

Gartner Various Gartner & Forrester Reports 2024 FORRESTER®

Data Driven DevEx Improvement Model



Driving action across ecosystem

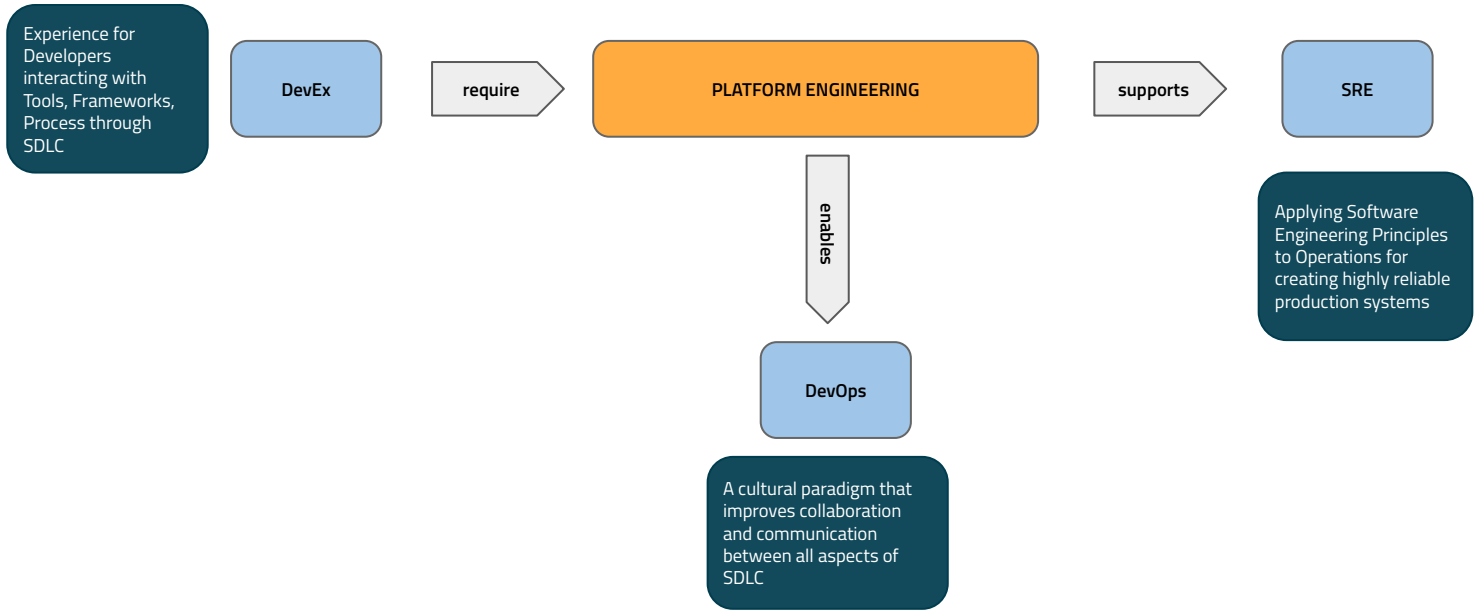


Platform Engineering

Platform engineering is the practice of designing, building, and maintaining the **underlying ecosystem** that enables the development and delivery of software applications and services.

brillio

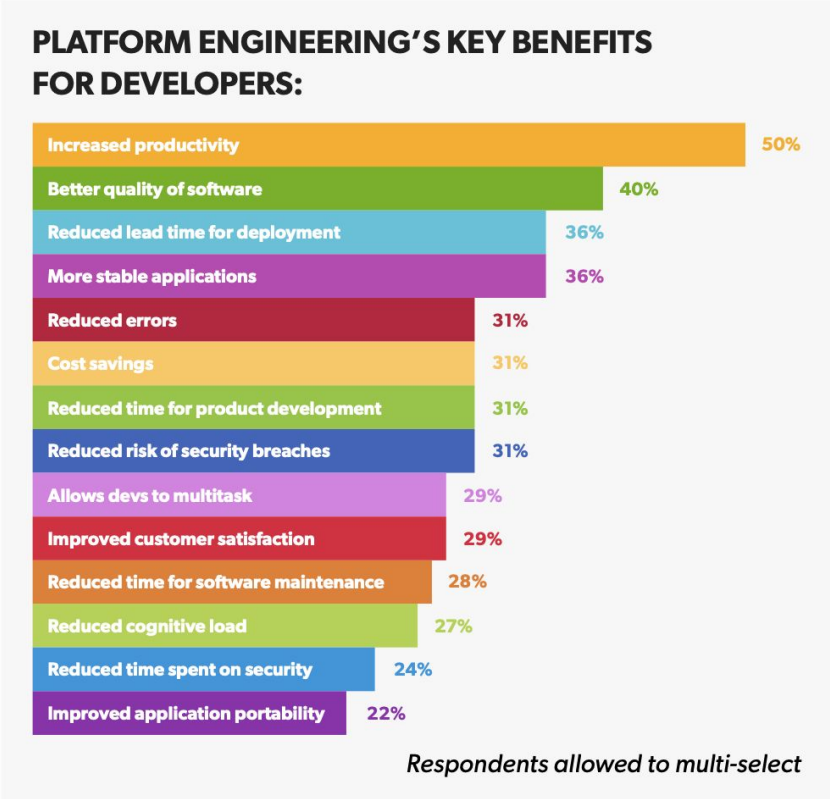
Platform Engineering Vision



Platform Engineering Notional View

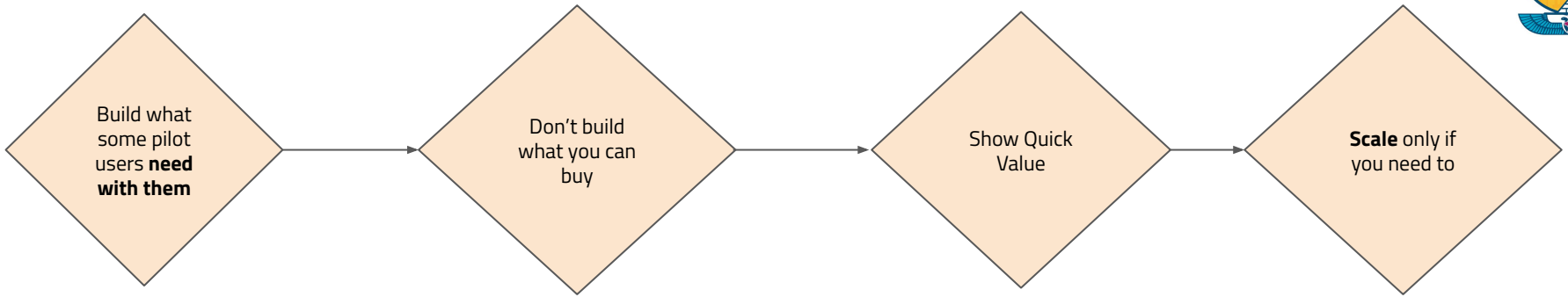
Platform Engineering - Overall Value Proposition

How Platform Engineering Helps DevEx?



Source: State of Platform Engineering - 2024 - https://www.puppet.com/resources/state-of-platform-engineering

Value Modeling in Platform Engineering

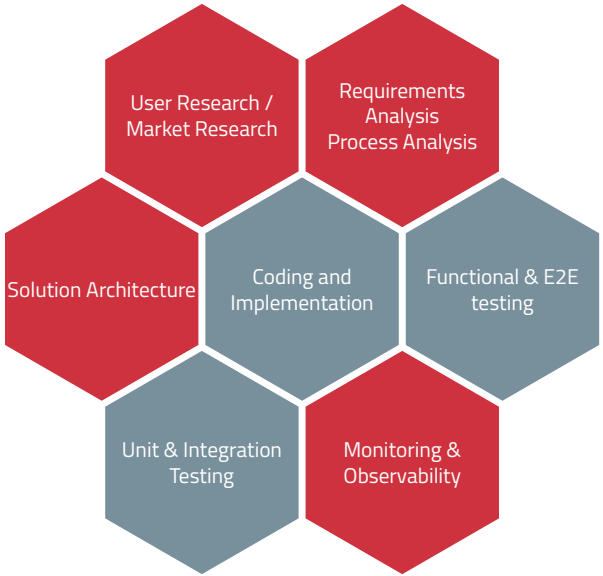


brillio

Why modern techniques?

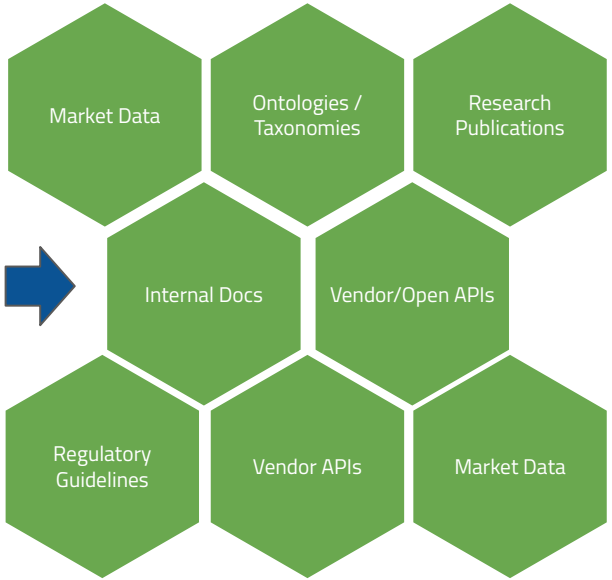
As traditional platform engineering techniques become table stakes, more ideas, also driven by GenAI, need to come to the fore to keep improving value, differentiating and moving up the abstraction layer.

#1 AI Driven Automation



What you need to do?

Potential Retrieval Augmented Generation (RAGs) to contextualize your AI models

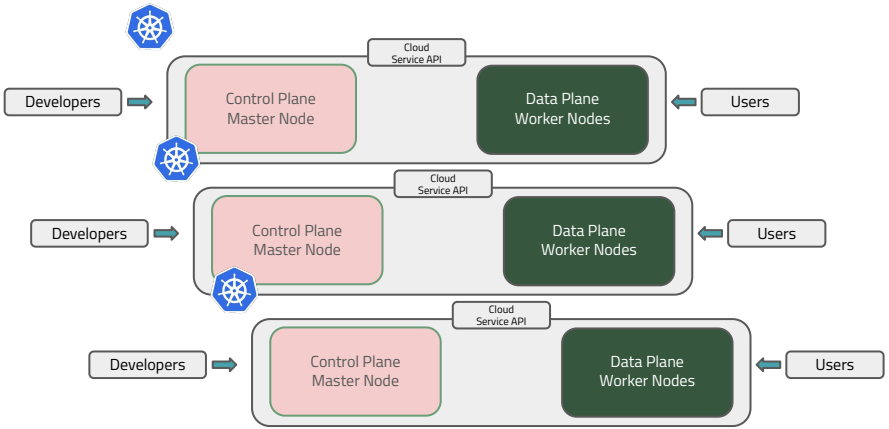


Quick Wins

High ROI

RAGs

#2 Multi Cluster Management



Pulumi

Crossplane

KUBERNETES OPERATORS

Unified Control Plane
Centralized Management, MC orchestration

Automated Cluster Lifecycle Management
Cluster Provisioning / Scaling
Self-Serve Platforms

Workload Orchestration
Optimized Resource Utilization
Multi cluster Networking

Cost Optimization
Quotas / Visibility
Capacity planning

Observability
Cross Cluster Logging
Centralized Monitoring

brillio

#3 Ephemeral Environments

#4 Causal AI Techniques

App Instrumented for Observability

Observability Platform proactively detects patterns

Temporary self-healing

Sourcegraph Cody

Deep Insights into the codebase

Code Refactor and generate code

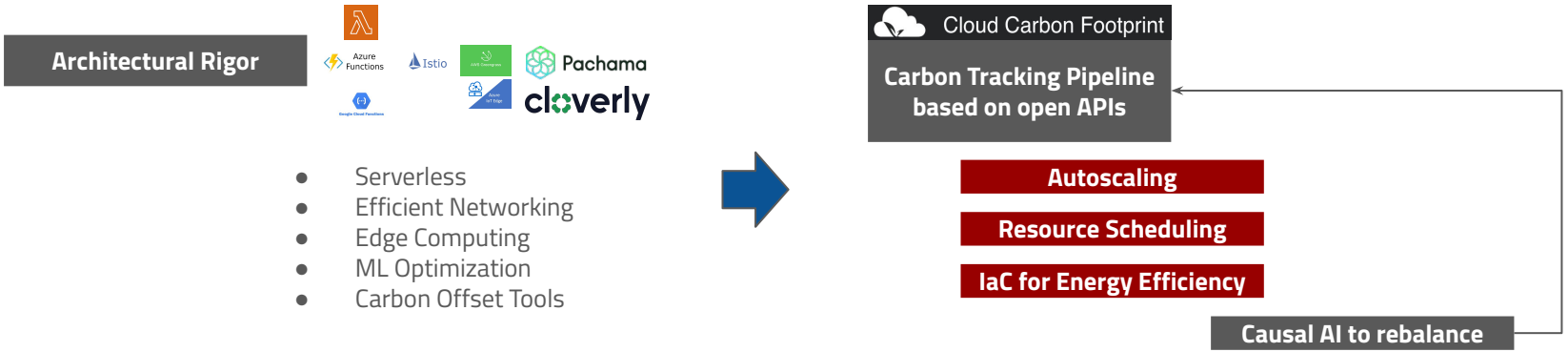
openrewrite/rewrite

Automated mass refactoring of source code.

Real-Time Test Generation and Gating

Developer PR

#5 Sustainability as a first class citizen



Orchestrated Through an IDP

Case Study # 1

Problem:

The bank's SRE functions were heavily focused on infrastructure, leading to inefficiencies and high developer overhead. Developers were frequently called upon to handle L3 incidents, which slowed down development cycles and impacted overall product quality and turnaround time.

Solution:

To address this challenge, the bank shifted from a traditional **Platform SRE** model to a **Product SRE** model, integrating **Platform Engineering techniques** to better align SRE with the needs of product development teams. This approach ensured SREs became more embedded within the product life cycle, focusing on system reliability while removing the need for developers to be involved in L3 issues.

Key changes included:

- Establishing dedicated Product SRE teams for specific business lines.
- Automating infrastructure and environment management using tools such as **Terraform** and **Ansible**.
- Leveraging **Sourcegraph Cody** for code navigation and **ReWrite** for reducing technical debt.
- Using **Dynatrace** for proactive monitoring and incident detection to minimize downtime.

Outcomes:

The shift to Product SRE delivered substantial improvements across key metrics:

1. **L3 requirements eliminated**: Developers were **no longer required to handle L3 incidents**, freeing them to focus solely on development.
2. **Quality improvement**: There was an **81% improvement in the quality of fixes**, reducing bugs and system failures.
3. **Turnaround time**: Incident resolution time improved by an **average of 400%,** drastically reducing response times and accelerating development cycles.
4. **Developer Experience (DX)**: The **DX happiness index improved by a factor of 8X**, indicating a dramatic enhancement in the developer's overall workflow and satisfaction.

Case Study # 2

Problem:

The CPG chain store was facing ESG/regulatory pressure to **reduce its carbon footprint** and align with sustainability goals. They set an ambitious target to cut carbon emissions, but existing processes lacked the technology and automation required to achieve this without increasing developer friction or impacting operational efficiency.

Solution:

The company incorporated **sustainability techniques** powered by **modern platform engineering technologies** to create a more carbon-conscious development and operational environment. The key innovation was leveraging **automated architectural review board (ARB)** systems that ensured any new architectural decisions were aligned with carbon-negative or carbon-neutral approaches. Additionally, automated workflows were introduced to reduce friction for developers while embedding sustainability as a core design principle.

Key changes included:

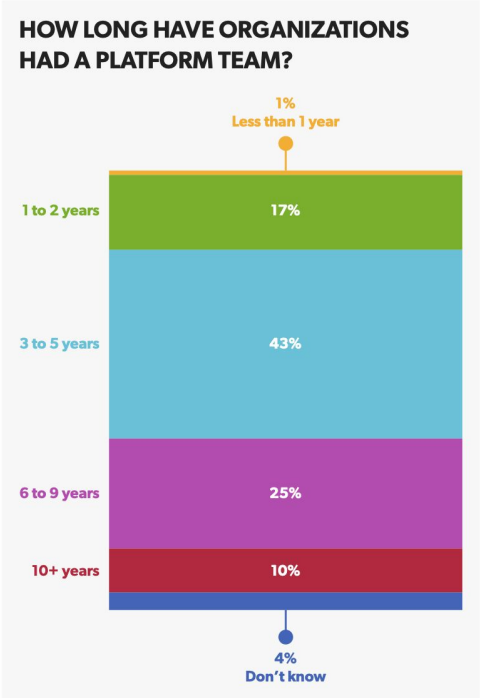
- **Automating sustainability assessments** in the development pipeline using **CCF** and **Pachama** to monitor and manage carbon impact.
- **Pulumi** for provisioning and managing infrastructure with automated sustainability audits.
- Utilizing **AWS Lambda** and **Istio** to optimize cloud usage, reduce waste, and improve resource allocation.
- **Cloverly** was implemented to help offset carbon emissions through seamless integrations into operational workflows.
- **Backstage** was used to centralize and simplify the developer experience, ensuring that sustainability checks did not add unnecessary complexity or slow down development cycles.

Outcomes:

1. **Carbon footprint reduced by 39%**: The organization successfully met its sustainability goals while continuing to innovate and expand.
2. **Automated architecture sustainability assessments**: By embedding carbon-conscious techniques directly into the ARB, decisions about infrastructure, applications, and deployments automatically aligned with sustainability targets, without the need for manual intervention. **ARB approvals were faster by 88%**
3. **Developer friction minimized**: Despite the added focus on sustainability, automation and smart tooling ensured that developers faced reduced friction, improving efficiency and **satisfaction by 21%**

Takeaways

- Developer Productivity is an easy topic to complain about and hard to fix as it involves People | Process | Technology, precisely in that order
- Bringing in the rigor of Platform Engineering changes the equation right off the bat
- Traditional PE techniques are table stakes, even though there are lots of organizations still catching up
- Advanced techniques is what you need for the next 5 years to stay competitive



DEVOPS DAYS 2024 CAIRO

brillio

Stay Connected? Questions?


ajay.chankramath@brillio.com


/chankramath


chankramath.com

brillio