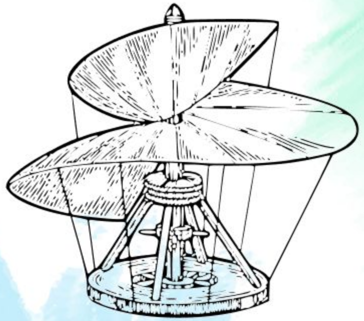
Platforming from the Trenches : Closing the Feedback Loop from Production to Trust

Sakshi Nasha



Platform Engineering × wescale
PLATFORM CON LIVE DAY PARIS

Who am I?

- Developer👩🏻‍💻
- OpenSearch Ambassador
- AWS community Builder ☁️
- Innovator : Hackathons
- Athlete at heart : 🏃‍♀️🏀⚽🏸

The Production Firefight

- On a customer call, we heard a clear ask:
  “We don’t want traffic from the internet or public sources. Only specific IPs should be allowed.”
- The request was about **IP Whitelisting :** a way to restrict access only to approved sources.
- Instead of letting it sit in a support ticket or backlogged notes, the **Support team + PM acted immediately**.
- The feature was **prioritized**, built, and integrated into the product life cycle. As a result, the **customer felt heard**.
- **When feedback flows from production to product, trust becomes a feature.**

The Problem: Gaps Between Users and Platforms

1. **Organizational Silos Create Friction :**
   a. Dev and CloudOps **operate in silos**, leading to misaligned priorities
   b. Developers struggle with tooling, platform inconsistencies, and unclear ownership
   c. Platform teams overwhelmed with support tickets instead of building scalable solutions
   d. CloudOps forced to step in **reactively with urgent patches and hotfixes**
   e. **Feedback gets lost, delayed, or mistranslated** across teams
2. **It's Not Just a Tech Problem**
   a. These are not technical gaps — they're cultural and systemic gaps
   b. Lack of shared goals, visibility, and trust between teams
3. **A Cultural Paradigm Shift Is Needed**
   a. Shift from siloed support to shared ownership
   b. Build systems that enable continuous feedback and collaboration

The Trench Lessons: Practical Wins

1. Small Automation, Big Win
2. Fixing the Friction, Not the System
3. What Feedback Loops Should Look Like
4. Shift Left -> Respond Right
5. Zoom Out : Feedback Loops as a Mindset

#1 – Small Automation, Big Win

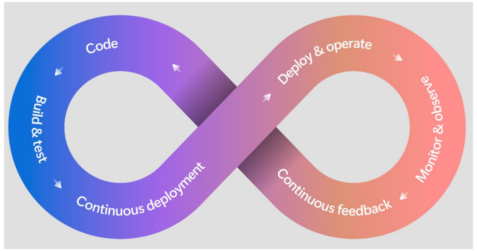
- Manual steps = friction in the SDLC
  Friction → slower velocity + higher frustration (Dev & Ops)
- Automated log trails and alerts → **smoother, faster deployments**
- Developers stay **secure & compliant** without slowing down
- **Templates as strategic levers:**
  - Enforce policies and security scanning
  - Support governance beyond bootstrapping
- Automation = **scalable, sustainable, and cost-effective** development

#2 - Fixing the Friction, Not the System

- "Not my job" mindset holds teams back
- Developers can influence platform experience ,even if they don't own it
- **Collaboration > Ownership**
- Align Devs & SREs **around shared goals**, not just ticket queues
- Introduce biweekly check-ins / retros to surface pain points early
- Small conversations → Big shifts in trust, usability, and velocity

What Feedback Loops Should Look Like

- Types of Feedback:
  - **Developer Feedback**: Direct input from devs on tool usability, pain points, or friction in workflows.
    - Example: Surveys, slack threads, platform office hours.
  - **Platform Telemetry**: Data from usage metrics, error rates, and performance logs.
    - Example: Monitoring IDP adoption, tracking CI/CD pipeline times.The feedback loop isn't optional anymore. It should be built in, measured and visible.



What Feedback Loops Should Look Like

Case study : Microsoft 365 - Listening to Customer and Internal Feedback

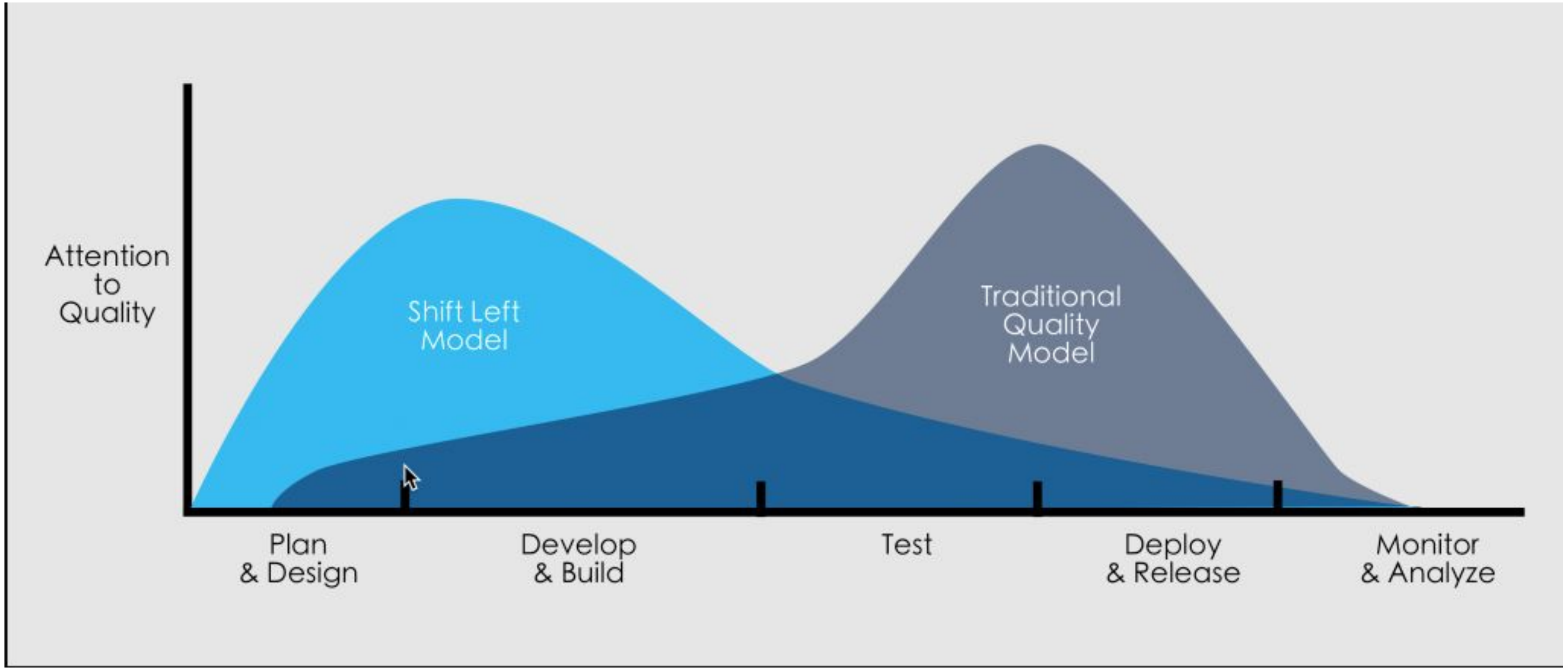
What they did:
The Microsoft 365 engineering org made a concerted push to incorporate user feedback (both internal users and external customers) at every stage of product development.

They used customer insights, telemetry, performance data, etc., to spot and prioritize what needs fixing.

Feedback loop in practice:

- Customer/user feedback triggers bug / pain point → engineers or product teams assess → fix deployed → monitor if satisfaction or usage improves.
- Also internal feedback (from employees) helps shape tools and workflows.

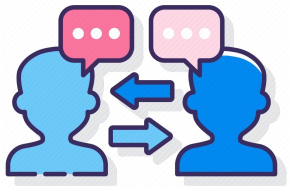
#3 Shift Left -> Respond Right



Respond rather than reactive feedbacks act as catalyst
Fewer surprises in production because reliability concerns are surfaced ahead of time.

#4 Zoom Out: Feedback Loops as a Mindset

1. See the **bigger picture** : it's not just about fixes, it's about flow
2. Feedback loops are a process and a **mindset shift**
3. **Everyone has a voice** in shaping the platform experience
4. You **don't have to own** the platform to influence it
5. Create **intentional touch points** between users and platform teams
6. Use feedback to drive **trust,** not just tickets
7. Systems improve when **collaboration is systemic,** not reactive



One fix at a time

Connect with me