PLATFORM ENGINEERING

Platform Engineering on the Brink: Breakthrough or Bust?

Is platform engineering at a crossroads? Will it fulfill its promise or fade away? At State of Open Con, a panel weighed what's ahead for the movement.

Feb 18th, 2025 9:42am by Jennifer Riggins

Photo of Paula Kennedy, of Syntasso (from left), Nikki Watt, of OpenCredo, Leena Mooneeram, of Chainalysis and Shweta Vohra, of Booking.com at State of Open Con by Jennifer Riggins.

As platform engineering sits on the edge of a precipice, is it on the brink of falling down into Gartner's trough of disillusionment? Will it finally bridge the chasm between business and tech? Or will it fall from grace and into disinvestment in 2025? What will the next two years look like?

If 89% of organizations are already using some form of internal developer platform (according to the latest DORA report), the future of platform engineering lies in how you deliver what your developers want. It's all about developing a platform as a product mindset, said Booking.com's Shweta Vohra, OpenCredo's Nicki Watt, Chainalysis's Leena Mooneeram and Syntasso's Paula Kennedy, during a panel at State of Open Con in London earlier this month.

And of course, the longevity of your platform engineering program depends on measuring its impact on the software development life cycle.

The panelists predicted that, over the next two years, platform engineering innovation will be about what it means to build and maintain a platform. Platform engineering teams, the panelists said, will look to help devs reduce their own cognitive load by getting them to adopt and extend the internal developer platform (IDP).

Which Problems Does Platform Engineering Address?

As the trend moves into its third year, the industry is only starting to coalesce around a definition of platform engineering.

If platform engineering is trying to accomplish one thing, said Watt, CEO and CTO at OpenCredo, it's to offload the cognitive burden that developers experience while trying to ship software.

Platform engineering, she said, is "an evolution of DevOps that looks to try and take some of the heavy lifting away from developers, so you can concentrate on just building the software that gets shipped out to users." This non-differential work can range from spinning up infrastructure and cloud environments to consolidating and standardizing the dev tool sprawl, so that there's "a paved path or sensible way to move forward, so that we can move forward as an engineering group."

SHARE THIS STORY

TRENDING STORIES

1. Kubernetes Complexity Realigns Platform Engineering Strategy
2. Is Swag The Secret to Getting Developers to Use Your Platform?
3. Driving Platform Adoption: The Missed Opportunity of Marketing
4. How SPS Commerce Built Its Internal Developer Portal
5. Is Datadog Becoming a Platform Engineering Company?

TRENDING STORIES

1. Kubernetes Complexity Realigns Platform Engineering Strategy
2. Is Swag The Secret to Getting Developers to Use Your Platform?
3. Driving Platform Adoption: The Missed Opportunity of Marketing

distractions away.

"We try and look at those common patterns — drudge tasks that you're doing repetitively — and how can we abstract them away and remove the cognitive load," said Mooneeram, senior engineer at Chainalysis.

As her organization scales well past startup status, she said, a platform engineering strategy enables alignment.

"We're looking at a landscape where engineers have a lot of autonomy and we have many, many tools. But for platforms to actually be able to contribute and be that multiplier, we have to find that alignment and that commonality."

In order to succeed, a platform engineering team also has to support and measure business goals.

"One of the biggest challenges in our industry today is our fragmented technology ecosystem. We are rapidly evolving, constantly updating, and, on top of that, we are dealing with the ever-evolving business models," said Vohra, enterprise architect at Booking.com and author of "Decoding Platform Patterns."

"Platform engineering is helping us in abstracting at least two of these problems, if not all. It should definitely be managing the fragmented technology for us and keeping pace with the evolving, constant updates that businesses deal with."

If a platform doesn't successfully hide away the technical complexity, she argued, it fails to meet its purpose.

How Does a Platform Team Measure Its Success?

Platform engineering is, first and foremost, about engineering, which makes it a science, which means you can't improve what you can't measure.

"If we are not accelerating the digital transformation and slowing it down, maybe we are not engineering the right thing," Vohra said. On the other hand, she added, "We need to stop treating our internal platforms or all this platform engineering stuff as secondary. We need to mirror it like any business platform."

For Chainalysis — and many platform teams — one of those key metrics is IDP adoption and usage rates.

"That's a proxy for making sure that we're actually delivering solutions that our engineers want," Mooneeram said. "And without perhaps deeper, qualitative informational surveys, we're not able to find that information quickly."

Truly, it takes a combination of quantitative and qualitative measurements, the panelists agreed. These could include:

- DORA metrics.
- SPACE framework.
- DevEx metrics.
- DX Core4 framework.

Look for oppositional metrics, recommended Watt, like making sure to not focus on speed at the expense of quality. Also, she said, you don't want to measure so many things at once that you can't identify the thing that's causing the most harm to developer experience.

You also need to be clear on your goal, which at some companies could be that of a more traditional platform's focus on security and standards enforcement, which means you will be measuring very differently than if you are optimizing for developer productivity.

What Does 'Platform as a Product' Mean?

Is the promise of the platform being delivered right now? The panelists unequivocally agreed: No, it's not.

Platform teams are faltering when they focus on building technological solutions that don't match their internal developer customer needs. They aren't treating their platform as a product.

"It's the biggest problem that I think we face in platform engineering at the moment, but I also think it's our biggest opportunity," Mooneeram said, even advocating to have product owners or product managers on the platform team.

"I do wish they hadn't codified the concept of the platform team. Naming is important — it pushes a direction — 'We exist to build a platform,'" Newman wrote. However, he added, as the book Team Topologies outlines, that's not really the purpose of a platform team: "The mission of the platform team, should you choose to accept it, is about enabling these stream-aligned teams to get things done."

When teams get stuck building a platform for platform's sake, Kennedy said, product mindset isn't a consideration.

"We're incredibly lucky. We work in the same company as our customers, and so we need to work really hard to get that information and to create those relationships," Mooneeram said, "so that when you are then showing a new product: One, they already know that it's coming. And two, it's something that they actually need and is filling a void."

Watt recommended asking your internal developer customers:

- What's holding you back the most?
- If you could change one thing about your job, what would it be?

The Future State of Platform Engineering Technology

As platform engineering has gained traction, unsurprisingly, so has the breadth of platform tooling, and the open source or proprietary, build-versus-buy debate will only continue over the next couple years.

"There's a lot of competition and combinations around the open source tooling going on, and that's where we are dealing with this technology mindset first, rather than abstracting that technology," Vohra said.

She sees most organizations learning toward DIY solutions, including at Booking.com. "This gives us flexibility. Somewhere, engineers want to handle and learn and introduce AI on their own terms. Strictly giving them some abstraction, which just hides away all the internals from them, doesn't work for them."

Build-your-own platforms, she continued, enable:

- Flexibility.
- Abstraction.
- Centralized governance.
- Decentralization.

So far, Software as a Service (SaaS) products, she argued, simply don't bring these same benefits as home-built solutions do.

Plus, "Engineers are always going to engineer," Kennedy remarked, which is why build-your-own tends to be compelling for most platform engineers.

True, Mooneeram agreed, but platform teams are also typically understaffed. That's why, over the next two years, she predicts a move toward adopting existing open source or even proprietary tooling that allows for some flexibility and extendibility — "an organization specific wrapper" — without building from scratch.

And while the Cloud Native Computing Foundation (CNCF) landscape, like the universe, seems in infinite expansion, Watt said that's in part because lots of different tools work for very specific situations.

However, "there will probably be one or two that will become a little bit more prominent, that are more helpful in certain situations," she added. "I think there may well be SaaS offerings that are actually quite helpful. Maybe they build on some of the open source tools. But for organizations that are just starting out, or that are actually quite small, it's really easy for them to get up and up and running."

But Where Are the People and Processes?

Like all things engineering, technology is only part of the story.

With more than a decade's experience running a cloud native computing consultancy, Watt has observed a pattern of organizations starting with one platform team to reduce friction. But as they scale, so does the number of platform teams.

> "We're incredibly lucky. We work in the same company as our customers, and so we need to work really hard to get that information and to create those relationships, so that when you are then showing a new product: One, they already know that it's coming. And two, it's something that they actually need and is filling a void."
> **– Leena Mooneeram, Chainalysis**

Even Matthew Skelton, co-author of the engineering management guide "Team Topologies," has said he wishes he had called it a "platform grouping," as it is rarely just one platform team.

"We're effectively a platform team within a data platform organization," Mooneeram said, of her platform team at Chainalysis. "So it feels like we're nested anyway, so lots of the teams actually within where they train others."

On the other hand, for startups, a single platform engineer can accomplish a lot.

As you grow, Mooneeram said, you need an internal developer advocate who does some sort of product owner or platform internal marketing role. This Platform as a Product role should also increase awareness so that the work can be inner sourced, with app teams contributing into the platform.

Of course, two years from now, it won't just be developers building the platform. Watt predicted that AI agents would be embedded throughout the process. Platforms, she said, will continue to become more composable, too.

Vohra offered more predictions: "I can see platform engineering is on the same trajectory as we have seen DevOps, from concept to movement to reality. What I would like to see is more of consolidation of efforts, more of experience building, because the future doesn't belong to those who just build fast with the AI.

"The future belongs to those who build experience, for the engineers and the developers alike, so that it works for business, because it will accelerate for them the business and, of course, for the engineers."

TNS

Jennifer Riggins is a culture side of tech storyteller, journalist, writer, and event and podcast host, helping to share the stories where culture and technology collide and to translate the impact of the tech we are building. She has been...

Read more from Jennifer Riggins →

TNS DAILY NEWSLETTER

ARCHITECTURE

Cloud Native Ecosystem
Containers
Databases
Edge Computing
Infrastructure as Code
Linux
Microservices
Open Source
Networking
Storage

ENGINEERING

AI
AI Engineering
API Management
Backend development
Data
Frontend Development
Large Language Models
Security
Software Development
WebAssembly

OPERATIONS

AI Operations
CI/CD
Cloud Services
DevOps
Kubernetes
Observability
Operations
Platform Engineering

CHANNELS

Podcasts
Ebooks
Events
Webinars
Newsletter
TNS RSS Feeds

THE NEW STACK

About / Contact
Sponsors
Advertise With Us
Contributions

roadmap.sh

Community created roadmaps, articles, resources and journeys for developers to help you choose your path and grow in your career.

Frontend Developer Roadmap →
Backend Developer Roadmap →
Devops Roadmap →