Improving your Developer Experience

Through Platforms & Generative AI

Ajay Chankramath, Head of Platform Engineering
Chris Westerhold, Head of Developer Experience

May 6th, 2024

What helps my experience as an FP developer ?

Immutability	Less Bugs	Concurrency / No Race Conditions
Modularity	Reusable	Easier to Scale
HOFs	Flexible Pass Alongs	Powerful abstractions
Declarative	Readability	Better reasoning
Pure Functions	Predictable	Testability
Lazy Evaluation	Optimized Perf	Infinite Data Structures
Robustness	Robust Code	Less prone to bugs
Static Typing	Error Reduction	Reduce the burden of typed inferences

What hurts my experience as an FP developer ?

Learning curve	New concepts	Different methods
Performance Issues	Memory Usage	Copy due to avoiding mutable data
Miss my imperative	Database Operations	Too Verbose
Resource overhead	Recursion	Higher CPU
Tooling ecosystem	Limited lib support	Better reasoning
Industry adoption	Not mainstream	Skilled programmers

What is DevEx?
&
Why it Matters?

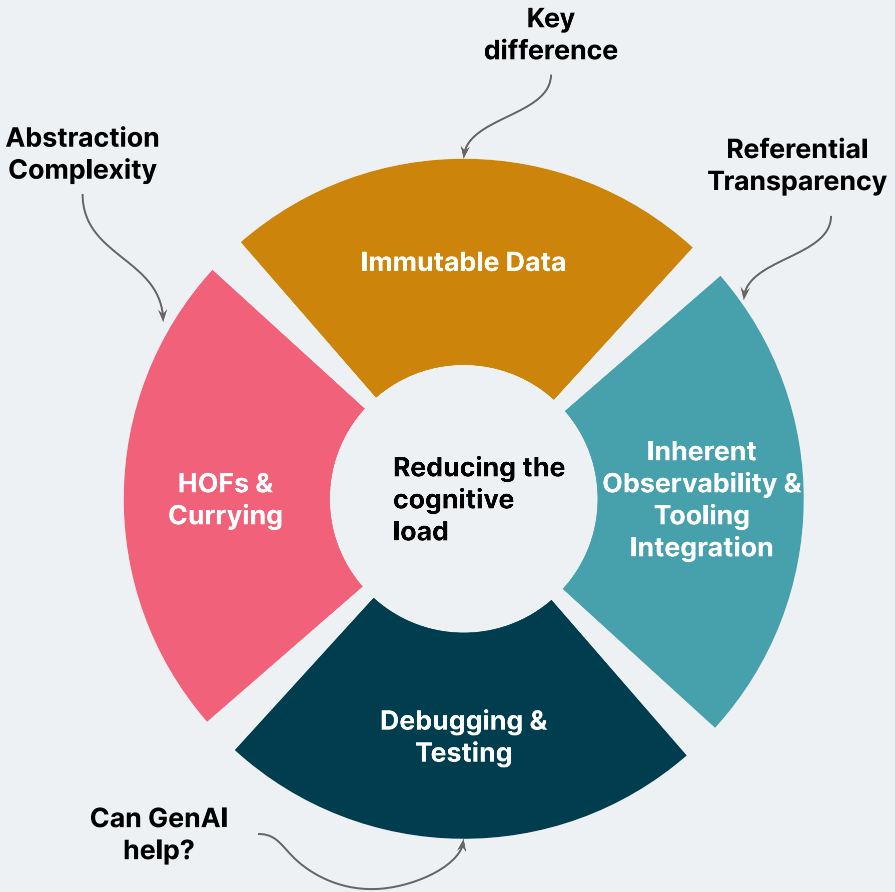
Value of human time at most software companies is 20-100x the value of machine time.

Capability optimization should almost always focus on saving developer time

Developer experience is the **key** to modern software development success

Lack of clarity in **optimizing** DevEx

Developer experience challenges in Functional Programming are not unique, but there are some differences from imperative programming



DevEx Principle # 1 - KISS

Clear and concise documentation, examples, and error messages. APIs and libraries developers is intuitive

Avoid unnecessary complexity and abstraction

DevEx Principle # 2 - Consistency

Well thought out and **documented** patterns for naming conventions, coding styles, and design patterns

DevEx Principle # 3 - Fail Fast

Strive for quick build times, automated testing, and instant previews so that you catch problems early in the process.

DevEx Principle # 4 - Tooling Ecosystem

Invest in an Engineering Platform abstraction with a composable, replaceable, self-serve paradigm

DevEx Principle # 5 - Learning

Collaborative approach at its core

Leaning on supervised learning and models that support

The outcome of DevEx is *NOT* about having good metrics… it's about delivering on business value

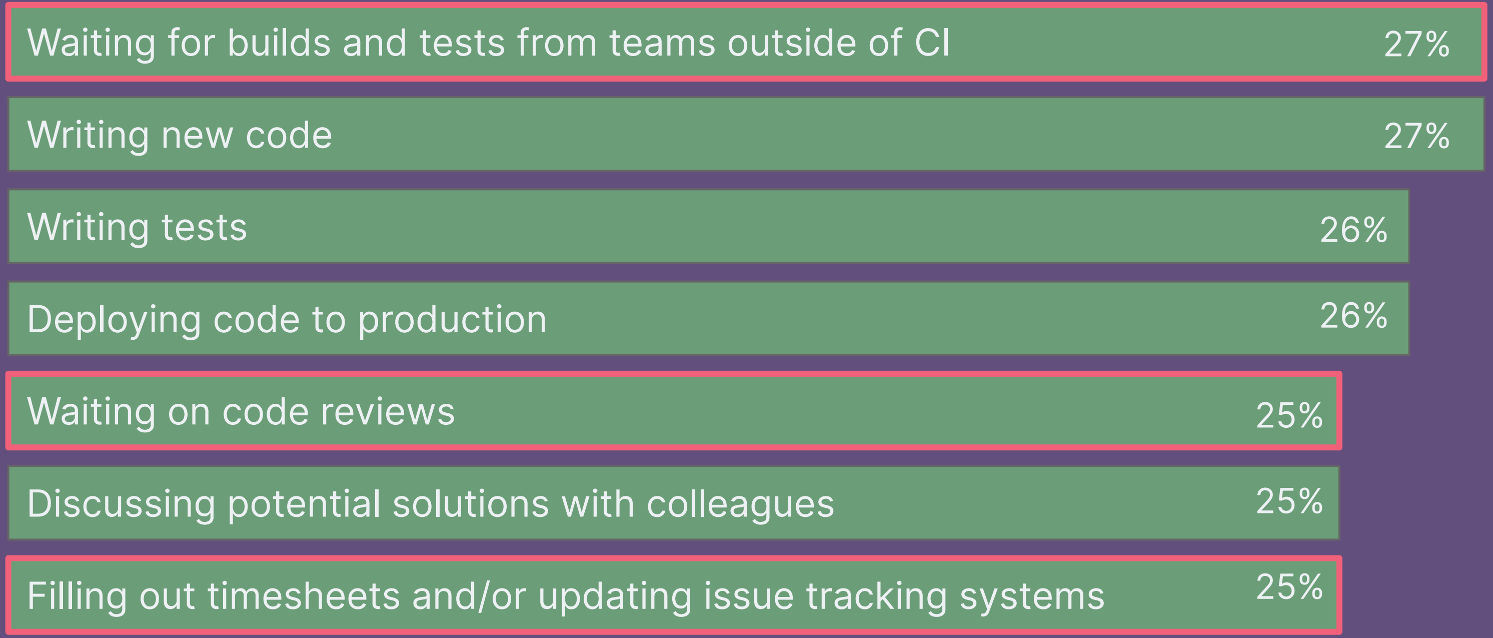
The engineering metrics we see all of the time (Lagging)

	Low	Mid	High
Deployment frequency	Monthly or less often	Between daily and weekly	On demand (whenever we want)
Lead time for changes	Between a week and 6 months	Less than a week	Less than an hour
MTTR	Less than a week	Less than a day	Less than an hour
Change failure rate	Less than 15%	Less than 15%	Less than 5%

The symptoms of the problem

The actual root of the problems (Leading)

Which of the following do you spend the most time doing in any given day?

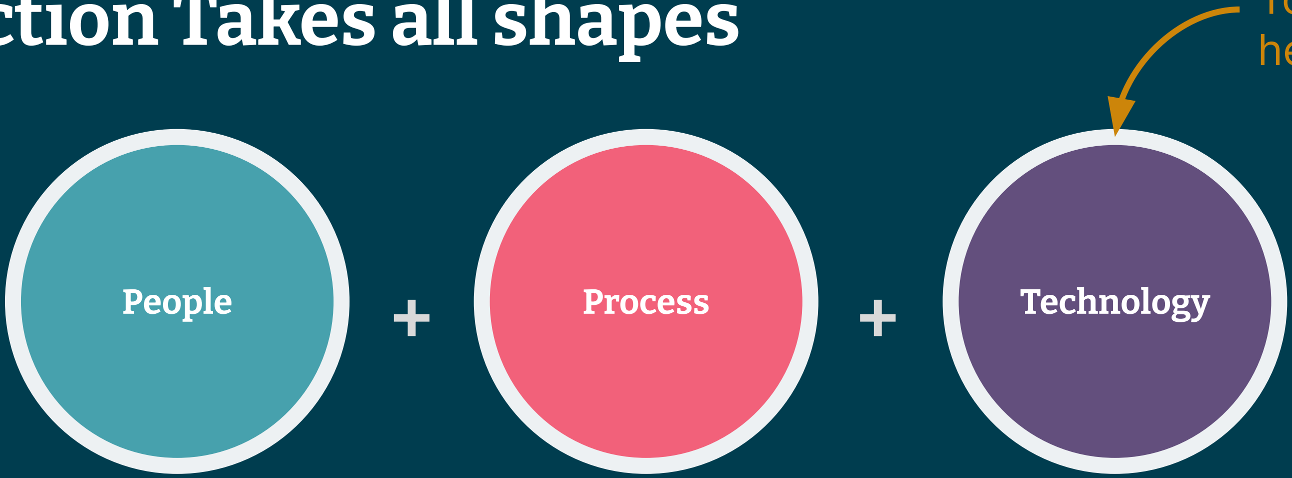


Top 3 ranked responses, top responses show, N=500

* Data credits: Github.com blogs/surveys

If you wait for the metrics to tell you there is a problem... it’s too late

DX Friction Takes all shapes



Too much focus here

People + Process + Technology

Understand pain and friction points

Understanding of skill set challenges

Org structure friction

Be metrics driven

Understand that all friction is not tech based

Value stream map to identify lead, wait, and overall cycle times

Right tools for the job

Well defined ownership

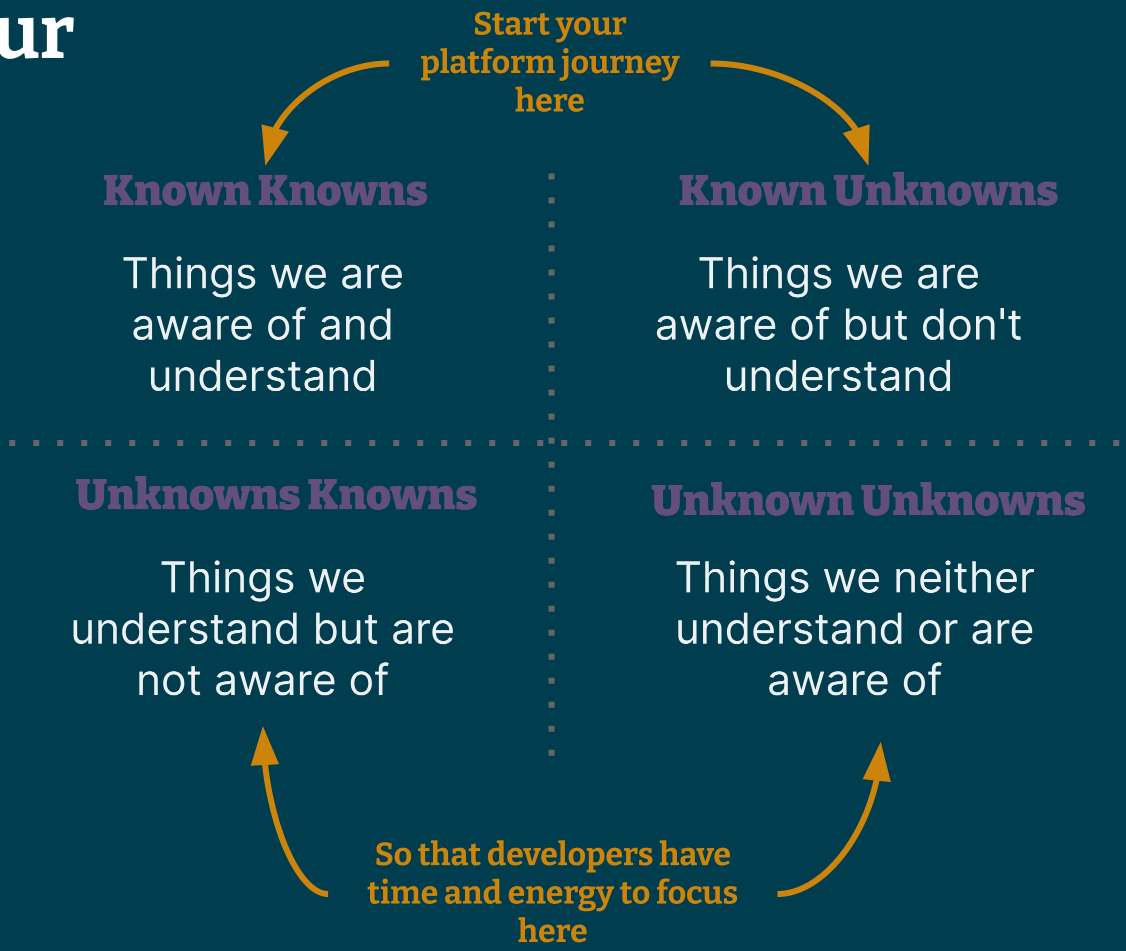
Continuous improvement based on feedback

Platform Engineering to the rescue

Platform engineering can accelerate the delivery of applications and the pace at which they produce business value (Gartner)

It helps developer experience by providing self-service capabilities that are composable and replaceable

Leverage your platform



Start your platform journey here

Known Knowns

Things we are aware of and understand

Known Unknowns

Things we are aware of but don't understand

Unknowns Knowns

Things we understand but are not aware of

Unknown Unknowns

Things we neither understand or are aware of

So that developers have time and energy to focus here

How do you build a platform?

Developer Centric

❏ Easier to code, test, deploy
❏ Additional Support needs!

HEROKU

Backstage

Operations Centric

❏ Better stability, uptime and reliability
❏ Devs still do infra abstractions!

kubernetes

Open App Model

KubeVela

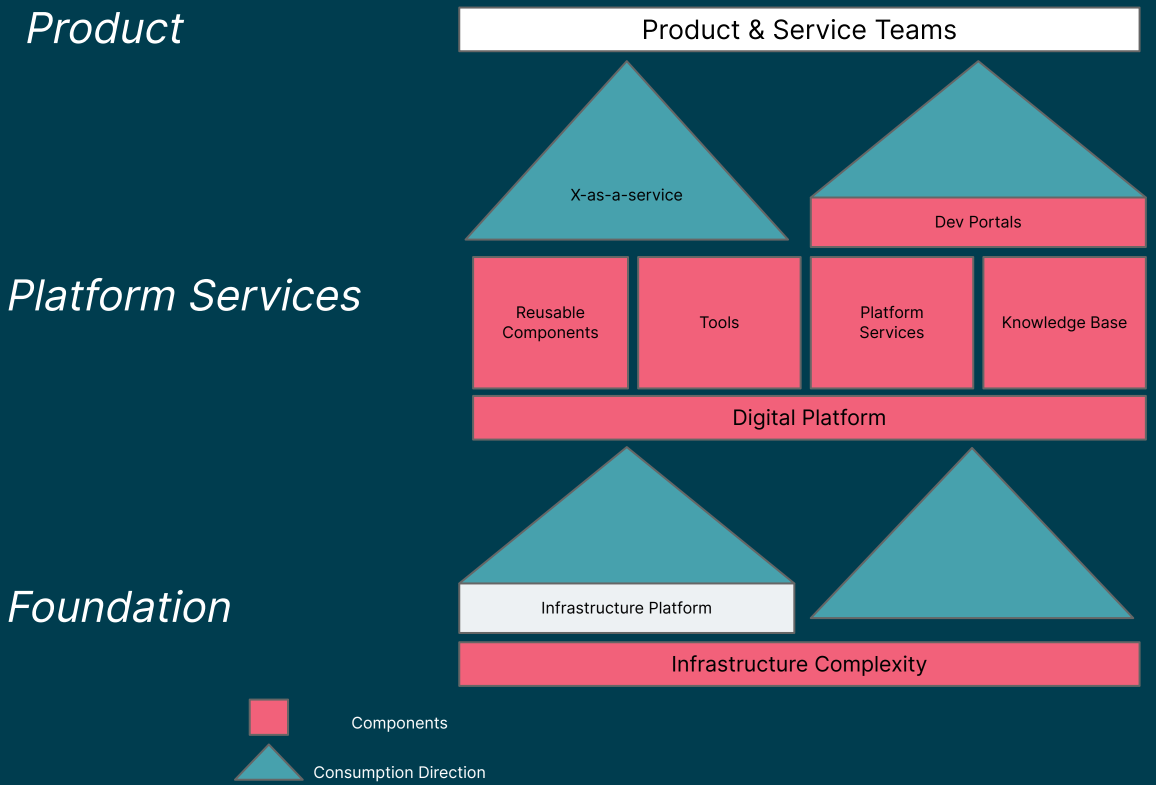
Platform Engineering Centric

❏ Serves both devs & ops
❏ Customizable and integrates with existing toolset as an IDP/ abstraction layer on top of K8S

humanitec

KRATIX

Platform Engineering - Gartner View



Product

Product & Service Teams

X-as-a-service

Dev Portals

Platform Services

Reusable Components

Tools

Platform Services

Knowledge Base

Digital Platform

Foundation

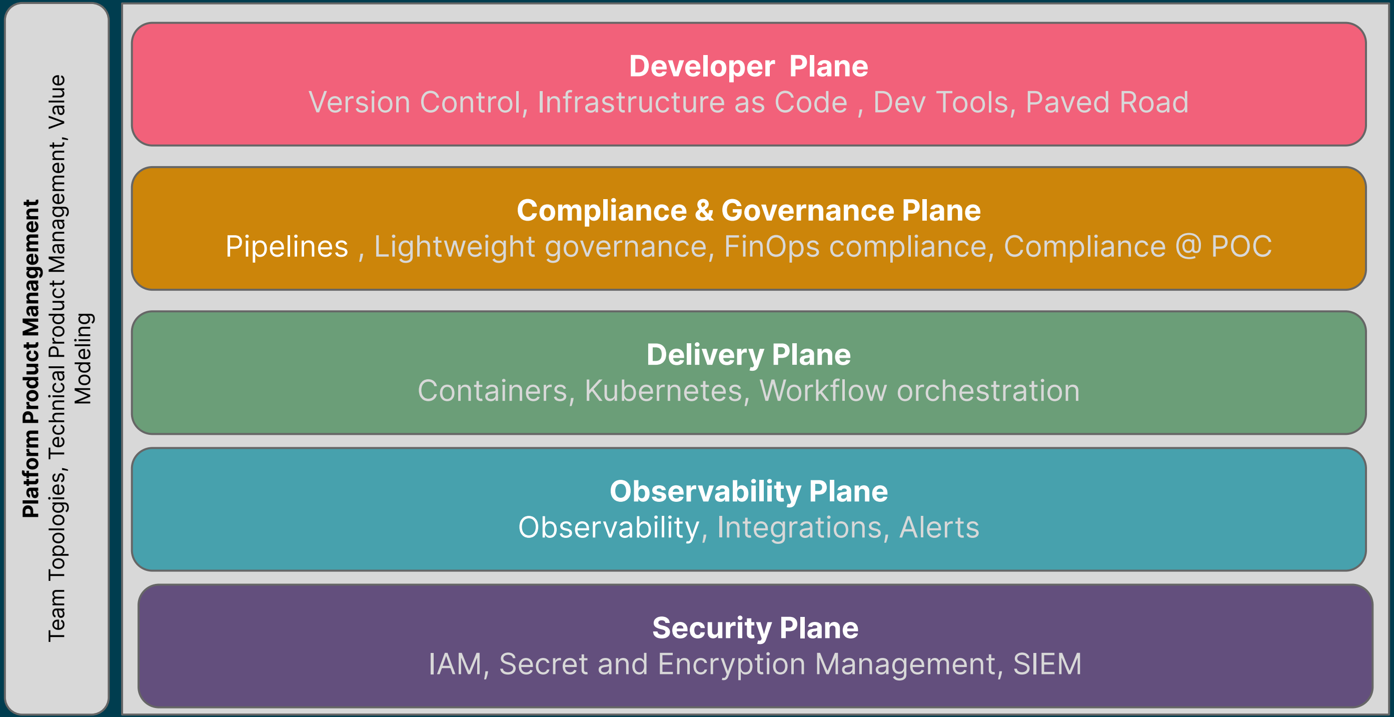
Infrastructure Platform

Infrastructure Complexity

🟥 Components

▲ Consumption Direction

Scope of Engineering Platforms



Platform Product Management
Team Topologies, Technical Product Management, Value Modeling

Developer Plane
Version Control, Infrastructure as Code , Dev Tools, Paved Road

Compliance & Governance Plane
Pipelines , Lightweight governance, FinOps compliance, Compliance @ POC

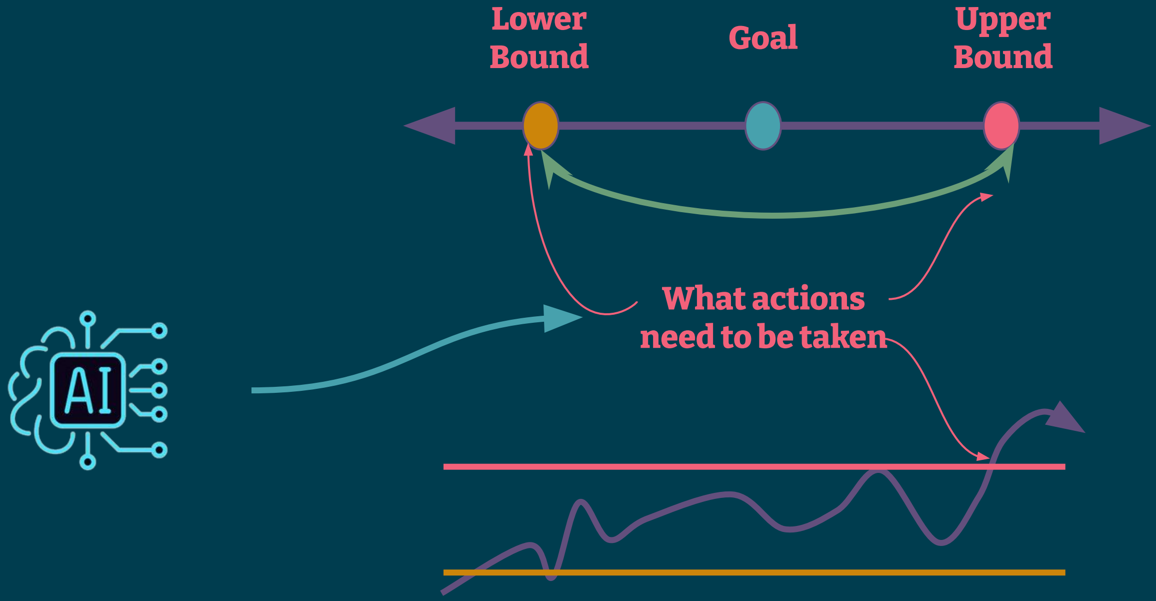
Delivery Plane
Containers, Kubernetes, Workflow orchestration

Observability Plane
Observability, Integrations, Alerts

Security Plane
IAM, Secret and Encryption Management, SIEM

Where is GenAI having a positive impact?

Use AI to find the signal through the noise



And to take action

AI Code Assistants for Functional Programming

codeium

tabnine

AI Assistant For Software Developers

GitHub Copilot

replit

BLACKBOX AI

AI Code Reviews for Functional Programming

Codacy

deepsource

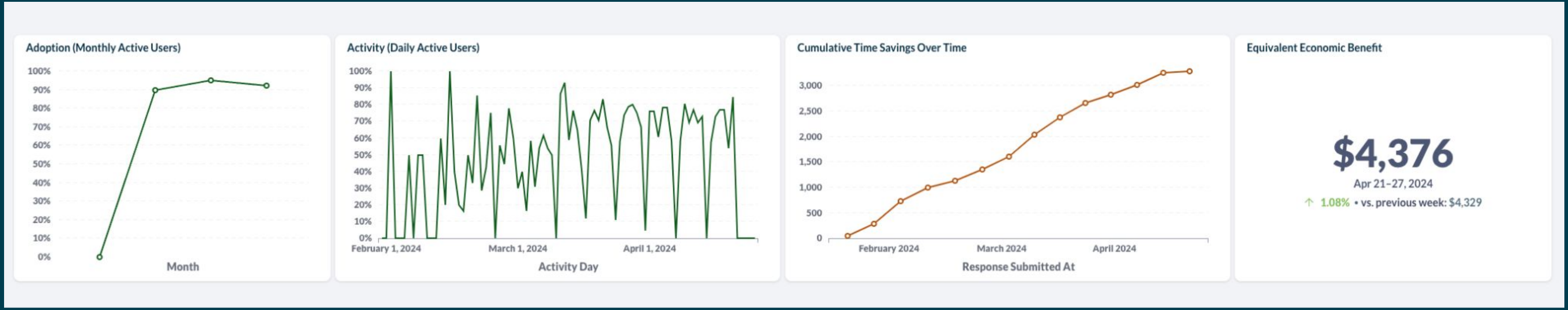
sonarqube

CodeScene

GitHub Copilot

Know what good looks like

Measure the ROI of AI Assistance



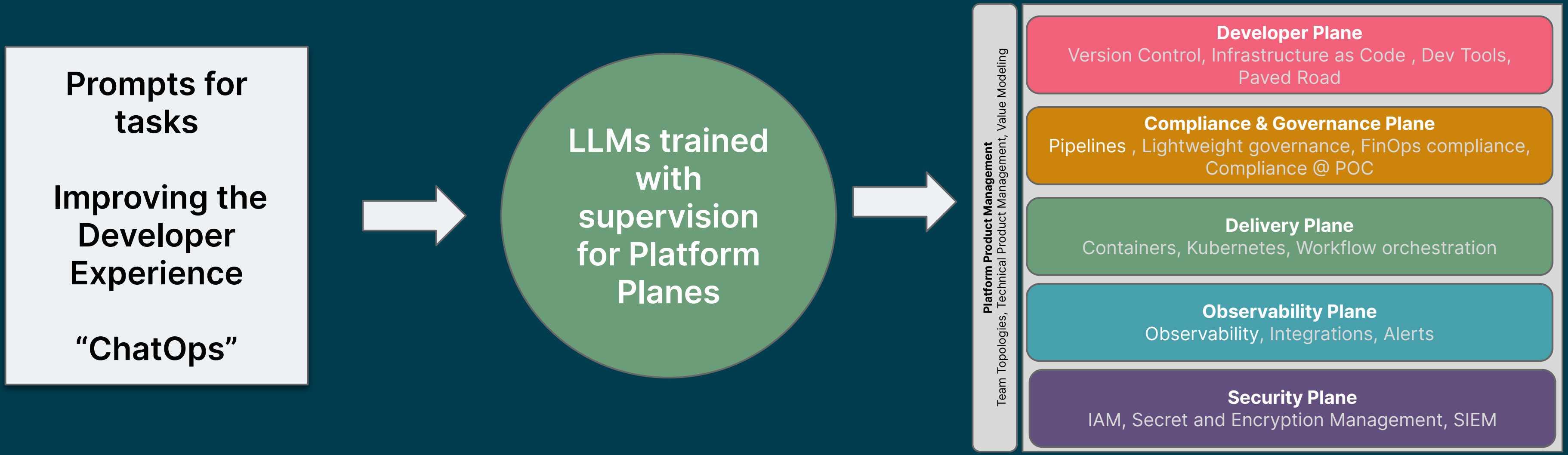
$4,376

Apr 21–27, 2024

↑ 1.08% • vs. previous week: $4,329

Courtesy: FarosAI

Applying Prompts and Supervised training



Prompts for tasks

Improving the Developer Experience

“ChatOps”

LLMs trained with supervision for Platform Planes

Platform Product Management
Team Topologies, Technical Product Management, Value Modeling

Developer Plane
Version Control, Infrastructure as Code , Dev Tools, Paved Road

Compliance & Governance Plane
Pipelines , Lightweight governance, FinOps compliance, Compliance @ POC

Delivery Plane
Containers, Kubernetes, Workflow orchestration

Observability Plane
Observability, Integrations, Alerts

Security Plane
IAM, Secret and Encryption Management, SIEM

What are these tasks?

- AUTOMATED CODE REVIEWS
- AUTOMATED CODE ANALYSIS

- BUILD PROCESS OPTIMIZATION
- DEPLOYMENT OPTIMIZATION
- ROLLBACKS & ERROR RECOVERY
- SECURITY & COMPLIANCE

- FUNCTIONAL TESTING
- LOAD TESTING
- OBSERVABILITY INSIGHTS
- LOG ANALYSIS

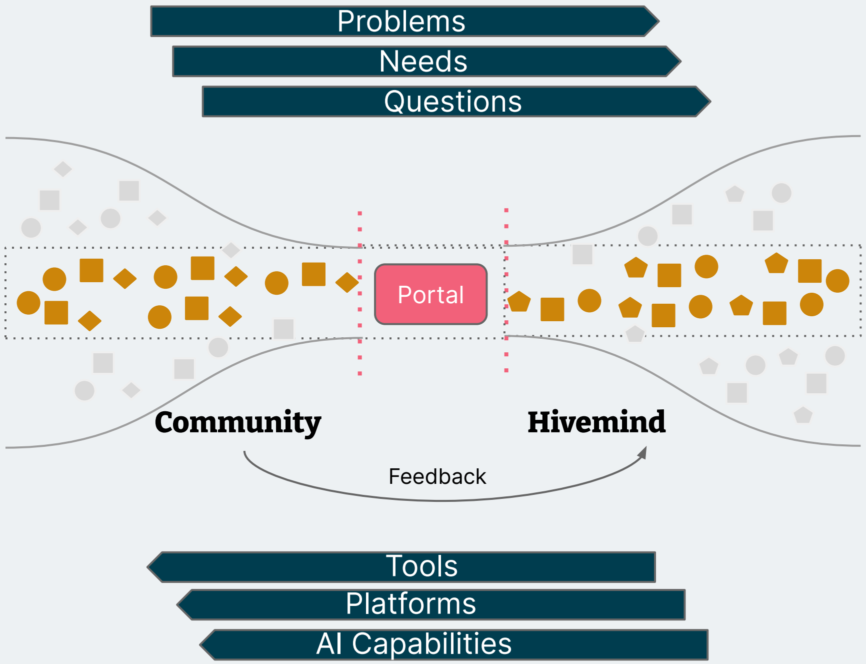
- ABSTRACT IAC
- PREDICTIVE RESOURCE MANAGEMENT

Introduction of an AI Portal

A cohesive platform allows for a central point for the community to find new and existing capabilities across tooling, platform, and AI capabilities.

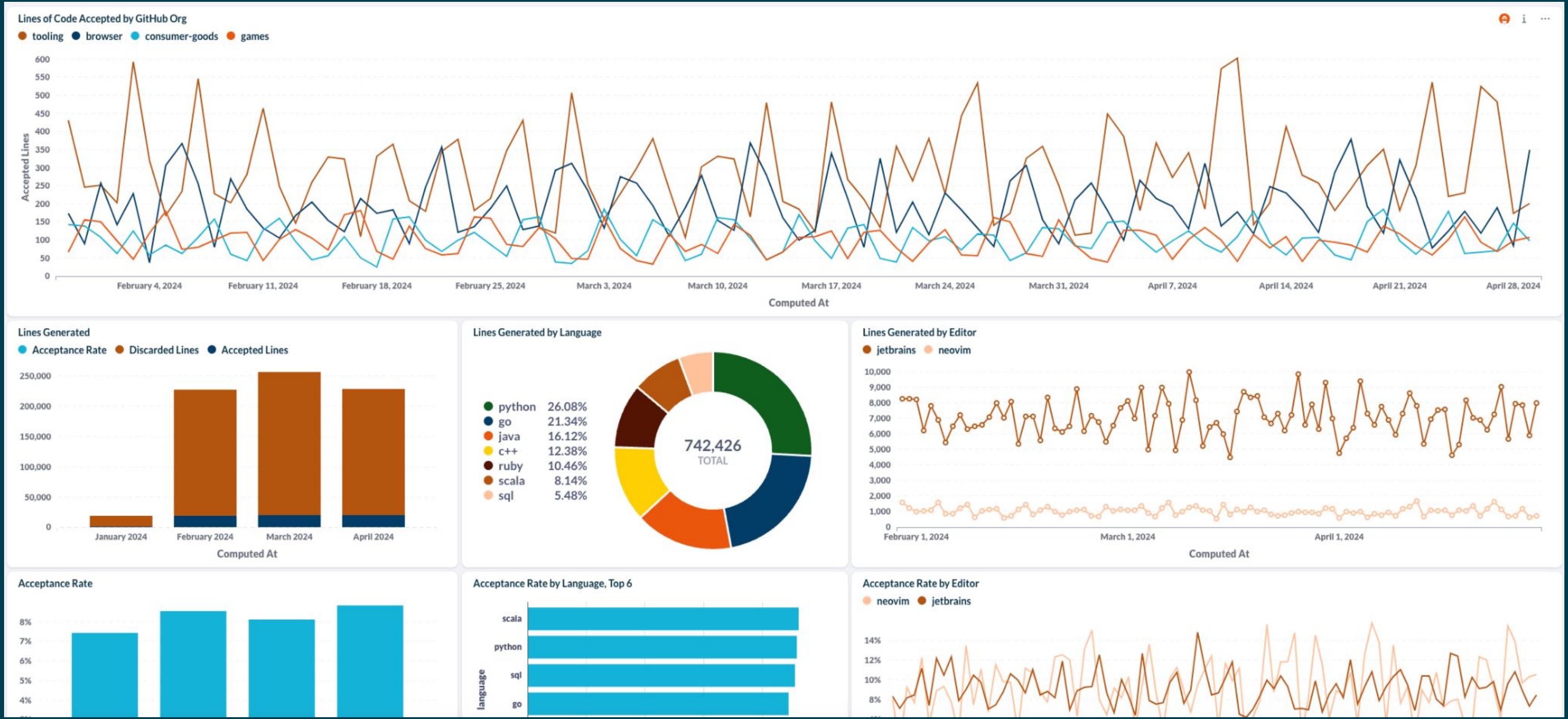
In the fast and changing AI landscape, having fast feedback loops and understanding the realization of value are key to driving overall success.

This should be a part of your developer portal/platform strategy not another stand alone tool.



A cohesive platform

An example of a real-life portal (Faros.AI)



Generative AI in platform engineering catalyzes a transformative shift in developer experience for Functional & Imperative Programmers alike

By automating the mundane and illuminating new possibilities, it empowers developers to innovate at the frontier of technology, redefining the boundaries of what can be built and how swiftly it can be brought to life.

/thoughtworks

Ajay Chankramath

Head of Platform Engineering

ajayc@thoughtworks.com
/chankramath
@ajayctw
chankramath.com
thoughtworks.com

Chris Westerhold

Head of Developer Experience

chris.westerhold@thoughtworks.com
/chriswesterhold
thoughtworks.com
https://www.thinkbigcodesmall.io/