Accelerating Development at Airbnb

768

Agenda

Development at Airbnb

High value recent investments

What’s next?

Anna Sulkina

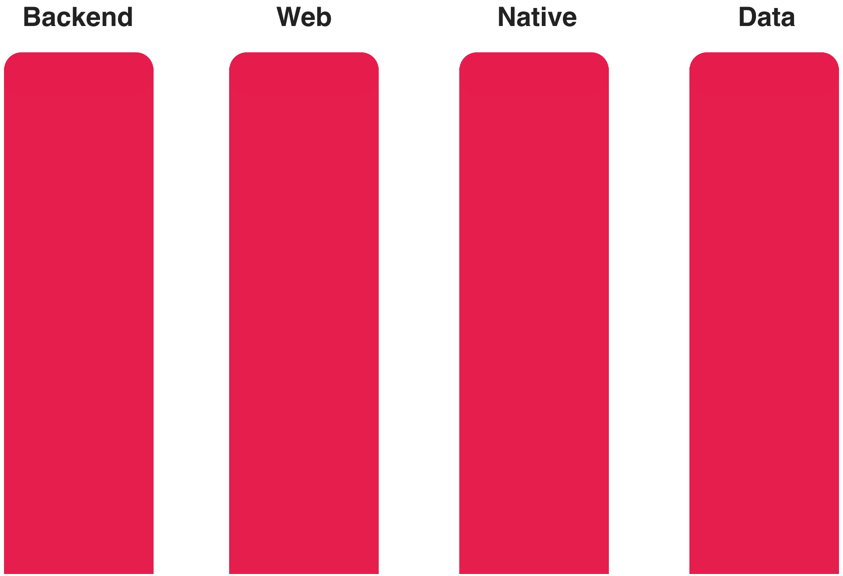
Head of Developer Platform & Core Services

Airbnb

Development at Airbnb

What does development look like at Airbnb?

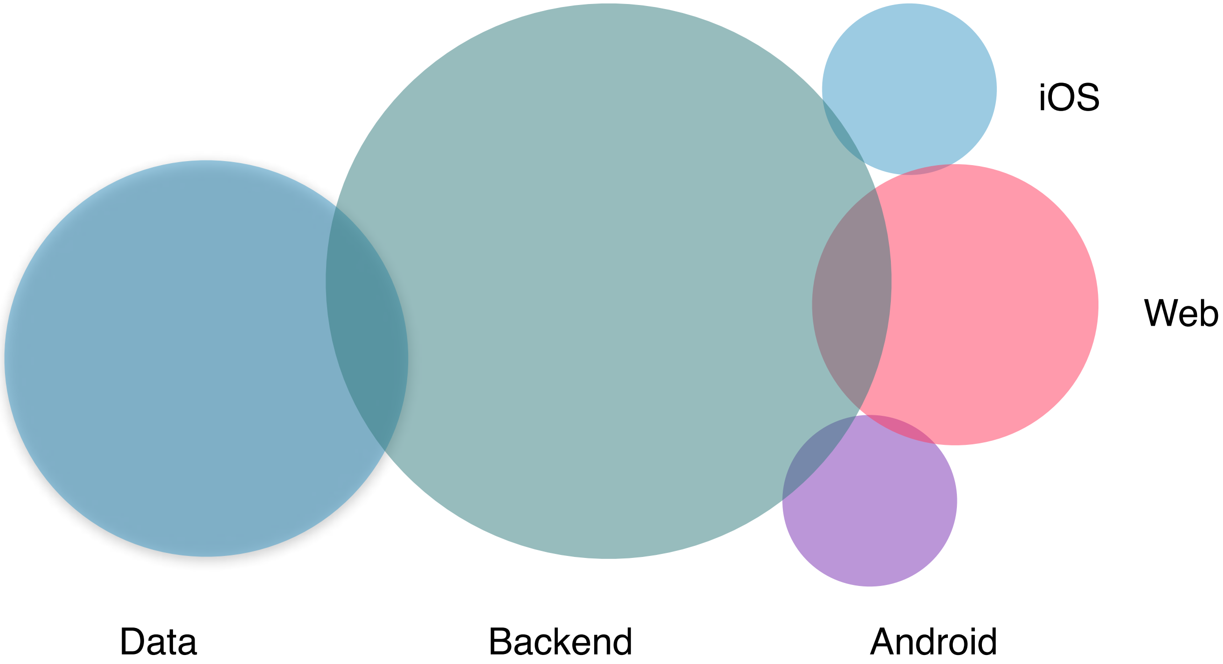
Multi-monorepo



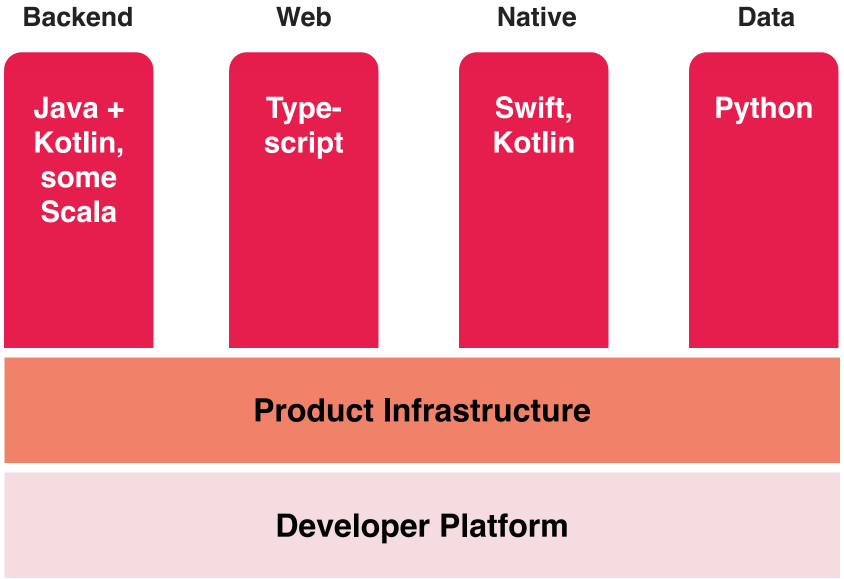
Multi-monorepo

Backend	Web	Native	Data
Java + Kotlin, some Scala	Type-script	Swift, Kotlin	Python

Share of Developers



Development Infrastructure



Backend

Java + Kotlin, some Scala

Web

Type-script

Native

Swift, Kotlin

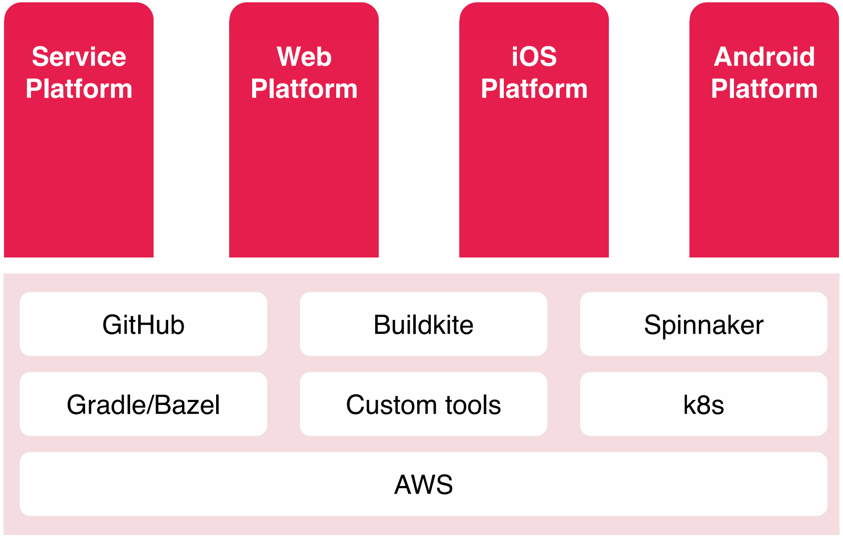
Data

Python

Product Infrastructure

Developer Platform

Developer Platform



Service Platform

Web Platform

iOS Platform

Android Platform

GitHub

Buildkite

Spinnaker

Gradle/Bazel

Custom tools

k8s

AWS

7k

PRs per week

50k

CI builds per week

Sept 2023

Work From Anywhere

Service owners carry the pager

High Value
Recent Investments

Challenges

Lack of clear paved path

Slow builds & IDE

Hard to test code changes

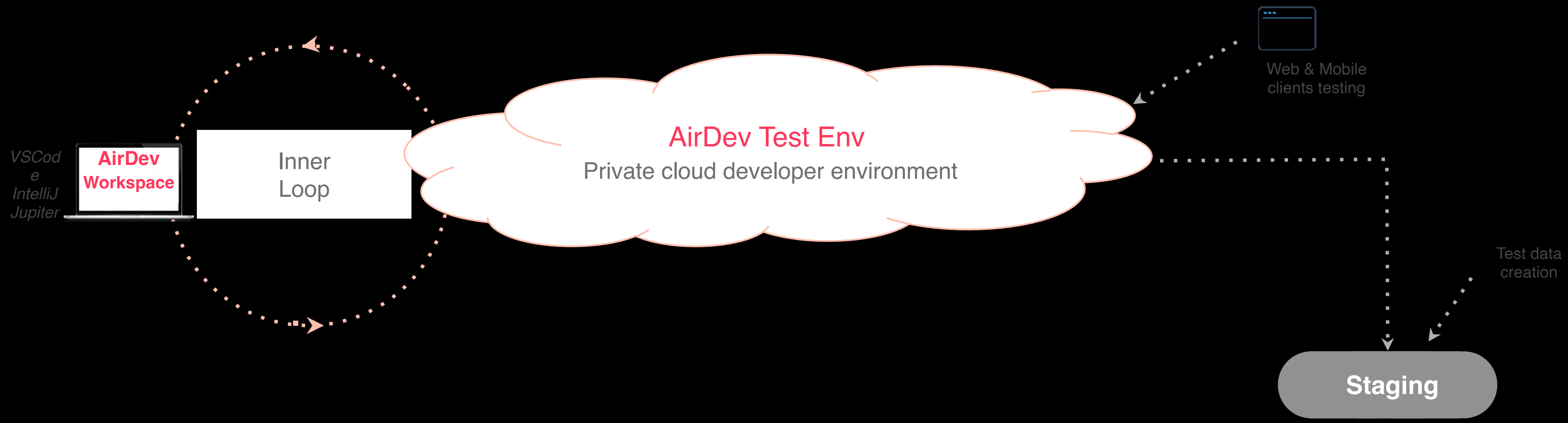
Successful Investments

Cloud development environment

More opinionated paved path

Understanding developers better

Cloud Development Environment



VSCode
IntelliJ
Jupiter

AirDev Workspace

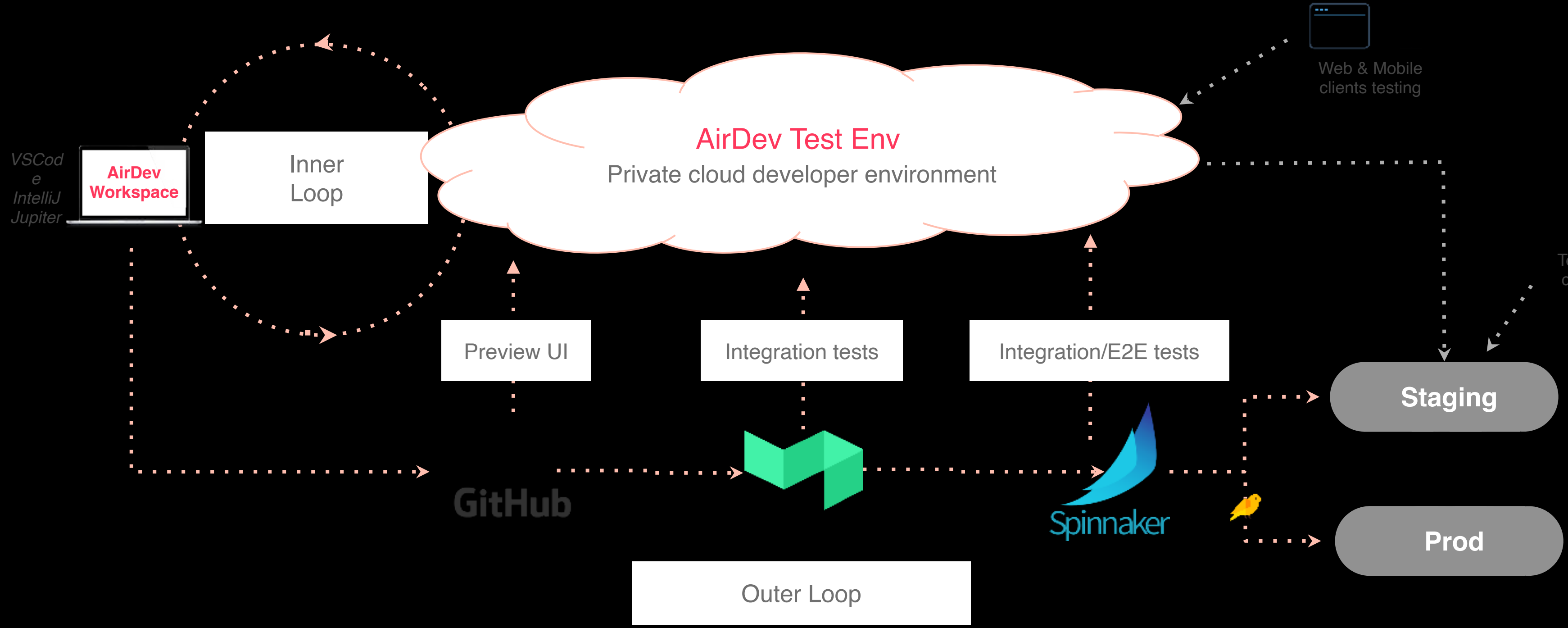
Inner Loop

AirDev Test Env
Private cloud developer environment

Web & Mobile clients testing

Test data creation

Staging



VSCode
IntelliJ
Jupiter

AirDev Workspace

Inner Loop

AirDev Test Env

Private cloud developer environment

Web & Mobile clients testing

Preview UI

Integration tests

Integration/E2E tests

Test data creation

Staging

Prod

GitHub

Spinnaker

Outer Loop

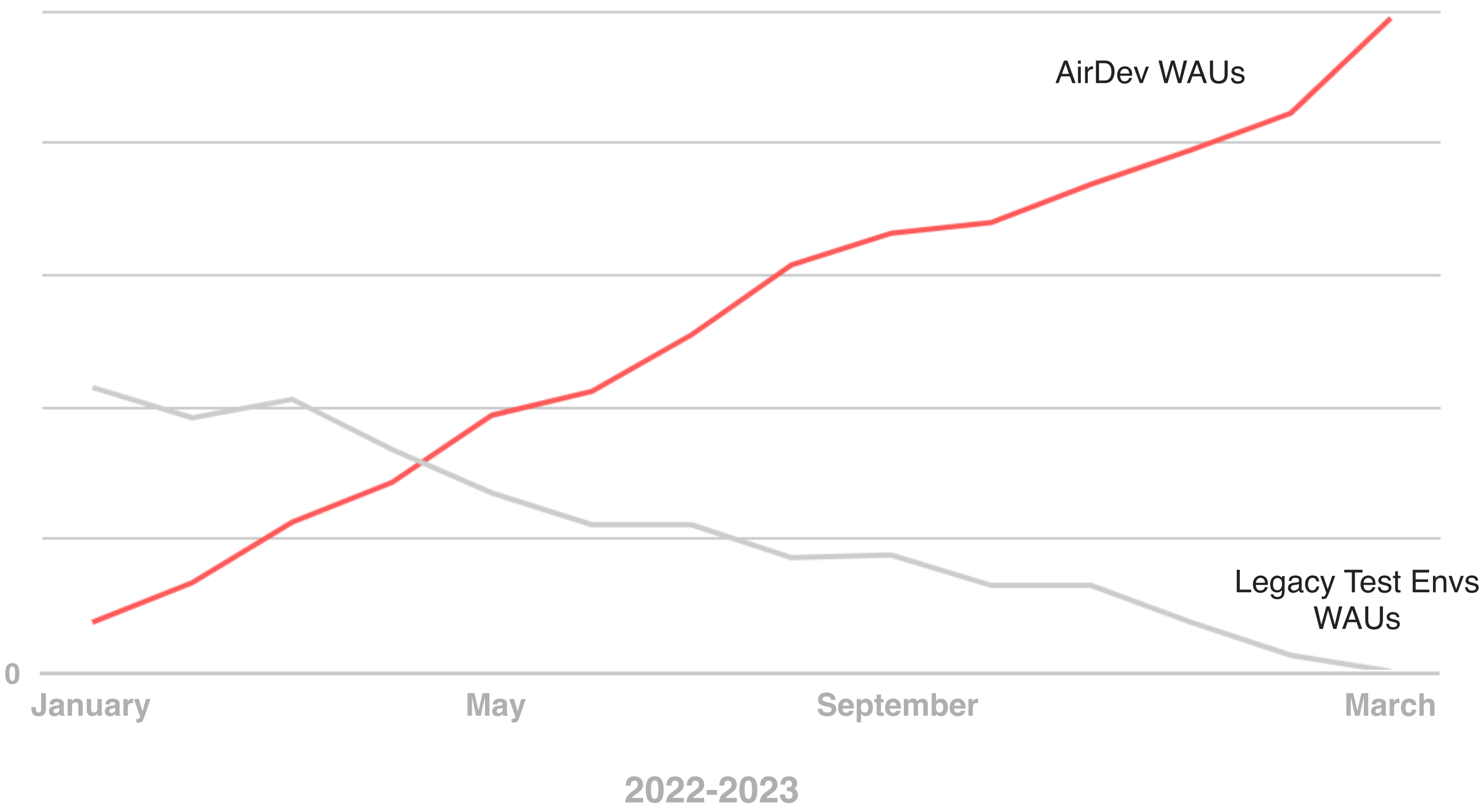
2-4x

Faster Deploys

2.2x

Faster Builds

Early-mid 2022


AirDev WAUs
Legacy Test Envs WAUs
0
January
May
September
March
2022-2023

Opinionated Paved Path

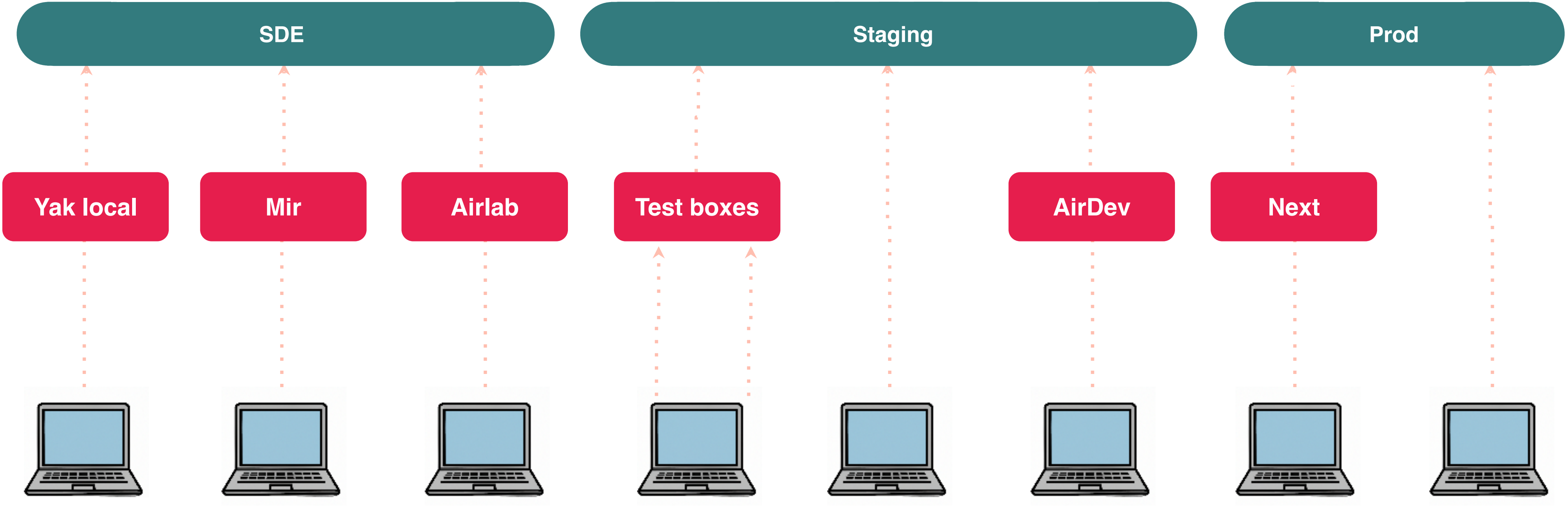
Paved path

Opinionated

Platformized

Explicit standards

Developer Testing



SDE

Staging

Prod

Yak local

Mir

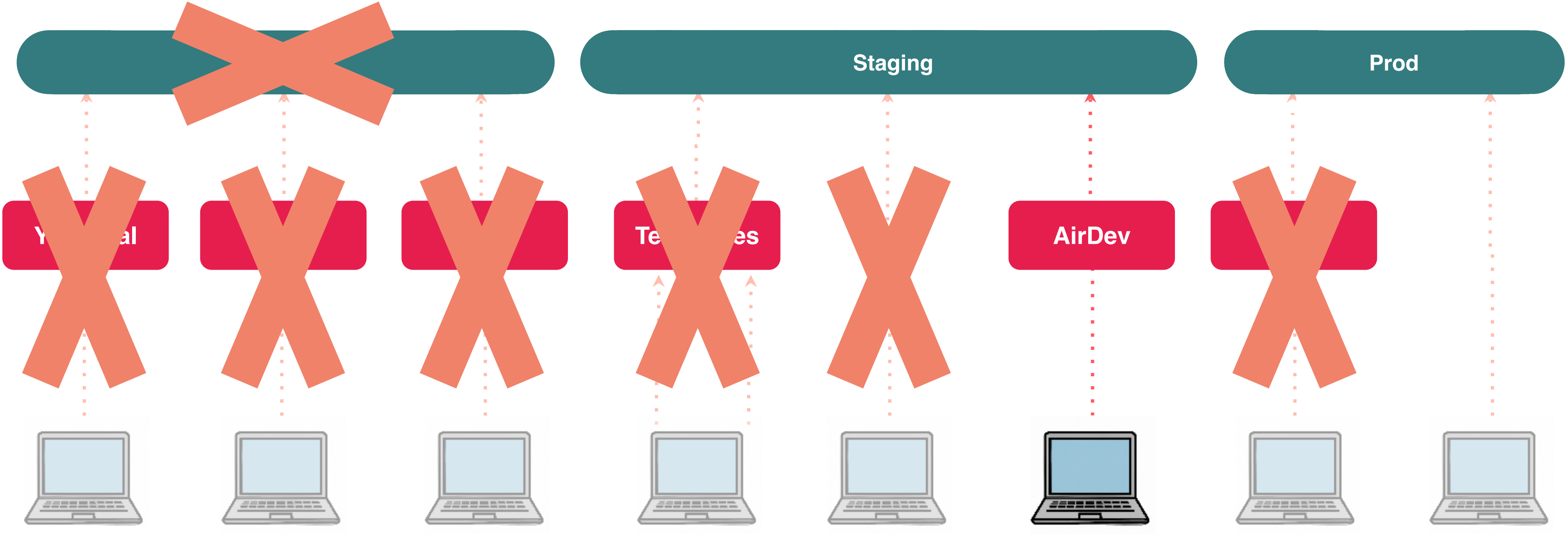
Airlab

Test boxes

AirDev

Next

Developer Testing



Understanding Our Developers Better

Collaboration

X-org partnerships
Early prototypes
Alpha users

DevX Metrics

Quantitative data
DORA metrics
SPACE Framework

DevX Survey

Bi-annual
<5 min to complete
Consistent structure
Summarize with AI

What’s Next?

Goals

Faster feedback loops

Lower failure rates

Happier developers

What’s Next?

Migrating to Bazel

Faster inner loop

Safe testing against production

AI-powered development

Migrating to Bazel

Bazel

Parallelized and incremental

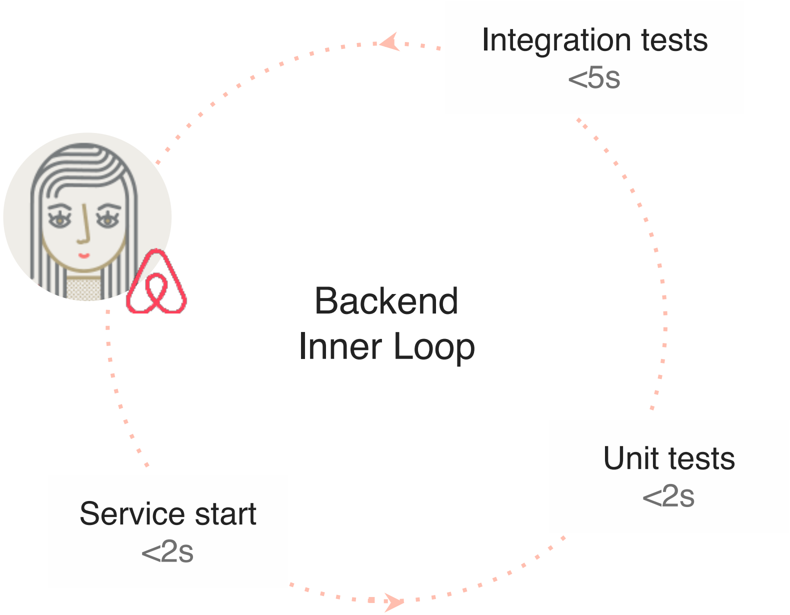
Automation and incentives

Forked IntelliJ Bazel plugin

Custom Bazel files generator

Faster inner loop

Future



Integration tests
<5s

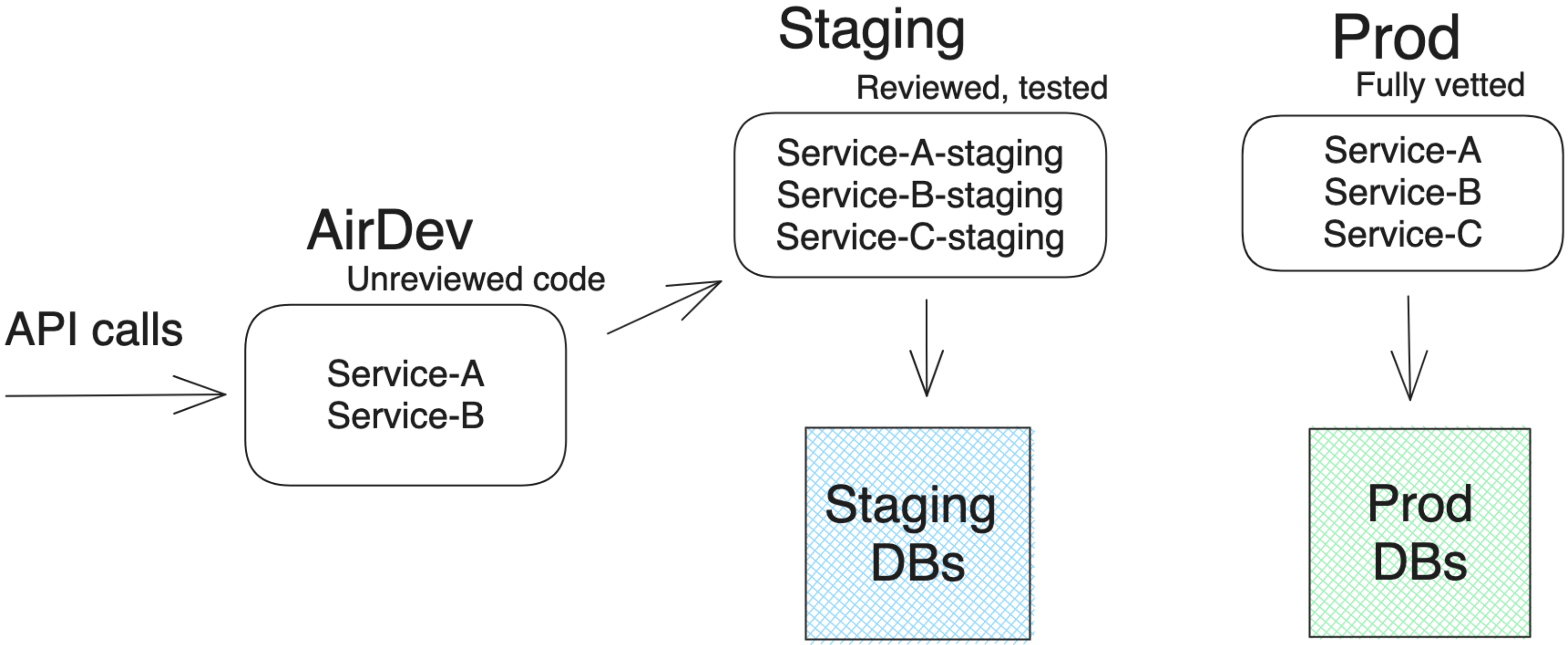
Backend
Inner Loop

Unit tests
<2s

Service start
<2s

Safe testing against production

Current paved path



Staging
Reviewed, tested

Service-A-staging
Service-B-staging
Service-C-staging

Prod
Fully vetted

Service-A
Service-B
Service-C

AirDev
Unreviewed code

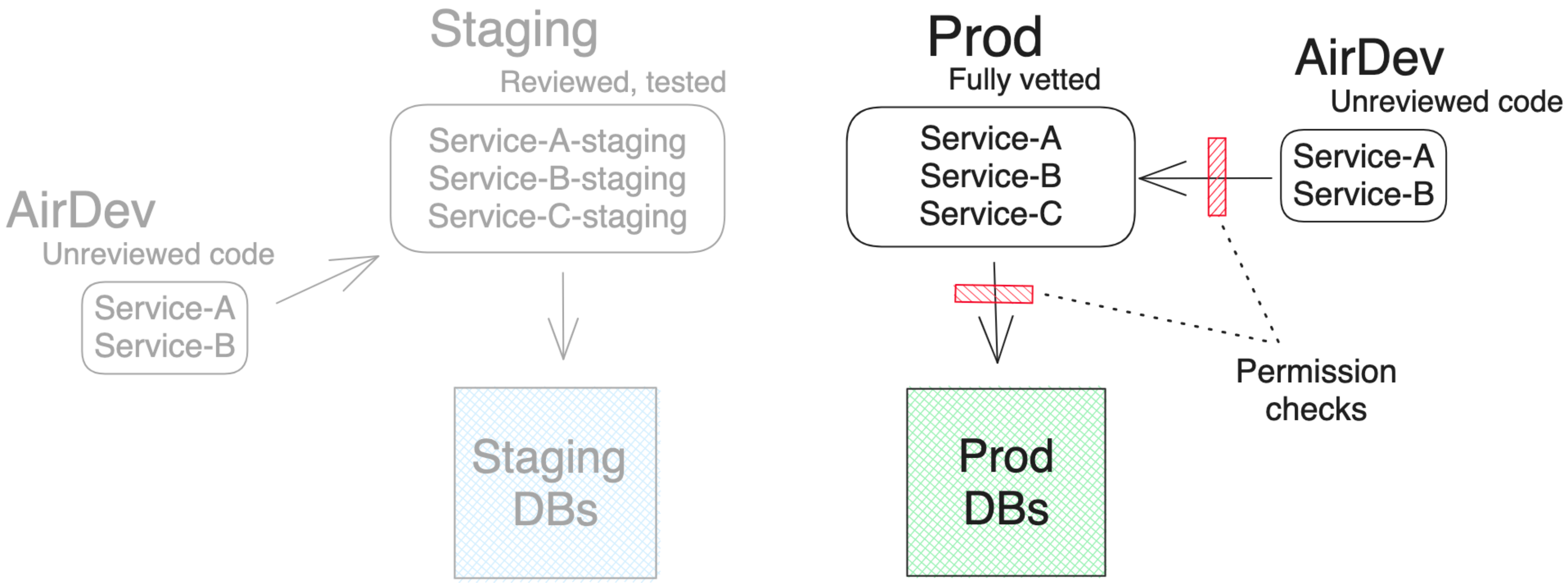
API calls

Service-A
Service-B

Staging DBs

Prod DBs

AirDev + Production North Star



AI-powered development

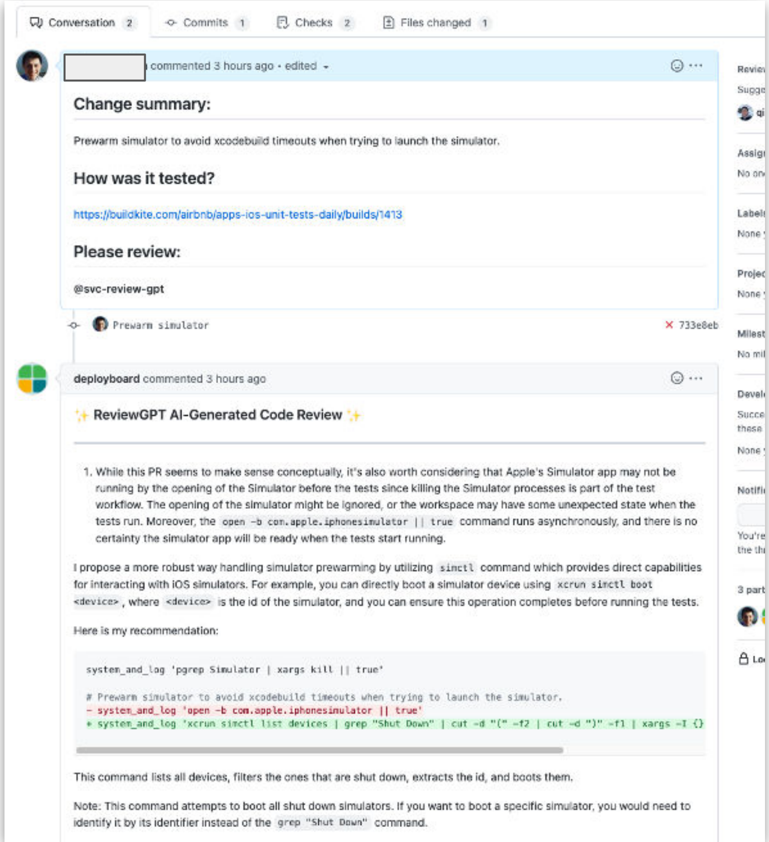
AI-powered

ChatGPT

Github Copilot

Build your own

Review GPT



[redacted] commented 3 hours ago • edited

Change summary:

Prewarm simulator to avoid xcodebuild timeouts when trying to launch the simulator.

How was it tested?

https://buildkite.com/airbnb/apps-ios-unit-tests-daily/builds/1413

Please review:

@svc-review-gpt

`Prewarm simulator` ✕ 733e8eb

deployboard commented 3 hours ago

✨ ReviewGPT AI-Generated Code Review ✨

---

1. While this PR seems to make sense conceptually, it's also worth considering that Apple's Simulator app may not be running by the opening of the Simulator before the tests since killing the Simulator processes is part of the test workflow. The opening of the simulator might be ignored, or the workspace may have some unexpected state when the tests run. Moreover, the `open -b com.apple.iphonesimulator || true` command runs asynchronously, and there is no certainty the simulator app will be ready when the tests start running.

I propose a more robust way handling simulator prewarming by utilizing `simctl` command which provides direct capabilities for interacting with iOS simulators. For example, you can directly boot a simulator device using `xcrun simctl boot <device>`, where `<device>` is the id of the simulator, and you can ensure this operation completes before running the tests.

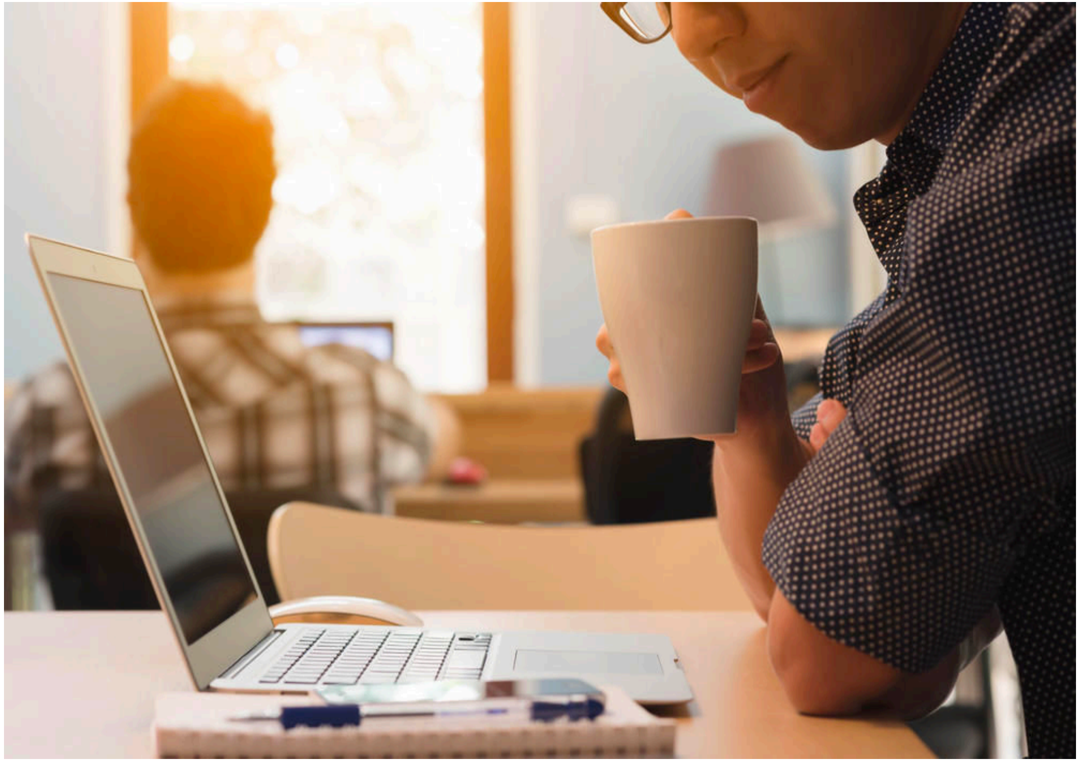
Here is my recommendation:

```
system_and_log 'pgrep Simulator | xargs kill || true'

# Prewarm simulator to avoid xcodebuild timeouts when trying to launch the simulator.
- system_and_log 'open -b com.apple.iphonesimulator || true'
+ system_and_log 'xcrun simctl list devices | grep "Shut Down" | cut -d "(" -f2 | cut -d ")" -f1 | xargs -I {}
```

This command lists all devices, filters the ones that are shut down, extracts the id, and boots them.

Note: This command attempts to boot all shut down simulators. If you want to boot a specific simulator, you would need to identify it by its identifier instead of the `grep "Shut Down"` command.



Thank you!