

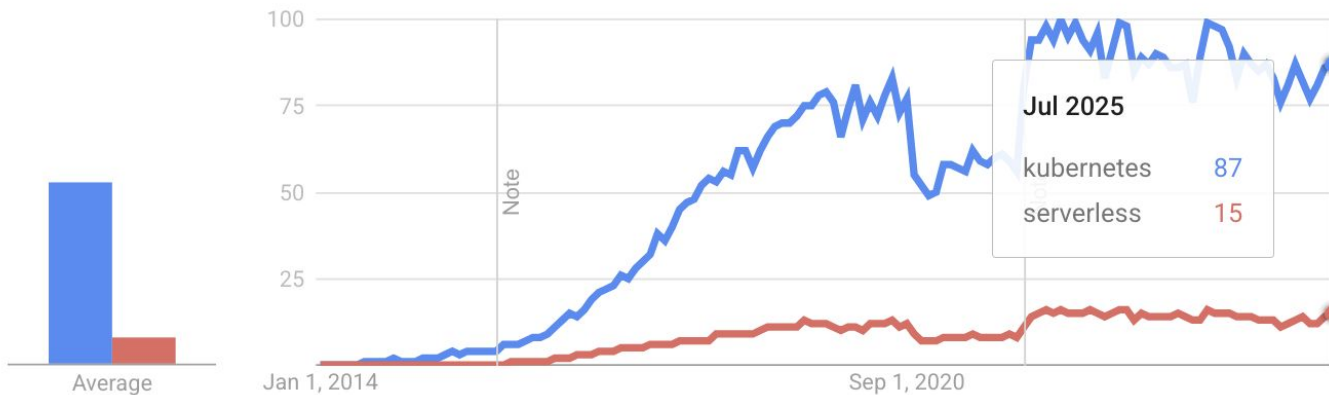
THE
**LIGHTWEIGHT
APPROACH**

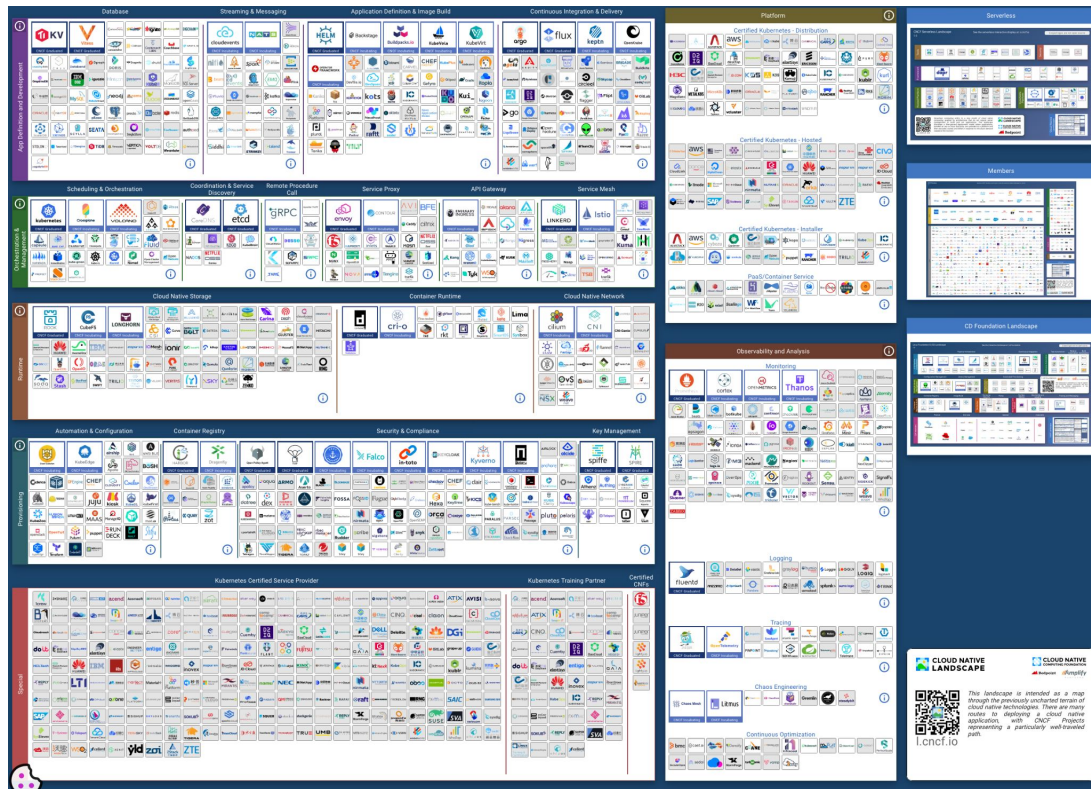
TO BUILDING INTERNAL
DEVELOPER PLATFORMS



82%

Interest over time ?



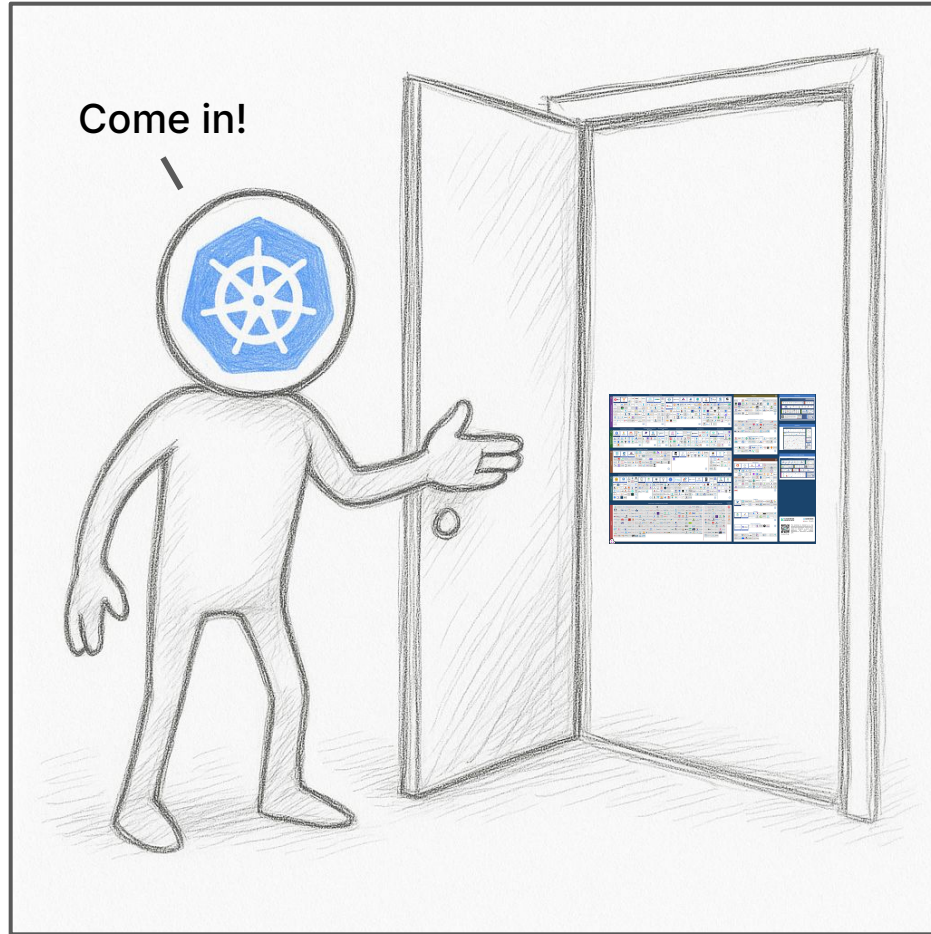


<https://landscape.cncf.io>

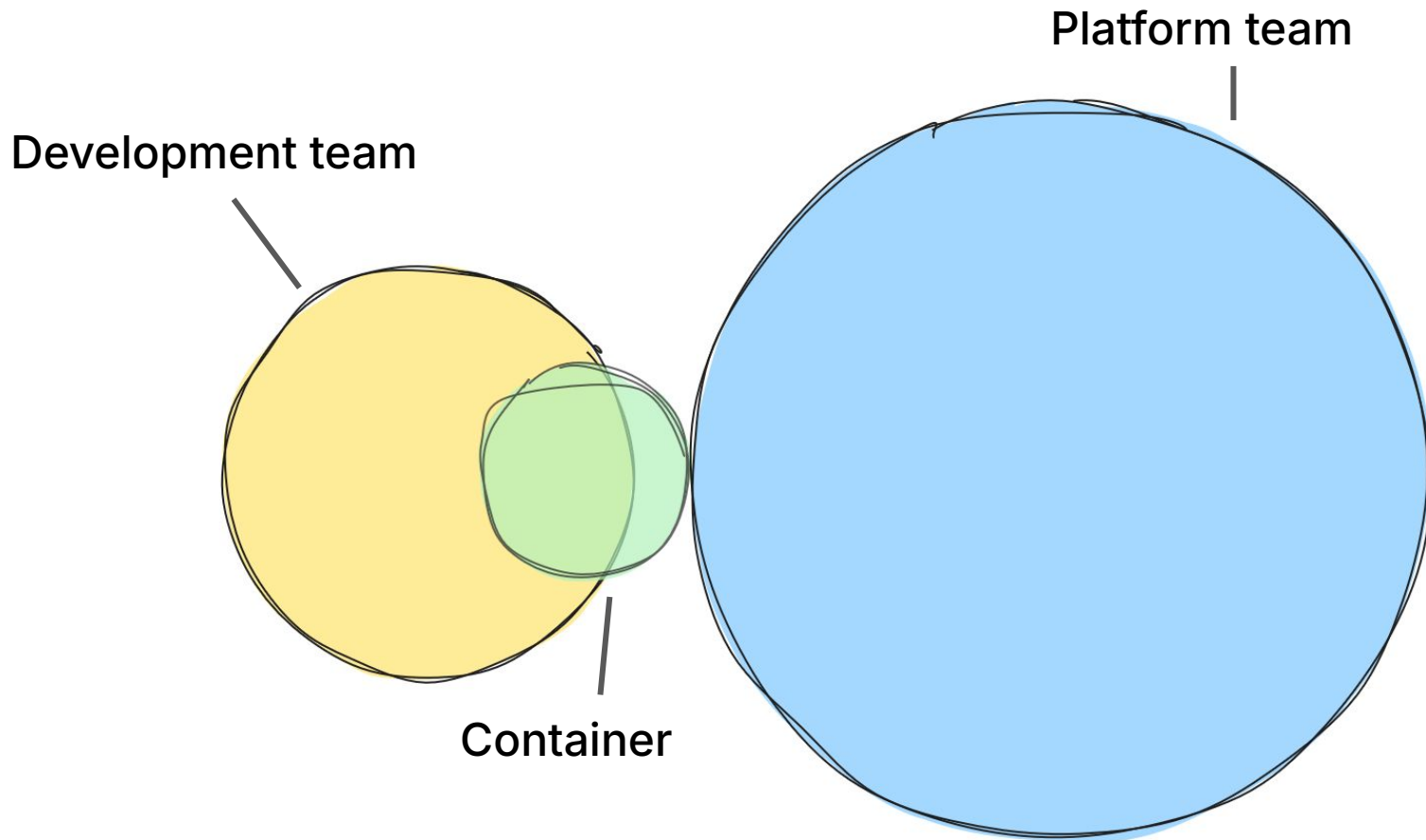
@erlendekern



**BE AFRAID OF
THE ENORMITY OF
THE POSSIBLE**



@erlendekern



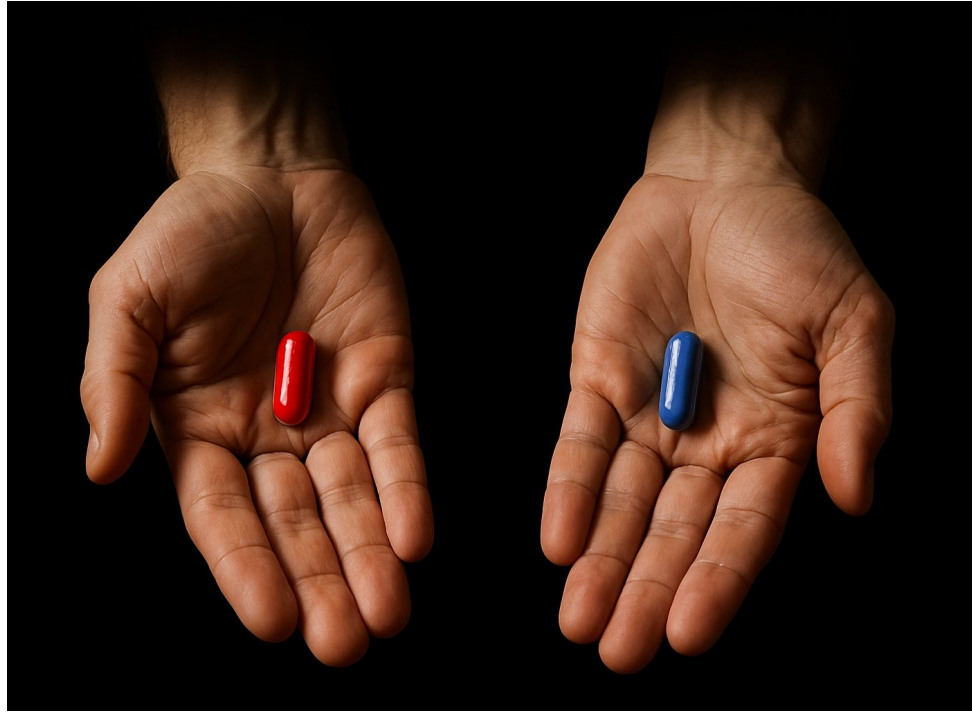
Inspired by devopstopologies.com

@erlendekern





Dev Ops



@erlendekern

Serverless is a State of Mind

The point is focus — that is the why of serverless



Ben Kehoe

Following ▾

12 min read · Mar 17, 2019

"Thinnest Viable Platform"



Real-world examples

The blueprint

Concluding thoughts



Erlend Ekern
AWS Community Builder
Cloud Architect

@erlendekern





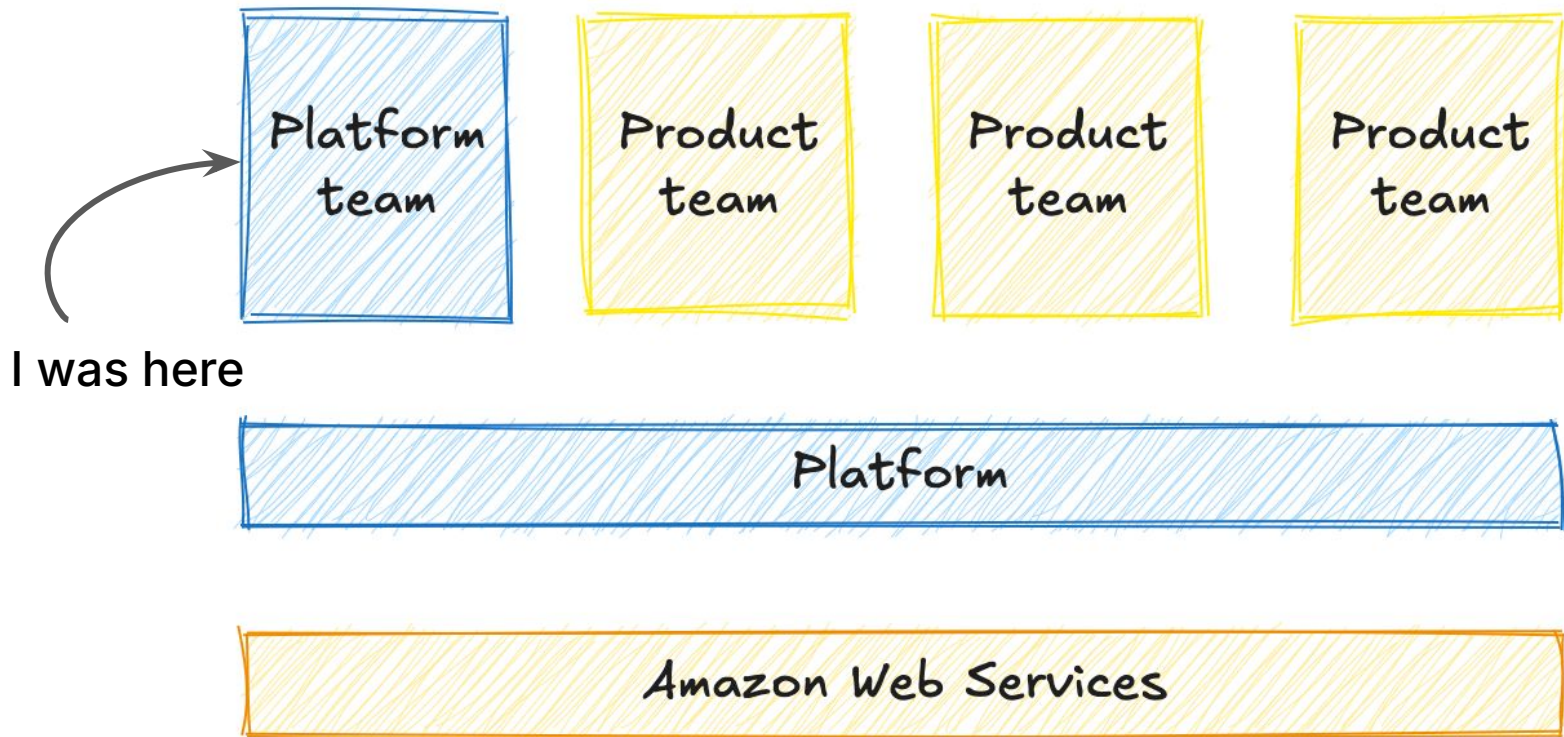
Liflig

@erlendekern

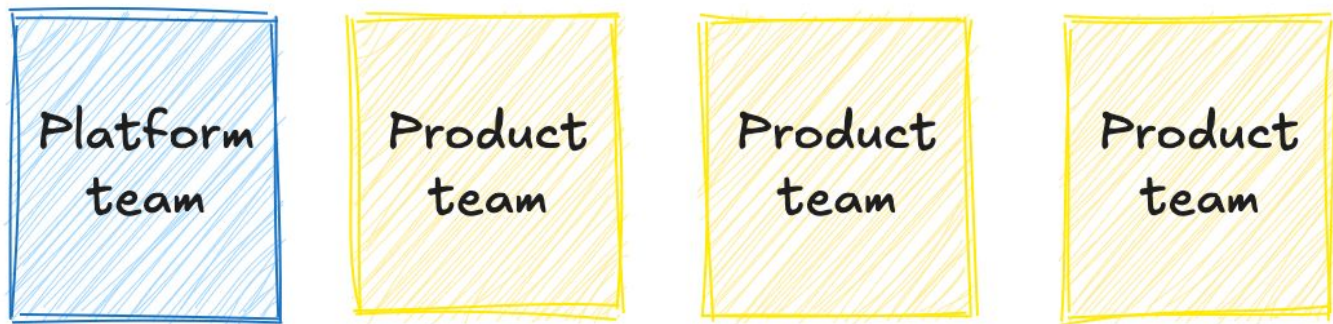
 Capra

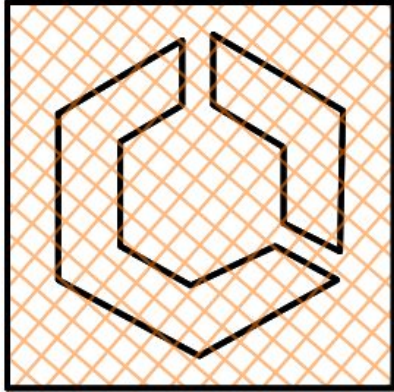
End-to-end responsible



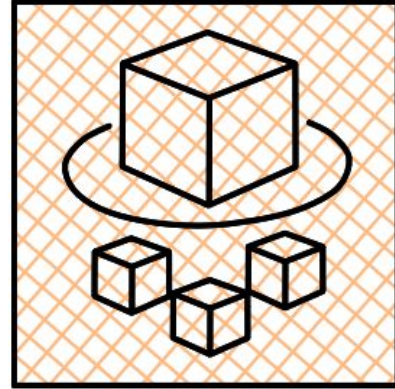


Applying a
serverless
mindset





ECS

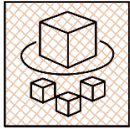


Fargate

Compute



ECS

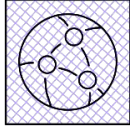


Fargate

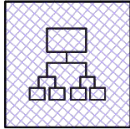


Lambda

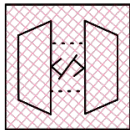
Front doors



CloudFront

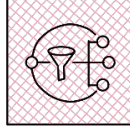


ALB

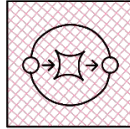


API Gateway

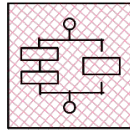
Integration



SNS

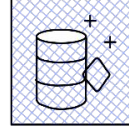


SQS

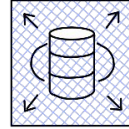


Step Functions

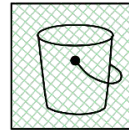
Storage



Aurora

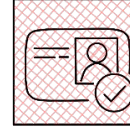


RDS



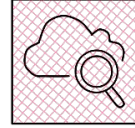
S3

Identity



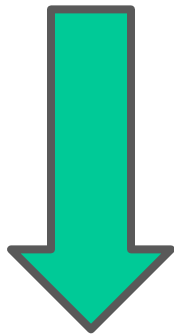
Cognito

Observability



CloudWatch





- ⚡ Onboarding to a production-ready foundation in **1 hour**
- 💪 Empowered teams that deployed **daily**
- 🧩 Lean platforms maintained by **2-3 people**

THE BLUEPRINT

You build it, you run it!

You build **it**, you run it!

@erlendekern



Limited
responsibility



"Stay inside
the container."

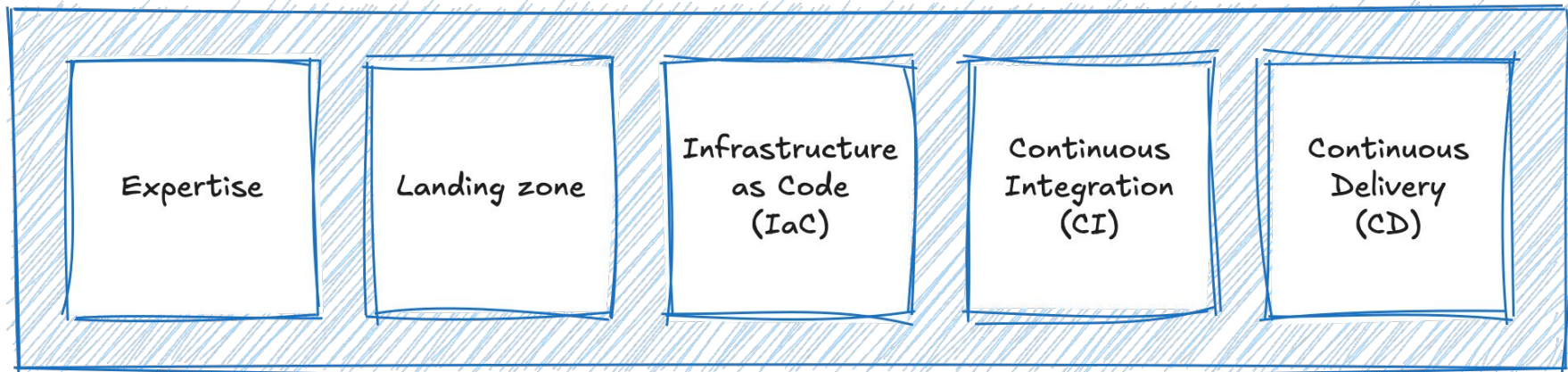


The lightweight
approach

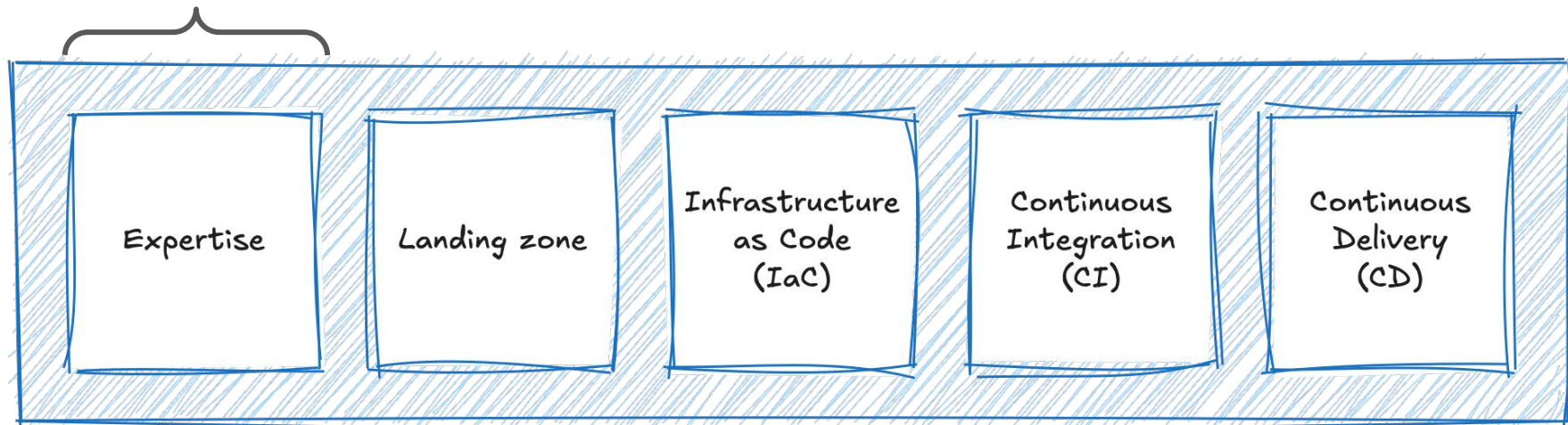
Excessive
responsibility



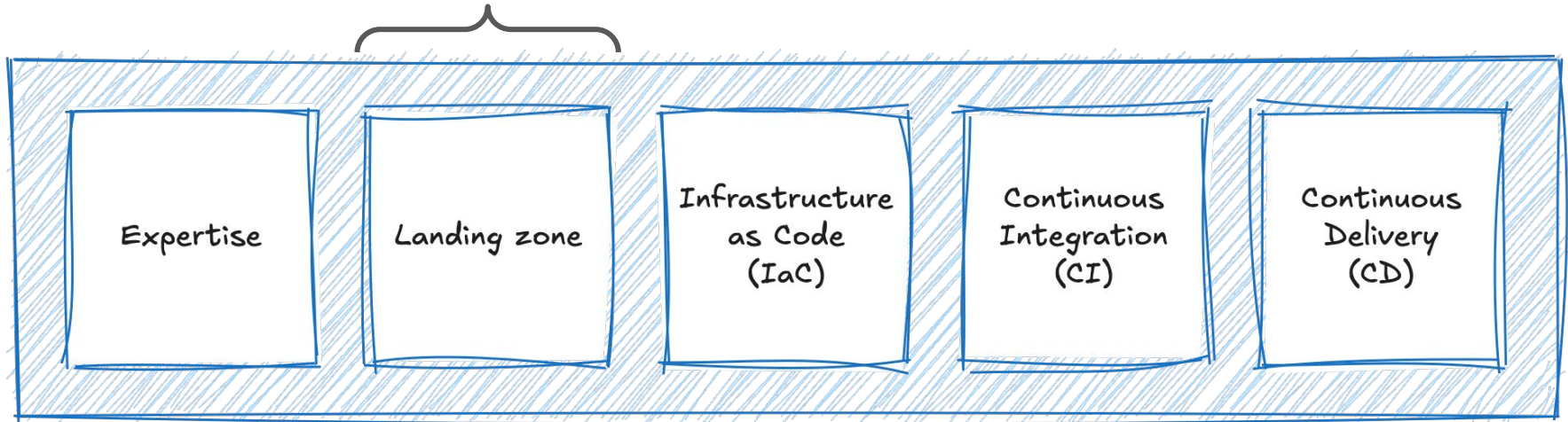
"Here's a cloud account.
Good luck."



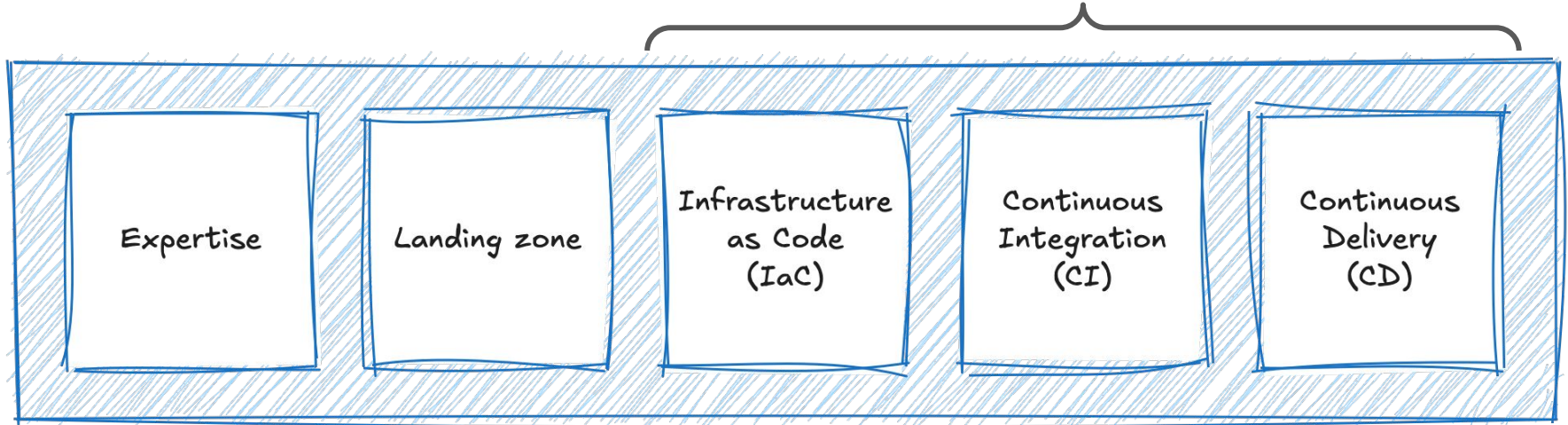
Support and
documentation



Production-ready cloud environments



Tools and code libraries





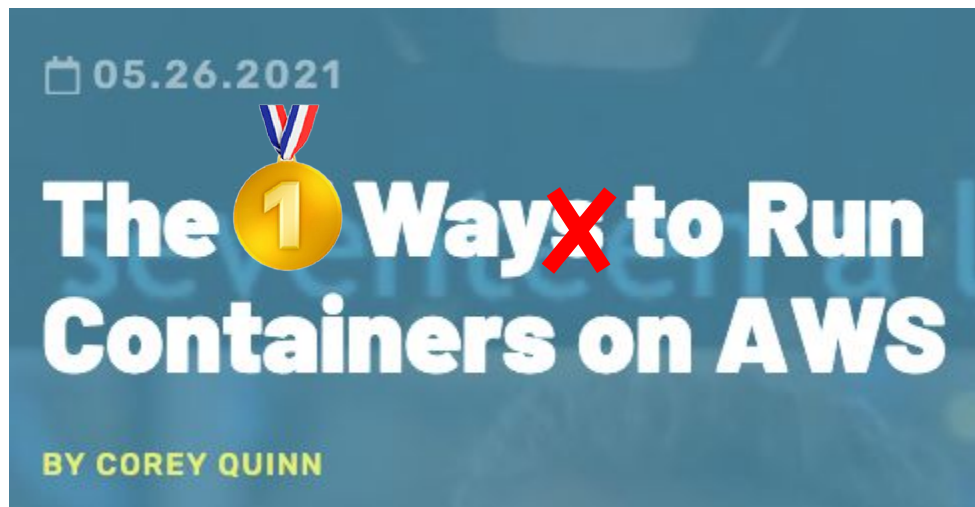
Expertise

Expertise



@erlendekern

Expertise



<https://www.lastweekinaws.com/blog/the-17-ways-to-run-containers-on-aws>

@erlendekern







Landing zone

management

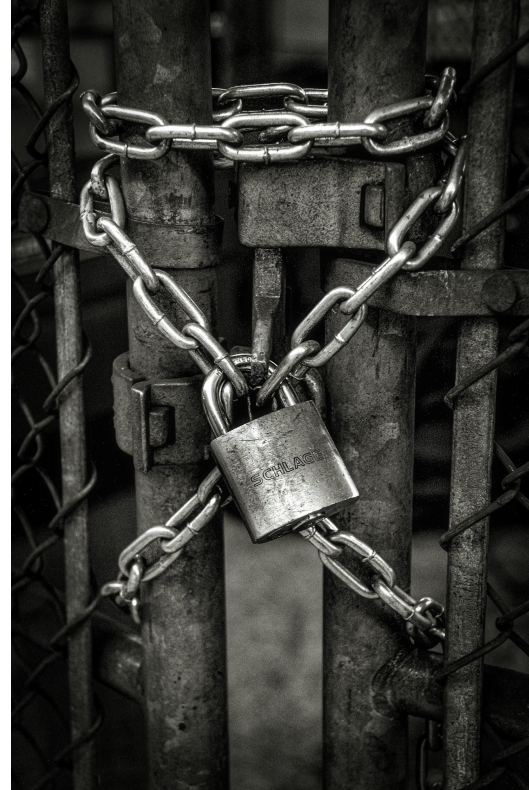
governance

@erlendekern

Landing zone

management

governance



@erlendekern

Landing zone



"We might've overdone it with the negations..."

management

governance

```
{
  "Effect": "Deny",
  "NotAction": [ /* ... */ ],
  "Condition": {
    "StringNotEquals": { /* ... */ },
    "ArnNotLike": { /* ... */ }
  },
  "Resource": "*"
}
```

Landing zone

Bounded context

"Facilitating" account

Workload accounts

service

management

development

staging

governance

production



Landing zone

"AWS Control Tower offers a straightforward way to set up and govern an AWS multi-account environment"



Landing zone

"AWS Control Tower offers a ~~straightforward~~ way to set up and govern an AWS multi-account environment"

Landing zone

HOW STANDARDS PROLIFERATE: (SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)

AWS Landing Zone
AWS Control Tower
AWS Account Factory for Terraform
AWS Landing Zone Accelerator
org-formation
superworker
...

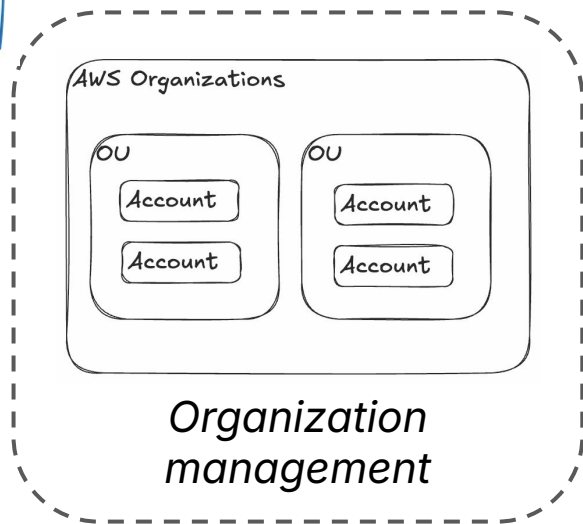


<https://xkcd.com/927>

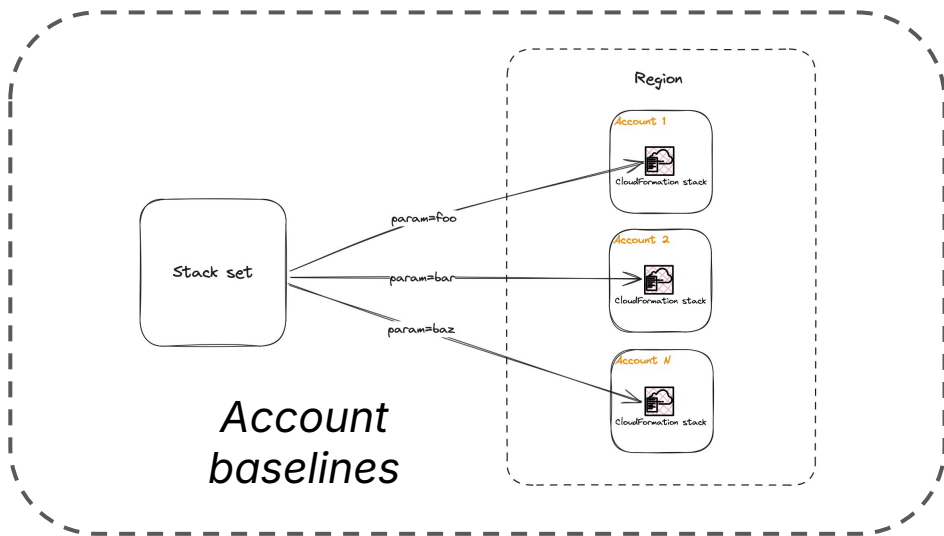
@erlendekern

 Capra

Landing zone



Terraform



CloudFormation StackSets

Landing zone

```
import { /* ... */ }  
moved { /* ... */ }
```

No changes. Your infrastructure matches the configuration.

Landing zone



primeharbor / org-kickstart

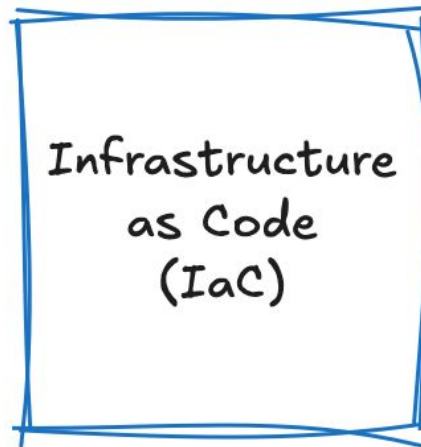
blog.ekern.me

Feb 27, 2025

What's the deal with AWS CloudFormation StackSets?

@erlendekern





Infrastructure
as Code
(IaC)

AWS CloudFormation

Terraform

Serverless Framework

AWS SAM

Pulumi

AWS CDK

cdktf / cdktn

SST

Alchemy

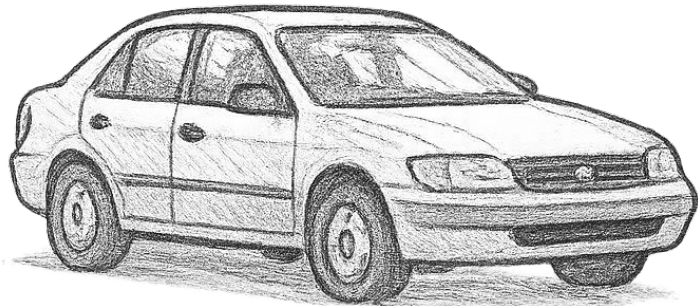
Swamp

 Capra

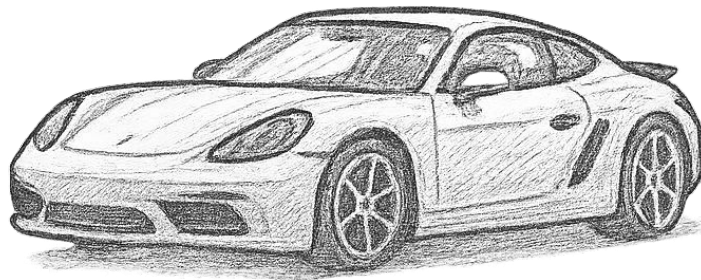
???

@erlendekern

Infrastructure
as Code
(IaC)



Terraform



AWS CDK

Infrastructure
as Code
(IaC)

Keep It Simple™

@erlendekern



Infrastructure
as Code
(IaC)

Thin **library** on top of cloud primitives

@erlendekern



Infrastructure as Code (IaC)



ECS



CloudFront



SNS



Aurora



Cognito



CloudWatch



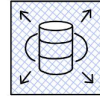
Fargate



ALB



SQS



RDS



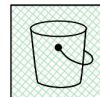
Lambda



API Gateway



Step Functions



S3

Infrastructure as Code (IaC)



ECS



cloudFront



SNS



Aurora



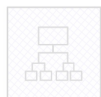
Cognito



cloudWatch



Fargate



ALB



SQS



RDS



Lambda



API Gateway



Step Functions



S3

Infrastructure
as Code
(IaC)

Library

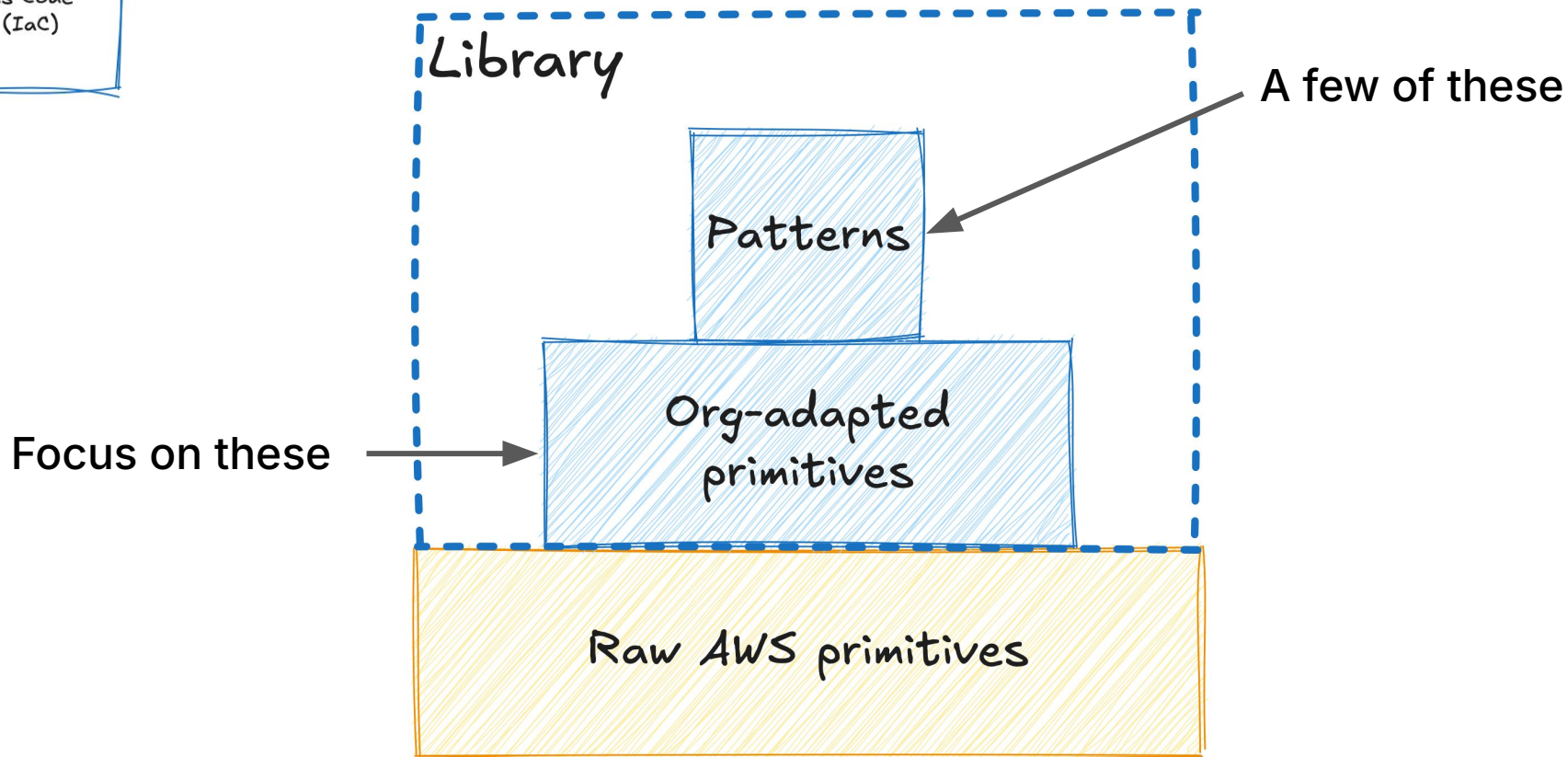
Org-adapted
primitives

Raw AWS primitives

Focus on these

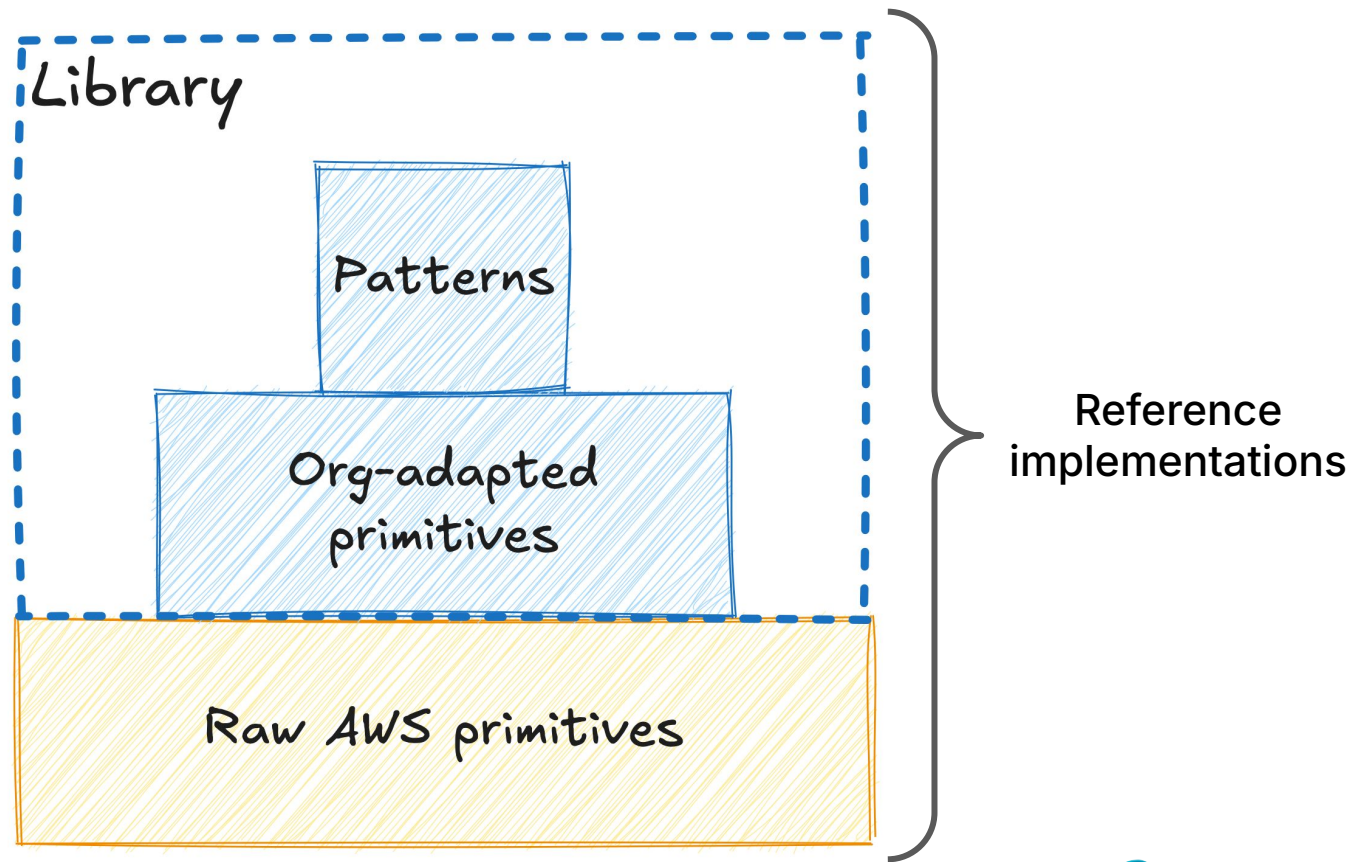


Infrastructure
as Code
(IaC)



@erlendekern

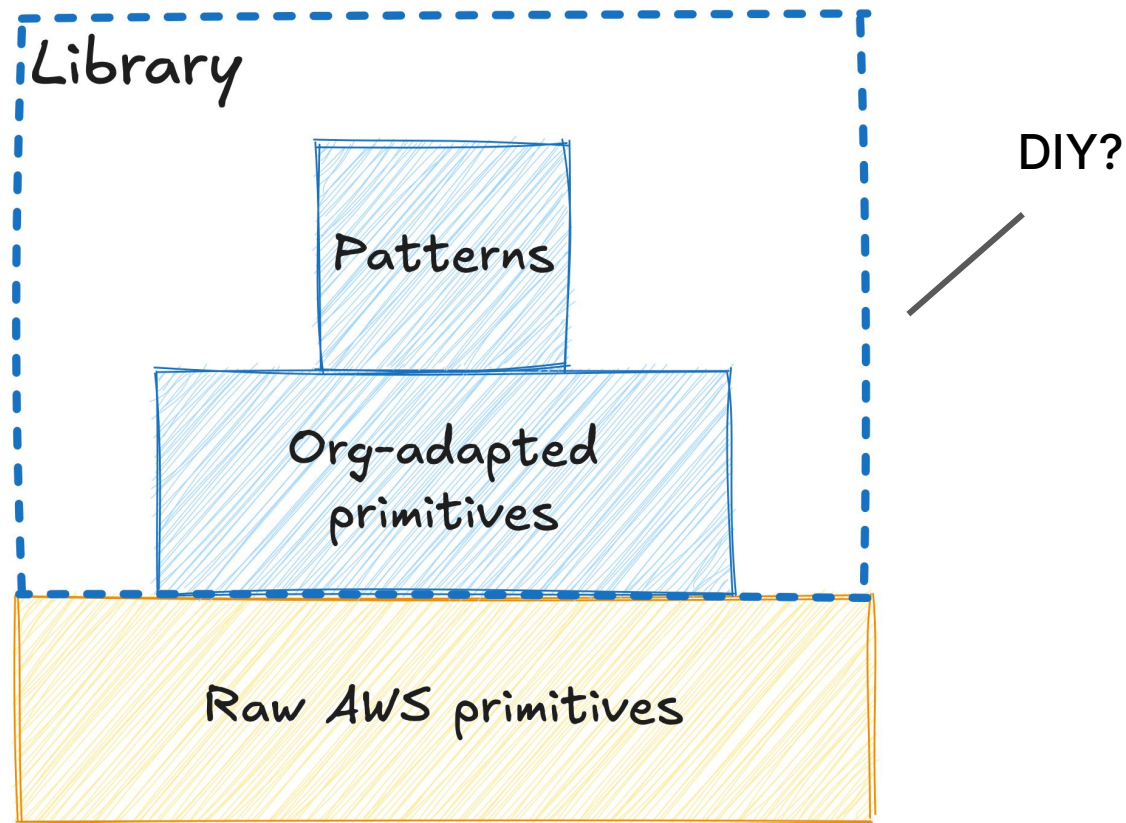
Infrastructure
as Code
(IaC)



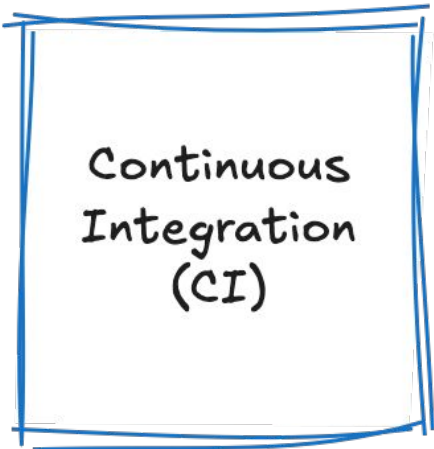
@erlendekern



Infrastructure
as Code
(IaC)



@erlendekern



Continuous
Integration
(CI)



Continuous
Delivery
(CD)

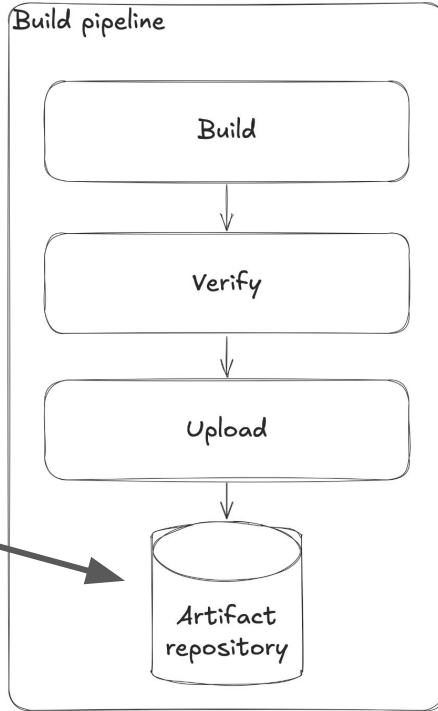
**"If it hurts, do it more frequently,
and bring the pain forward"**

— Jez Humble & David Farley, Continuous Delivery



Continuous
Integration
(CI)

Continuous
Delivery
(CD)



@erlendekern

Continuous
Integration
(CI)

Continuous
Delivery
(CD)

Build pipeline

Build

Verify

Upload

Artifact
repository

Trigger

Deployment pipeline

Deploy to
development

Deploy to
staging

Run tests

Deploy to
production

Run tests

```
$ terraform apply  
$ pulumi up  
$ cdk deploy
```

App deploy
through infra

@erlendekern



Continuous
Integration
(CI)

Continuous
Delivery
(CD)

Decoupled pipeline

Build pipeline

Build

Verify

Upload

Artifact
repository

Trigger

Deployment pipeline

Deploy to
development

Deploy to
staging

Run tests

Deploy to
production

Run tests

Continuous
Integration
(CI)

Continuous
Delivery
(CD)



@erlendekern



Expertise

Landing zone

Infrastructure
as Code
(IaC)

Continuous
Integration
(CI)

Continuous
Delivery
(CD)

Starter kit

Hey, could you help us out with cloud environments for *foobar*?



Sure can do!



Hi 🖐️

The AWS account set for bounded context **foobar** has been created, and the accounts are ready to use:

- *foobar-service* 123456789012
- *foobar-dev* 234567890123
- *foobar-staging* 345678901234
- *foobar-prod* 456789012345

Each account has its own DNS zone under `foobar.example.com` (e.g., `prod.foobar.example.com`).

Check out our [Getting started](#) guide to configure access to AWS and get started with your very own Infrastructure as Code (IaC) repository.

You'll be up and running on a production-ready, multi-account foundation in no time 🚀

If you have any questions at all, don't hesitate to reach out in [#platform-support](#)!

aws-starter-kit-template Private template

Watch **0** Fork **0** Star **0** Use this template

main 2 Branches 0 Tags Code

stekern initial commit 3e80cc1 · 2 minutes ago 1 Commit

.github/workflows	initial commit	2 minutes ago
examples	initial commit	2 minutes ago
src	initial commit	2 minutes ago
.gitignore	initial commit	2 minutes ago
README.md	initial commit	2 minutes ago
apply-template.sh	initial commit	2 minutes ago
cdk.json	initial commit	2 minutes ago
package-lock.json	initial commit	2 minutes ago
package.json	initial commit	2 minutes ago
tsconfig.json	initial commit	2 minutes ago

About

Get started with a production-ready, multi-account foundation in AWS in no time 🚀

- [Readme](#)
- [Activity](#)
- 0 stars
- 0 watching
- 0 forks

Languages

- TypeScript 100.0%

@erlendekern



Usage

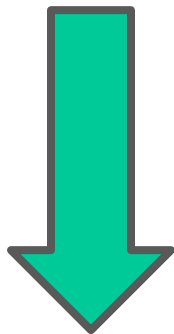
1. Create a new repository using this template by clicking here: ✨ [Use template](#) ✨
2. Name the repository using the following convention `<bounded-context>-infra` (e.g., `example-infra`).
3. Click **Create repository**, clone the repository to your local machine, and follow the steps below 📄
4. Install the npm packages: `$ npm ci`
5. Log in to the `service` account of the AWS account set: `$ assume <bounded-context>-service-admin`
6. Replace placeholders and create the deployment pipeline by running the `apply-template.sh` script:

► Show option details and example values



```
$ ./apply-template.sh \
--bounded-context    "<bounded-context>" \
--dns-zone           "<dns-zone>" \
--default-aws-region "<default-aws-region>" \
--service-account-id "<service-account-id>" \
--dev-account-id     "<dev-account-id>" \
--staging-account-id "<staging-account-id>" \
--prod-account-id    "<prod-account-id>"
```

7. Commit your changes and push them to trunk
8. Let the CI/CD pipeline take care of the rest 😎



- ★ Their own IaC codebase covering all environments
- ★ Their own deployment pipelines, ready-to-go
- ★ Full access to their AWS accounts
- ★ DNS zones and TLS certificates
- ★ A demo application in each environment
- ★ Documentation on where to go next

The perfect platform?

No, but an **effective** one



Challenges

Skills and maturity

Ownership boundaries

Variation and consistency

Shadow platforms

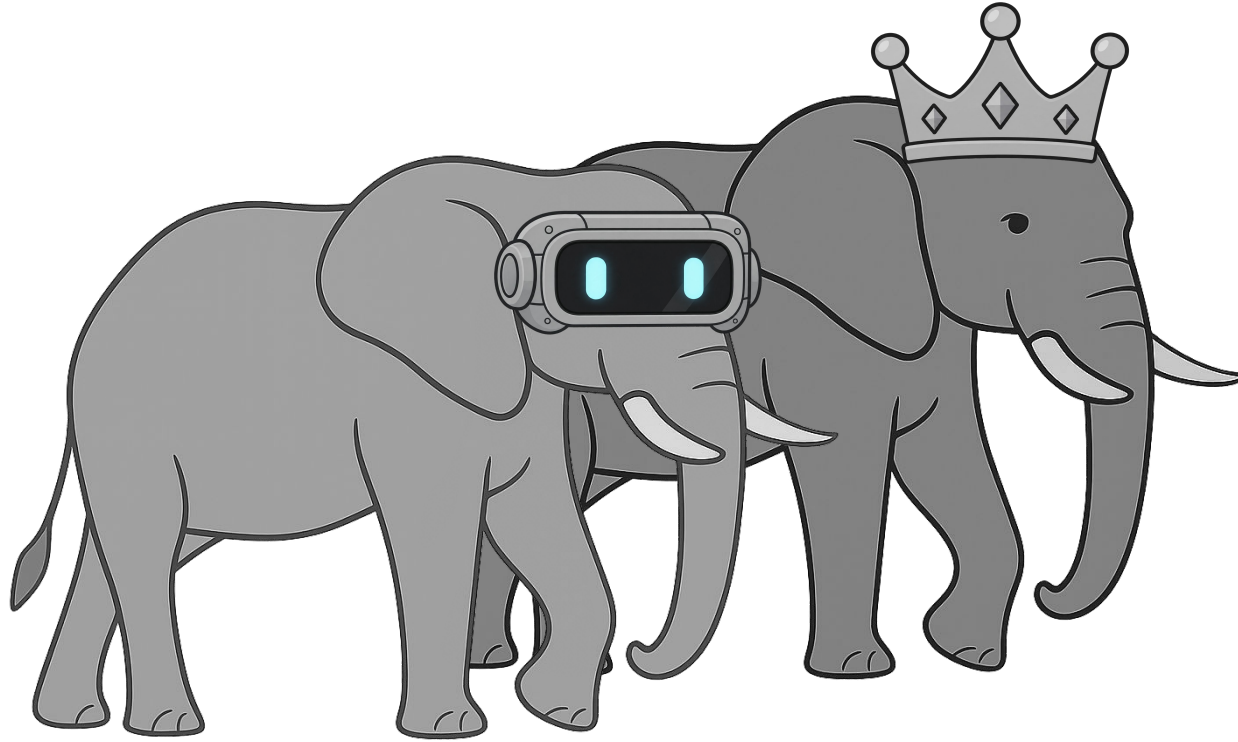
tl;dr

Enablement > restriction

Outcome > technology

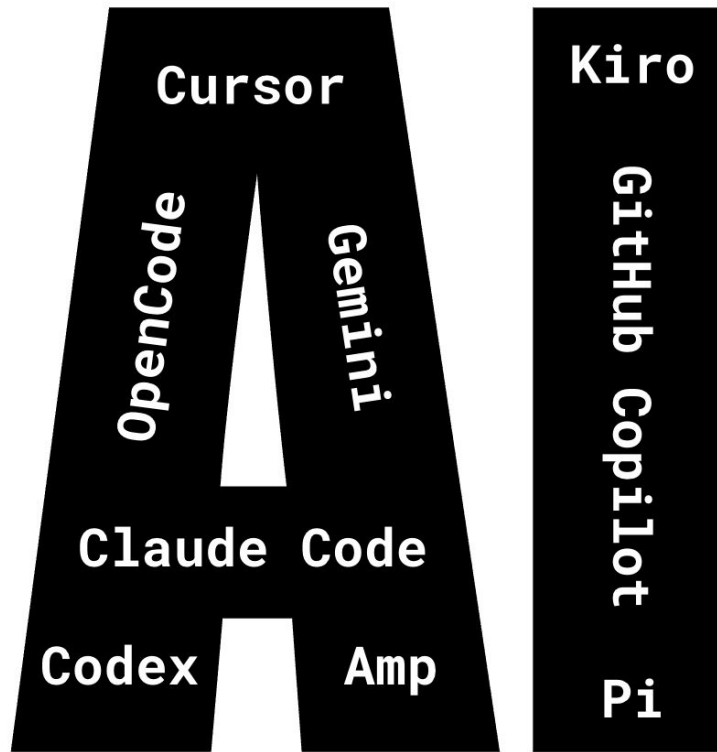
Constraints > flexibility

Composition > abstraction



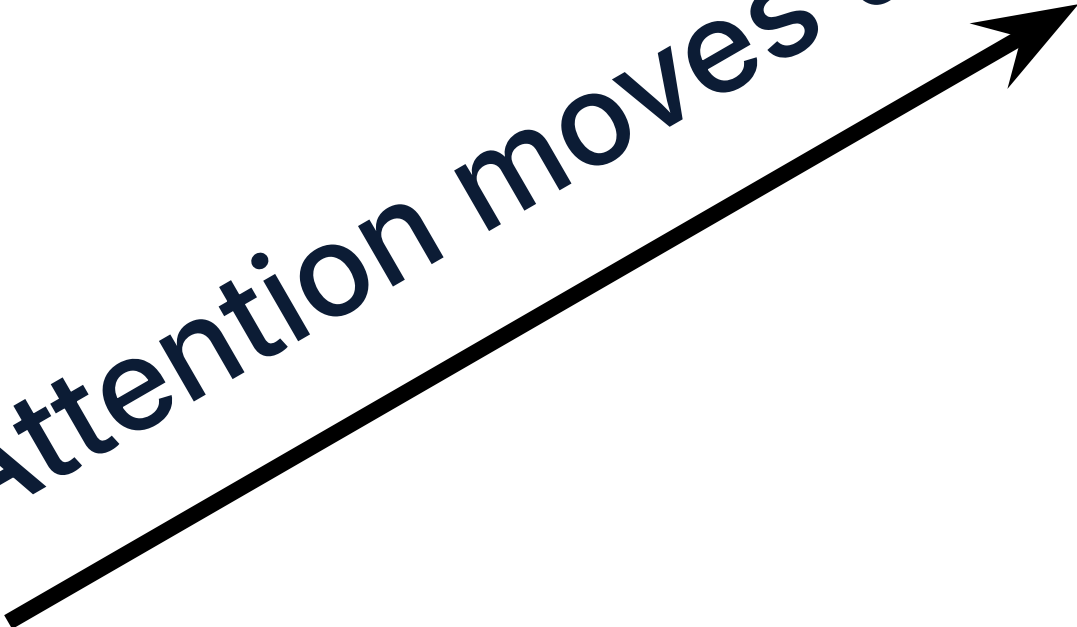
@erlendekern





AI is an **AMPLIFIER**

Attention moves up

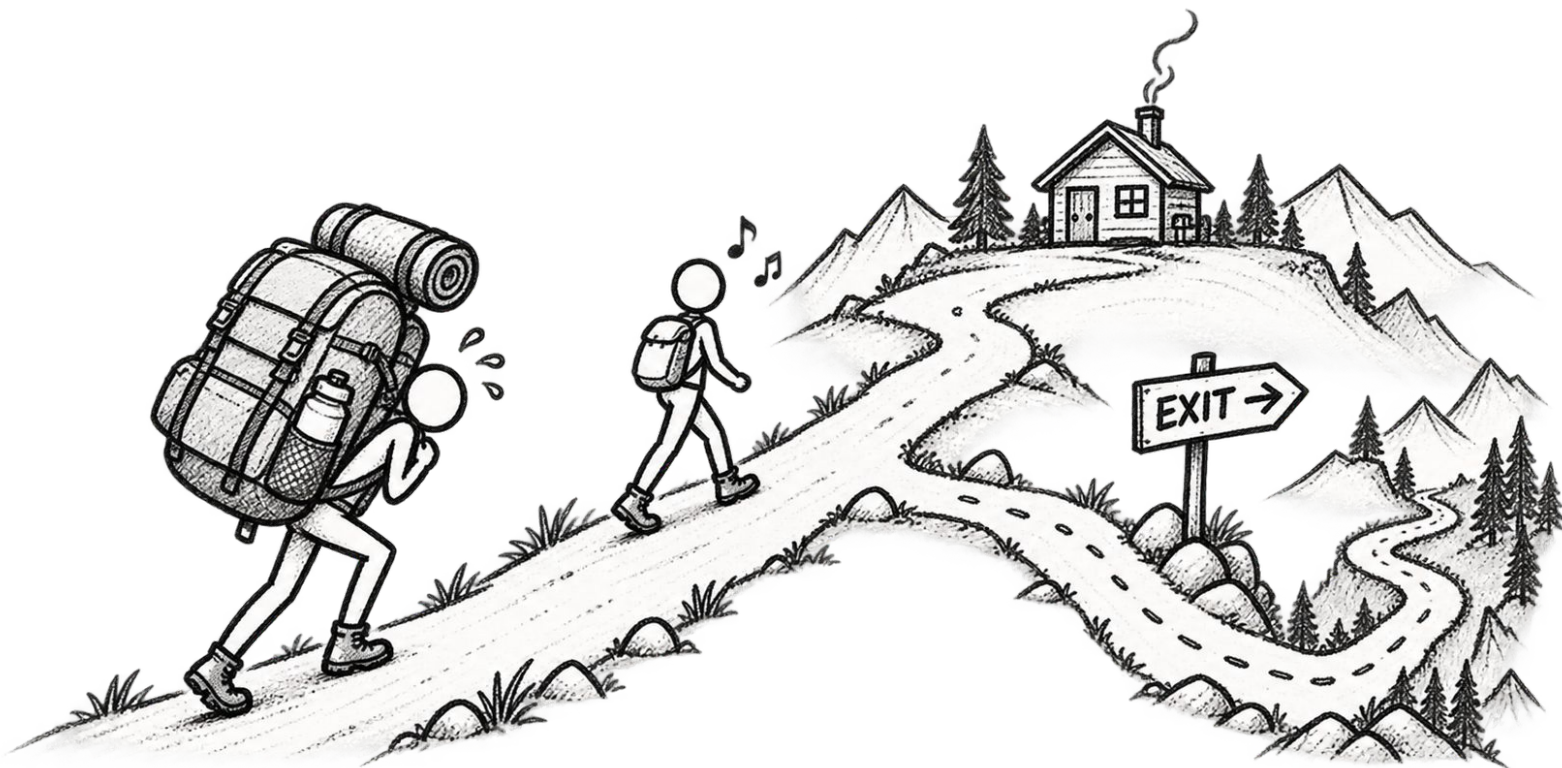


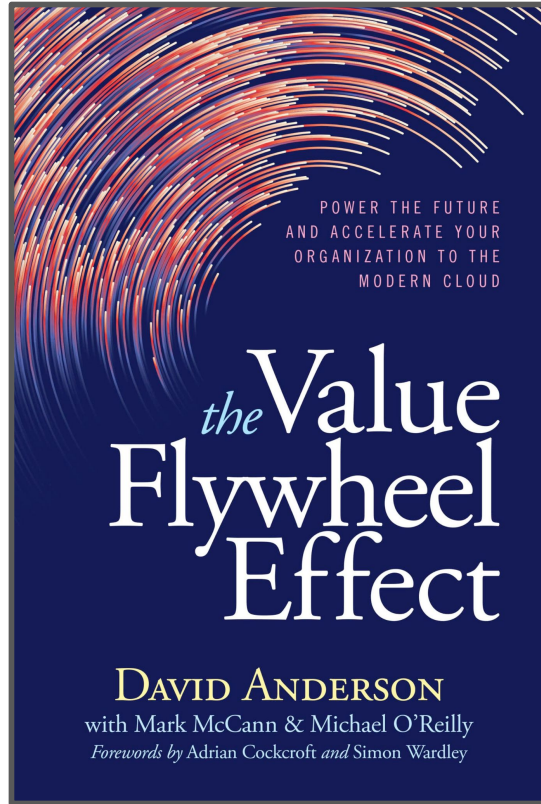
Platforms become more important

Code is [still] a **liability**



@erlendekern





@erlendekern





@erlendekern



Thank you!

Slides available at talks.ekern.me



ekern.me

linkedin.com/in/erlendekern

twitter.com/erlendekern

@erlendekern

