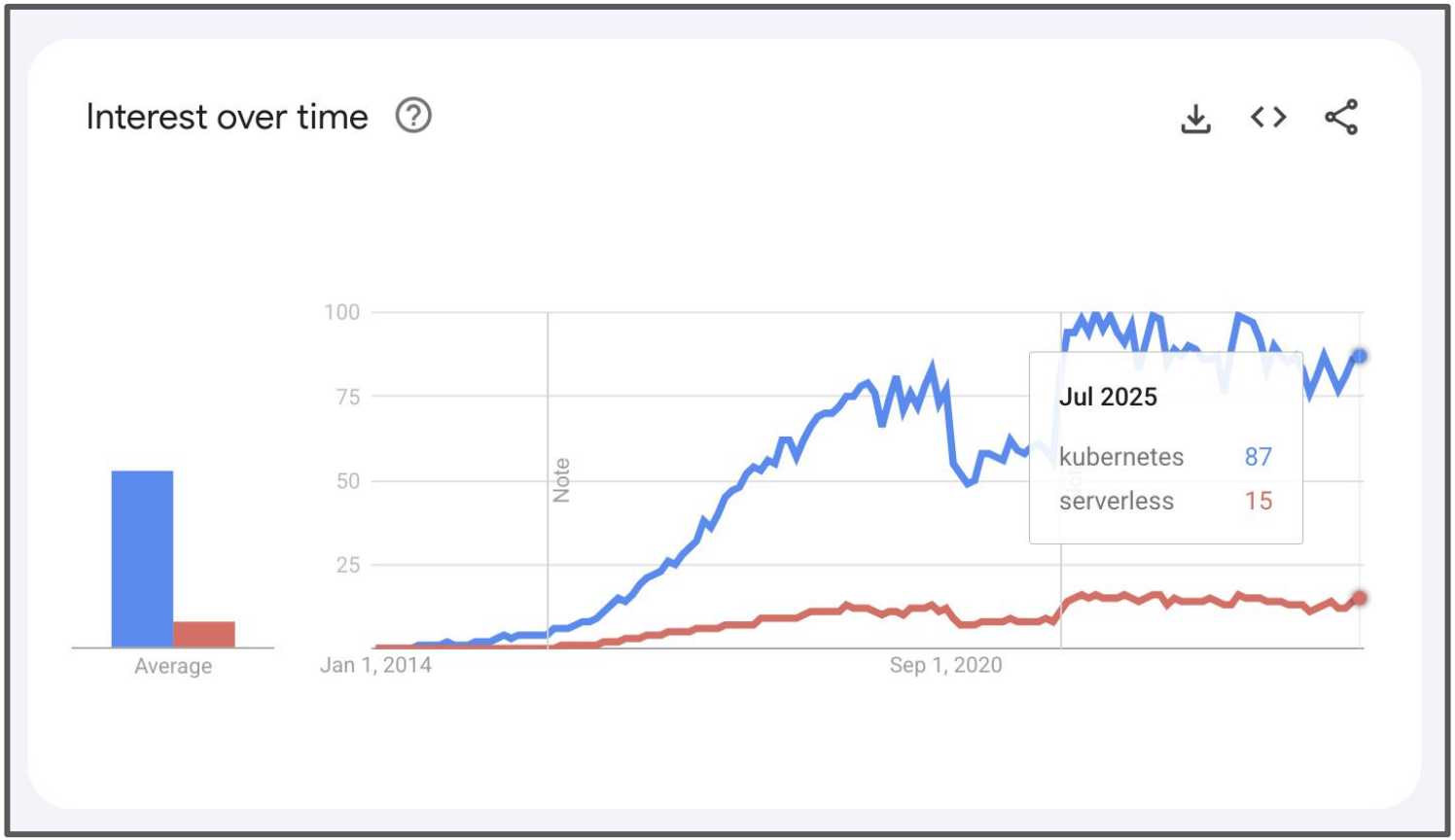
The lightweight approach to building internal developer platforms

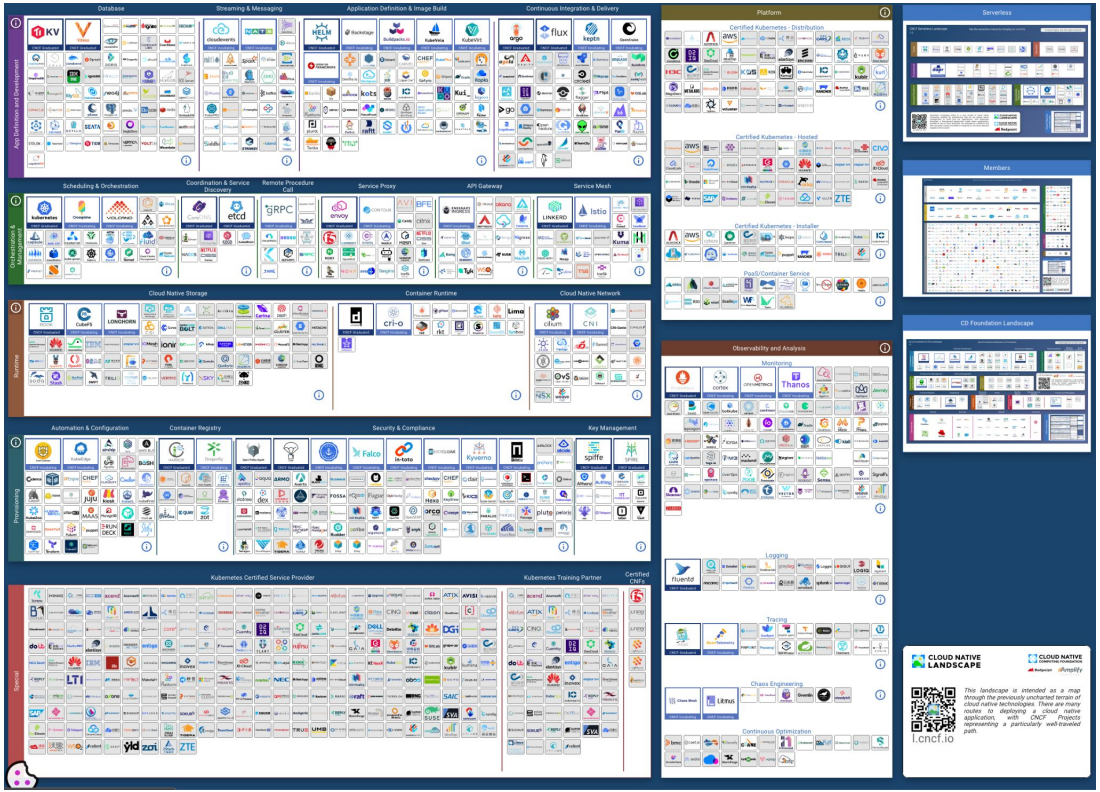
Erlend Ekern
AWS Community Builder
Cloud Architect

82%



@erlendekern

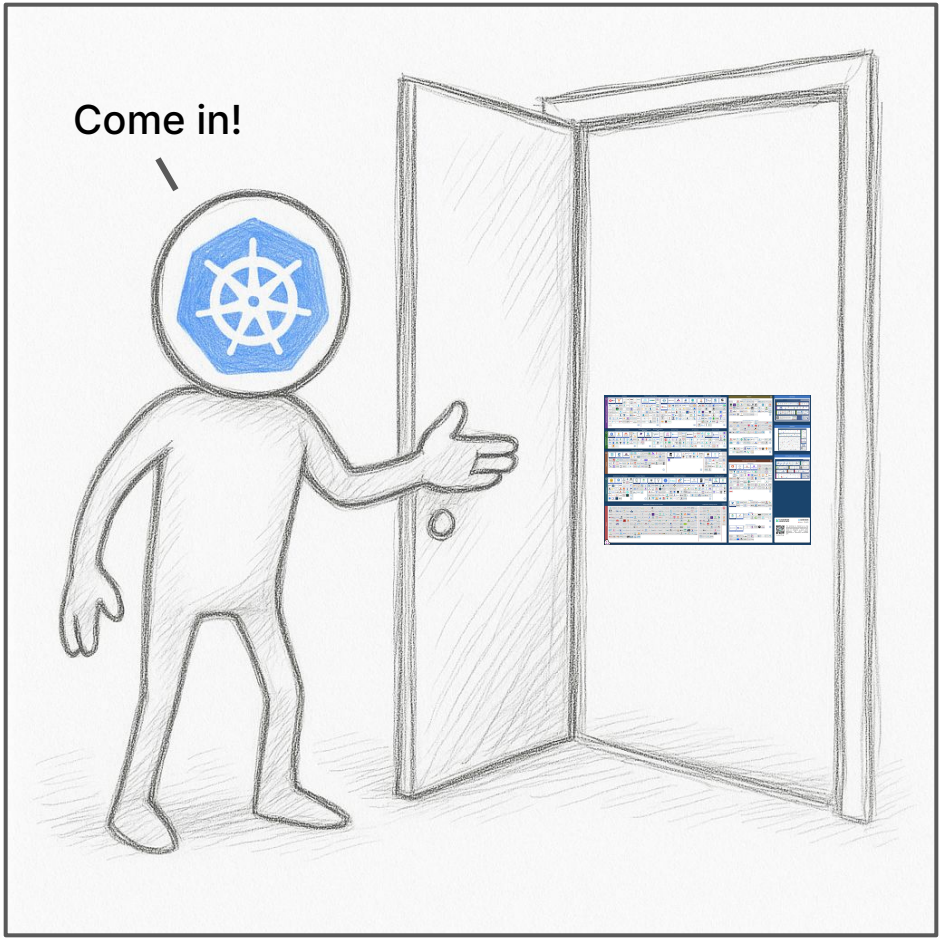
Capra



https://landscape.cncf.io

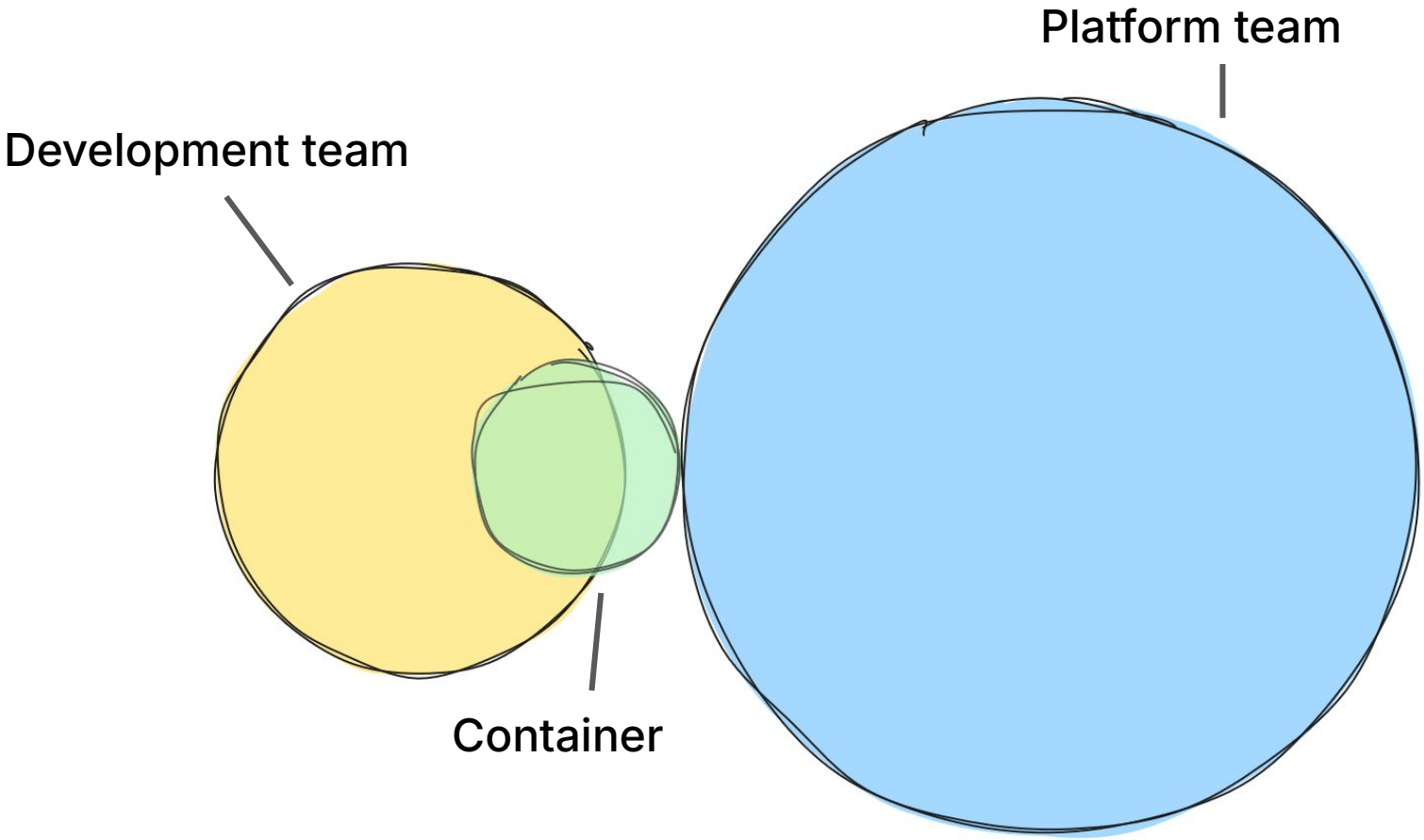
@erlendekern

Capra



@erlendekern

Capra



Inspired by devopstopologies.com

@erlendekern

Capra

@erlendekern

Capra

Dev 💥 Ops

@erlendekern

Capra

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

@erlendekern

Capra

"Thinnest Viable Platform"

Real-world examples

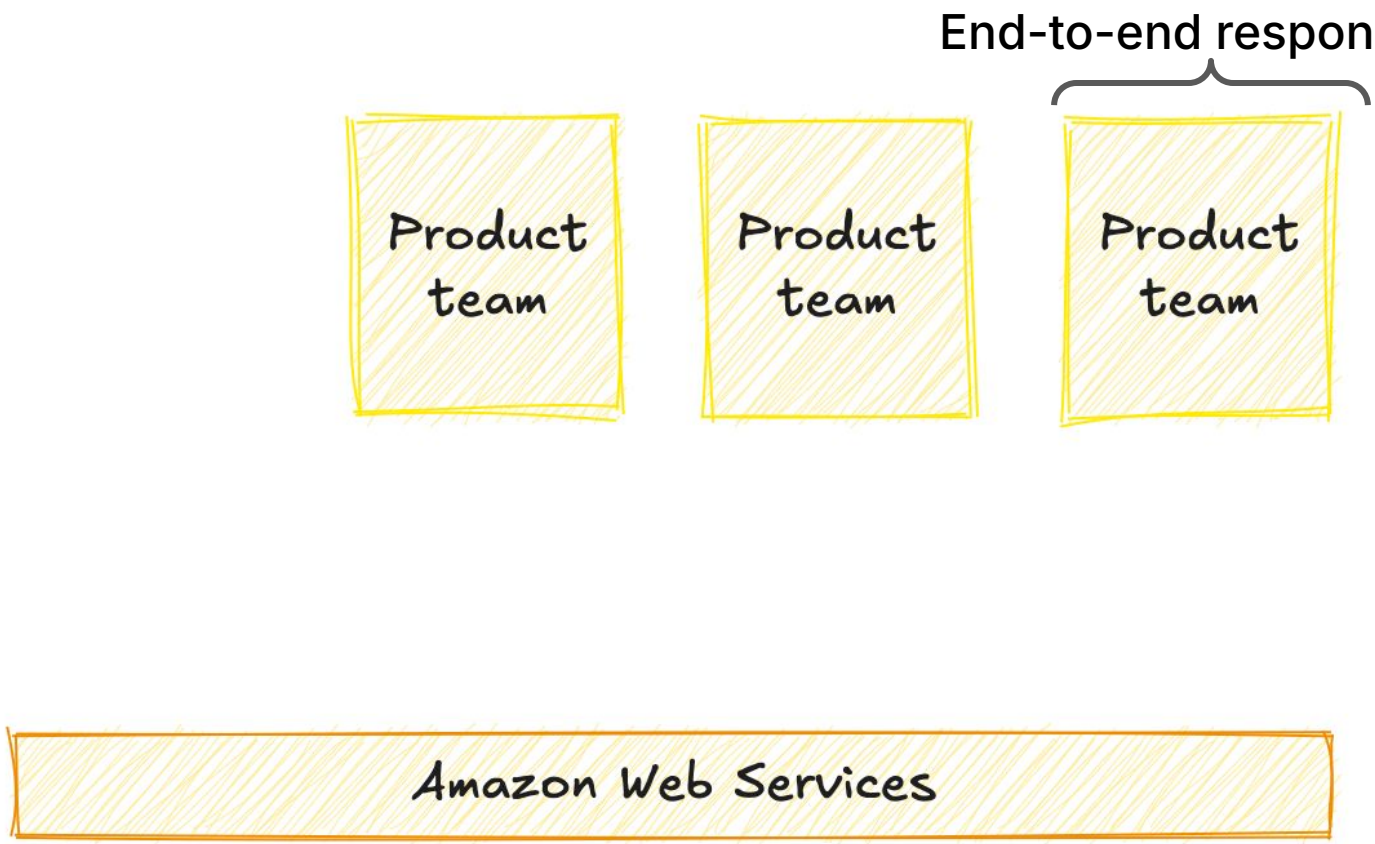
The blueprint

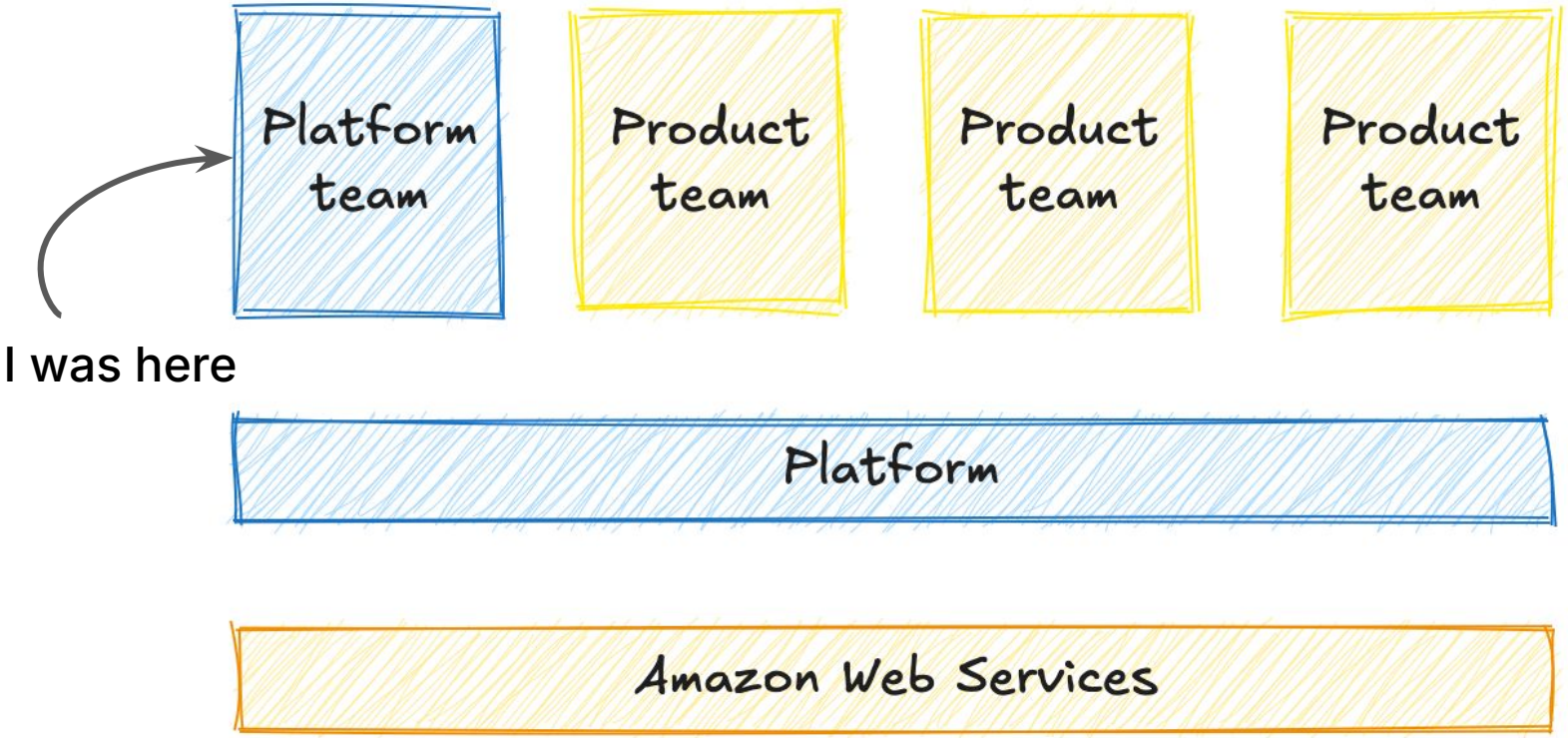
Concluding thoughts

Liflig

@erlendekern

Capra


End-to-end responsible
Product team
Product team
Product team
Amazon Web Services

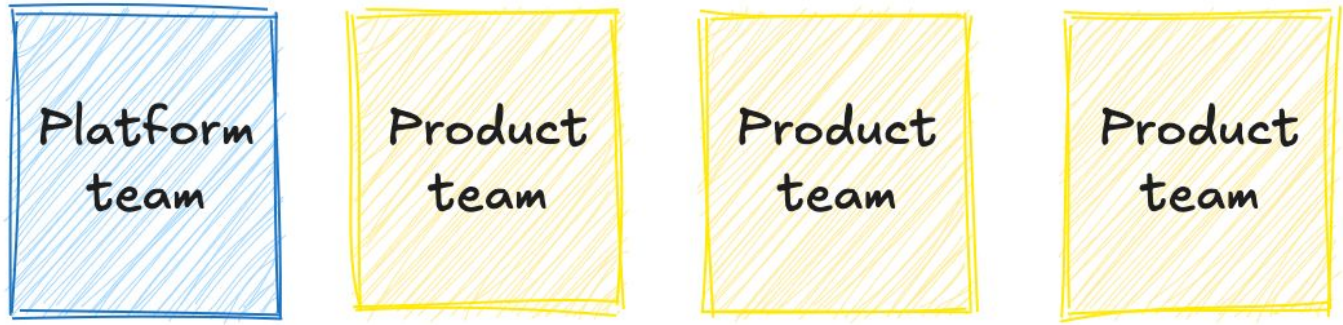


@erlendekern

Capra

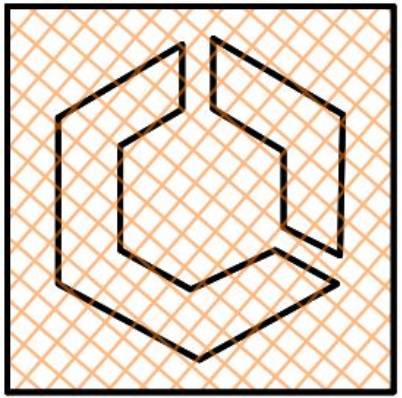
Hmm.. There's probably a managed service for this!

Applying a serverless mindset

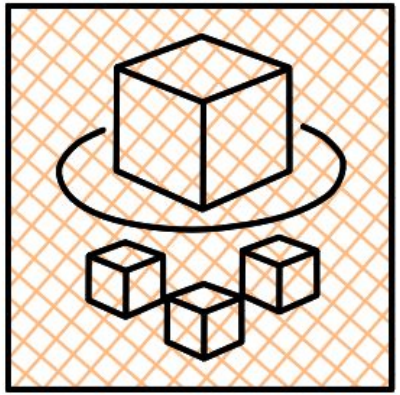


@erlendekern

Capra



ECS



Fargate

ECS

CloudFront

SNS

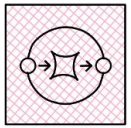
Aurora

Cognito

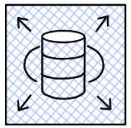
CloudWatch

Fargate

ALB



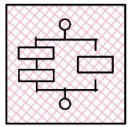
SQS



RDS

Lambda

API Gateway

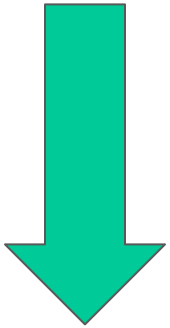


Step Functions

S3

@erlendekern

Capra



⚡ Onboarding to a production-ready foundation in **1 hour**

💪 Empowered teams that deployed **daily**

🧩 Lean platforms maintained by **2-3 people**

The blueprint

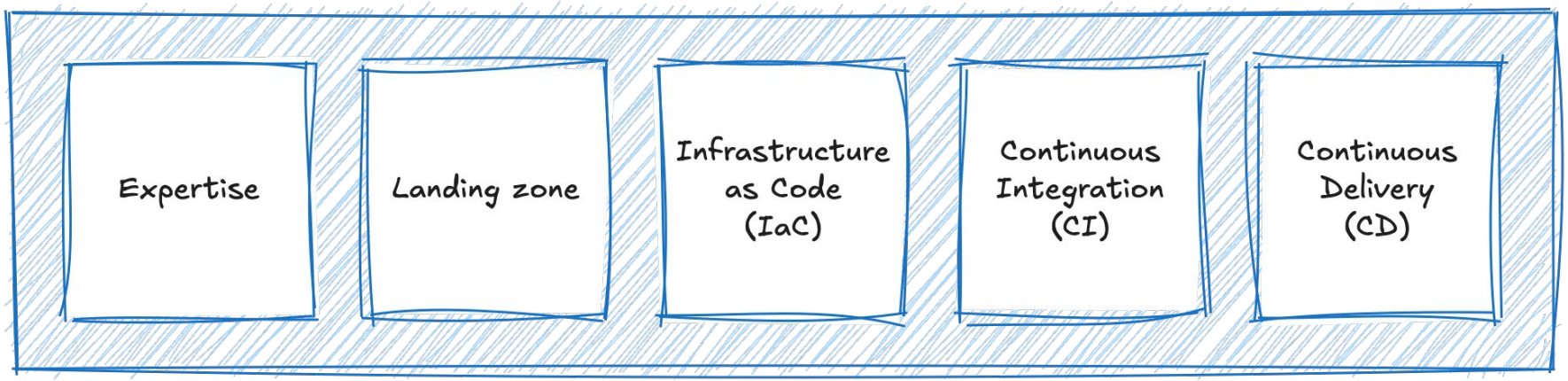
You build it, you run it!

@erlendekern

Capra

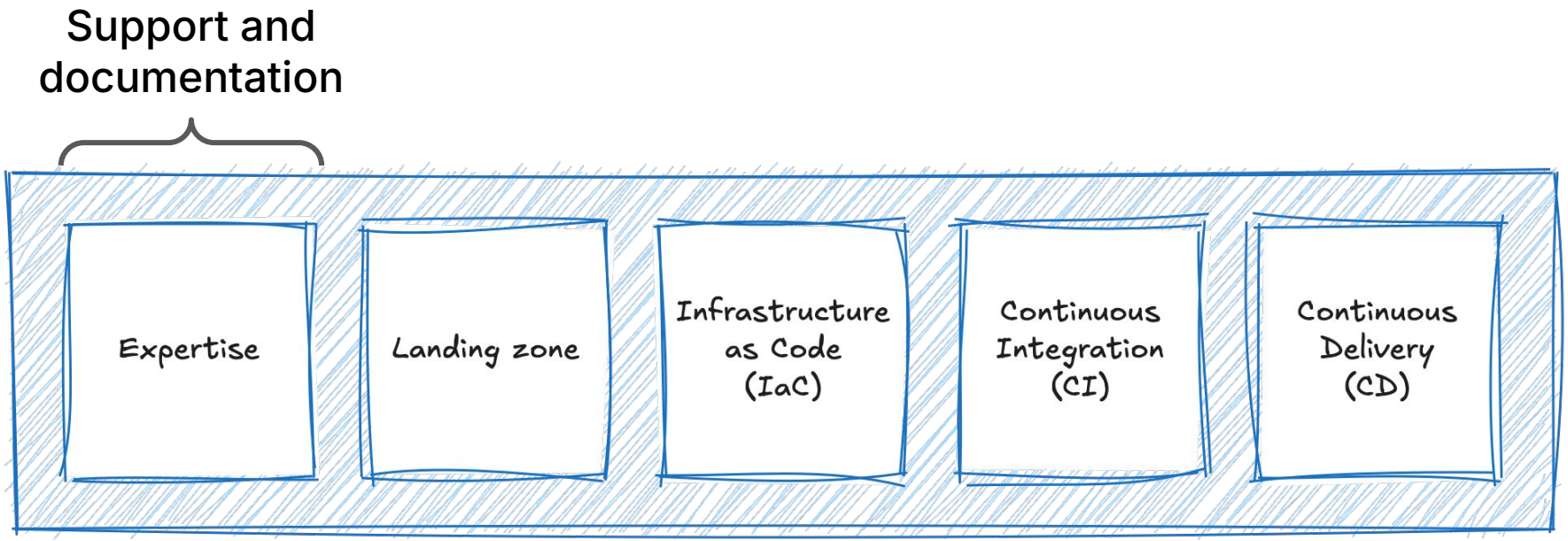
@erlendekern

Capra

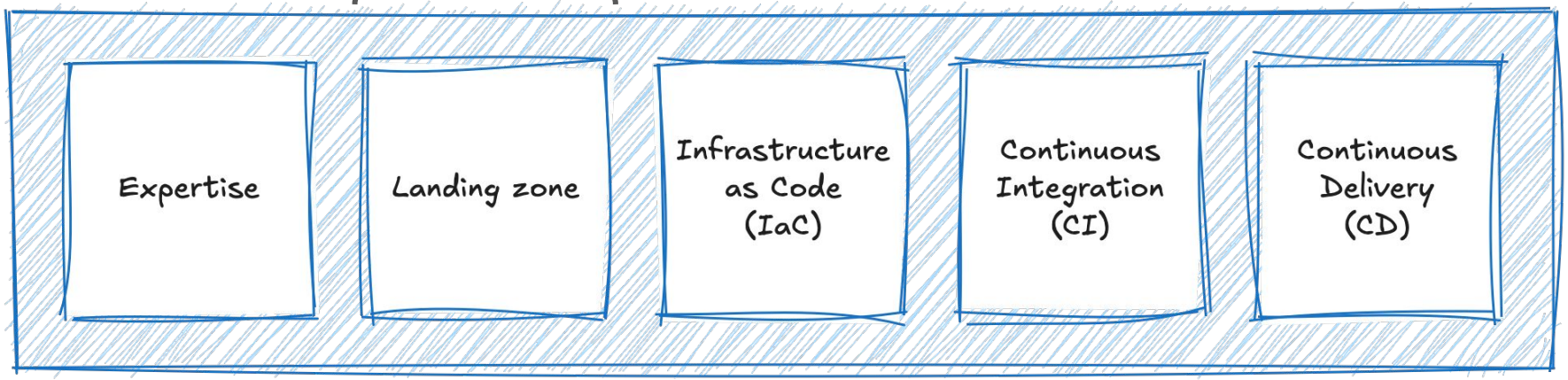


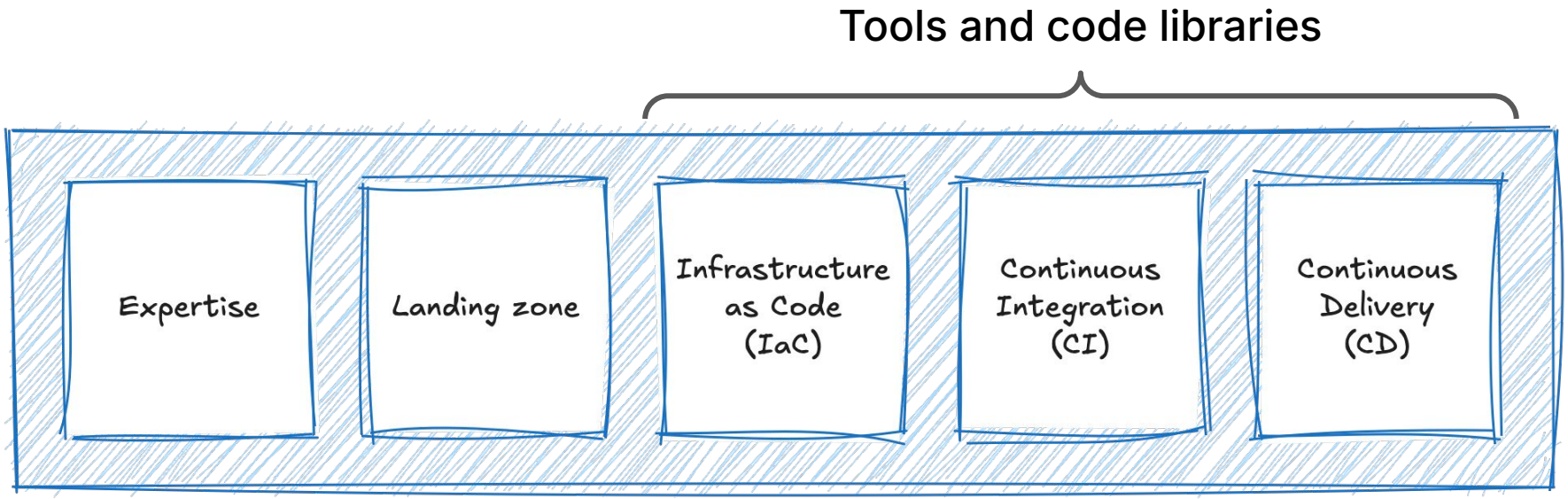
@erlendekern

Capra


Support and documentation
Expertise
Landing zone
Infrastructure as Code (IaC)
Continuous Integration (CI)
Continuous Delivery (CD)

Production-ready
cloud environments




Tools and code libraries
Expertise
Landing zone
Infrastructure as Code (IaC)
Continuous Integration (CI)
Continuous Delivery (CD)


Expertise

Expertise

@erlendekern

Capra

Expertise

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

@erlendekern

Capra


Landing zone

management

governance

Capra

Landing zone

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

management

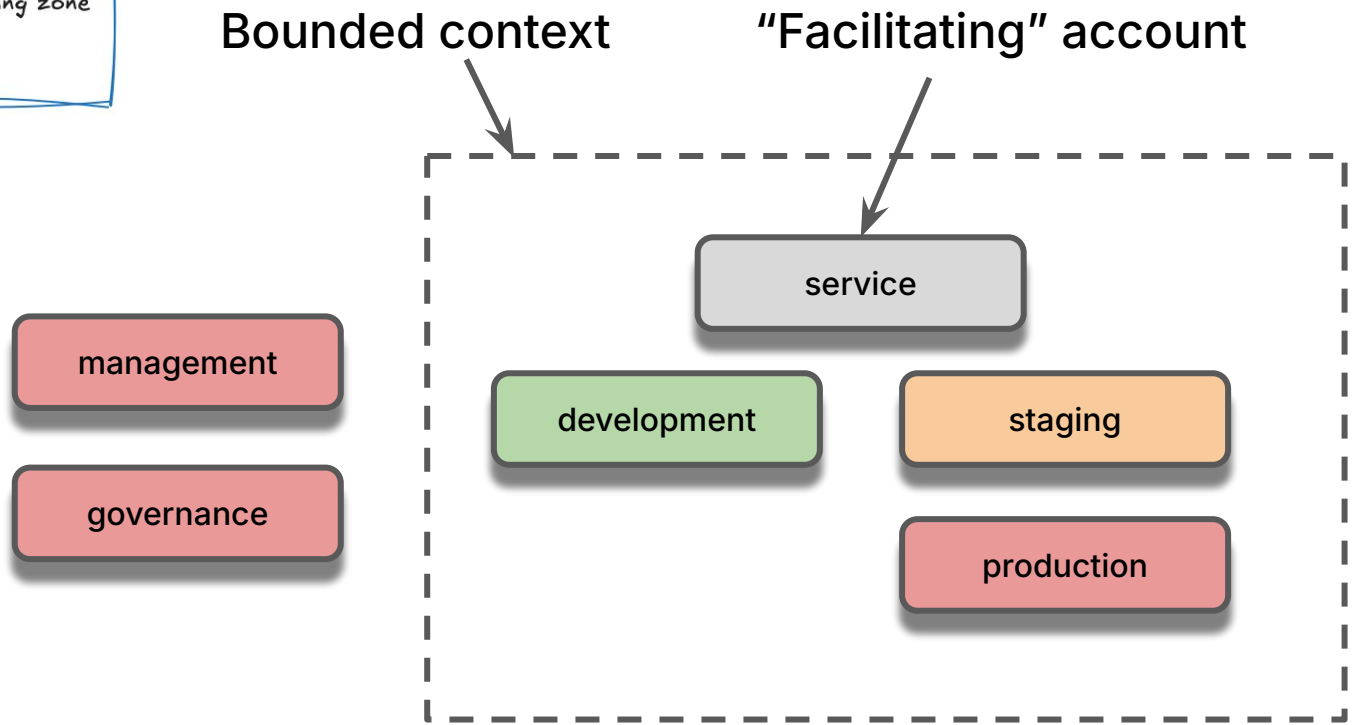
governance

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

@erlendekern

Capra

Landing zone



@erlendekern

Capra

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

AWS Control Tower supports automatic enrollment of accounts

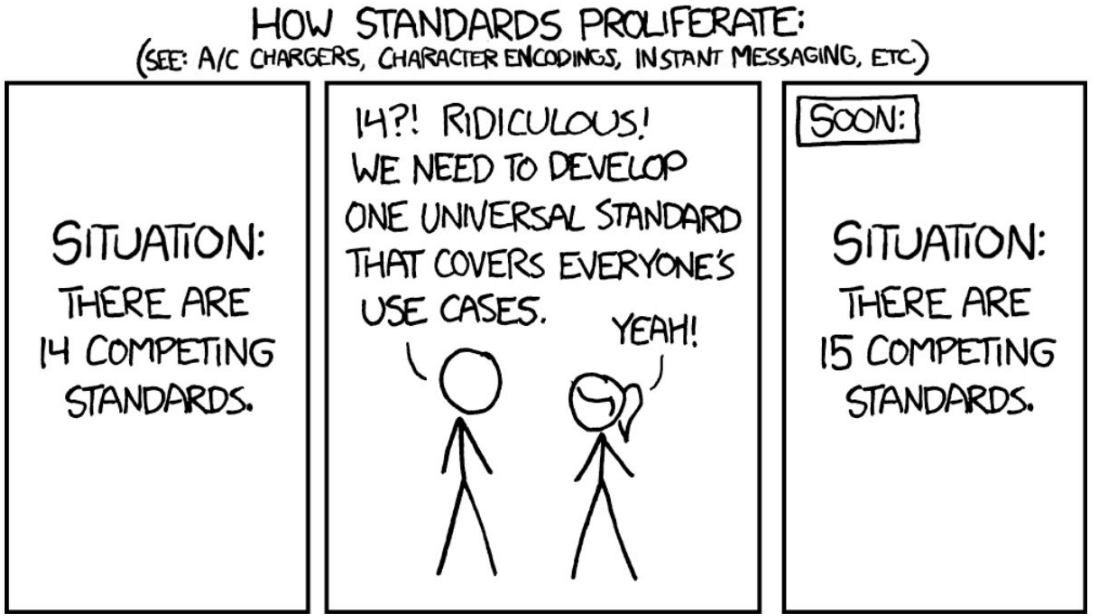
Posted on: Nov 10, 2025

AWS Control Tower introduces a controls-dedicated experience

Posted on: Nov 21, 2025

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

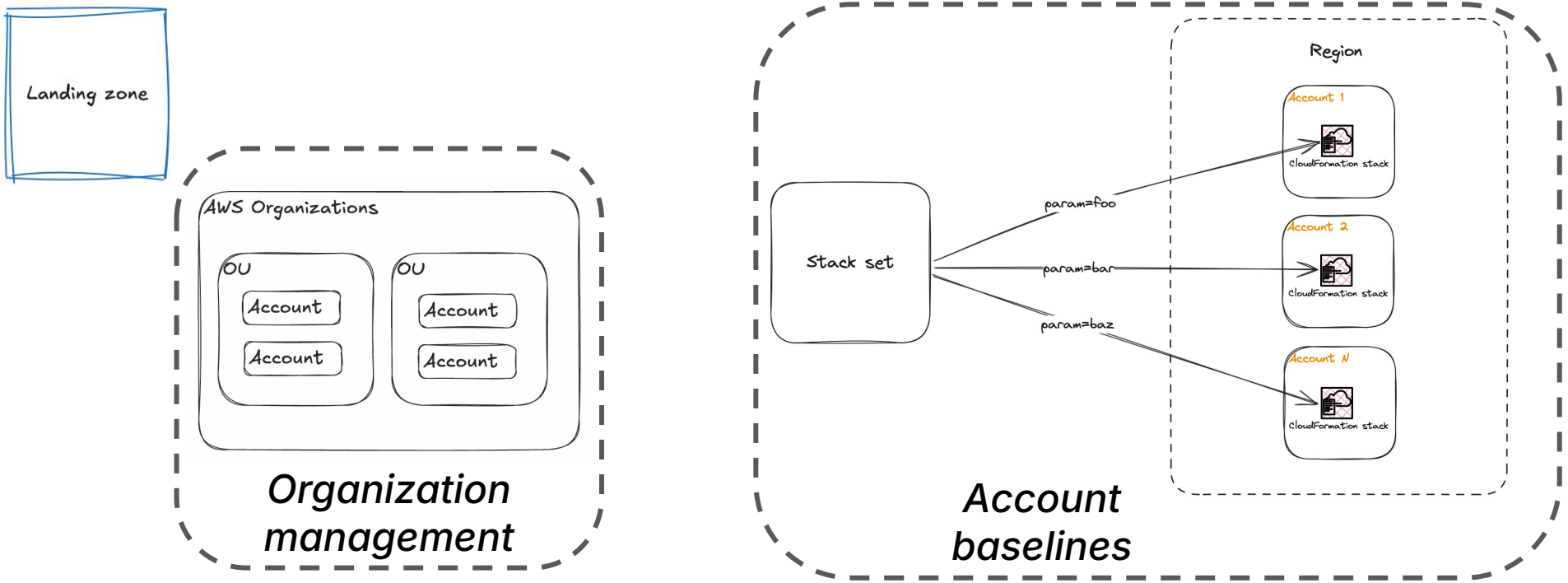
AWS Landing Zone
AWS Control Tower
AWS Account Factory for Terraform
AWS Landing Zone Accelerator
org-formation
superwerker
++



https://xkcd.com/927

@erlendekern

Capra



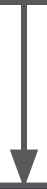
Terraform + CloudFormation StackSets?

@erlendekern

Capra


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

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



```
No changes. Your infrastructure matches the configuration.
```

@erlendekern

Capra

blog.ekern.me

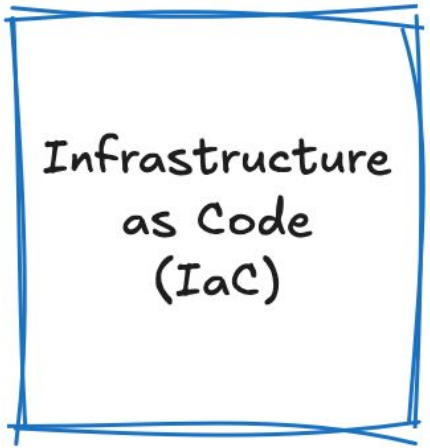
Feb 27, 2025

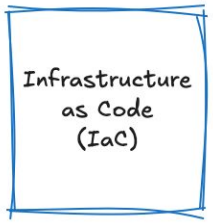
What's the deal with AWS CloudFormation StackSets?

primeharbor / **org-kickstart**

@erlendekern

Capra


Infrastructure
as Code
(IaC)



AWS CloudFormation

Terraform

Serverless Framework

AWS SAM

Pulumi

AWS CDK

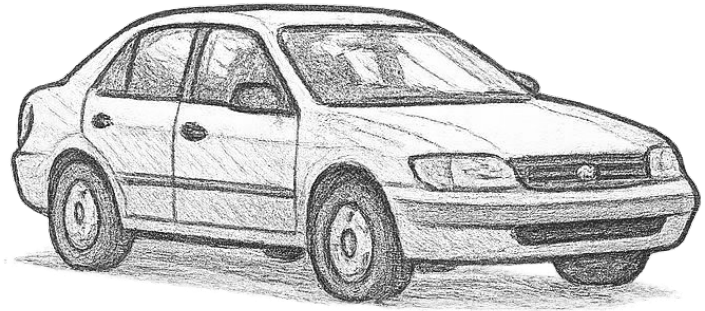
cdktf / cdktn

SST

Alchemy

???

Infrastructure as Code (IaC)

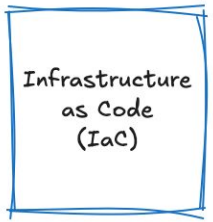


Terraform

AWS CDK

@erlendekern

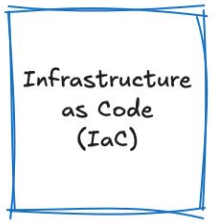
Capra



Keep It Simple™

@erlendekern

Capra



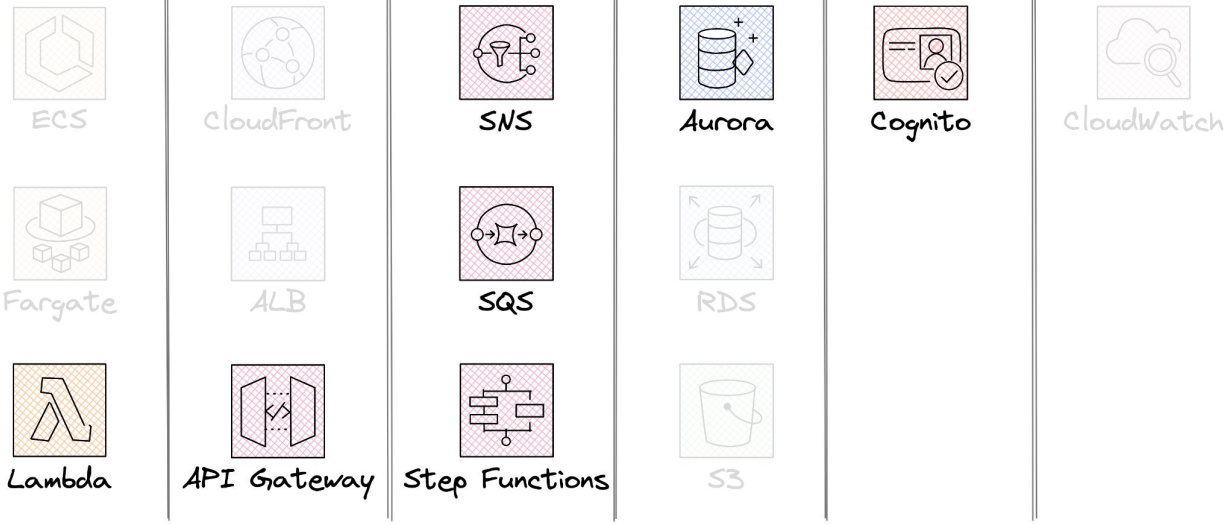
Thin **library** on top of AWS primitives

Infrastructure as Code (IaC)

@erlendekern

Capra

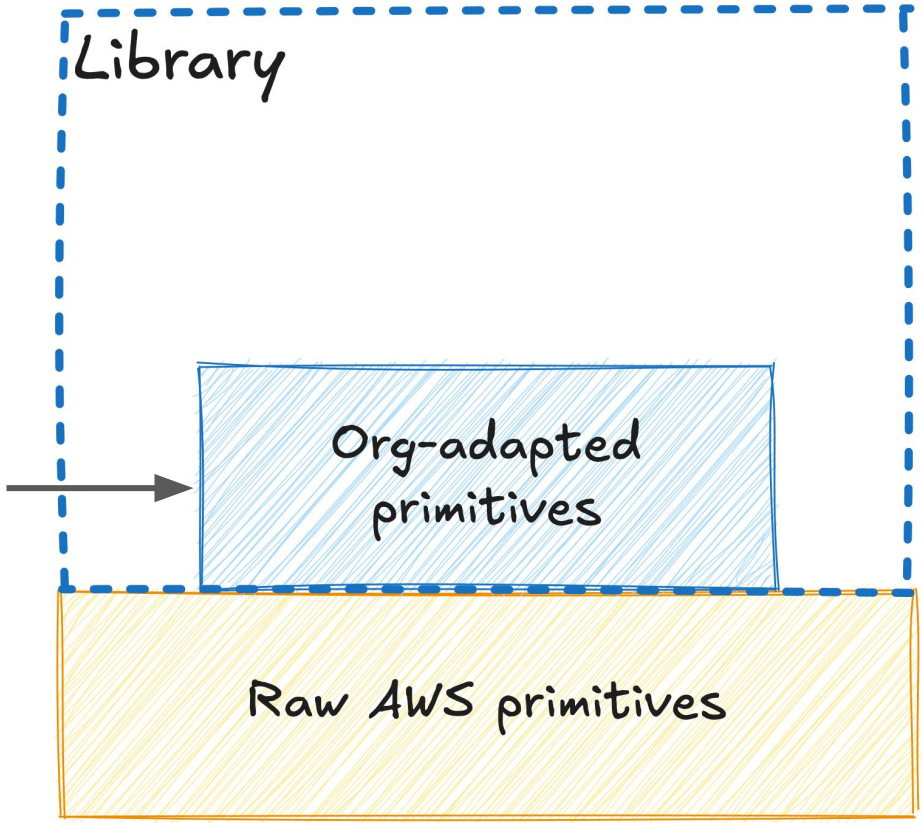
Infrastructure as Code (IaC)



@erlendekern

Capra

Infrastructure as Code (IaC)

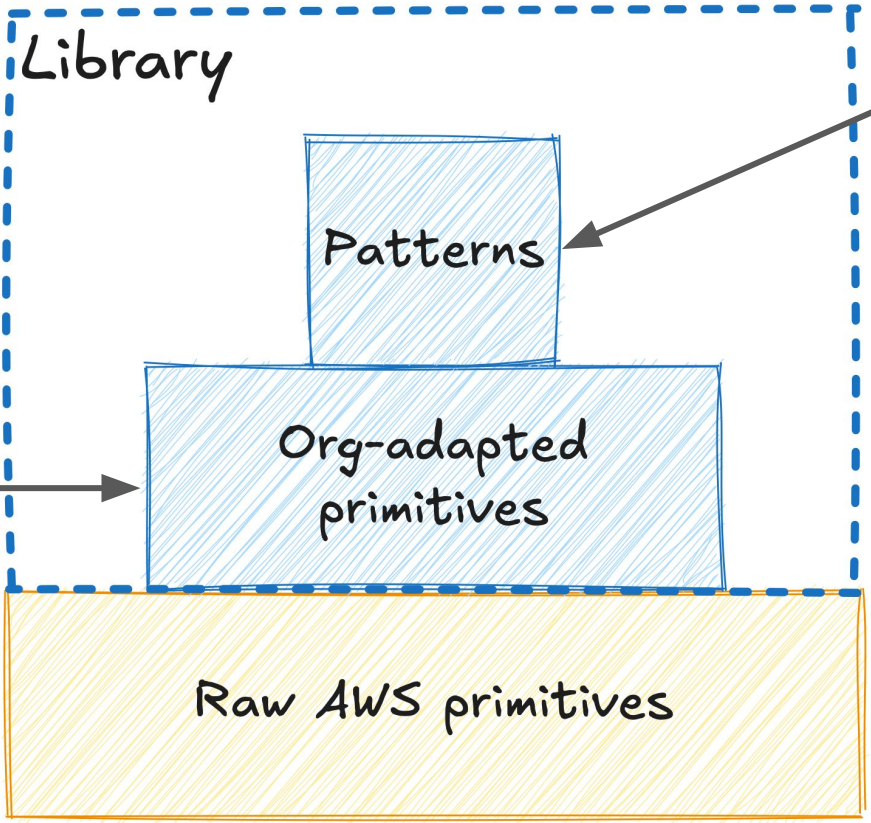


Focus on these

@erlendekern

Capra

Infrastructure as Code (IaC)



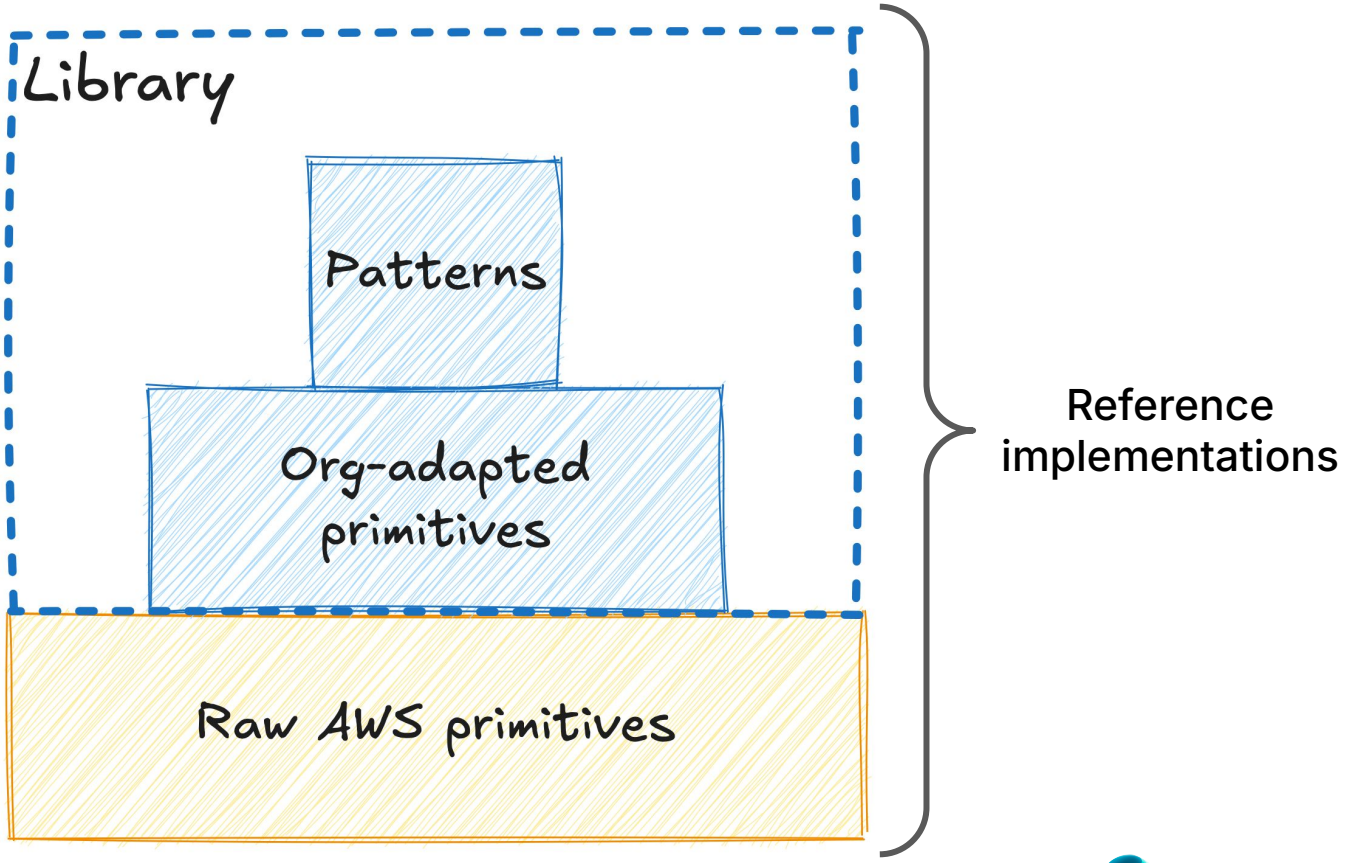
A few of these

Focus on these

@erlendekern

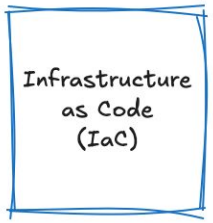
Capra

Infrastructure as Code (IaC)

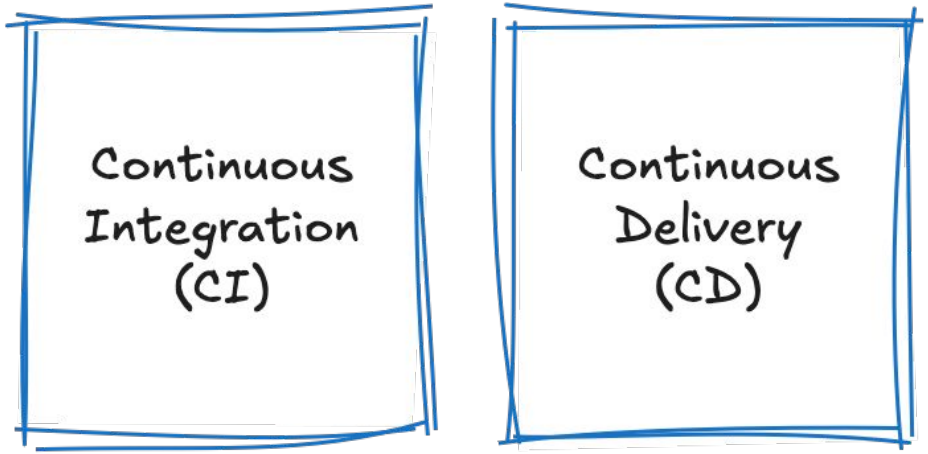


@erlendekern

Capra

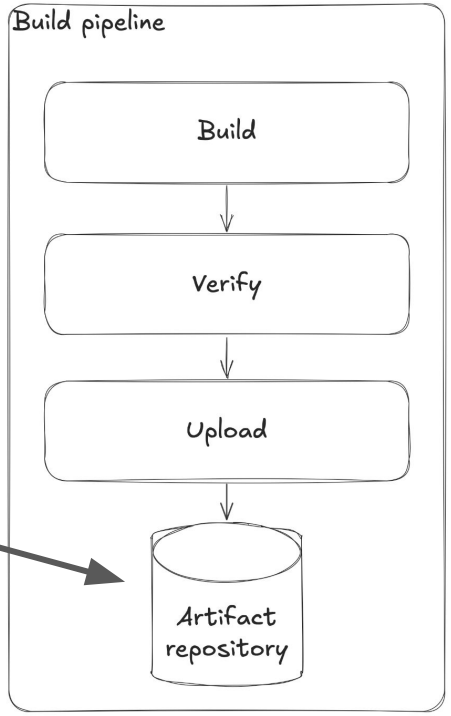


A forcing function for expertise?


Continuous
Integration
(CI)
Continuous
Delivery
(CD)

Continuous Integration (CI)

Continuous Delivery (CD)



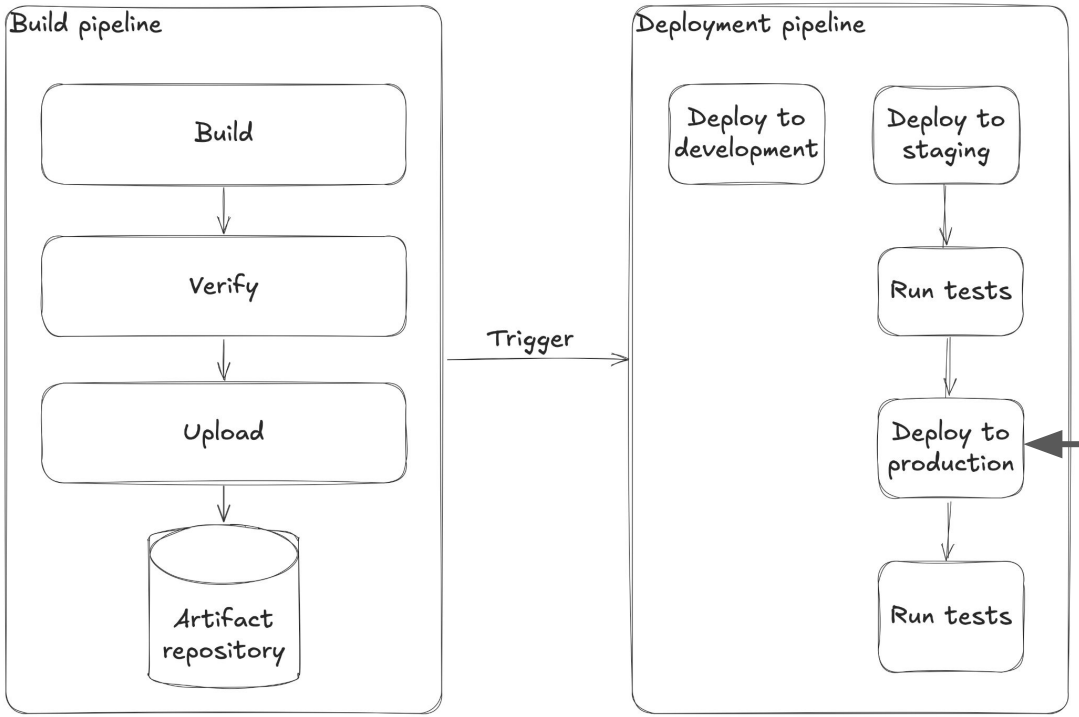
Build-once-deploy-many

@erlendekern

Capra

Continuous Integration (CI)

Continuous Delivery (CD)




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

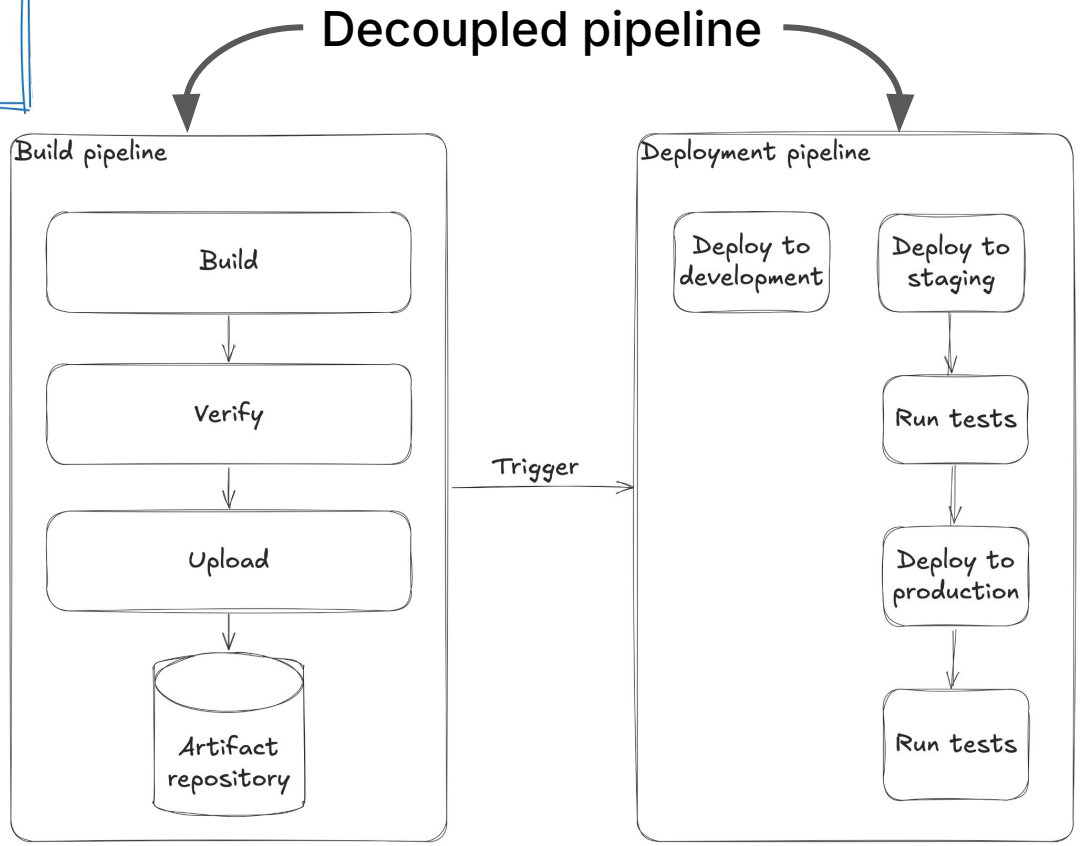
App deploy through infra

@erlendekern

Capra

Continuous Integration (CI)

Continuous Delivery (CD)



@erlendekern

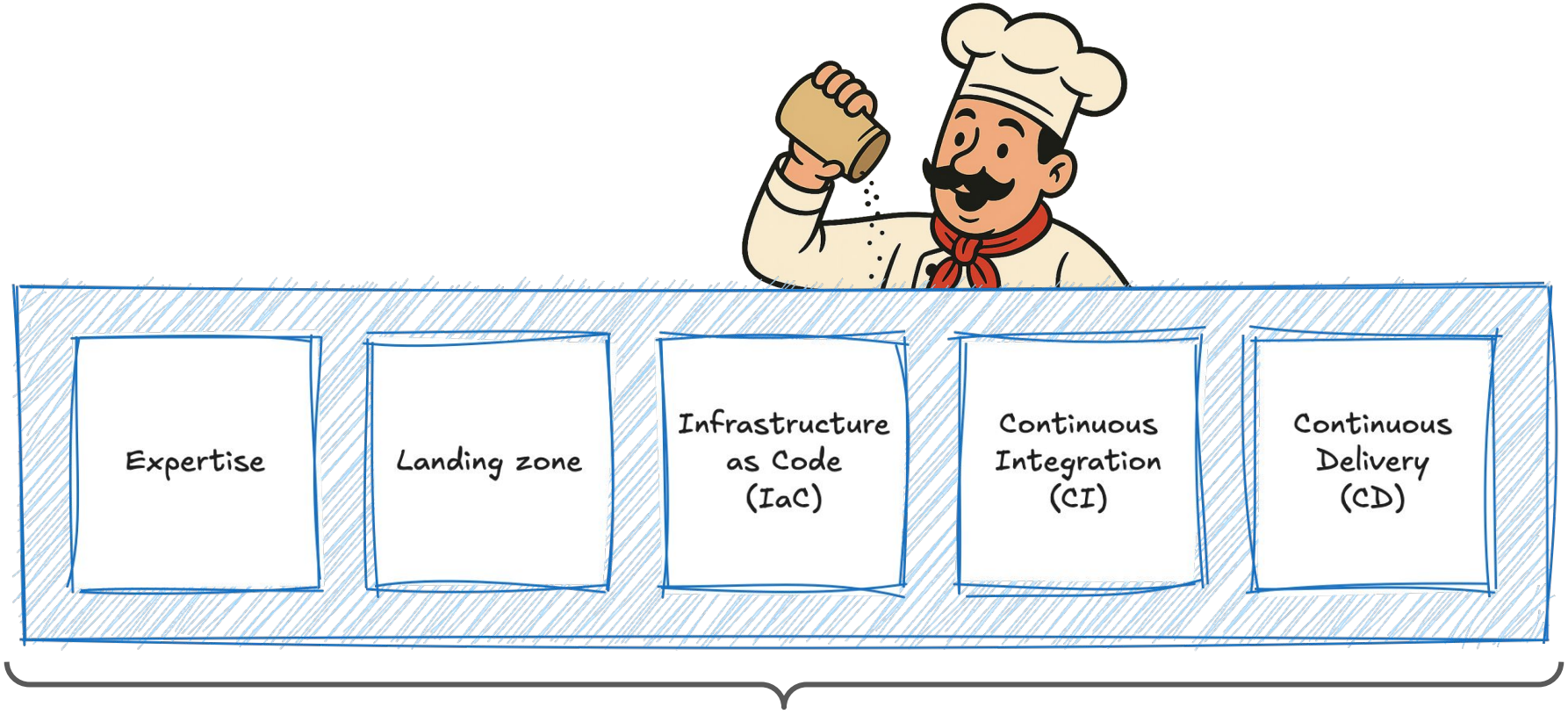
Capra

Continuous Integration (CI)

Continuous Delivery (CD)

@erlendekern

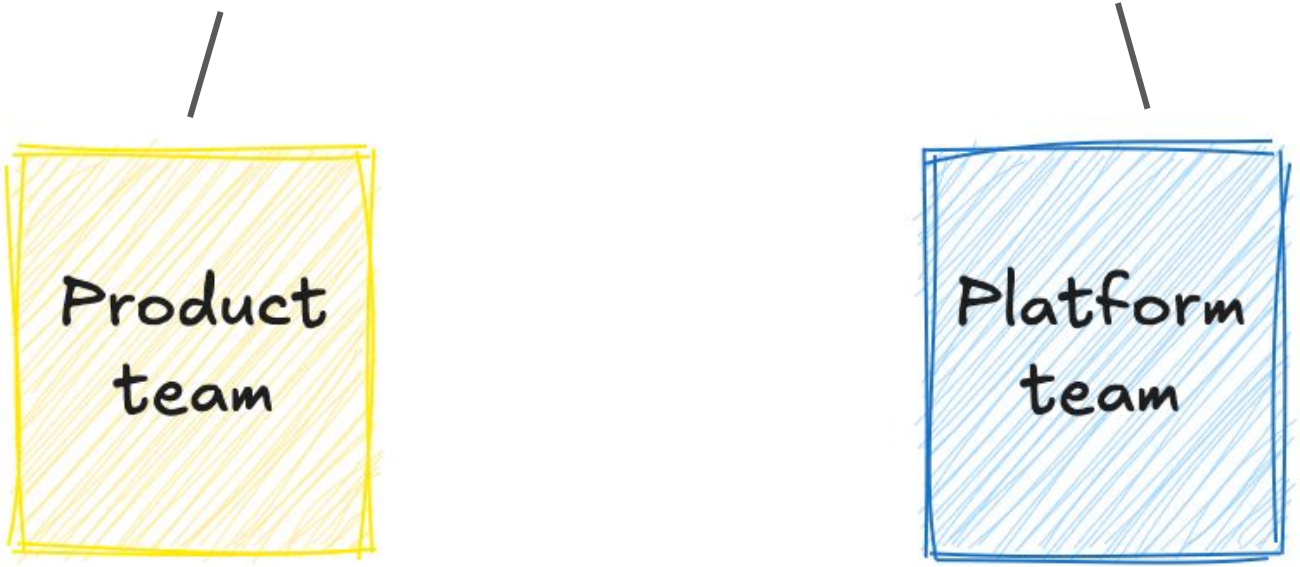
Capra


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`).

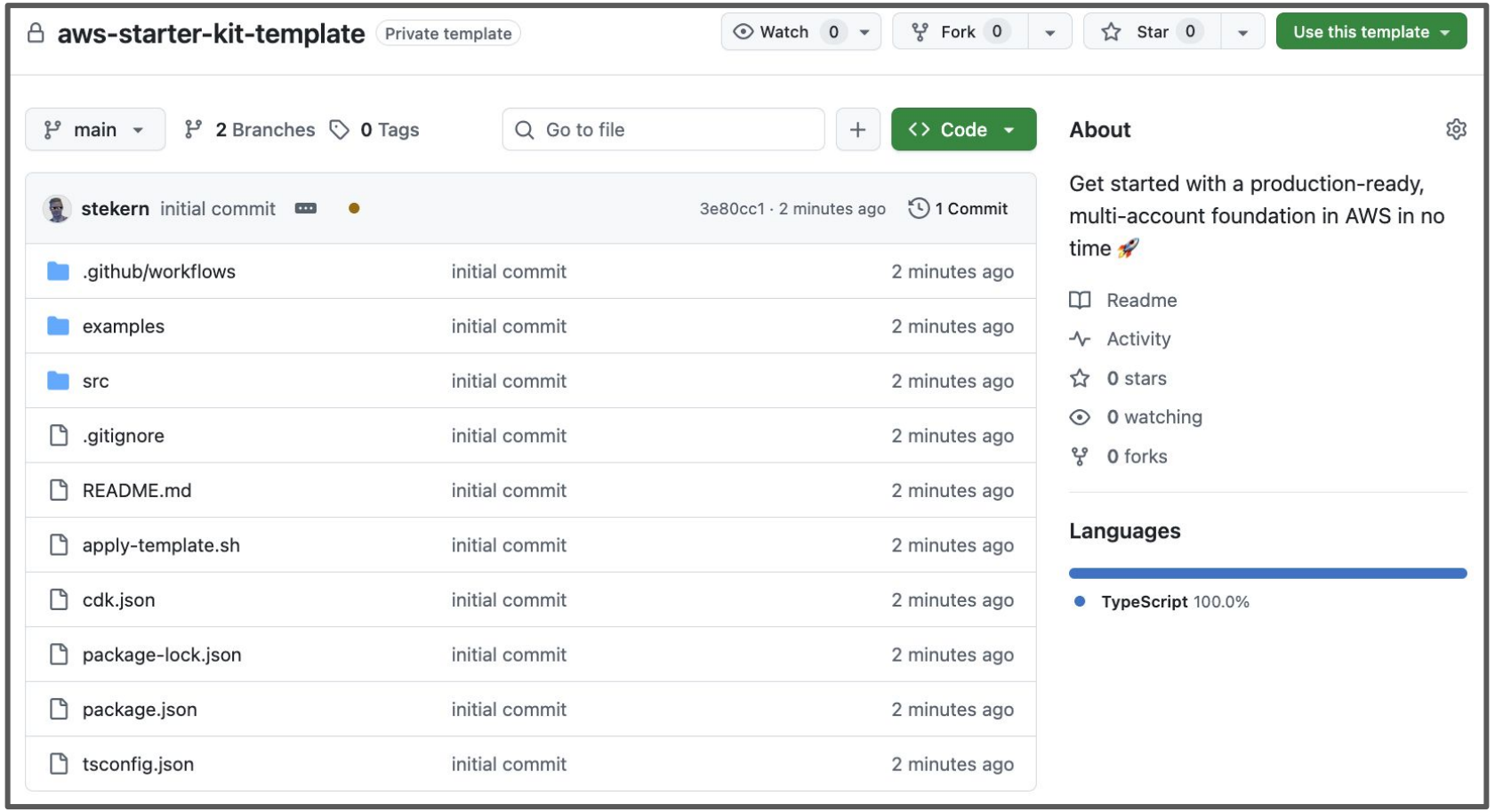
Check out our Getting started guide to configure access to AWS and get started with your very own Infrastructure as Code (IaC) repository.

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!

@erlendekern

Capra



@erlendekern

Capra

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:

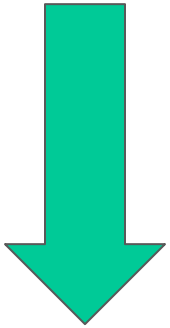
   ▸ **Show option details and example values**

```
$ ./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 😎

@erlendekern

Capra



- ★ 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

@erlendekern

Capra

Challenges

- Skills and maturity
- Ownership boundaries
- Variation and consistency
- Shadow platforms

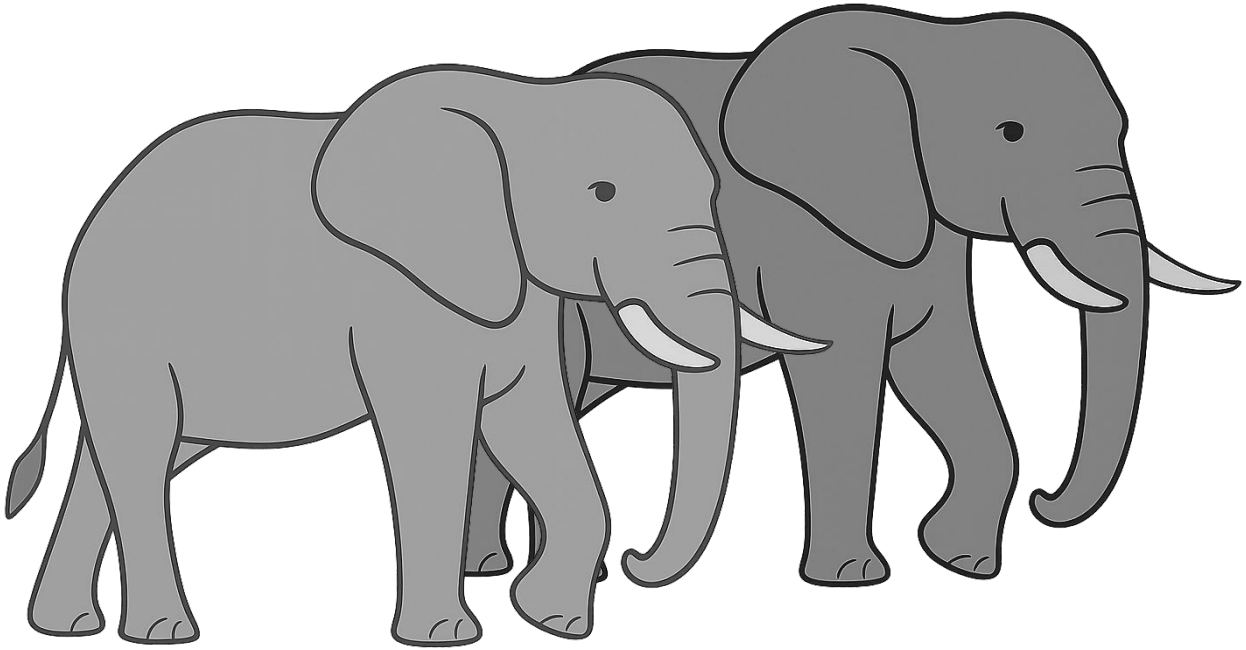
tl;dr

Enablement > restriction

Outcome > technology

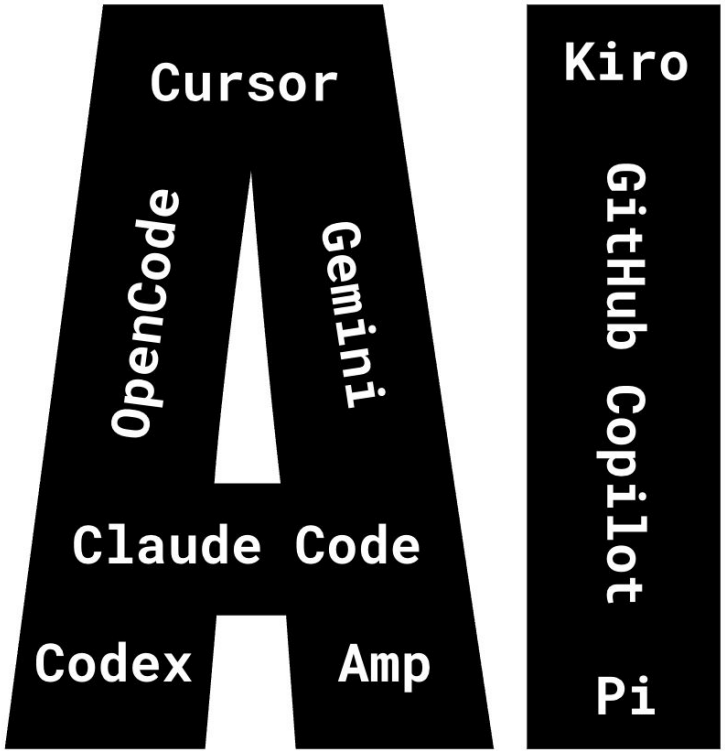
Constraints > flexibility

Composition > abstraction



@erlendekern

Capra

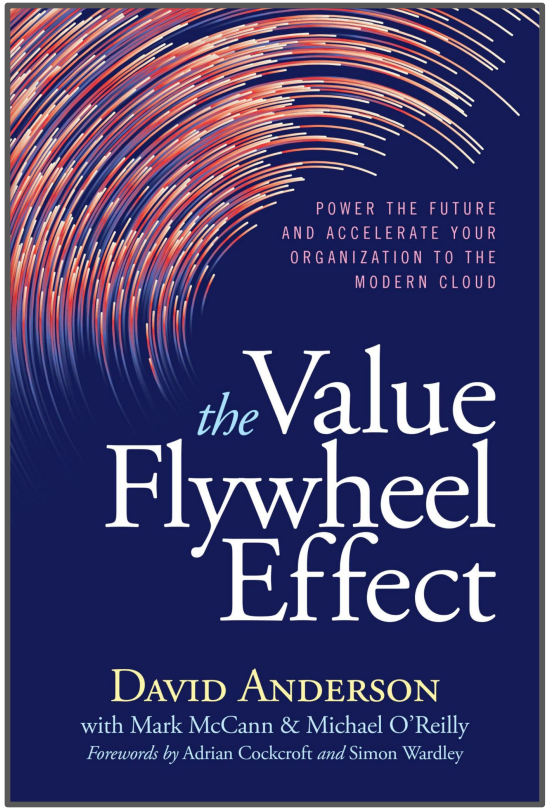


@erlendekern

Capra

@erlendekern

Capra



@erlendekern

Capra

Thank you!

🔗 ekern.me
🔗 linkedin.com/in/erlendekern
🔗 twitter.com/erlendekern

@erlendekern

Capra