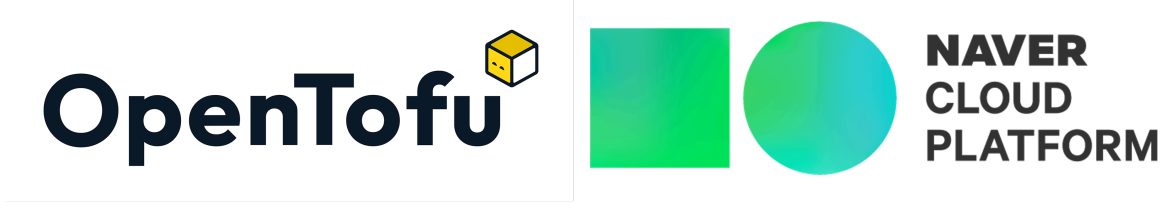
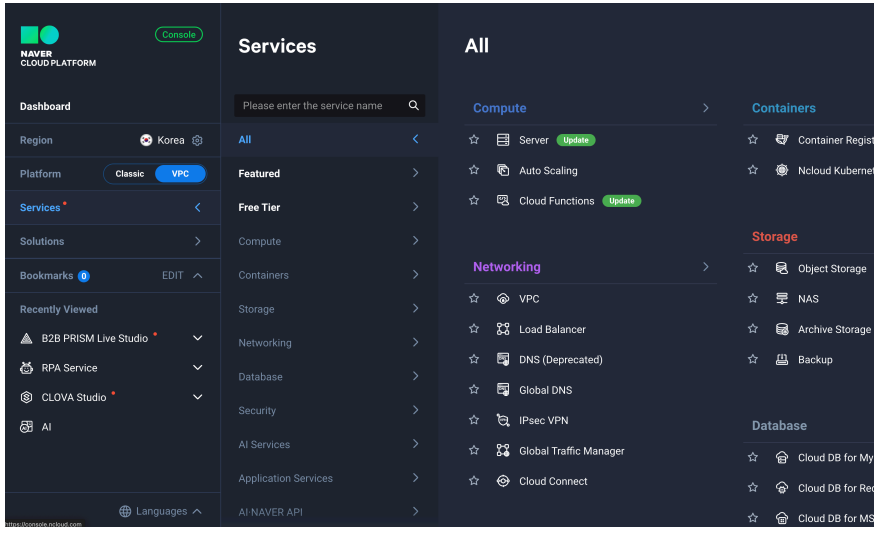
10 OpenTofu best practices for NCloud



Wojciech Barczynski | VPE | Spacelift

NCloud

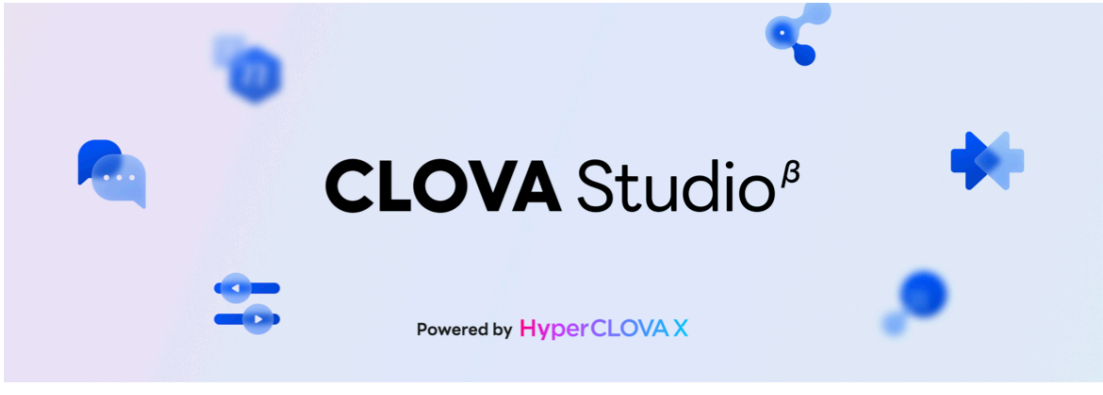


- Hyperscaller for **AI** and cloud services;
- Powers 65k companies;
- Samsung, SK Telecom, PUBS...;
- 6 regions;
- 300,000₩ companies;
- 100,000₩ individuals.

console.ncloud.com

The right hyperscale AI development tool for your business

With CLOVA Studio, you can train your HyperCLOVA X language model on your company's large datasets to tailor your AI service to your customers.
Optimized for businesses, CLOVA Studio allows companies to improve their productivity, stay ahead of their competitors, and solve a wide range of challenges.



Create new user experiences and business opportunities.

With HyperCLOVA X, developing AI models is easy.
1,000+ businesses across fields that had previously faced barriers to using AI are now embracing the technology, creating their own hyperscale AI-powered services using CLOVA Studio.

- Founding partner of OpenTofu,
- Top contributor to the project,
- TF/OpenTofu automation and collaboration tool,
- Infrastructure orchestrator.

What is OpenTofu/Terraform?

Demo:

```
variable "repository_name" {
  type    = string
  default = "my_app"
}

resource "github_repository" "my_repo" {
  name        = var.repository_name
  description = "my repo for my NCloud App."
  visibility = "public"
}
```

What is OpenTofu/Terraform?

Demo:

```
resource "ncloud_server" "app1" {
    name = "app-vm1"
    server_image_product_code = "SPSW0LINUX000032"

    tag_list {
      tag_key   = "team"
      tag_value = "billing_and_revenue"
    }
}
```

Why Infrastructure-as-Code (IaC)?

- Clickops 💀,
- CLI scripts with `if`/`else` 😐,
- Hard to track changes,
- Hard to work on Infra in a team,
- Gitops 🎉.

Why OpenTofu?

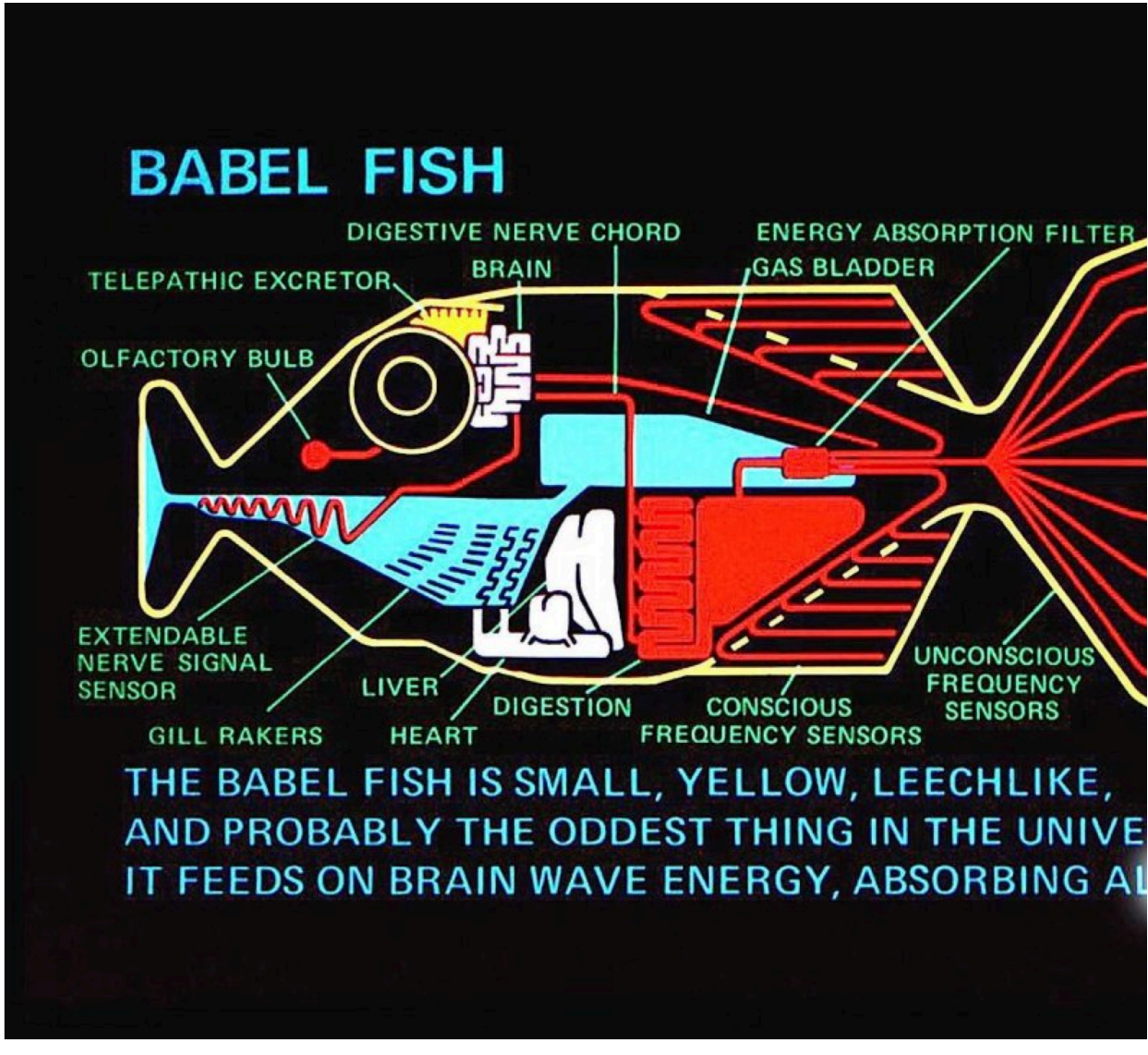
1. `tofu plan` – you know in advance about changes,
2. `tofu apply`,
3. state file – benefits (demo):
    - shift left X;
    - policies, linters, and scanner;
    - drift detection 🎉.

Common

Language

Artifacts

Platform



Why OpenTofu/Terraform FOSS?

- Solid documentation,
- Vibrant ecosystem.

IaC in your company

Ultimate goal in mind, our north star:

- Engage other groups of engineers;
- Everybody should have access to IaC and contribute;
- Empower - IaC is close to the product teams.

Infrastructure-as-Code

Early mistakes will lead to difficulties later.

1/10: Remote state

Implementation with locks:

- Cloud provider: Naver, Azure, AWS, GCP;
- TACOS, for example, Spacelift or Env0.

Turn on the encryption: server-side or client-side with opentofu 1.7.x

1/10: Remote state

Client-side encryption (coming in OpenTofu 1.7):

```
terraform{
  state_encryption {
    statefile {
      key_provider {

      }
      method {

      }
      enforced = true
    }
 }
```

1/10: Remote state

Remember the basics.

- `.gitignore`
- `.dockerignore`

2/10: Project structure

Mono-repo (per env/app):

```
infrastructure.git/
 |- modules/
 |   |- network
 |   \- service-a/
 |
 |- production/
 |    \- service-a/
 |
 ...
```

2/10: Project structure

Close to the application:

```
my_app.git/
 |- app/
 |- infra/
 |   \- ...tf
 |
 ...
```

2/10: Project structure

- Copy&paste is the easiest debt/loan to pay back;
- You do not have modules from day 1;
- Start with the community modules;
- Do not rash standard modules.

3/10: Small stacks/states

- Smallest that makes sense, e.g., per component;
- You SHOULD not read state files of other workspaces/stacks;
- Watch out to not get rigid infrastructure 💀.

3/10: Small stacks/states

Rigid infrastructure:

- You need to be a super admin;
- Hard dependencies between workspaces/stacks;
- Too opinioned modules from the start;
- Cyclic dependencies.

3/10: Panacea

	Description	+	-
`data`	read from resources	Easy to implement	Slow
key/value;	my default	Fast	Cycles
TACOS env vars	k/v overlay Spacelift context	Reusablity	Tracing cycles
Graph /workflow	Spacelift Dependencies	Visible dependencies	Not standard

4/10: Naming Convention

1. Underscores(_) as a separator and lowercase letters in names.
2. Try not to repeat the resource type in the resource name.
3. Single-value vars and attr - use singular nouns.
4. Lists or maps - use plural nouns.

4/10: Naming Convention

Learn more:

- Cloudposse best practices;
- Developer Terraform Docs;
- 20 best practices for OpenTofu/Terform workflow;
- see Cloudposse terraform-null-label procider.

5/10: Pin versions

```
terraform {
  required_version = ">= 0.12.26"

  backend "s3" {}
}
```

- Providers – minimum version
- Modules & version for OpenTofu/Terraform - exact for 3rd party

6/10: Tools

1. `tofu plan` - you can validate the changes before apply.
2. shift-left X, where X = best practises, security, and policies.

6/10: Tools

In your CI/CD or git .pre-commit:

- Linters: `tofu fmt` & `tflint`;
- Security: `tfsec`, `kics`, or `checkov`;
- Cost estimation: `tf-cost-estimation` & `infracost`.

6/10: Tools

In your CI/CD or git .pre-commit:

- Policies: conftest & OpenPolicyAgent:
  - Warning, when a resource will be deleted or recreated,
  - Enforce tagging;
  - Prevent expensive resource configurations.

TACOS: Spacelift, Env0, and Scalr — have out-of-the-box integrations

7/10: Running from your laptop

https://twitter.com/mtcderek

- Uff, luckily it was my test env
- ...
- ...
- 🚨

7/10: Continuous Deployment

1. Start with Continuous Integration,
   even if you run your OT/TF from your laptop;
2. For the IaC democratization & team empowerment
   GitOps & Continuous Deployment is a MUST;

7/10: Continuous Deployment

Options for this maturity of your IaC play:

- Generic CI/CD;
- Atlantis (open source);
- TACOS: Env0, Scalr, and Spacelift.

8/10: Democratization

Just a matter of time:

- Development teams grow fast;
- Hard to hire Platform or System Cloud/DevOps engineers;
- Budget?
- CTO, why devs are waiting again for X?
- Product teams!

8/10: Democratization

How to start:

1. Start with one person per product team — Infra/IaC/Cloud Engineering champion;
2. Alternative - assign one DevOps / platform team engineer to the team;
3. First iteration, copy&paste plan as Pull Request comment;
4. Few common (small) modules.

8/10: Democratization

It will be much easier when:

1. No large stacks/workspaces and hard dependencies;
2. You do not need to have god permissions to do cloud resouces.

8/10: Democratization

Right tooling:

- GitOps — PR workflow with TACOS;
- GitOps through Kubernetes CRDs with, e.g.: tofu controller, crossplane or Spacelift operator.

8/10: Democratization

Right tooling:

- A platform that uses Infrastructure Orchestrator (APIs of TACOS) and other tools under hood.

8/10: Democratization

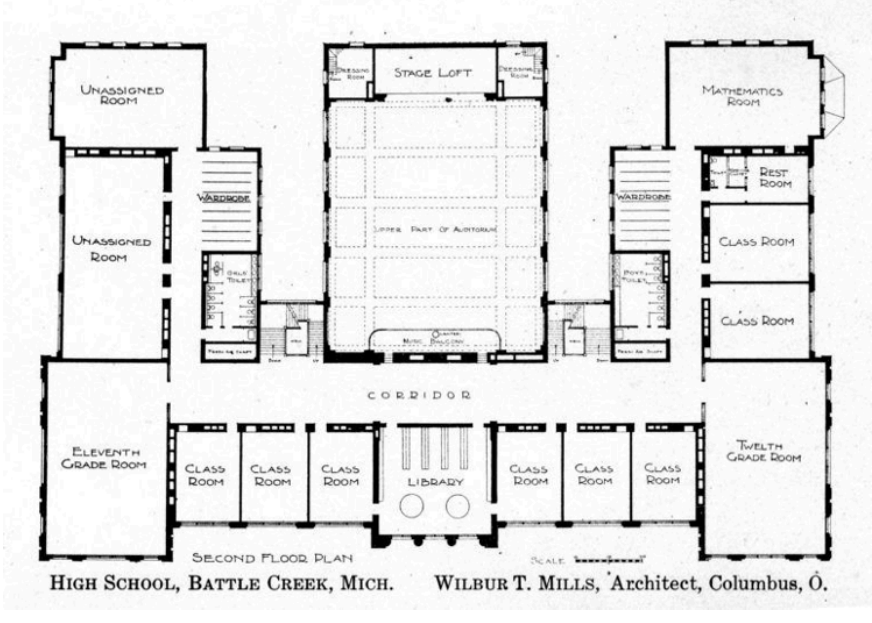
Benefits:

- The opening of IaC will change the dynamics between DevOps and developers;
- Encourage ownership and improve communication;
- Much less finger pointing and leaner IaC.

8/10: Democratization

This does not mean all engineers will suddenly create infrastructure themself.

8/10: Democratization



- Start with common conventions... later tools/modules/reusable XYZ

9/10: Scaling

1. Streamline the chanages;
2. Sharig best practises;
3. Governance and security.

9/10: Scaling

Streamline the chanages:

- Promote infrastructue self-service (GitOps/K8S Operators);
- Support short-living environments;
- Decide what changes need review amd which not (approval policies).

9/10: Scaling

Sharing best practises:

- Landing zones and templates;
- OpenTofu/Terraform modules and providers;
- Tooling, e.g., `conftest` or `tfsec`

9/10: Scaling

Generators and common modules:

- Conventions over code;
- Avoid black magic;
- Light platform with YAML or JSON (not big fan);
- Developers should be able to run apps on their workstation.

9/10: Scaling

Governance and security.

- Testing of common code;
- Enforcing module updates;
- Enforcing policies;
- Security and lintering tools;
- **Drift**.

9/10: Scaling

Drift detection (managed and unmanaged):

- Catch ClickOps;
- Ensure your rarely touched project, still works;
- Clean up after experiements;
- Track a migration from ClickOps to IaC.

9/10: Scaling

Mono repozytory or repo per component:

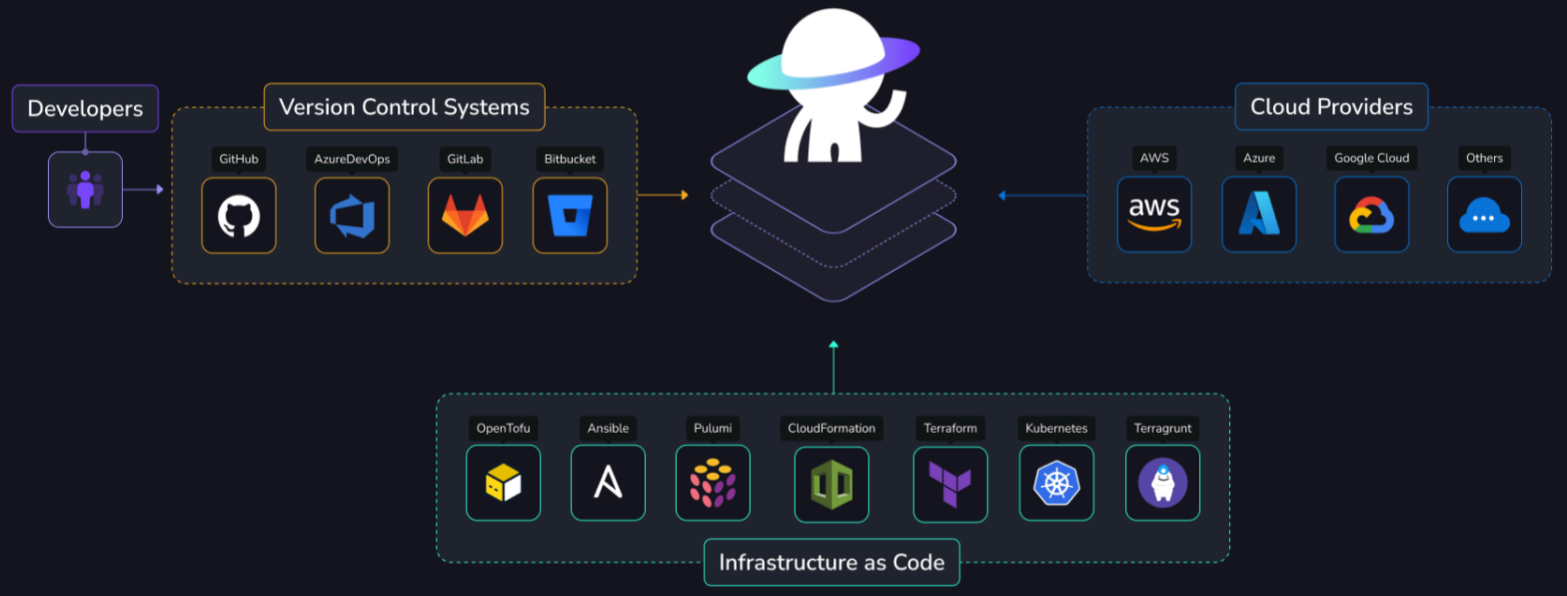
- Depends on your use case, company structure, etc.

9/10: Scaling

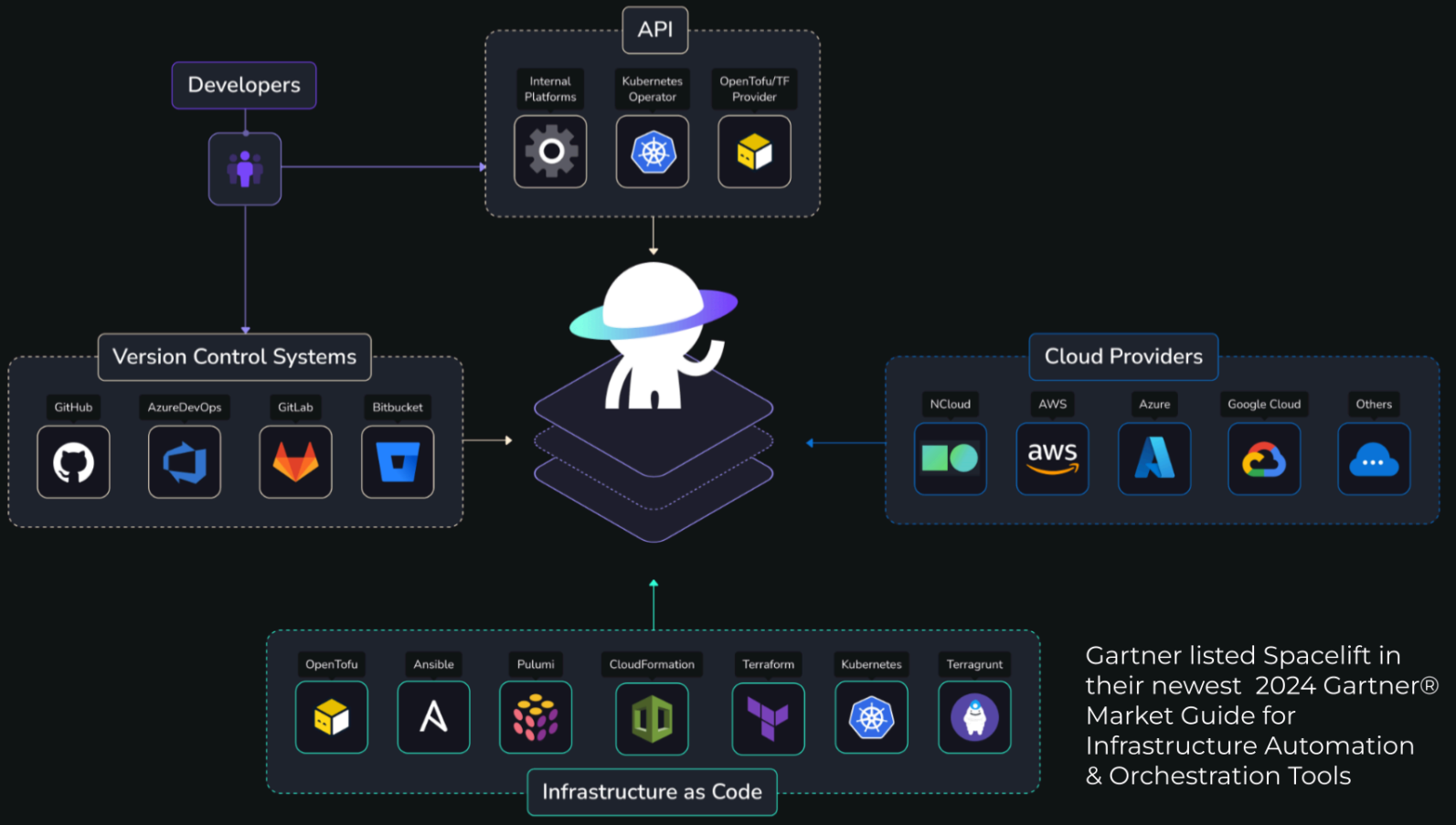
Demo of GitOps with Spacelift:

1. Pull-request model;
2. Policies;
3. OpenTofy plan in the PR comment.
4. Custom tools;
5. Account resources.

Infrastructure-as-Code automation and collaboration tool



Infrastructure orchestrator



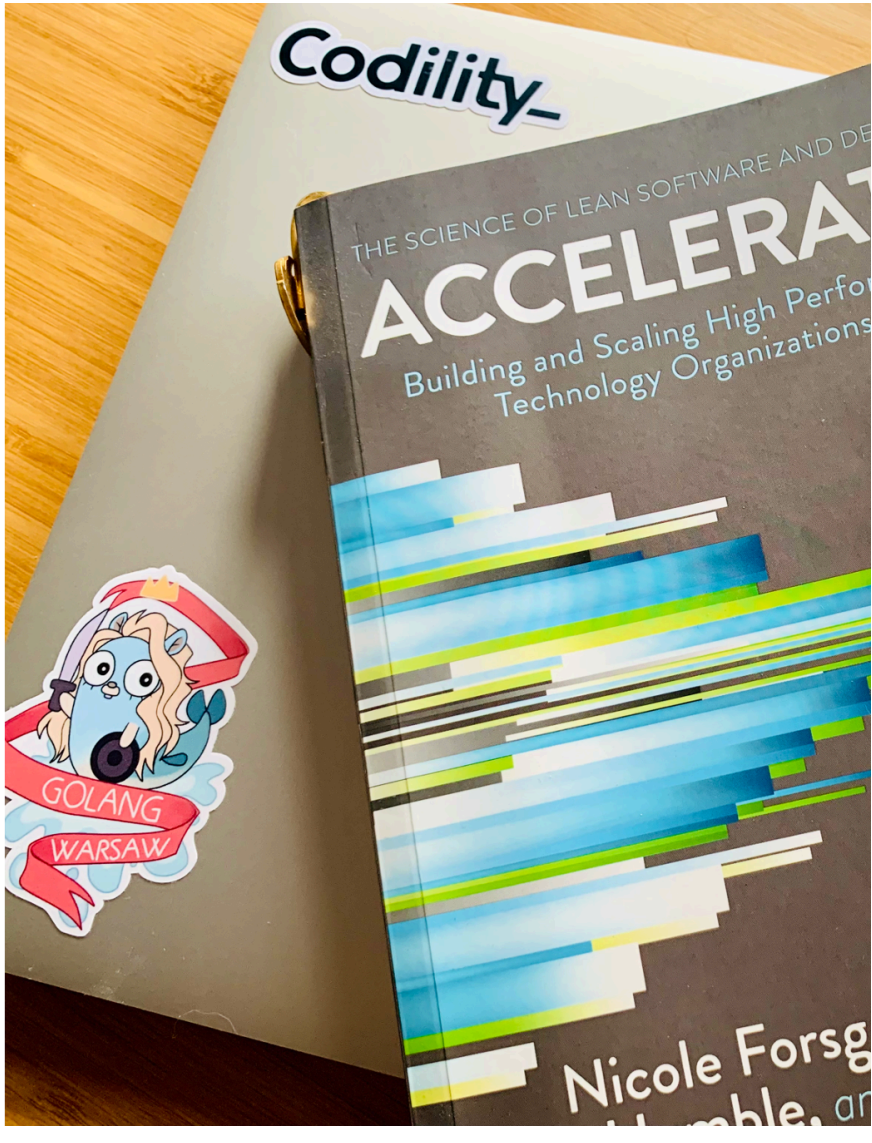
Gartner listed Spacelift in their newest 2024 Gartner® Market Guide for Infrastructure Automation & Orchestration Tools

10/10: Metrics

- The first successes are easy to feel,
- Later, it is harder to measure progress,
- So, what should we measure?

High delivery performace

- Lead Time
- Deployment frequency
- Mean time to Recovery
- Change Fail Percent



10/10: Metrics

Deployment frequency:

1. Easy to capture with GitOps;
2. Completion of the pipeline or simply merging into the production branch.

10/10: Metrics

Lead time:

1. From openening Pull Request as DRAFT;
2. To production deployment
   (merge to master/prod).

10/10: Metrics

To consider:

1. Number of contributions to IaC from outside Platform/Infra/DevOps teams;
2. The rest of DORA metrics - mttr and failure-rate - and apdex.

10/10: Metrics

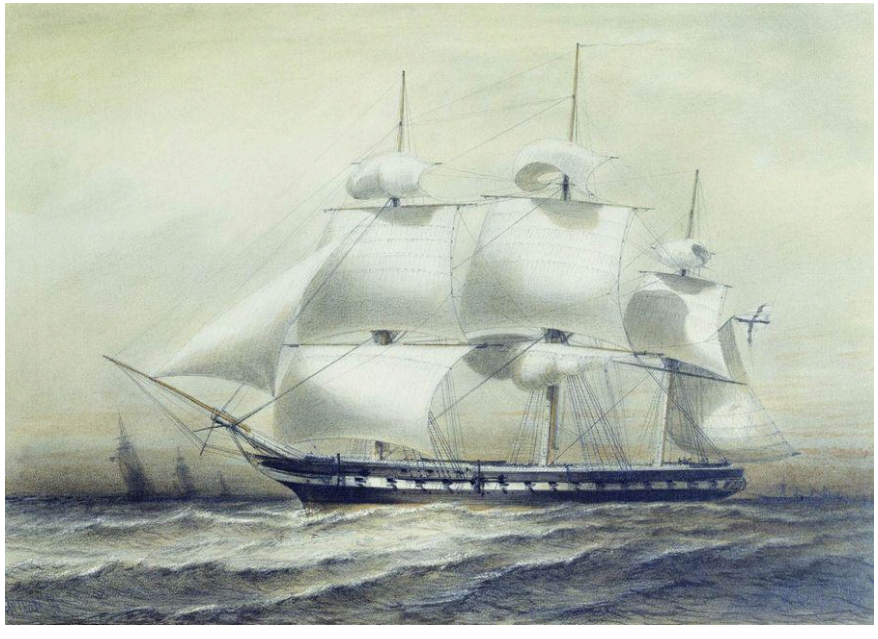
I use at this moment:

- metabase & dbt & python exporters;
- before: Google Colab Notebook, BigQuery, and ad-hoc python exporters.

DevLake looks promising.

Summary

- Assume you need to democratize your IaC;
- OpenTofi/Terraform is the most mature IaC;
- Most of best practices apply to other IaC, e.g., AWS CloudFormation, CDK, CDK-TF, to Kubernetes.

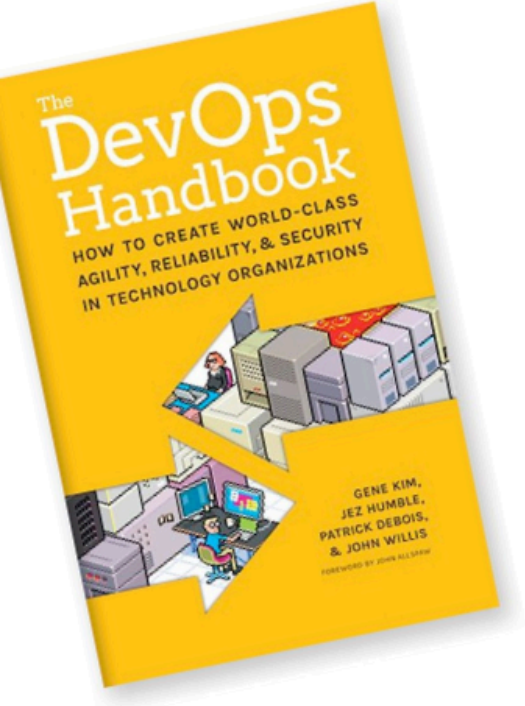


Thank you

github.com/wojciech12/talks

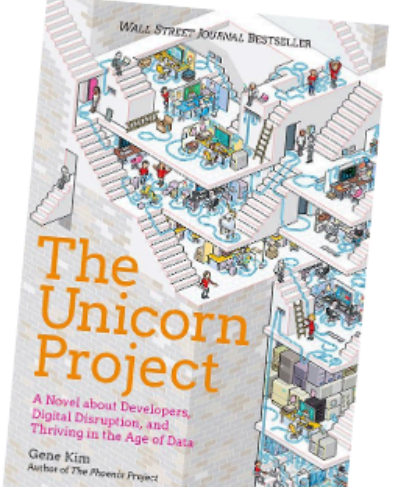
Backup Slides

Recommended read



1 Concrete, do A, do B,
because C
(DevOps here as culture ☺)

2 The first 4 chapters,
the rest if you have time.



3

A story,
not as straight forward
as (1)

Rabbit holes everywhere...

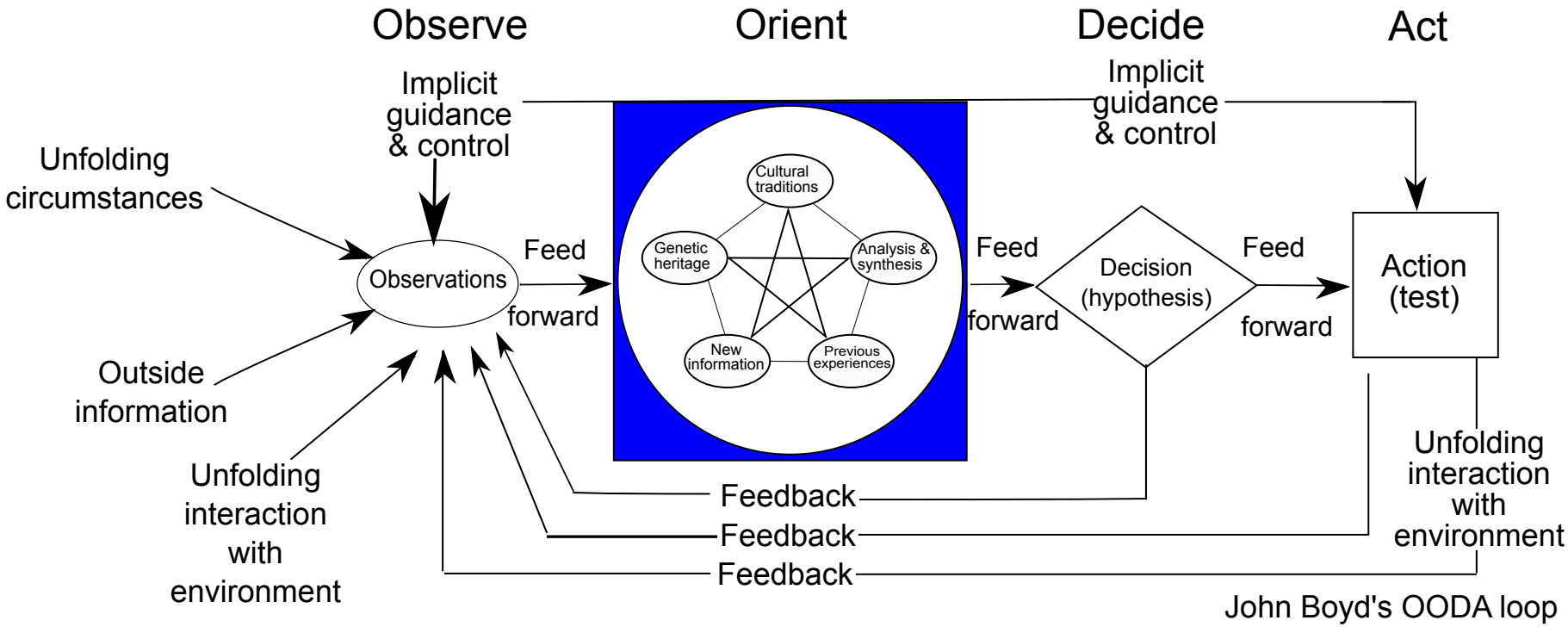
Approach:

- The iteration, decision, and deliver;
- As soon as possible to get into the cycle Patch Patch Patch.

Alternative take:

- Tracer bullet development;
- Lean v1/v2.

OODA



John Boyd's OODA loop