Step 1
It’s easy to comprehend
But it's a fixed pipeline

Doesn't scale well for complex features: rollbacks, feature builds, cleanup
Your deployment audit is your CI history
No central control
Step 2
You get the core properties of GitOps

Git interface, desired state is versioned, audit log
Still working in the boundaries of your CI engine
Requires a paradigm shift

Additional machinery, eventual deployments
Step 3
Environments captured declaratively with the application source code
Do you need a new environment?

Add a new environment file
Some config needs change?

Change it in your app repo, make PR and review
Upgrade to a newer application template?

Bump the chart version
Drawbacks?

Updating configs in all apps becomes an automation / coms problem
Still working in the boundaries of your CI engine
Step 4
No central control

Cannot enforce policies, difficult to share logic across repos
CI is rigid

Predefined pipelines, only so many workflows, triggered by code change
GitOps write collisions
Detaches the release workflow from CI

Unlocking advanced features, adding flexibility to refactor workflows, on-demand deploys
Adds central control to the release workflow

Policy based deploys, fine grained ACL model
Want to get involved?

https://github.com/gimlet-io/gimletd