

Go2Group

Rasmus Praestholm

Sr. DevOps Architect


Rasmus@go2group.com

Demo code and dev workspace at:

<https://github.com/Cervator/modern-jenkins>



GO2GROUP

A laptop screen is shown in the background, displaying a line graph with a blue line and a pie chart. The text is overlaid on the screen in a large, white, sans-serif font. The text reads: "Target Audience: Complex enterprises, or the consultants that serve them".

Target Audience:
Complex enterprises,
or the consultants
that serve them

The problem

Rarely are all teams in sync. Maturity varies

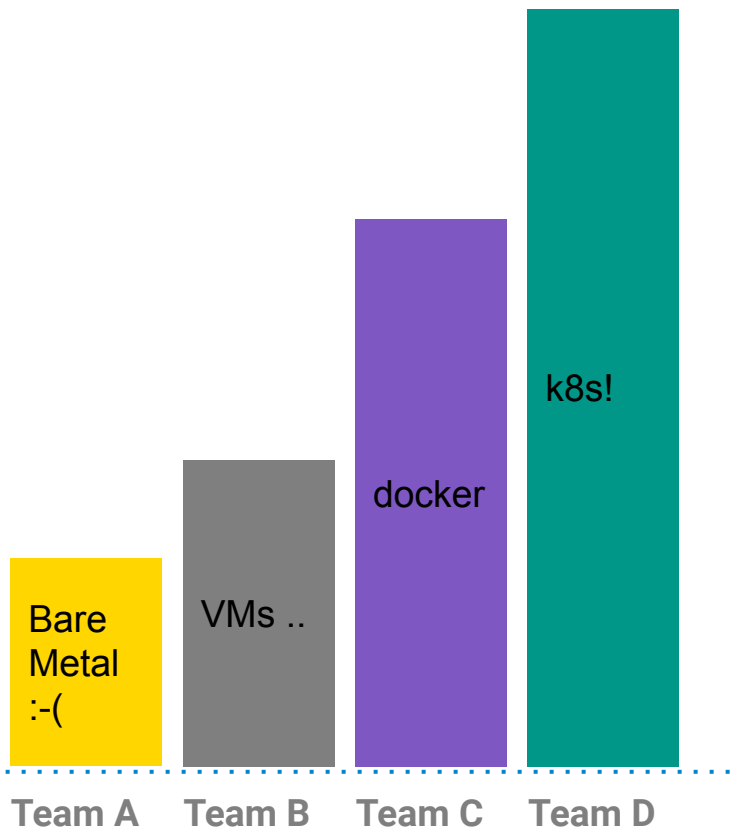
Target state: To the moon! That's great but ..

CodeFresh? Drone? WeaveWorks?

This one team uses X, another Y ..

We aren't even on Docker. What's k8s?

What can GitOps do for me as a caveman?



A close-up photograph of a person's hand holding a purple marker, drawing on a whiteboard. The background is blurred, showing some office equipment and lights.

The solution

Hobo GitOps!

Teams come in all shapes
and sizes. May come and go

Need something that helps
them all move forward

The teams

All use cases welcome!



Alice

We're fully on Kubernetes using the WeaveWorks Flux Git Operator. Our CI builds run out of a heavily customized Jenkins



Bob

I majored in theater. Why am I even here? Oh, I have an old server that hosts a website. I don't know how it works



Carol

Our old accounting software still runs on VmWare. It is what it is. A Jenkins agent sits on there for deploys.



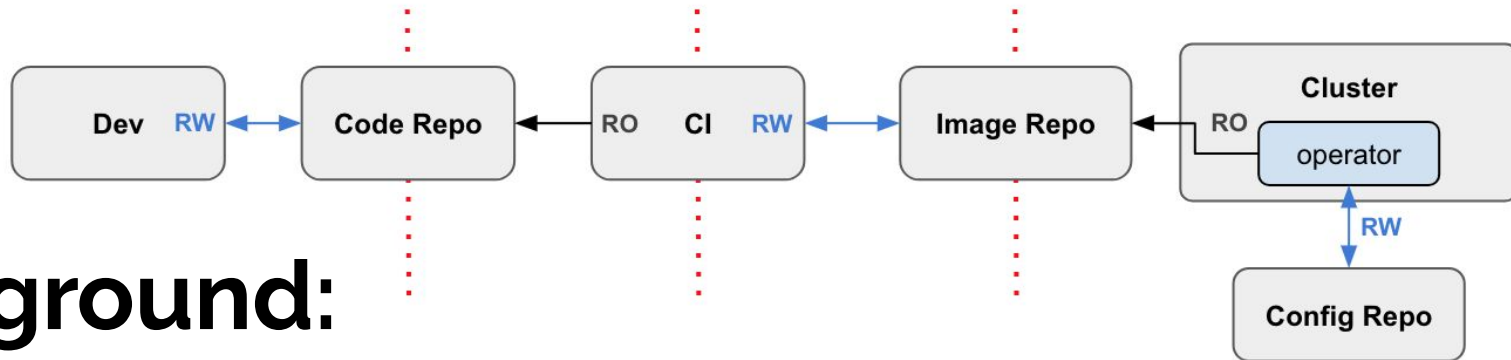
Donny

Docker on its own may not be so shiny anymore, but it works, dangit! Just give me a registry and I'll push stuff there and deploy it myself

YA KNOW I VASS NORMAL ONCE.
I CAME TO WORK VITH A TIE AND A SUIT
AND SOMETIMES PANFS.



What is GitOps
and how do
hobos relate?



Background: GitOps according to WeaveWorks

- <https://www.weave.works/blog/gitops-operations-by-pull-request>
- <https://www.weave.works/blog/what-is-gitops-really>
- <https://www.weave.works/blog/the-official-gitops-faq>

GitOps vs Infrastructure-as-Code (IaC)

Infrastructure-as-code

- Version control config for infrastructure
- Cookbooks maybe with app source
- Bare metal, VMs, Docker ...
- Provision stuff using SCM as source of truth
- Puppet, Chef, Ansible, etc
- Trigger convergence via some process
- Segregate access to things to some degree

GitOps

- Opinionated IaC
- Cloud native
- Almost certainly Kubernetes
- No direct actions - only modify config in SCM
- Uses a “GitOps Operator”
- Operator converges infrastructure
- Strong access segregation

GitOps is not “CiOps”

“You got your CI on my CD!”

Strong separation between CI and CD - not old school Jenkins deploy jobs attached directly to your CI jobs

- <https://www.weave.works/blog/kubernetes-anti-patterns-let-s-do-gitops-not-ciops>

But we use CI and CD jobs now!

Well, for the teams that actually
have any sort of CD to begin with ..

Your teams are all over the map.
That's okay! Grab the best of each
world, it is okay if the result looks a bit
messy - to begin with

GitOps vs Hobo GitOps

Hobo GitOps

- Using IaC for configuring Jenkins itself
- Allow teams to run the show, wherever
- Ideally at least Docker, maybe Kubernetes
- Direct access on the left, full-auto on the right
- Uses environment tiers, propagate via SCM
- Utility jobs in Jenkins automated via Job DSL
- Access segregated between tiers
- Provides “hooks” for teams to DSL into
- Could support teams using True GitOps

GitOps hobos live under the on-ramp
that leads to True GitOps adoption

True GitOps

- Opinionated IaC
- Cloud native
- Almost certainly Kubernetes
- No direct actions - only modify config in SCM
- Uses a “GitOps Operator”
- Operator converges infrastructure
- Strong access segregation

Only works for truly mature teams

Tools

Jenkins remains the king of flexibility
It may not always be glamorous work, but it works!



- Jenkins Job DSL - <https://github.com/jenkinsci/job-dsl-plugin>
- Layered seed jobs in different Git repositories - <https://github.com/Cervator/GitOpsUtilityJobs.git>
- Manifest repositories
 - Hobo GitOps manifests - for Jenkins - <https://github.com/Cervator/GitOpsManifest>
 - Environment manifests - for dev team apps
- Local Jenkins dev workspace for easy testing - <https://github.com/Cervator/modern-jenkins>

**SMALL
STARTING
TEAM**



Team X

Infrastructure As Code
Jenkins development & testing
Groovy/pipeline scripting

**GROWING
ENTERPRISE**



X



Y



Z

Team differentiation
Shared utility
Empower teams to self-configure

**FULL
HOBO
GITOPS!**



X

X

X



Y

Y

Y



Z

Z

Z

Segregated tiers for access control
Specialized masters (i.e. lab tier)
CD via PR merging (GitOps)



?????

Horrible cloning experiment went awry
Formalize things via manifest
Start identifying reuse potential

Alice is ahead!

How about sharing with the class?

- Codify customizations in Git
- Prepare generic DSL utility
- Adopt manifest repos for CI
- Prepare True GitOps “hooks”



We're fully on Kubernetes
using the WeaveWorks Flux
Git Operator. Our CI builds
run out of a heavily
customized Jenkins

Bob is, uh, Bob

Learn from the class. Slowly

- Get a fresh new lowest denominator Build Jenkins
- Get some code metrics and analytics going
- Plug in where the code lives and where it goes via manifest



I majored in theater. Why am I even here? Oh, I have an old server that hosts a website. I don't know how it works

Carol is cool

But can get cooler

- Adopt useful build stage stuff
- Get on the Hobo train with tiers
- Segregate access and version releases better
- Prepare for the future



Our old accounting software still runs on VmWare. It is what it is. A jenkins agent sits on there for deploys.

Donny dances

Close to the fire

- Add structure with better CD
- Set up Docker-heavy CI with more testing and analytics
- No more “Whoops that was meant for Dev, not Prod”
- Maybe k8s soon? Alice has a cool GitOps operator ...



Docker on its own may not be so shiny anymore, but it works, dangit! Just give me a registry and I'll push stuff there and deploy it myself

What to manage

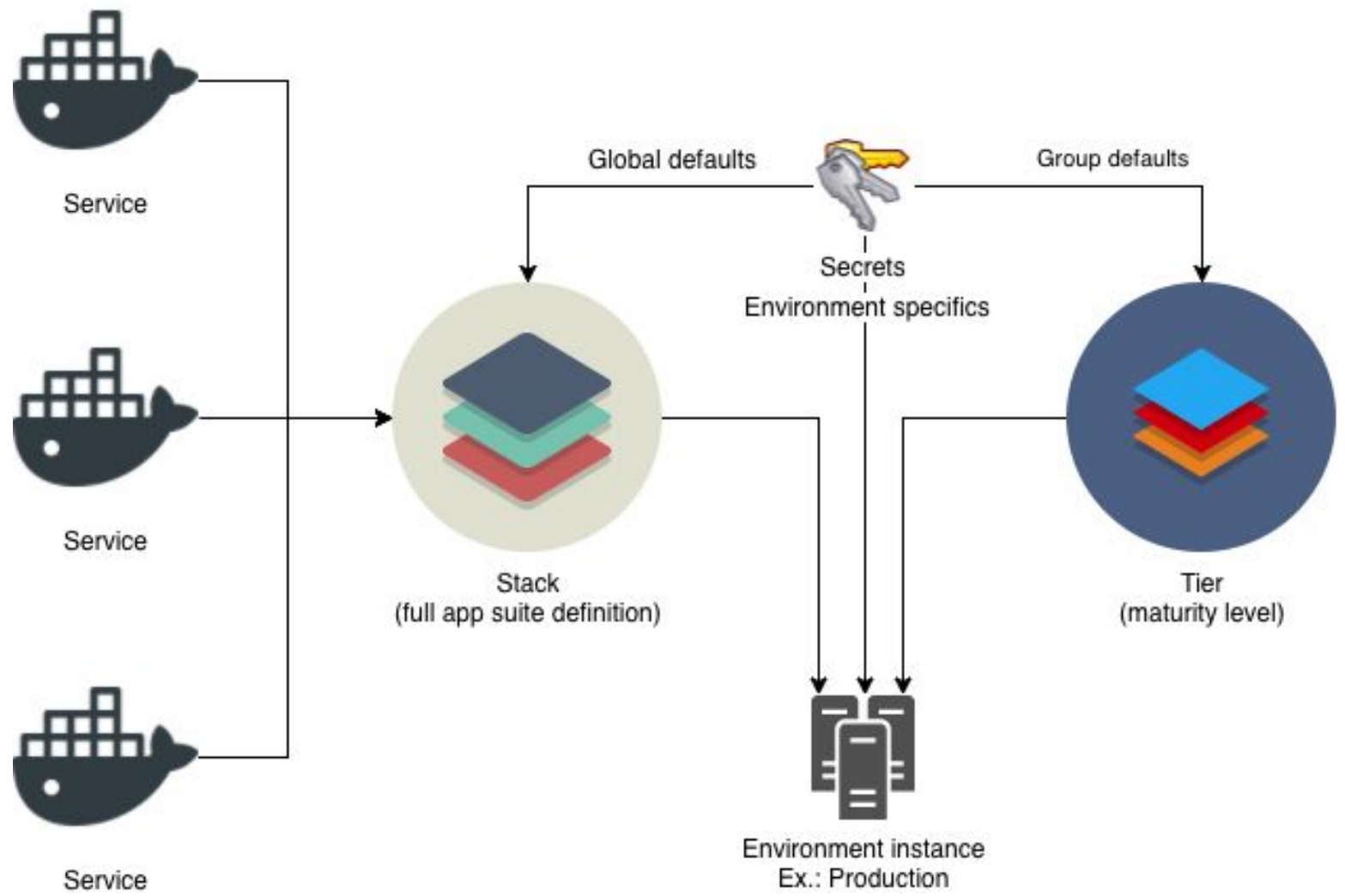
Where, why, and who?

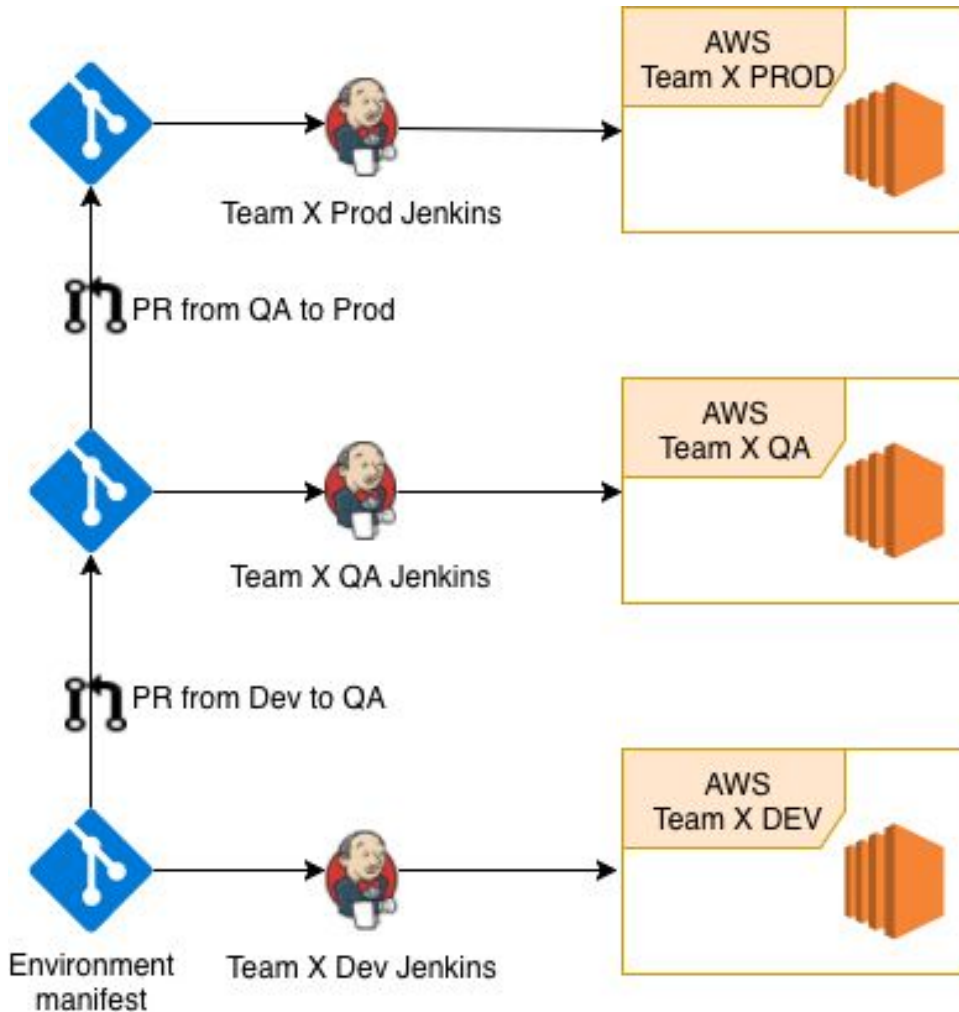


Stacks!

Are things with use

Have parts
& tiers
& envs





Stacks go to environments

They can live in maturity tiers

Manifest repos define what's where

Promotion done via PRs (GitOps)

Jenkins watches

And triggers ... something



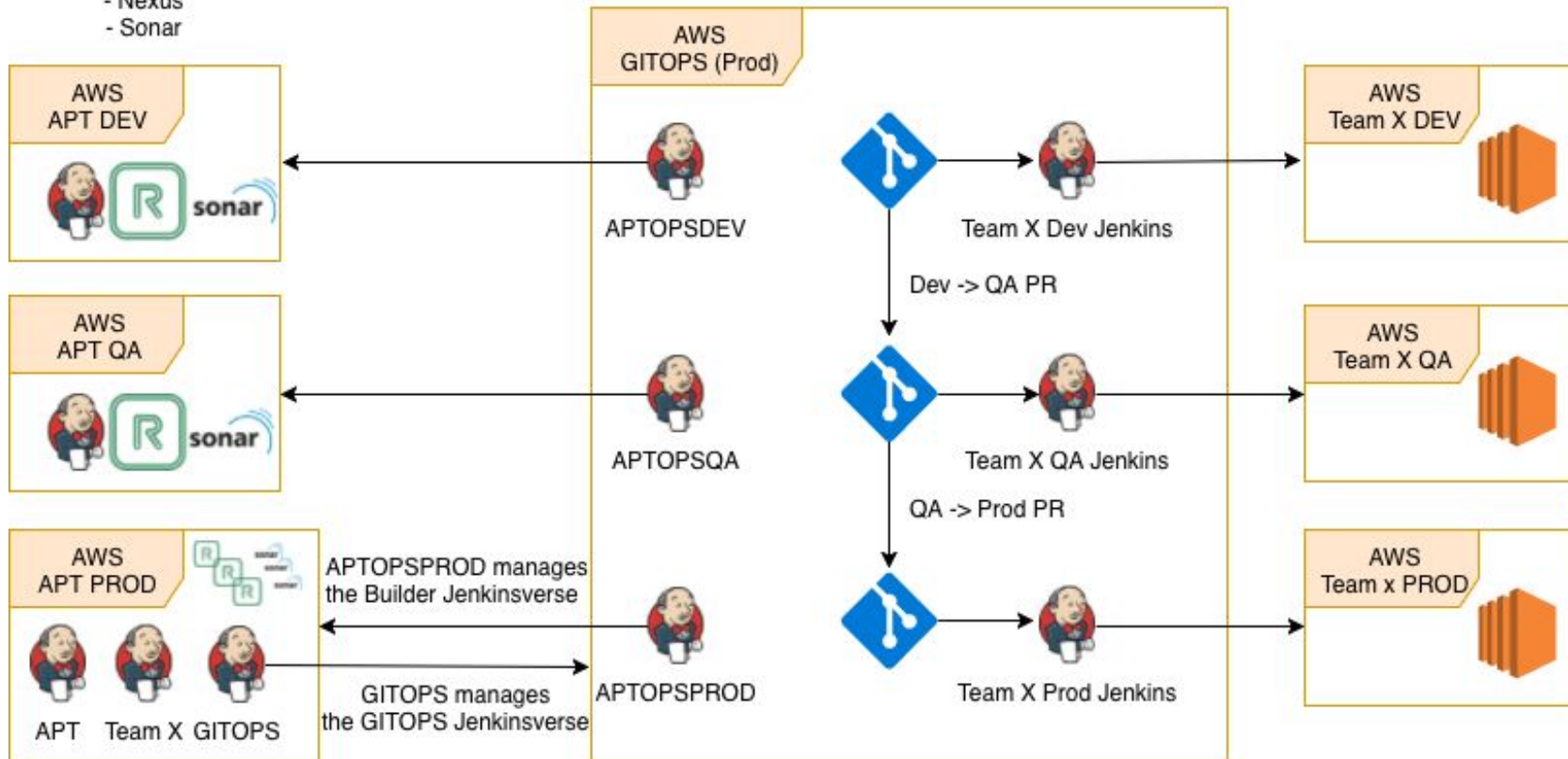
Stack
"CI toolchain"
- Jenkins
- Nexus
- Sonar



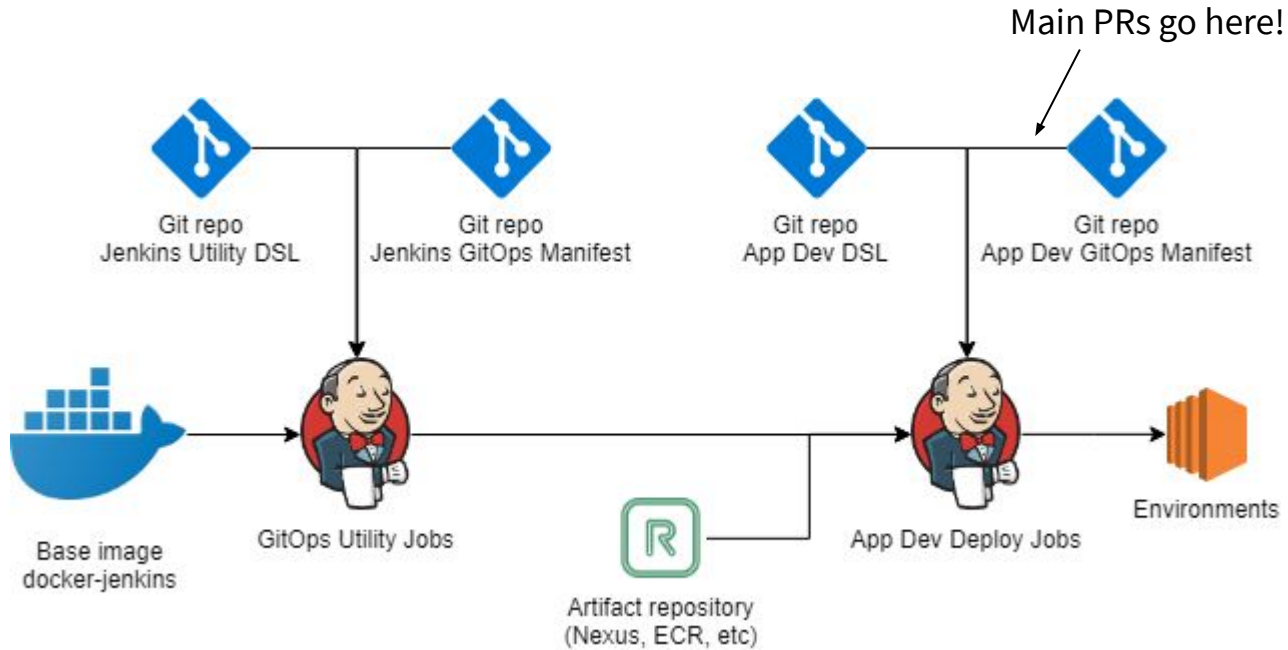
Stack
"Hobo GitOps toolchain"
- Jenkins



Stack
App Team X
- ???



High level view of involved pieces and the flow between them



Okay that was a lot of diagrams

Lets dial it back a bit

In short, you can do a lot

It varies on your needs

Who are your hobos?

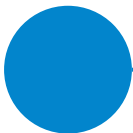
What do they want?

What can be shared?

How it works

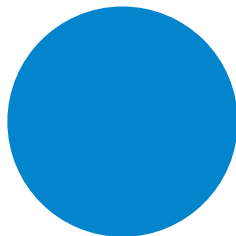
Step 1

Set up some minimal
Hobo GitOps structure



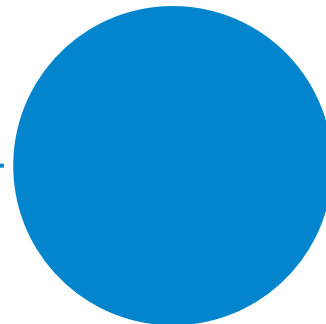
Step 2

Snare in pilot teams
(you're in yourself!)



Step 3

Grow the circle. Build
shared libraries of stuff.
Achieve True GitOps for
the teams able to make it



The Future!

What's next?



Possible hooks and other improvements

Any of this stuff sound a little bit like Jenkins X or Cloudbees Core (CBC)?

- Manage multiple Jenkins X clusters with Hobo GitOps
- Upgrade to Jenkins CasC
- Hook in other Git Operators as adapters for compatible teams (like Flux)
- Manage tools beyond Jenkins itself (Nexus, Sonar ...)
- Improve dev workspace to better handle multiple concurrent efforts
 - (teams, clients, stacks, sites, tools ...)
- More utility jobs, like a plugin de-conflictizer (Hobo CBC BeeKeeper)
- More samples like a fully Hobo GitOps managed app
- Option to use headless Jenkins for CD (some jobs really don't need the UI)