

# The Seven Deadly Sins *of GitOps*

Koray Oksay

Kubernetes Consultant @ Kubermatic





**Koray Oksay**



Consultant and Trainer @Kubermatic



Working remotely from Istanbul



CNCF Ambassador and Kubestronaut

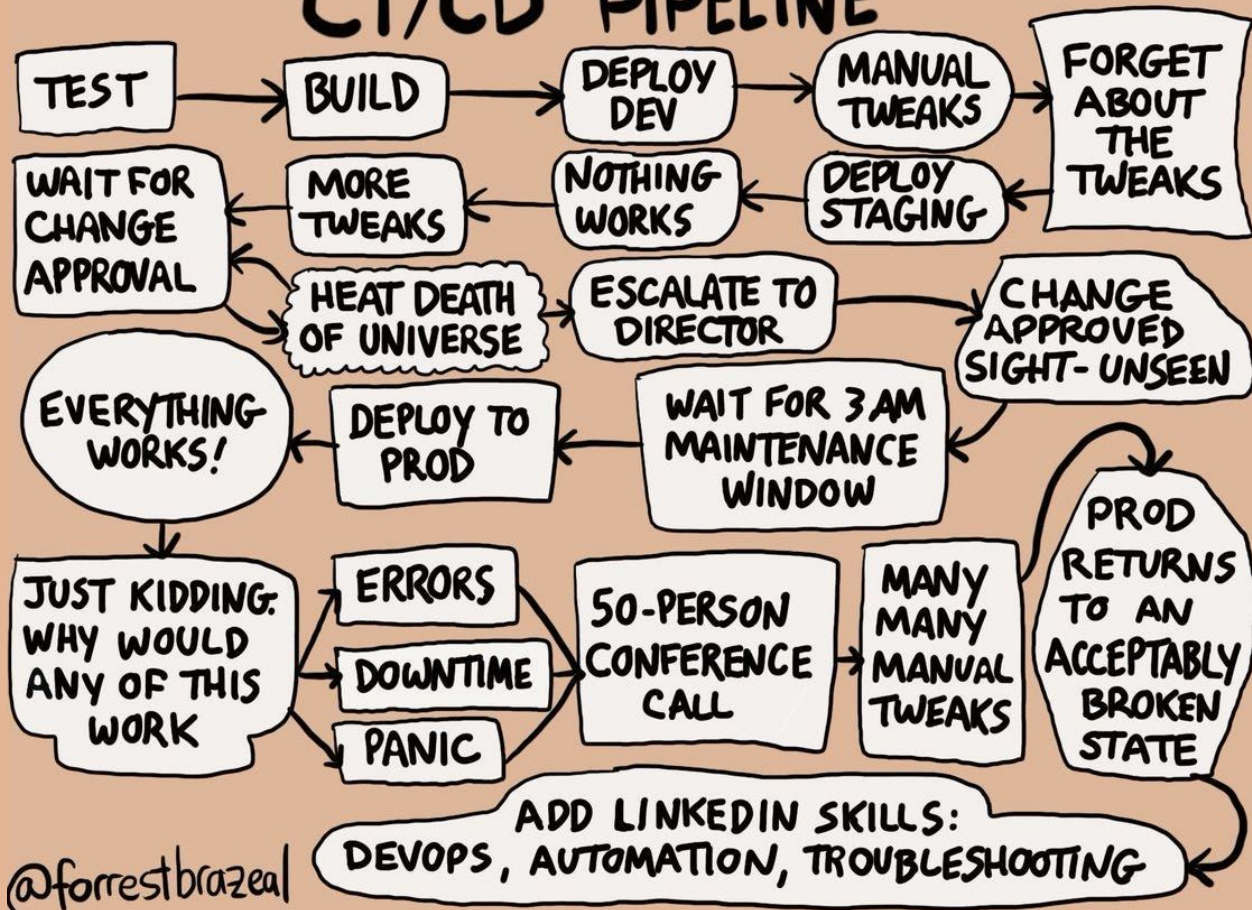


Kubernetes Contributor: SIG K8s Infra  
and K8s conformance testing



KCD and DevOpsDays Istanbul  
Organizer

# CI/CD PIPELINE



@forrestbrazeal

# The Promise of GitOps

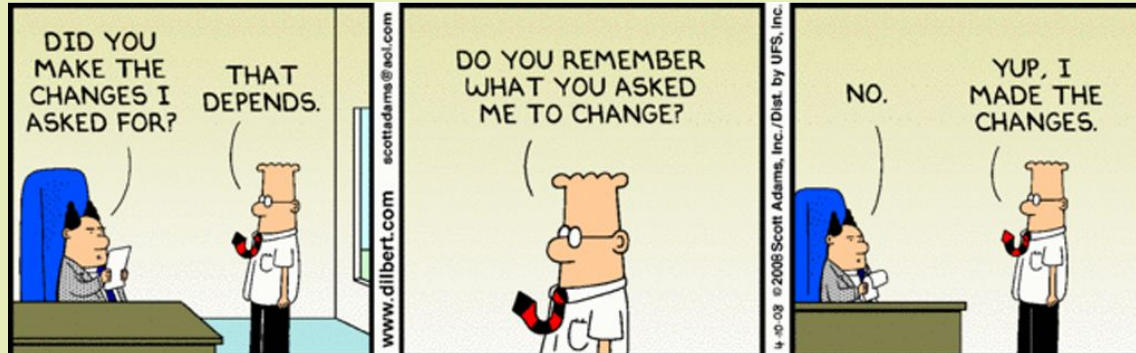
*It's not a 'set it and forget it' solution; it's a living, breathing system.*

GitOps Principles:

1. Declarative
2. Versioned and Immutable
3. Pulled Automatically
4. Continuously Reconciled

..that provides:


- Simplicity
- Consistency
- Automation
- Single Source of Truth





# Sin #1: PRIDE

*or*

*We know better than the system!*

 **Mistake:** The belief that your Git repo always reflects reality, leading to complacency


 **Consequence:** Inconsistency, non-reproducible environments, "works on my machine" for production


 **Redemption:** Monitor for drift, automate reconciliation, process for hotfixes


# **Sin #2: ENVY**

*or*

*Blindly Copying Other GitOps Setups*

 **Mistake:** Desiring another org's "perfect" GitOps setup without understanding your own context

 **Consequence:** Over-engineering, unnecessary complexity, frustration, slowed adoption

 **Redemption:** Start simple, understand your needs, Adapt instead of just copying




# Sin #3: SLOTH

*or*

*Ignoring Reconciliation Errors*

 **Mistake:** Laziness in addressing errors reported by your GitOps engine, letting them pile up

 **Consequence:** Accumulation of technical debt, masked critical issues, eventual system failure

 **Redemption:** Alerting and monitoring, automated remediation where appropriate

# Sin #4: WRATH

*or*

*Blaming GitOps for Non-GitOps  
Problems*

 **Mistake:** Expressing frustration at the GitOps system when the root cause lies elsewhere

 **Consequence:** Misdiagnosis, wasted time, undermining trust in the GitOps approach

 **Redemption:** Holistic monitoring, structured debugging, collaboration

# Sin #5: LUST

or

## *Obsessive Feature Chasing*





**Mistake:** An insatiable desire for the latest tools & features without a clear need



**Consequence:** Instability, increased complexity, team burnout, lack of consistent understanding





**Redemption:** Define your reqs, stability over novelty, controlled experimentation


# Sin #6: GLUTTONY

*or*

*One Repo to Rule Them All*

 **Mistake:** Consolidating all app and environment config into a single, monolithic Git repository for perceived simplicity

 **Consequence:** Slow operations, frequent merge conflicts, long sync times, and eroded team autonomy

 **Redemption:** A multi-repo strategy; each app has its own repo, enabling isolated deployments and clearer ownership




# **Sin #7: GREED**

*or*

*Hoarding Knowledge*

 **Mistake:** The selfish accumulation of GitOps knowledge by a few individuals, leading a lack of transparency

 **Consequence:** Bus factor of one, tribal knowledge, slow onboarding, increased risk of errors

 **Redemption:** Documentation, knowledge sharing sessions, automate setup

# Key Takeaways



GitOps is a discipline, not just a tool choice.



Always watch for drift.



Simplicity > complexity.



Don't blindly follow, design what fits your team.



Observability is not optional.



Reconciliation is the heartbeat, don't ignore it.

# Thank you!



[koray@kubermatic.com](mailto:koray@kubermatic.com)



[linkedin.com/in/korayoksay](https://linkedin.com/in/korayoksay)



[@koksay](https://github.com/koksay)