Ten Ways to Doom Your DevOps

Gene Gotimer, DevOps Engineer

@OtherDevOpsGene

Praeses, LLC

DOOM 1

Slow onboarding

It takes days/weeks/months for a new team member to contribute meaningfully to the team.

Makes them feel like a burden and/or bores them.
Sets the expectation of pace.
Shows them how important we consider them.

- Have a laptop or workstation ready
- Install the necessary tools
- Grant the necessary permissions
- Pair or mob or ensemble,
  and practice it ahead of time

DOOM 2

“Cloud”

We run our data center in the cloud instead of leveraging the cloud.

Ignores the advantages of the cloud.
So there are no advantages of the “cloud.”

- On-demand self service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

DOOM 3

“Agile”

Our organization has standardized the way teams will be agile, and there is no flexibility to change.

But we have a standup for an hour every day!
Agile is supposed to encourage agility.

- Individuals and interactions over processes and tools
- Responding to change over following a plan
- **Get rid of the status meetings, status reports, status rollups, and monthly summaries of status**

DOOM 4

Change control board

Every production change must be approved by a group that has no idea what we do.

Can’t release multiple times a day if it needs approval from a group that meets multiple times a month. What value are they adding?

- Disciplined following of process
- Pull request approvals
- Quality gates
- Principle of least privilege
- **Infrastructure as code**
- **Immutable infrastructure**

DOOM 5

Afraid to fail

A failed experiment means we'll have a lot to explain to management.

No failure means no learning.
We can't get different results if we don't change what we are doing or how we are doing it.

- Aggressively and ruthlessly protect psychological safety
- Strive for a generative culture (Westrum)
- Reduce/constrain the blast radius
- Be loud and public about learning from experiments – both good and bad

DOOM 6

Unchanging culture

We'll do anything to get to DevOps, but we aren't allowed to change our culture.

DevOps is a culture, not a set of tools or practices. We can't get the benefits of DevOps if we restrict change to only within the team.

- "If you don't manage your culture, your culture will manage you."
- Evangelize to other teams- make them want to have what our team has
- **Find a champion- someone to keep us from getting into trouble for doing things differently**

DOOM 7

Unchanging policies

We'll do anything to get to DevOps, but we aren't allowed to change our policies.

Existing policies are often designed around infrequent changes.
We can't get the benefits of DevOps if we restrict change to only within the team.

- Find out why the policy exists and use automation to generate evidence that we are meeting the intent
- **Show how our process reduces risk via automation rather than fear of getting caught**

DOOM 8

Too many decision makers

Every decision means meeting with multiple stakeholders to get consensus.

If multiple people are responsible for a decision to be made, then no one is responsible.
Often goes back to being afraid to fail.

- Find a champion
- Highlight some early wins to make them want to take credit and responsibility
- A generative culture would be even better

DOOM 9

Too many metrics

We can’t do that or it will hurt our monthly/quarterly/yearly numbers.

“When a measure becomes a target, it ceases to be a good measure.” - Goodhart

- Metrics should measure desired outcomes
- Too many metrics in a complex system leads to management whack-a-mole
- Pick a target metric, improve it, and add process to protect it.
- Then, pick the next metric

DOOM 10

Unrealistic expectations

All our problems will disappear as soon as we get DevOps.

DevOps won’t make problems vanish – it usually just shines a light on them.

- We will need more discipline, not less
- We will fail more often, not less
- Pick what we want to improve, and work until that is better
- Then pick the next thing

SUMMARY

Ten Ways to Doom Your DevOps

1. Slow onboarding
2. “Cloud”
3. “Agile”
4. Change control board
5. Afraid to fail
6. Unchanging culture
7. Unchanging policies
8. Too many decision makers
9. Too many metrics
10. Unrealistic expectations