Continuous Build and other DevOps anti-patterns, and how to overcome them


Tom Stiehm
@ThomasStiehm
About Tom Stiehm

• CTO of Coveros, Inc
• Agile Practitioner since 2002
• Worked in Application Security since 2004
• DevOps Practitioner since 2012
• Works at the intersection of Agile, DevOps, and Application Security
Why Anti-Patterns are bad

The insidious thing about anti-patterns isn’t that they fail outright but that they work in the short term while causing long term failure and pain.
Don’t let perfect be the enemy of good

“In some cases, even the anti-patterns are better than not doing it at all. It is important to recognize this and not kid yourself into a false sense of security” – J. Richard Mills
Anti-Pattern Format

✗ Name of Anti-Pattern
✗ Symptoms & Consequences

✓ What to do instead
Never changing the incentive system

- What gets you promoted
- What gets you a bonus
- What gets you fired
- No change in incentives, no change in culture

Change your incentive system to reflect what you want
Continuous Build (CB)

- Using a Continuous Integration (CI) server without Tests or Static Analysis
- No CI feedback on build health
- No feedback in integration health

Put the confidence back into CI with tests and analysis
DevOps, the third silo

- One way to get DevOps started
- Don’t let it become permanent

- Start with an end goal and date in mind
- Stick to them

https://web.devopstopologies.com/
Never asking the users about the software

- Ignoring the most valuable feedback
- It is arrogant to think you know more than your users

- On a regular basis talk to your customers
- Watch them use the software
Don’t automate that, it is my job

⚠️ Automating the wrong things for the wrong reasons.

⚠️ Objections to automating certain tasks

✔️ Pick criteria for selecting tasks to automate

✔️ Stick with the criteria

Focusing on Dev more than Ops

Dev\textsubscript{Ops}

- Build, test, and deployment is automated
- No or few production feedback systems in place
- Prod remains hard to maintain and troubleshoot

- Include Ops concerns from beginning
Automation until Production

- Automated processes with manual processes in production
- Manual processes in production seen as separation of concerns
- Or configuration management requirement

Use the same automation in all environments
Only automating build & deployment, not testing

- Cargo Cult #DevOps hallmark is no test automation
- No test automation, no confidence in the software

- Start test automation from the beginning
Turning unit tests off to build the release

- Special process for creating a release.
- No “release build” feedback on quality
- Minimal perceived value of feedback

- All builds are release candidates
- Use the same process for all builds
Reduce quality checks to pass a build

- Reducing coverage or static analysis gates to get a build to pass
- Believe that build results don’t reflect quality
- Under too much pressure to deliver

- Passing quality checks builds confidence
- Focus on delivering high quality increments
Different build processes for environments

Wrong:

- Having a different build process for developers and the CI Server. Sometimes even different build tools.
- Different processes have different results, you don’t know that your builds are the same

Right:

- Use the same build tool and process for all builds that the team does.
Different deployment process

Different deployment processes for different environments

Production only deployment processes

Processes that are never tested will fail.

Same goes for provisioning

Use the same provisioning and deployment processes in all environments

The same deploy process means your process improves quicker since it is focused on one process

@ThomasStiehm #Agile2019
No production-like test environment

🚫 Not testing in an environment configured like production
🚫 You are putting unproven software in production
🚫 It will fail at some point

✅ Create prod-like test environment
✅ Use canary releases to test in production

https://memegenerator.net
Saving security testing for the end of a release

❌ You AppSec posture will be poor, no time to remediate all issues

❌ Guaranteed to put vulnerabilities into prod

✔ Build security testing into your pipeline from the start

✔ Take the time to triage and remediate issues
Running security analysis and never using it

- Never looking at or acting on the findings
- Cargo Cult #DevOps loves to fill checkboxes
- Security concerns are minimized or ignored
- Security issues will be negotiated into production

✔ Triage and fix security finding from the beginning

@ThomasStiehm #Agile2019
Saving performance testing to the end

No time to fix issues
User retention affected
Performance testing is the first time you see a performance problem

Build performance indicators into the tests you run all the time
Create and performance testing from the beginning
Not having retrospectives

- No Value, No Retro
- Nothing changes
- Retro is only a complaint session

- Try different retro formats
- Select action items for team to fix
- Mix it up with team building

<table>
<thead>
<tr>
<th>No value</th>
<th>Great Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lot of fun</td>
<td>Great time</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No value</th>
<th>Great value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boring</td>
<td>Boring</td>
</tr>
</tbody>
</table>
Restricting retros to only development

- No Ops in retro
- No interest in improving Ops
- Focus on Dev only practices
- Without feedback other parts of the team, the team will stagnate

✅ Include everyone in product value stream
Questions?

• Join me on the TechWell Hub
• https://hub.techwell.com/
• #devops