Is your DevOps ‘tool tax’ weighing you down? How can a single application across the lifecycle help

Brandon Jung
VP Alliances, GitLab
Open source business models

Joseph Jacks
Founder and General Partner of OSS Capital

https://medium.com/open-consensus/3-oss-business-model-progressions-dafd5837f2d
Values that make a difference

Collaboration

Transparency
From idea to production: 200% faster
Aspiration: Shift-left

“Consumers want to interact with us in new, different ways. Shifting left enables us to fix problems earlier and be competitive in creating digital experiences quickly... It’s table stakes.”

Michael, Director of Product Integrations
S&P 500 Telecommunications, Mass Media

Listen to live broadcast
Problem: Tool tax

“Our feedback cycle took two weeks before we consolidated. It would have taken us 10 plus people to manage multiple tools.”

Michael, Director of Product Integrations
S&P 500 Telecommunications, Mass Media

Listen to live broadcast
Solution: Fast results

“We now do 1000s of deployments per week to our front-end with GitLab. Our GitLab team is 2 people and they support 1000 users. The feedback life cycle went from 2 weeks to seconds.”

Michael, Director of Product Integrations
S&P 500 Telecommunications, Mass Media

Listen to live broadcast
Why is this relevant?
Cycle time compression may be the most underestimated force in determining winners & losers in tech.

— Marc Andreessen
Full SDLC completed 2016
From Dev to DevOps
The Toolchain Crisis Leads to the DevOps Tool Tax

Different teams are using unique set of tools and integrations
Integration complexity of toolchains slows down teams: Integration Tax

Plan
Create
Verify
Package
Release
Configure
Monitor
Secure

Portfolio mgmt
Issue tracking
Version control
Code review
Continuous integration
Container registry
CD/Release automation
Configuration Management
Monitoring
Security testing

https://about.gitlab.com/sdlc/#interfaces
Traditional DevOps toolchain: Developer Tax

Microsoft Word

- One person edits at a time
- Multiple copies (hand offs)
- Version conflicts
- Waiting for feedback
- Sequential
What could it be like?

**Microsoft Word**
- One person edits at a time
- Multiple copies (hand offs)
- Version conflicts
- Waiting for feedback
- Sequential

**Google Docs**
- Many people edit at the same time
- One copy
- No conflicts
- Real time feedback
- Concurrent
Incident Management: Data Tax

High rate of 500 errors on API nodes

Triggered alert: All relevant alerts

Recommended runbook: All relevant runbooks, Create runbook

Postmortem
Timeline
Alerts
Deployments
Similar incidents
Runbooks

Write a comment or drag your file here...

Status page set to Investigating by Chloe Williams @khowilliams - 7 minutes ago

Chloe Williams @khowilliams - 1 minute ago

I’m now looking into the root of the problem so I’m updating the status page to “investigating”.

Elexis Willis -3 minutes ago via Slack

If this is enough to cause a problem where is the appropriate place for rate limiting? I would suggest we rate limit, as there is not a legitimate reason that I can think of for this many (22 in a short interval) clone requests.

Lonnie Garcia @lorniegarcia - 15 minutes ago

I’ve verified that this issue is currently a problem.

I wrote that we open up a feature request, if one does not exist currently to provide the necessary options into the application, I can’t think of anything our team can really do in situations such as this.

Acknowledged by Carlos Mills @carlosmillos - 29 minutes ago

Acknowledged by Lonnie Garcia @lorniegarcia - 30 minutes ago

4 responders notified by PagerDuty - 32 minutes ago

Created from alert: High rate of 500 errors on API nodes by GitLab Bot - 32 minutes ago.
Ops flow: Incident Management

Triggered alert

Recommended runbook
Ops flow: Incident Management

People
Incident manager, assignees and responders

Manager
Chloe Williams @chloewilliams

3 assignees

Responders 4 paged
Ops flow: Incident Management

- Chat, video, and status page
- Join the incident Slack channel and Zoom call, update your public status page
Ops flow: Incident Management

Timeline
Status page, comments and Slack messages, responder updates

Status page set to Investigating by Chloe Williams @chloewilliams · 7 minutes ago

Chloe Williams @chloewilliams · 1 minute ago
I’m now looking into the root of the problem so I’m updating the status page to “investigating”.

Elsie Willis @elsie · 3 minutes ago via Slack
If this is enough to cause a problem where is the appropriate place for rate limiting? I think I have a legitimate reason that I can think of for this many (22 in a short interval) clone requests.

Lonnie Garcia @lonniegarcia · 15 minutes ago
I’ve verified that this issue is currently a problem.

I vote that we open up a feature request, if one does not exist currently to provide the rate limiting. Anything our team can really do in situations such as this.

Acknowledged by Carlos Mills @carlitosmoinhos · 29 minutes ago

Acknowledged by Lonnie Garcia @lonniegarcia · 30 minutes ago

4 responders notified by PagerDuty · 32 minutes ago
Ops flow: Incident Management
Ops flow: Incident Management

Recommended runbook

- Troubleshoot API latency at api.xyz.com

Updated 1 wk ago
Ops flow: Incident Management

Timeline
Post updates and stay on top of important events

Troubleshoot with interactive runbooks
Plot graphs, run database queries, run terminal commands
Follow-up issues
Fixing merge requests
Link to related issues and merge requests

Timeline
Import events and get a clinical overview

Postmortem
Collaborative editing with templates

Ops flow: Incident Management
Ops flow: Incident Management

Overview metrics:
- Number of incidents: 84 (21%)
- Mean time to acknowledge: 12m (+17%)
- Mean time to resolve: 4h 1m (+40%)
- Average postmortem issue age: 2w 6d (-50%)

Discover root causes:
- Loudest services

Understand incidents impact over time:
- Incident volume
- Time to resolve
- Postmortem issues
Data is the lifeblood of software: Data Tax

Plan
Create
Verify
Package
Release
Configure
Monitor
Secure

Dashboard???

Portfolio mgmt
Issue tracking
Version control
Code review
Continuous integration
Container registry
CD/Release automation
Configuration Management
Monitoring
Security testing

https://about.gitlab.com/sdlc/#interfaces
Integration/Support Tax

+ Developer tax

+ Data tax

DevOps tool tax
Cloud-based, end-to-end innovation, development and production platform for financial services. Enable clients to accelerate technology

Uses unique “transparent source” model: access to ALL code

Moving to GitLab (SCM and CI) allowed them to "freeze" base code while ensuring all new code met quality, security, and documentation standards.
Values that make a difference

Convention over Configuration
Opinionated DevOps: commit your code, GitLab does the rest

CREATE
- Merge
- Build

VERIFY
- Code Quality
- Test

SECURE
- SAST
- Dependency
- Container
- License
- DAST

PACKAGE
- Container Registry

RELEASE
- Review App
- Deploy

CONFIGURE
- Infra Config
- Scale

MONITOR
- Response
- System
- Custom
- Perf Testing
Auto DevOps: a brief overview

Auto Build
Auto Test
Auto Code Quality
Auto SAST
Auto Dependency Scanning
Auto License Management
Auto Container Scanning
Auto Review App
Auto DAST
Auto Deploy
Auto Browser Perf Testing
Auto Monitoring

If there is a Dockerfile, it will use `docker build` to create a Docker image.

Otherwise, it will use [Herokuish](https://www.heroku.com) and [Heroku buildpacks](https://github.com/heroku/heroku-buildpacks) to automatically detect and build the application into a Docker image.
Auto DevOps: a brief overview

Auto Build
Auto Test
Auto Code Quality
Auto SAST
Auto Dependency Scanning
Auto License Management
Auto Container Scanning
Auto Review App
Auto DAST
Auto Deploy
Auto Browser Perf Testing
Auto Monitoring

Based on the Herokuish and Heroku buildpacks, Auto DevOps runs appropriate tests and code quality scans.
Auto DevOps: a brief overview

Multiple security scans are built into Auto DevOps:

**Static Application Security Testing (SAST)** runs static analysis on the current code and checks for potential security issues.

**Dependency Scanning** runs analysis on project dependencies and checks for potential security issues.

**Vulnerability Static Analysis** for containers runs static analysis on the Docker image and checks for potential security issues.

**Dynamic Application Security Testing (DAST)** performs an analysis on the current code and checks for potential security issues.
Auto DevOps: a brief overview

Auto Build
Auto Test
Auto Code Quality
Auto SAST
Auto Dependency Scanning
Auto License Management
Auto Container Scanning
Auto Review App
Auto DAST
**Auto Deploy**
Auto Browser Perf Testing
Auto Monitoring

**Review Apps** are temporary application environments based on the branch's code so reviewers can actually see and interact with code changes as part of the review process. Auto Review Apps creates a Review App for each branch in to the Kubernetes cluster.

**Deploys** the application to a production environment in the Kubernetes cluster.
Auto DevOps: a brief overview

Auto Build
Auto Test
Auto Code Quality
Auto SAST
Auto Dependency Scanning
Auto License Management
Auto Container Scanning
Auto Review App
Auto DAST
Auto Deploy
Auto Browser Perf Testing

Auto Monitoring

Auto Monitoring makes it possible to monitor your application's server and response metrics right out of the box. Powered by Prometheus.
Thank you!

Stay in touch:

@brandoncjung
bjung @gitlab.com
Everyone can contribute
about.gitlab.com
info@gitlab.com