Oh No, DevOps is Tough to Implement!

Hasan Yasar
Oh No, DevOps is Tough to Implement!

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.


DM18-0408
Oh No, DevOps is Tough to Implement!

Misconceptions
1. DevOps is only Dev&Ops
Dev

• Follow Agile methodologies
  - Using Scrum, Kanban and modern development approaches
  - Self directing, self managed, self organized

• Using any new technology
  - Each Dev has own development strategy
  - OpenSource,

Dev wants to deliver software faster with new requirements...

Ops &

• Operations
  - Runs the application
  - Manages the infrastructure
  - Support the applications

• Operations provides
  - Service Strategy, Design, Operations
  - Secure Systems

Ops wants to maintain stability, operations up-time...
It is more than Dev and Ops

IT Operations
- Scalability
- Infrastructure
- Maintenance
- Deployment
- Networks
- Updates
- Technical Documentation
- Release Review
- User Documentation

Developers
- Performance
- Functional Requirements
- Programming
- Testing
- Code Review
- User Interface
- Quality Assurance / Tester
- User Documentation
- Monitoring
- Security
- Intrusion Detection

Business Analyst
- Business Constraints
- User Requirements
- Legal Issues
- Market Needs
- Budgets / Timelines

Information Security
- Data Privacy
- Incident response
- Monitoring
- Security
- Intrusion Detection

Carnegie Mellon University
Software Engineering Institute

© 2018 Carnegie Mellon University

Oh No, DevOps is Tough to Implement!
2. DevOps is FAD
<table>
<thead>
<tr>
<th>IT performance metrics</th>
<th>2016</th>
<th>2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment frequency</td>
<td>200x more frequent</td>
<td>46x more frequent</td>
</tr>
<tr>
<td>Lead time for changes</td>
<td>2,555x faster</td>
<td>440x faster</td>
</tr>
<tr>
<td>Mean time to recover (MTTR)</td>
<td>24x faster</td>
<td>96x faster</td>
</tr>
<tr>
<td>Change failure rate</td>
<td>3x lower (1/3 as likely)</td>
<td>5x lower (1/5 as likely)</td>
</tr>
</tbody>
</table>

* 2017 state of DevOps report by puppet and DORA
3. DevOps is all about tools
Oh No, DevOps is Tough to Implement!

© 2018 Carnegie Mellon University
• Many tools can help you achieve your DevOps goals…

• But don’t get distracted!

• Integration and communication, even among tools, is key.

• Redundant tooling is worse than no tooling at all.
4. DevOps is Product!
DevOps is About Culture and Quality

• Early involvement of experts
  • Ops = experts in maintainability and deployability

• Complete engagement
  • Don’t bring Ops Engineers in as consultants – make them first-class team members with same success criteria as devs

• Break down organizational silos
  • Enable and require constant communication
Without a Collaborative Culture, You Don’t Have DevOps

Ask yourself:

- Do your Devs know **exactly** what **actual** production looks like?
- Does Ops know how Devs package a build?
- Is it **consistent**?
- Can both Dev and Ops collaborate on server configuration and apply it automatically to both **development and production environments**?
- Do business analysts **know the cost** of feature addition or modification?
- Can project managers measure project status **at any point in time**?
- Can the customer measure project status **at any point in time**?
5. DevOps is same for all: One Size Fits all
DevOps Requires Customization to Meet Your Unique Needs

Example: How should I configure my CI server?

- What is your project technical requirements?
- Want 90% test coverage?
  - Fail the build if code base is < 90% covered
- Want all DB queries < 2sec?
  - Test them, and fail the build otherwise
- What does quality mean to your organization?
6. DevOps is a Team & Title
Oh No, DevOps is Tough to Implement!

© 2018 Carnegie Mellon University

Dev
Ops
QA
Analysts
DevOps Breaks Down Silos

- Development Team
- IT Operations Staff
- Quality Assurance
- Analysts
7. DevOps replaces Ops (No Ops!)
Effective Teams Need Dedicated Experts

• Primary attributes of your system require
  • dedicated expert team members
    - Security, Usability, Deployability

• DevOps does not mean telling developers to learn / automate operations tasks
The SDLC is Full of Decision Points

Without Ops knowledge, developers continually make uninformed decisions, causing eventual risk or inefficiency.
The SDLC is Full of Decision Points

How many users?
Payment model?
Who is the Target Market?
Which regions?
8. DevOps is only Automation (CI &CD)
Not Everyone Needs to Achieve Continuous Deployment

• Your DevOps goals should be designed around business needs
  • Deployment frequency?
  • Production environment.
• Do frequent deployments give you a competitive edge?
• Establish traceability, visibility and collaborative environment
Oh No, DevOps is Tough to Implement!

So DevOps is…
DevOps and How it started

**DevOps** is a set of principles and practices emphasizing collaboration and communication between software development teams and IT operations staff along with acquirers, suppliers and other stakeholders in the life cycle of a software system [1]

- Patrick Debois “Agile infrastructure and operations: how infra-gile are you?”, Agile 2008 Conference

- John Allspaw “10+Deploys per Day: Dev and Ops Cooperation”, Velocity 2009

- DevOpsDays, October 30th 2009, #DevOps term born

DevOps has four Fundamental Principles

**Collaboration:** between project team roles

**Infrastructure as Code:** all assets are versioned, scripted, and shared where possible

**Automation:** deployment, testing, provisioning, any manual or human-error-prone process

**Monitoring:** any metric in the development or operational spaces that can inform priorities, direction, and policy
Oh No, DevOps is Tough to Implement!

Challenges
1. Culture
Incentivizing Behaviors

- Blame-Free Culture
  - No Hiding of Problems
  - Culture of shared responsibility
  - Collective decision and continuous learning

- Cross-Silo Goals
  - Incentivize Collaboration
  - Reduce “Not My Job”
  - Increase Sense of Purpose

- Optimize Ease-of-Use
  - Tools: Chat, ChatOps, Wiki
  - Integrated Pipelines
2. Organizational Structure
Conway’s Law:

“How to organize our teams affects how we perform our work”

- Share common goals from top to bottom
- Enable business value oriented team
- Functional Team
- Share responsibilities (like Security is everyone’s job)
- Keep team size small
3. Legacy Systems
Apply DevOps to migrate Legacy Systems

- Ancient systems should be replaced.
- Installing new systems to fit in
- Build a new version instead of maintaining
- Re-architect to support incremental and iterative development
- Enable dynamic integration of systems
4. Tools complexity

<table>
<thead>
<tr>
<th>Gt</th>
<th>Ssl</th>
<th>Gt</th>
<th>Dt</th>
<th>Lb</th>
<th>Bb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gt</td>
<td>Ssl</td>
<td>Gt</td>
<td>Dt</td>
<td>Lb</td>
<td>Bb</td>
</tr>
<tr>
<td>Gt</td>
<td>Ssl</td>
<td>Gt</td>
<td>Dt</td>
<td>Lb</td>
<td>Bb</td>
</tr>
<tr>
<td>Gt</td>
<td>Ssl</td>
<td>Gt</td>
<td>Dt</td>
<td>Lb</td>
<td>Bb</td>
</tr>
<tr>
<td>Gt</td>
<td>Ssl</td>
<td>Gt</td>
<td>Dt</td>
<td>Lb</td>
<td>Bb</td>
</tr>
</tbody>
</table>

[Periodic Table of DevOps Tools (V2)](https://example.com/table)

Follow @xebialabs
Publication Guidelines

Carnegie Mellon University
Software Engineering Institute

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.
Tools Quality Attributes

- Integrate-ability
- Interoperability
- Usability
- Portability
- Resilience
- Security/Permissions
- Availability (Error handling)
- Scalability

- Performance
- Modifiability
- Configurability
- “Automate-ability” (of manual tasks)
- “Approvability” (allows for manual approval)
- Measurability?
- Other?
4. Lack of Metrics and Measurements
Decide what to measure

- Deployment frequency
- Lead time
- # of work items (tickets)
- Defect escape rate
- Mean time to detection (MTTD)
- Mean time to recovery (MTTR)
- Application performance
5. Process Challenges
DevOps Enabler..

Establish a process to enable people to succeed using the platform to develop Secure application

Such that:

• Constant communication and visible to all
• Ensures that tasks are testable and repeatable
• Frees up human experts to do challenging, creative work
• Allows tasks to be performed with minimal effort or cost
• Creates confidence in task success, after past repetitions
• Faster deployment, frequent quality release
6. DevOps and Acquisition
Apply DevOps Mindset

Understand many portfolios of work as a continuous flow of smaller efforts

Expand the collaboration, iteration, distributed (automated) governance constructs of Agile and DevOps to acquisition, needs analysis, certification, etc…
7. DevOps and Governance
Compliance as Code

- Plan from beginning and carry-out throughout the lifecycle
- Enable audit log
- Design DevOps pipeline to comply with governance
- Make policy available to all stakeholders
- Implement configuration management and keep track every changes.
7. Inconsistent environments
Use Infrastructure as Code (IaC)

- Environment parity throughout the development pipeline
- Develop and treat provisioning scripts as part of code repository
- Share IaC amongst the developer and IT operational teams
8. Security (RMF, ATO)
Integrate Security across DevOps Lifecycle
SLS team GitHub Projects

• Once Click DevOps deployment
  https://github.com/SLS-ALL/devops-microcosm

• Sample app with DevOps Process
  https://github.com/SLS-ALL/flask_api_sample
  • Tagged checkpoints
  • v0.1.0: base Flask project
  • v0.2.0: Vagrant development configuration
  • v0.3.0: Test environment and Fabric deployment
  • v0.4.0: Upstart services, external configuration files
  • v0.5.0: Production environment

• On YouTube:
  https://www.youtube.com/watch?v=5nQlJ-FWA5A
For more information...

DevOps Blog: https://insights.sei.cmu.edu/devops
Webinar: https://www.sei.cmu.edu/publications/webinars/index.cfm
Podcast: https://www.sei.cmu.edu/publications/podcasts/index.cfm
Any Questions?

Hasan Yasar

Technical Manager,
Secure Lifecycle Solutions

hyasar@sei.cmu.edu
@securelifecycle