Deployment and Operations for Software Engineers 2nd Ed

Chapter 9—What is DevOps?

Outline

- **Introduction**
- Motivation
- Introduction to technical aspects
- Culture
- Organization
- Metrics

©Len Bass and John Klein 2022

DevOps

- DevOps is a combination of Development and Operations. The original intent was to break down walls between the two groups.
- Historically,
  - developers would deliver completed systems to operations
  - Operations would then be responsible for provisioning hardware and running the system
  - Caused tensions
- Recently the term DevSecOps has been used to indicate that security should be considered in DevOps activities

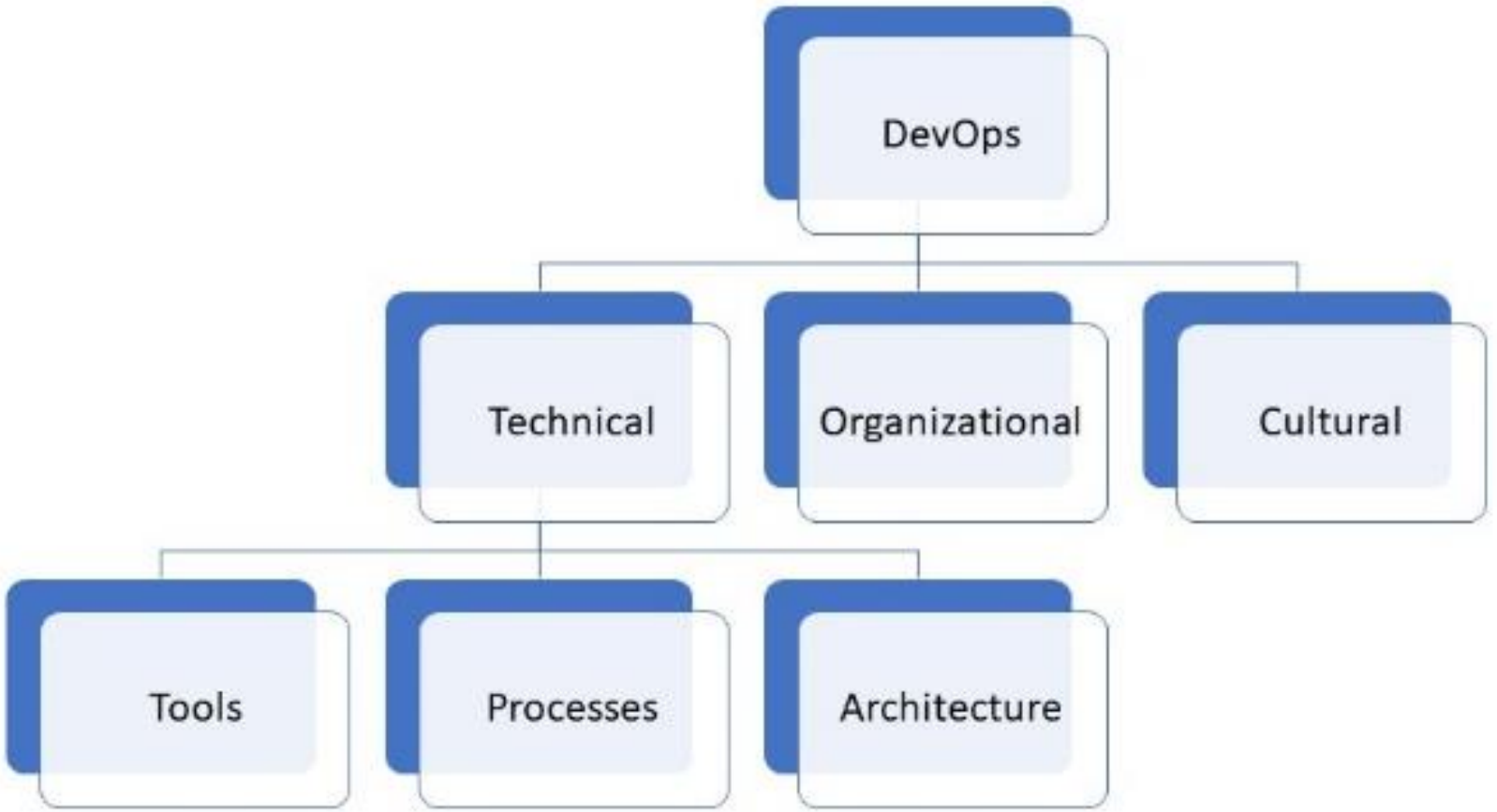
©Len Bass and John Klein 2022

Definition

- “a set of practices that aims to shorten the systems development life cycle and provide continuous delivery with high software quality”. – Wikipedia
- Definition ignores continuous deployment
- **Continuous deployment** – automated testing, automatic deployment after developer commits
- **Continuous delivery** – automated testing, deployment decision is manual. Regulations may require human sign off on deployment decisions

©Len Bass and John Klein 2022

DevOps decomposition



©Len Bass and John Klein 2022

Discussion questions

1. Have you used a DevOps tool? Which one? Does it fit the definition?
2. These days, Kubernetes is considered a DevOps tool. How does Kubernetes reduce the time to production?

©Len Bass and John Klein 2022

Outline

- Introduction
- **Motivation**
- Introduction to technical aspects
- Culture
- Organization
- Metrics

©Len Bass and John Klein 2022

Breaking down barriers is not enough

- It takes investments of time and money to implement DevOps
- Organizations must be convinced that they will recoup their investment.
- Two classes of payoffs
  - Reducing the time between the committing of code by the developers – code complete--and the placing of that code into production – deployment time.
  - Reducing the time to detect and repair problems after the code goes into production – incident handling.

©Len Bass and John Klein 2022

Traditional process for constructing or modifying large systems

- Developers are divided into teams.
- A new function or modification to an existing function is analyzed by the project management and portions are allocated to one or more teams.
- Each team develops their portion according to predefined project processes and checks the results into a version control system.
- After all of the teams have completed their portion, the portions are integrated into a complete executable, tested, and placed into production.

©Len Bass and John Klein 2022

Sources of delay - processes

- Defining project processes takes time.
- Agreement must be reached on
  - a common language,
  - versions of libraries,
  - interfaces,
  - supporting tooling,
  - coordination processes.
- Getting this agreement takes time, documenting the agreement takes time, and not all teams will remember or abide by the agreements.

©Len Bass and John Klein 2022

Sources of delay - integration

- Incorrect use of interfaces,
- inconsistent library versions,
- inconsistencies in development environments .
- Integration cannot be completed until all the teams have finished their portion and until all third-party dependencies are available.

©Len Bass and John Klein 2022

Sources of delay - testing

- The testers must define the test cases and their expected results.
- Manual testing takes time since humans are involved.

©Len Bass and John Klein 2022

Sources of delay – errors in production

- access control
- Inconsistent configuration parameters
- inconsistencies between the development environment and the production environment.

©Len Bass and John Klein 2022

Consequence of delays

- The time involved in deployment led to scheduled releases.
- Monthly, quarterly, or yearly were common.

©Len Bass and John Klein 2022

Traditional incident handline

- An operator observed or was notified of the incident.
- If repairing the incident was within the operator's ability, then it was repaired, a report was filed, and the system continued operations.
- If repairing the incident was not possible for the operator,
  - a ticket describing the incident was filed
  - the ticket was routed to developers who then took responsibility for repairing the problem.
- Diagnosing and repairing took time
- For many problems, the repair was included in the next scheduled release.

©Len Bass and John Klein 2022

Security incidents

- The number of securaity breaches has been increasing dramatically year to year.
- Over a third of such breaches are caused by configuration errors.
  - mistakes configuring the firewall
  - cloud systems and servers that were misconfigured

©Len Bass and John Klein 2022

Discussion questions

1. Which of the causes of delay in deployment have you experienced? How much time was spent identifying and repairing the errors?

©Len Bass and John Klein 2022

Outline

- Introduction
- Motivation
- **Introduction to technical aspects**
- Culture
- Organization
- Metrics

©Len Bass and John Klein 2022

DevOps processes

Release

Approve for deployment

Test

Ensure high test coverage & automate tests as much as possible

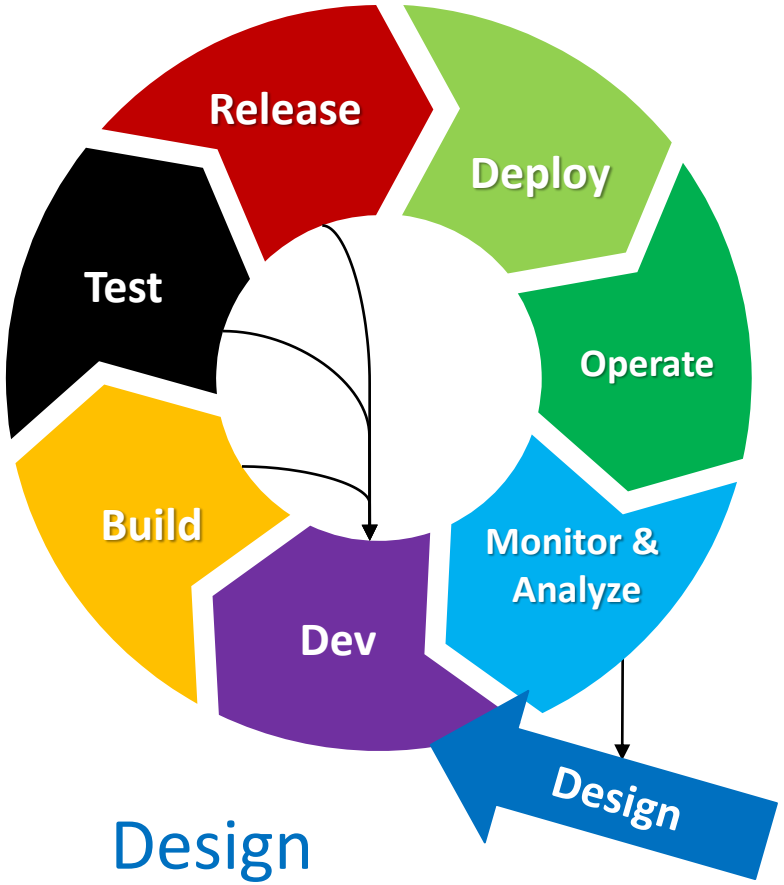
Build

Create an executable artifact

Dev

Perform normal development activities

Create scripts for other activities



Design

Design architecture to support other activities

Deploy

Move into production environment

Operate

Execute system and gather measurements about its operation

Monitor & Analyze

Display measurements taken during operation
& analyze the data

©Len Bass and John Klein 2022

Design

- The design must accommodate solutions to deployment problems and visibility for monitoring and analysis of the system in production.
- The design can also be subject to a security review

©Len Bass and John Klein 2022

Development

- DevOps automation involves a collection of scripts, and these are, typically, created and maintained by the developers.
- Static analysis can examine code created for bad security practices
- Scanning tools can examine scripts for various security flaws.

©Len Bass and John Klein 2022

Build

- An executable image is created automatically when code is checked into a version control system. Errors are returned to the developers.
- Tools can check whether any components have known vulnerabilities.

©Len Bass and John Klein 2022

Test

- Tests should be automated. Errors are returned to the developers.
- Both functional and quality tests are executed.

©Len Bass and John Klein 2022

Release

- Some domains require manual approval of a release. Others allow automatic release.

©Len Bass and John Klein 2022

Deploy

- The executable image is placed in production.
- Access and authorization functions are in place.

©Len Bass and John Klein 2022

Operate

- The system performs its functions. Data is collected while it is operating..
- Scanners can examine network connections and operation of some portions of the infrastructure for security purposes.

©Len Bass and John Klein 2022

Monitoring and Analysis

- The data collected during operation is monitored and analyzed.
- This may result in architectural modifications which are performed by returning to the design stage.

©Len Bass and John Klein 2022

Environments

- The Development, build, test, and operate stages all exist in their own environment.
- An *environment* is a collection of provisioned resources. The resources include an operating system, network connections, memory, CPU, and any software dependencies.

©Len Bass and John Klein 2022

Discussion questions

1. The various stages of the pipeline—build, test, deploy—should gather measures about their actions. What measures should be gathered?
2. What are some types of quality tests that can be done on the built system?

©Len Bass and John Klein 2022

Outline

- Introduction
- Motivation
- Introduction to technical aspects
- **Culture**
- Organization
- Metrics

©Len Bass and John Klein 2022

DevOps and Agile

- **Focus of movements**
  - DevOps has a focus both on development and operations
  - Agile has its focus on development.
- **Practice focus**
  - Agile focusses on management practices such as Scrum, Kanban, etc.
  - DevOps focuses on tools
- **Feedback**
  - Agile obtains feedback from customers
  - DevOps obtains feedback from measuring the running system and the various practices.

©Len Bass and John Klein 2022

Agile and DevOps commonalities

- Both sets of practices advocate a non-blaming, open, sharing culture.
- All members of the team should have access to the same information
- Communication channels provide for rapid sharing and feedback.
- Periodic retrospectives are advocated by both groups and problems are analyzed for the underlying causes without finger pointing and personal comments.

©Len Bass and John Klein 2022

DevOps specific cultural practices.

1. Mindset
   - Automation – always be alert to the possibility of automation
   - Measurement - self-service monitoring and alerting
2. Incentives. Incentives are adjusted so that Dev, Ops, and Sec all are incentivized to the same results. For example, uptime of new releases with no security breaches is a metric all can subscribe to and contribute to.

©Len Bass and John Klein 2022

Discussion questions

1. In your organization, how are developers incentivized? How are operators incentivized? Is there a tension between those two sets of incentives?
2. Who are the DevOps stakeholders? Who are the Agile stakeholders?

©Len Bass and John Klein 2022

Outline

- Introduction
- Motivation
- Introduction to technical aspects
- Culture
- **Organization**
- Metrics

©Len Bass and John Klein 2022

Responsibilities are “Shifting Left”

- “Shifting Left” in DevOps means that some people involved in early life cycle phases (i.e. developers) are performing tasks that tranditionally were done later in the life cycle (e.g. incident handling, acquiring resources)
- There is evidence that shifting left reduces costs. The earlier in the life cycle a problem is detected, the cheaper it is to fix it.
- Shifting left will cause changes in organizational units and add new responsibilities to existing roles.

©Len Bass and John Klein 2022

New and changed organizational units

- The quality assurance (QA) unit may disappear, or it may have its responsibilities altered.
- DevOps practices call for automated testing.
- Traditionally, the QA unit was responsible for creating and executing tests and deciding whether a system was suitable to be deployed.
- With automated tests and automated deployment, the responsibilities of the QA unit are substantially reduced if not eliminated.

©Len Bass and John Klein 2022

Incident handling

- A new unit may be created to deal with incident handling.
  - System Reliability Engineer (SREs) are the first responders when an incident occurs. They exist in a separate organizational unit that assumes responsibility for the reliability of systems.
  - Google initiated this unit, and it has been adopted in other organizations.
  - Meta (Facebook) has created a unit to perform Production Engineering. This unit's responsibilities are to "champion the Reliability, Scalability, Performance, and Security posture of production services."

©Len Bass and John Klein 2022

Creating tools

- A unit to manage and create tools has emerged in multiple organizations.
- Google and Netflix create their own tools and other organizations have a unit to manage their tool suite.

©Len Bass and John Klein 2022

New and changed responsibilities

- Traditionally
  - the operations group was responsible for resource acquisition and management.
  - A development team would create a system and the operations team would acquire resources to operate that system.
  - There are about 100 developers for every 10 operators for every security person.
- Developers now have responsibility for allocating resources and for reviewing and acting upon the monitoring information collected from operating the system.
- One model for incident handling is “You built it, you run it”. This model, introduced by Amazon, makes developers the first responders in the case of an incident.

©Len Bass and John Klein 2022

Open System Model

- Microsoft and Alibaba Cloud have announced the “Open System Model” which defines responsibilities for developers and operators in terms of platform-based tools
  - Developer builds systems in containers
  - System operator packages and deploys containers
  - Infrastructure operator tunes as necessary at run time.

©Len Bass and John Klein 2022

Discussion questions

1. Can you think of other roles affected by Shifting Left?
2. Will the ratios of developers to operators to security personnel change as a result of Shifting Left?

©Len Bass and John Klein 2022

Outline

- Introduction
- Motivation
- Introduction to technical aspects
- Culture
- Organization
- **Metrics**

©Len Bass and John Klein 2022

Metrics

- A process improvement effort must be able to measure whether the changes it introduces have a positive impact.
- Choosing the correct metrics is important since emphasizing the wrong metrics will lead to incorrect behavior.
- Also, having too many metrics makes it difficult to optimize any single one.

©Len Bass and John Klein 2022

DORA metrics

- DORA (DevOps Research and Assessment) proposes the following four metrics that have become widely adopted.
  1. Lead time for changes
  2. Change failure rate
  3. Deployment frequency
  4. Mean time to recovery

©Len Bass and John Klein 2022

Lead time for changes.

- Length of time between committing a code change to the trunk branch and when it is in a deployable state.

©Len Bass and John Klein 2022

Change failure rate

- .Percentage of code changes that require hot fixes or other remediation after production. This does not measure failures caught by testing and fixed before code is deployed.

©Len Bass and John Klein 2022

Deployment frequency

- The number of deployments into production per unit time. For example, Amazon reports thousands of deployments per day.

©Len Bass and John Klein 2022

Mean time to recovery

- . Mean time to recovery (MTTR) measures how long it takes to recover from a partial service interruption or total failure.

©Len Bass and John Klein 2022

Discussion questions

1. How are each of the DORA metrics collected.
2. Which organization role is interested in the results of each metric.

©Len Bass and John Klein 2022