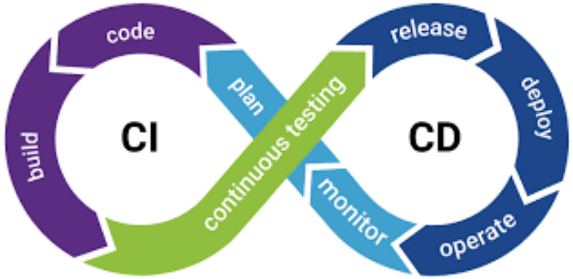
unison
Cost Engineering

Addressing Continuous Integration/Continuous Delivery with Automated Testing in Your Software Estimate

Arlene Minkiewicz

Agenda

- Introduction
- Definitions
- Continuous Integration/Continuous Delivery (CI/CD)
- CI/CD Pipeline
- Key Software Delivery Performance Metrics
- Cost Consideration for DevOps and CI/CD with Automated Testing
- Wrap Up



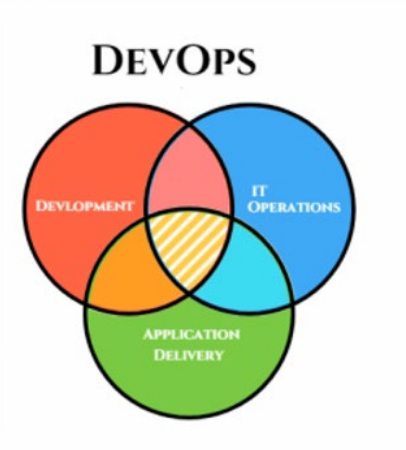
Introduction

- Agile development and DevOps/DevSecOps (Development (Security) Operations) practices are becoming increasingly common in software development projects
  - Commercial, government, government contractor organizations
- The State of the DevOps Report from 2021[1] found that 90% of respondents with highly evolved DevOps practices report:
  - Most repetitive tasks are automated
  - Automation improves the quality of their work
- Report published by Atlassian [2] on DevOps Trends report over half respondent organizations had dedicated DevOps Teams

[1] The 2021 State of DevOPs Report" available at https://www.puppet.com/success/resources/state-of-devops-report

[2] Armlin, Davc, "10 DevOPs Tools for Continuous Monitoring", ChaosSearch Blog, July 2021, available at https://www.chaossearch.io/blog/continuous-monitoring-tools

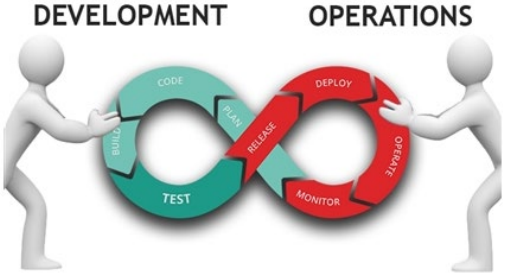
Introduction



- Key practices that enable DevOps Success are:
  - Continuous Integration
  - Continuous Delivery
  - Continuous Deployment
  - Continuous Monitoring
- Automation, tools and technologies make these practices possible
- This paper focuses particularly on
  - Continuous Integration (CI) and Continuous Delivery (CD)
  - The CI/CD Pipeline
  - Key Software Delivery Performance Metrics
  - Cost implications of the practices and technologies of CI/CD

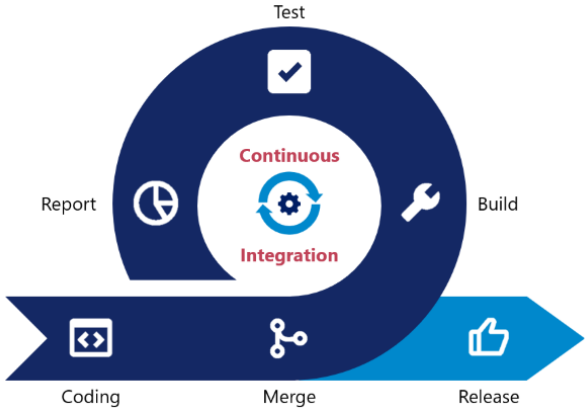
What is DevOps?

- DevOps combines cultural philosophies, practices and tools increasing chances of delivering quality with high velocity
- DevOps Approach merges the skills of ‘development’(Dev) with those of ‘Operations’(Ops) to optimize software delivery efficiency and customer satisfaction
- Focus on collaboration and communication between Dev and Ops
  - Ensuring these activities occur in tandem rather than serially
  - Requires developers to own application’s reliability and performance
  - Requires business and project leaders to provide support and encourage a DevOps culture



What is Continuous Integration (CI)?

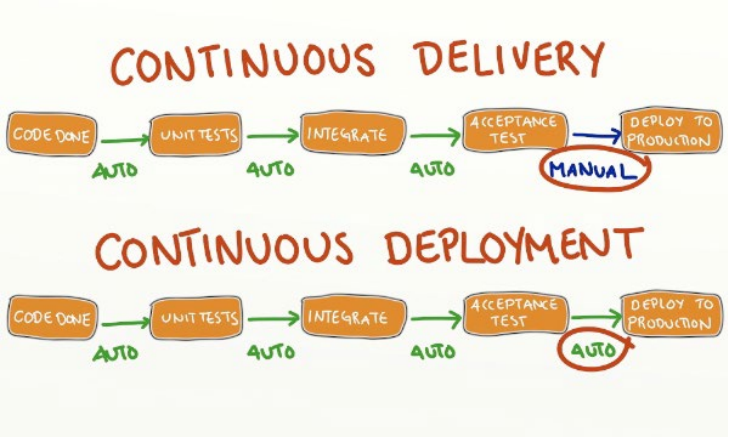
- One of the biggest success stories in automated software engineering!
- CI is the practice of automatic build and test of code changes
  - When they are committed to the code base or several times a day
  - Execution of test suites is automatic
  - Bugs and other integrations are discovered close to the source
- Benefits include:
  - Improved quality and velocity
  - Feedback from developers and stakeholders near real-time
  - Increased visibility within the team and the organization



What is Continuous Delivery (CD)?

- Extension of CI focused on automating the software delivery process
  - Teams can quickly, easily, and confidently deploy applications into production-like environments
  - Code base should always be in a ready to ship state at any time
  - Teams can quicken tempo from infrequent, big, risky deployments to ones that are frequent, small and safe
  - CD automates the steps between code commit and gatekeeper(s) decision that it's ready for production
- Benefits include
  - Faster time to market
  - Reduced Deployment failures
  - Improved visibility and collaboration between dev, ops and IT
  - Improved productivity and efficiency
  - Developers have more time to work on creative tasks

What is Continuous Deployment?



- Extension of Continuous Delivery
  - Automatically deploys builds into the production environment when tests pass
  - No human gatekeeper
- Benefits include
  - Faster Release cycles
  - Culture of continuous improvement
  - Improved productivity

What is Continuous Monitoring?

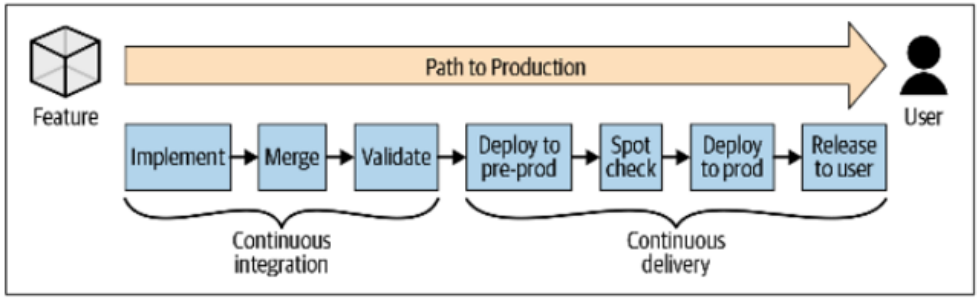
- Continuous Monitoring (CM) is the process of routinely and vigilantly scanning for performance degradations in:
  - Application Performance
  - Security
  - Compliance
- Benefits include
  - Improved security through automation
  - Early detections of performance issues
  - Reduced downtime
  - Better business performance, improved user performance
  - Increased clarity and collaboration

What is a CI/CD Pipeline?

- An organized and automated suite of tests
  - Used to ensure all software versions are up to date, compatible, and ready to use
  - Pipelines are driven by tests, services, outcomes, and people
- Core components of CD/CI pipeline:
  - Build scripts to automate the tasks necessary to build a release
  - Infrastructure elements such as development environments, platform(s), hosting services, etc.
  - Test elements to enable developers to create scripts to automate testing
  - Release elements allow for the automation of the release process
  - Validation elements to create automation for testing before release

What is a CI/CD Pipeline?

- Path to production across organizations
  - Engineers implement features and other changes
  - New features are reviewed and merged into CI/CD system
  - Build is created and automated tests validate the changes
  - Change is automatically delivered to a shared integration environment – production or production-like
  - Potentially controlled rollout to users

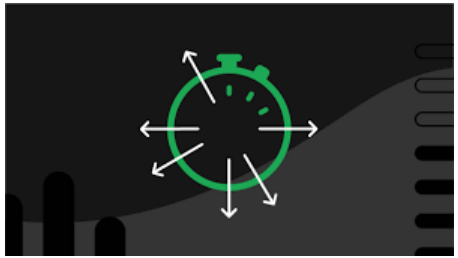


DevOps Metrics from Continuous Monitoring

- DevOps Research and Assessment (DORA)
  - Conducted largest academic study exploring DevOps principals and applications
  - Employs data driven insights to promote DevOps best practices
  - Over seven years DORA has identified four DevOps Success metrics thought to differentiate high performing DevOps teams with respect to software delivery performance
    - Deployment Frequency
    - Lead Time for Changes
    - Mean Time to Recovery
    - Change Failure Rate
  - In 2021 DORA added a 5th key metric – Operational Performance focused on measuring reliability – how well services meet user expectations such as availability and user expectations

DevOps Metrics from Continuous Monitoring

- **Deployment Frequency (measure of throughput)**
  - Number of deployments in a given time frame
  - The higher the number the better – more value delivered more often
  - To measure deployment frequency…
    - Use pipeline tool to determine total builds (TOTAL_BUILD) and successful deployment to production (DEPLOY_BUILD)
    - Development Frequency = (DEPLOY_BUILD/TOTAL_BUILD * 100)
  - 2022 State of DevOps Report from DORA
    - High performing teams deploy multiple times per day
    - Medium performing teams deploy between once a week and once a month
    - Low performing teams deploy between once a week and once every 6 months



DevOps Metrics from Continuous Monitoring

- **Lead Time for Changes (measure of throughput)**
  - Time it takes for an idea to progress from implementation to production
  - Entire lifecycle including development, testing, and delivery
  - To calculate Lead Time for Changes you need to identify start of work and its finish
  - This metric can be improved by shortening the deployment time through improved automation and test integration
  - The shorter the lead time the better the throughput
  - 2022 State of DevOps Report from DORA ….
    - High performing teams have a mean time between one day and one week
    - Medium performing teams have a mean time between one week and one month
    - Low performing teams have a mean time of between one month and six months

DevOps Metrics from Continuous Monitoring

- **Mean Time to Recovery (measure of stability)**
  - Time it takes an organization to restore a system when there has been a production failure
  - Mean Time to Recovery can be calculated by tracking average time between a defect report and a fix being deployed to production
    - CM tools can be configured to enable this tracking - providing alerts about potential problems
  - The shorter the Mean Time to Recovery, the better
  - 2022 State of DevOps Report from DORA ….
    - High performing teams average less than one day
    - Medium performing teams average less between one day and one week
    - Low performing teams average between one week and one month

DevOps Metrics from Continuous Monitoring

- **Change Failure Rate (measure of stability)**
  - Number of code changes that result in any sort of production failure
  - This can be calculated by dividing the number of deployment failures by the number of total deployments
  - Change Failure Rate can be decreased through robust monitoring and through strategies such as delivering in small increments and increased test automation
  - The lower the Change Failure Rate the better
  - 2022 State of DevOps Report from DORA ….
    - High performing teams are between 0 – 15%
    - Medium performing teams are between 16-30%
    - Low performing teams are between 46-60%

Cost Considerations for DevOps and CI/CD

- When estimating a CI/CD DevOps projects, it is important for the estimator to understand
  - Is there already CI/CD infrastructure and tools in place at an organizational or department level so initial deployment costs are not part of the project costs or …
  - If there are considerable investments that need to be made to start up the Pipeline?
- The majority of this cost discussion assumes initial investment costs are not part of the project
- The first thing an estimator needs to establish is how well entrenched the organization and DevOps teams are with CI/CD and the DevOps culture
- Teams that are new to CI/CD or are working on brand new projects are less likely to achieve maximum benefits of CI/CD

Question the Software Estimator should Ask

- How many CI/CD projects has the DevOps Team worked on?
  - If the team is brand new to CI/CD then:
    - Investments in training will be part of the project costs
    - Learning curve for adoption will be steep, impacting productivities
- How long has the DevOps team been working on the project being estimated?
  - If the project is brand new to a DevOps team, other cost considerations:
    - Investment in development and verification in automated test suites
    - Investment to prepare the test data
    - Investment to prepare the test environments
    - Cost associated with any additional infrastructure, tools or technology that will be specific to the project

Software Estimation Considerations for CI/CD

- Regardless of whether a project is waterfall, agile, DevOps, DevSecOps, or a hybrid, many cost estimating drivers don't change
- There are some agile, DevOps-specific project characteristics that should influence how to select input values most software estimating models:
- The following cost drivers should be reconsidered or added to the estimating equation:
  - Level of automation – not all tests can be automated. Questions to ask:
    - Automated test coverage (%)
    - Quality of testing tools (amt of manual intervention to run and interpret the automated test sets)
  - Project complexity – projects with more complex logic or workflows will have increased effort to develop and maintain the automated tests and may require more effort to manage the pipeline
  - Programming Language(s) – some languages are easier to interface with automation tools

Software Estimation Considerations for CI/CD (cont.)

- The following cost drivers should be reconsidered or added to the estimating equation:
  - Ongoing cost associated with infrastructure resources, tools, and technologies required for the project (licenses, support, maintenance fees, etc.)
  - DevOps Team Size
  - Integrations with third party or legacy applications and the CI/CD Pipeline. Questions to ask:
    - How many third part or legacy applications?
    - Complexity of these integrations?
  - Training or Retraining requirements – DevOps teams need to regularly get training to keep up with tools and technologies and to enable continuous process improvement
  - Security Requirements. Questions to ask:
    - Security Level (EAL, NIAP, etc.)
    - Security Process Level – formality and entrenchment of security process

Software Estimation Considerations for CI/CD (cont.)

- The DORA Key Software Delivery Metrics don't specifically cover cost
- Intuition tells us that it is likely that these metrics should be influential in cost models
- The following cost drivers should be reconsidered or added to the estimating equation:
  - Organizational Productivity – this value represents a comparison of the overall productivity of an organizations ability to deliver software efficiently. Questions to ask
    - How well has the organization embraced DevOps culture – How supportive and cooperative is the organization
    - How low is turnover for the DevOps team
    - Team score based on the Software Delivery Performance Metrics
  - DevOps Team Skill and Experience
    - Experience with the Project, Market, Programming Language
    - Team Continuity
    - Familiarity and Experience with CI/CD Pipeline
    - Team score based on the Software Delivery Performance Metrics

Software delivery performance metric	Low	Medium	High
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	Between once per month and once every 6 months	Between once per week and once per month	On-demand (multiple deploys per day)
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Between one month and six months	Between one week and one month	Between one day and one week
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Between one week and one month	Between one day and one week	Less than one day
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	46%-60%	16%-30%	0%-15%

Cost Considerations for DevOps and CI/CD

- CI/CD Pipelines when well implemented have cost, productivity and quality benefits
  - But CI/CD is not inexpensive
  - There are many types of projects where investments in CI/CD pays off but not all
- Some project factors make it less likely that the benefits will be worth the cost:
  - Projects where software deliveries are customized for individual customers
  - Projects where compliance and/or security requirements make automated testing complex or not acceptable
  - Projects that integrate many third-party solutions
  - Projects where Service Level Agreements (SLAs) require customized pipelines
  - Projects with a great deal of algorithm or workflow complexities

Wrap-Up

- DevOps success occurs when a culture is created where an organization embraces the value added to the software delivery process through automation, collaboration, continuous process improvement, rapid delivery, and better resource and cost management.
- More organizations are using CI/CD Pipelines – an organized and automated suite of tests to ensure software versions are up-to-date, compatible, and ready to use
- CI/CD Pipelines have many potential cost, productivity, and quality benefits
- There are costs associated with creating, building, and maintaining the pipeline
- There are costs associated with the fact that a project is embracing CI/CD and DevOps
- Cost considerations include:
  - How entrenched is the CI/CD and DevOps in the organization and development team?
  - How should CI/CD project influence a software cost estimate?
  - How to consider the influence of the DORA software delivery performance metrics on a cost estimate?

Backup

Cost Considerations for DevOps and CI/CD

- If a project estimate is required to include the cost of creation and implementation of CI/CD Pipeline there are additional cost considerations which are briefly discussed here:
  - Infrastructure may need to be purchased for new tools and technologies for the pipeline. This could include costs of servers, cloud services, network equipment, etc.
  - Tools may need to be purchased, if they are not available, for version control, test automation, deployment automation, etc.
  - Training will be required to teach the DevOps team how to build, use, and maintain the pipeline
  - Effort will be required for the DevOps and IT teams to build and maintain the pipeline – creating tests for the pipeline and addressing operational issues during startup
  - The pipeline needs to be integrated with other tools and systems within the organization
  - Ongoing costs associated with maintaining, upgrading and scaling the pipeline for the lifetime of the project

Continuous Monitoring Tools for DevOps

- A plethora of CM Tools for DevOps….
  - All types of monitoring
  - Various environments and platforms
  - Open source, licensed and pay per use models
- Way too many to discuss them all
- Multi-platform support (AWS, Azure, Kubernetes, Docker, etc.)
- Descriptions of several popular ones follows….

Continuous Monitoring Tools for DevOps

- **AppDynamics**
  - Application Monitoring
  - Cloud and on-premise
  - Enables optimization of applications by monitoring APIs, key business metrics and code level problems
  - Licensed with free lite version available

- **Dynatrace**
  - Application, Infrastructure and Network Monitoring
  - Hybrid and cloud
  - Artificial Intelligence (AI) powered automation monitoring cloud based PaaS and container technologies
  - Various pricing schemes

Continuous Monitoring Tools for DevOps

- **Splunk**
  - Application and Infrastructure Monitoring
  - Cloud, on-premise and hybrid monitoring
  - Automatic, continuous, real-time monitoring through application lifecycle
  - Various pricing models
- Nagios
  - Network, Application, and Infrastructure monitoring
  - Cloud, web and storage service monitoring
  - Services easily find network outages, protocol errors and provide for monitoring for audit and compliance issues
  - Open source (Nagios Core) and paid version (Nagios XI)