Introduction to DevOps

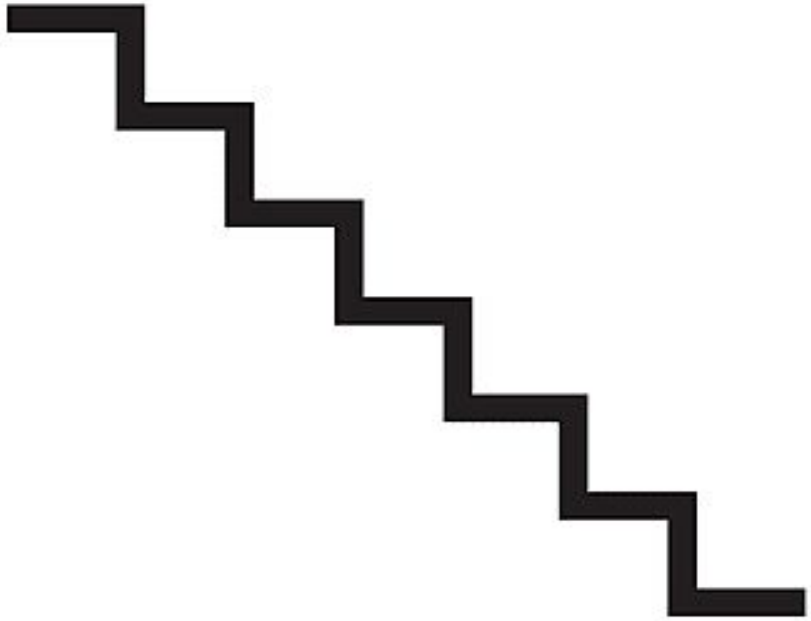
Some background …

SDLC?

Software Development Lifecycle

- Requirement Analysis
- Defining requirements
- Designing architecture
- Development
- Integration
- Testing
- Documentation
- Training
- Deployment
- Maintenance

Models …



Waterfall model

- Linear
- Large codebase

Waterfall model contd …

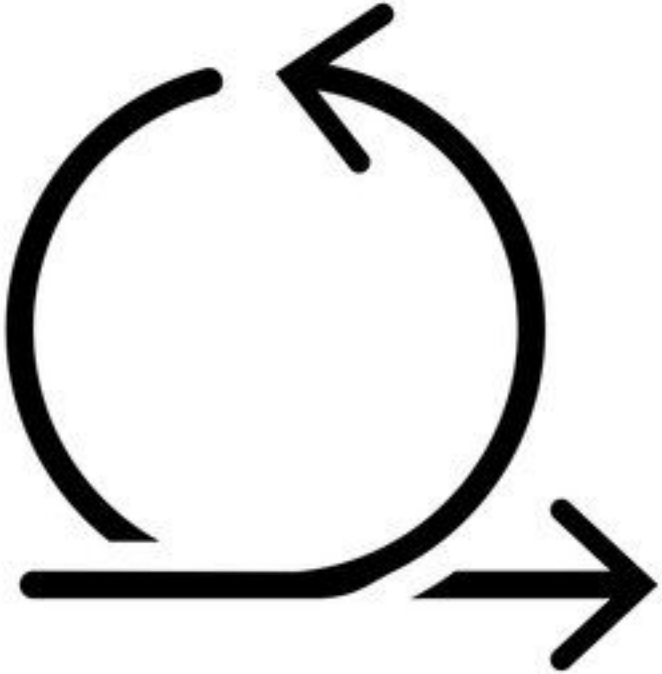
Drawbacks

- Silo role

  Development => Integration => Testing => Deployments => Operations …

- Massive release cycle
- Rigid to changes

The evolution …



Agile model

- Iterative
  - Develop
  - Deliver
- Sprints
- ~~Massive~~ Smaller release cycle
- ~~Rigid to changes~~ Continuous integration and delivery

But, …

Agile model contd …

The issue

- Organizational differentiation
  - Developers
  - Deploy and Support
  - Different KPI
- Still in silo
  - IT
  - Deployment
  - Operations
- Botched up releases
- Unhappy customers

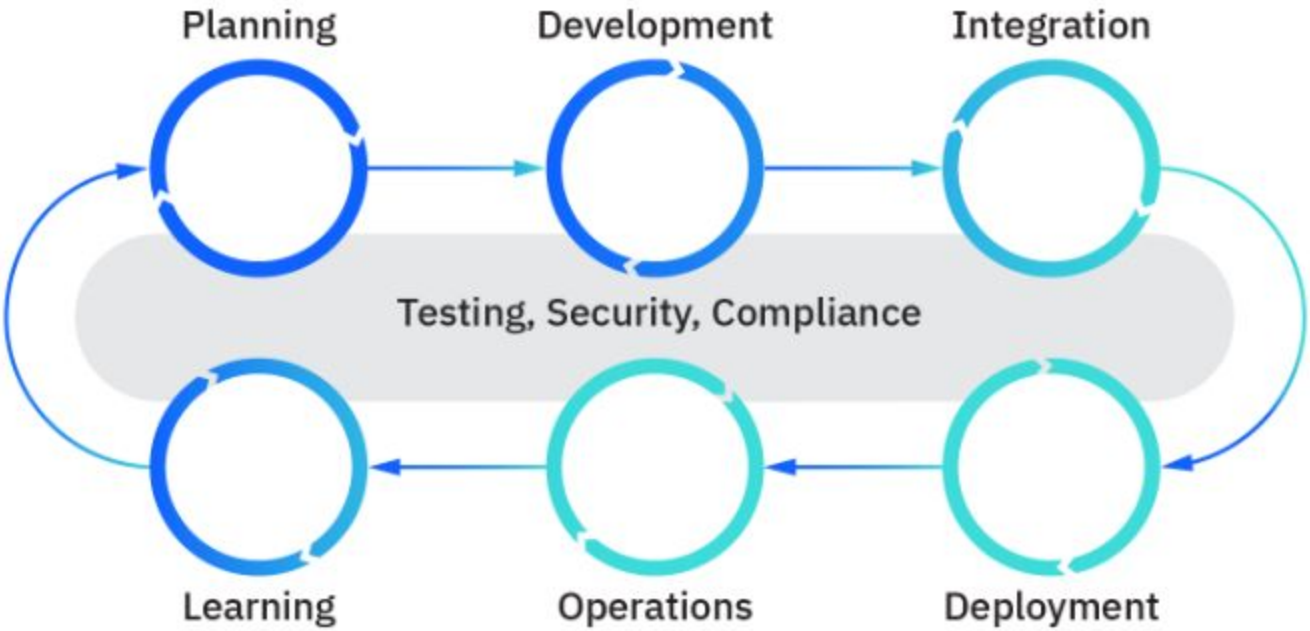
Leveraging agile …

DevOps

A culture

- Collaboration of roles
- You build it, you run it!
- Use of tools and technology

DevOps - Phases


Planning
Development
Integration
Testing, Security, Compliance
Learning
Operations
Deployment

DevOps - Phases

Planning

- Determine scope
- List features
  - Case studies
  - Input from stakeholder
  - Customer requirements
- Prioritize features
- Create backlog
- Tools
  - JIRA, Confluence, Slack

DevOps - Phases

Development

- Design and review
- Code feature
- Review code changes
- Test impact
  - Automated
    - Unit tests
    - Integration tests
  - Manual

Development contd …

- Tools
  - Environment
    - Kubernetes, Docker
  - Provisioning
    - Ansible, Terraform
  - Version Control
    - Git, Bitbucket, Gitlab
  - Test
    - Veracode, Sonar

DevOps - Phases

Integration

- Merge code changes to source
- Execution of automated checks
  - Quality
    - Unit tests
    - Integration tests
  - Compliance
    - Code coverage
    - Code smells
  - Security
    - Static code analysis
- Generate builds for later use
- Tools
  - Jenkins, AWS, Bitbucket, Circle CI

DevOps - Phases

Deployment

- Day 1
- Deploy builds generated in integration step
- Deploy progressively
  - Development
  - Testing
  - UAT
  - Production
- More stringent quality checks at each stage
- Break early and correct mistakes, if any
- Tools
  - Bamboo, Bitbucket, AWS CodePipeline

DevOps - Phases

Operations

- Day 2
- Monitor system health
  - Performance
  - Availability
  - Network
  - Storage

Operations contd …

- Incident management
  - Detect
  - Setup communication
  - Assess impact and determine severity
  - Communicate to customers
  - Escalate to responders
  - Resolve incident.
- Tools
  - Wavefront, Logz, Slack, PagerDuty, JIRA

DevOps - Benefits

- Smaller release cycles
- Improved collaboration
- Quality and reliability
- Security
- Improved time to resolution
- Management of unplanned work

DevOps - Best Practices

DevOps - Best Practices

Agile

- Iterative
  - Develop
  - Deliver
- Workflow
  - Todo => In Progress => Code Review => Done
- Items
  - Epics, Stories, Tasks, etc.

Agile contd …

- Frameworks
  - Scrum
    - Regular cadence
    - Fixed interval
    - Sprint planning, Daily sync, Retrospective
    - Scrum master
  - Kanban
    - Continuous flow
    - Limit work in progress

DevOps - Best Practices

Shift Left

- Testing during development
- Automated tests
  - Unit tests
  - Integration tests

DevOps - Best Practices

Toolset

- Use the right tools for each phase

DevOps - Best Practices

Automation

- From commit, to production
- Unit tests
- Generate build
- Deploy the build
- Integration tests
- E2E tests
- Performance tests

DevOps - Best Practices

Observability

- Sources
  - Logs
  - Traces
  - Metrics
    - Example: Used memory, Used CPU, Request latency, etc.
- Discover issues using the above sources

DevOps - Best Practices

Monitoring

- Broken builds
- Anomalies or issues
- Lack of performance
- Find it before the customer does

Continuous Delivery

Continuous Delivery

- From commit, to production
- Manual hops
  - Dev => Tools => Environment => Release => Ops => Customers
- Automated hops
  - Dev => Version Control => Automated Pipeline => Customers

Continuous Delivery - Benefits

- Velocity
- Productivity
- Sustainability

Continuous Integration / Continuous Delivery and Deployment (CI/CD)

Continuous Integration

- Merge code changes to source
- Generate build
- Validate the build
- Emphasis on automating tests

Continuous Integration contd …

How?

- Automated tests
- Automate test execution on every commits

Benefits

- Lesser bugs
- Smaller feedback loop
- Reduced testing costs
- QA focuses on larger picture

Continuous Integration / Continuous Delivery and Deployment (CI/CD)

Continuous Delivery

- Extension of Continuous Integration
- Deploy build to environments automatically
- Additional verification on deployed environments

Continuous Delivery contd …

How?

- Strong Continuous Integration foundation
- Good test suite
- Feature flags

Benefits

- No need to manually release
- Frequent releases

Continuous Integration / Continuous Delivery and Deployment (CI/CD)

Continuous Deployment

- Extension of Continuous Delivery
- Deploy to customers automatically

Continuous Deployment contd …

How?

- Strong Continuous Delivery foundation

Benefits

- Releases are less risky
- Frequent updates to the customers
- Continuous updates to the customers

Self Learning

- Feature flags
- CI/CD pipeline

How?

- Word document
- Illustrations help
- Quote references

Where?

- Keep it for now
- Submission opens 22nd Oct