

# DevOps Assembly Lines

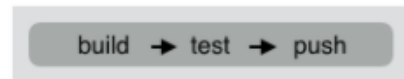
COMP 3104 - DevOps



# What do CI and CD mean?

- ***Continuous Integration***, is a software development practice in which all developers merge code changes in a central repository multiple times a day.
- CD stands for ***Continuous Delivery***, which on top of Continuous Integration adds the practice of automating the entire software release process.
- CD, Continuous Delivery includes **infrastructure provisioning and deployment**, which may be manual and consist of multiple stages.
- Here in CI/CD all these processes are fully automated, with each run fully logged and visible to the entire team.

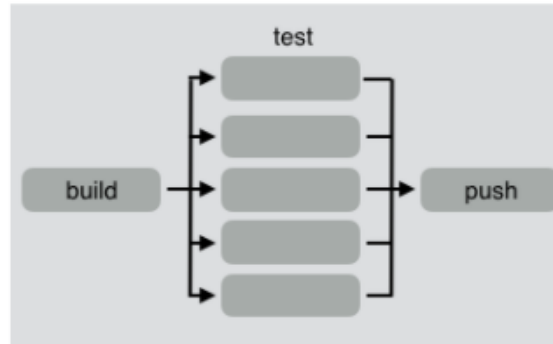
# What are CI pipelines?



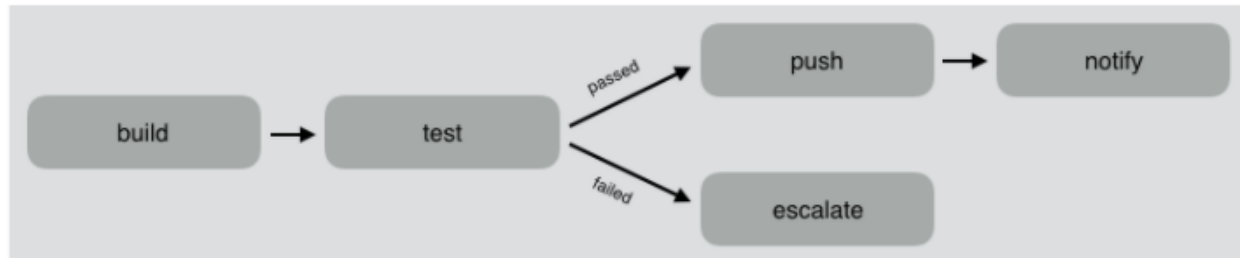
1. classic ci



2. ci with stages



3. ci with parallel stages

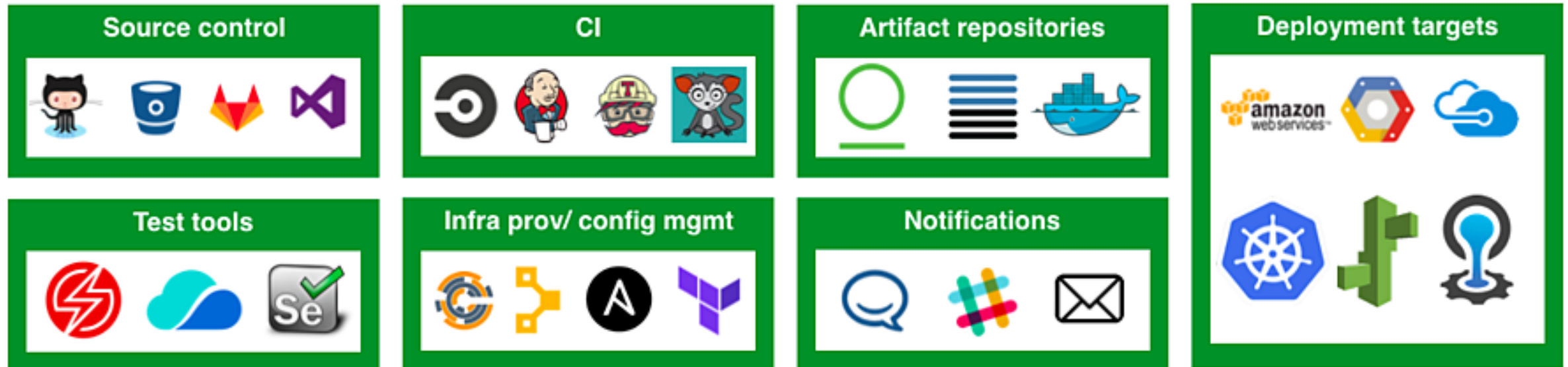


4. ci with forked stages

- Continuous Integration has evolved tremendously over the last few years.
- What started as a simple process of automating build and unit tests for each code change has evolved into a pretty complex workflow.

# What are DevOps Assembly Lines?

- DevOps Assembly Lines are focused on automating and connecting activities performed by several teams.
- For e.g.
  - CI for developers, infrastructure provisioning
  - config management for Ops,
  - test automation for Test,
  - security patching for SecOps,
  - semantic versioning and approval gates for Release Managers, deployments for multiple environments, and so on.



*DevOps Islands Of Automation*

# A DevOps Assembly Line is therefore a "Pipeline of Pipelines".

- Ability to easily define workflows across multiple pipelines.
- Versioned and reusable workflows, enabling rapid changes and scaling for multiple applications and/or microservices
- Integrations with all popular source control systems, clouds, artifact repositories, DevOps tools, languages, services, etc
- Runtime to execute every pipeline, including pre-installed tools and CLIs, and the ability to use the right runtime, depending on pipeline type
- Accelerators/playbooks for common pipelines and tools
- Ability to pass state and other information while triggering dependent pipelines
- Automatic triggers or manual approval gates between each pipeline

# A DevOps Assembly Line is therefore a "Pipeline of Pipelines".

- Configurable notifications for each stage of every pipeline
- Release automation features like semantic versioning of packages
- Audit trails for each pipeline, with the ability to go back or forward to a specific state
- Abstraction of all sensitive information like passwords, tokens, keys, etc for security reasons
- Roles and permissions restricting Assembly Line and pipeline actions
- Manage DevOps infrastructure, including spinning VMs and containers up and down as required
- Visibility into each pipeline and stage, including logs, status, and versioned data
- Metrics and Analytics across pipelines to help identify bottlenecks

# Managing DevOps Assembly Lines

- **Easy workflows and integration:**

- With the help of the DevOps assembly line, we can easily establish workflows across various pipelines.
- It integrates smoothly with the source control systems, DevOps tools, repository management, and artifacts management tools and helps us achieve collaboration amongst tasks and teams.
- And at the same time, it opens possibilities to automate and scale.

- **Configurable notifications, transparency, and monitoring:**

- Each stage within a pipeline can have configurable notifications. We can access each pipeline and the stages within it, along with the versioned data, logs, and status.
- Roles and permissions can help us monitor the assembly line in a better way.

- **DevOps infrastructure backed:**

- A DevOps assembly line can help us look after the entire DevOps infrastructure.

# DevOps Maturity Model

- A DevOps Maturity Model is a conceptual model in the form of a matrix of maturity levels at one side and areas (with subtopics) at the other side.
- Organizations use these models to determine the current (and desired) level of DevOps related topics.
- The DevOps Maturity Model acts as a “mental guide” to determine the level of these topics and it helps you lay out a path to advance from one level to the next one.
- There are various DevOps consultancy firms to guide you in your journey towards DevOps.



# Why Is Continuous Delivery Pipeline Maturity Important to DevOps?

Benefits of mature, well-engineered Continuous Delivery Pipelines include the following:

- A well-engineered Continuous Delivery pipeline provides visibility for software as it propagates through each stage.
- Quick lead times and more frequent releases provide quicker access to user feedback.
- Smaller, incremental, controlled releases are less painful, and failure events are lower risk.
- Reduced lead improves time-to-market for innovative new features.
- Software quality and stability are improved when each change is following a disciplined, well-engineered Continuous Delivery pipeline.
- Cost of software change is reduced when lean engineering practices are applied to the Continuous Delivery pipeline.
- Customer and employee satisfaction improve when they see positive results of the Continuous Delivery practices.

# Why Is Continuous Delivery Pipeline Maturity Important to DevOps?

Mature, well-engineered Continuous Delivery and Deployment avoids the following types of problems:

- Inefficient Change Review Board meetings required to approve releases
- Voluminous release documentation instead of automated deployments
- Reliance on manual error prone testing
- Frequent corrections to the manual release process
- Manual configuration errors
- Lengthy manual deployments
- Frequent release roll-backs caused by manual errors
- Unexpected interruptions during a long release
- Sitting bleary-eyed in front of a monitor during release hell nights and weekends

## Continuous Delivery Pipeline Maturity Level 1: Chaos

- Figure shows key characteristics of People, Process and Technology evident at this level of maturity.
- At this level there is little evidence of continuous delivery skills.
- Pipeline processes are not well integrated and pipeline automation technology has major gaps.
- Typical outcomes are releases are unpredictable, reactive, and the pipeline is wasteful.

### People

- ❑ Silo team organization
- ❑ Little communication
- ❑ Blame, finger-pointing

### Process

- ❑ Requirements, planning and tracking processes poorly defined and operated manually
- ❑ Unpredictable and reactive

### Tech

- ❑ Manual builds and deployments
- ❑ Manual quality assurance
- ❑ Environment inconsistencies

## Continuous Delivery Pipeline Maturity Level 2: Continuous Improvement

- Figure shows key characteristics of People, Process and Technology evident at this level of maturity.
- At this level there is some knowledge of continuous delivery pipelines.
- The builds, integrations, and build tests are well supported by processes and technology, End-to-end continuous delivery, especially delivery and deployment stages are not well supported with automation.
- Build quality is good, but typically there are deployment quality problems, and the infrastructure is used inefficiently.

### People

- ❑ Managed work backlog
- ❑ Communication between silos
- ❑ Limited knowledge sharing
- ❑ Ad hoc training

### Process

- ❑ Processes defined within silos
- ❑ Lack E2E process standards
- ❑ Can repeat what is known, but can't react to unknowns

### Tech

- ❑ Source code version management
- ❑ Automated builds, release artifacts and automated tests
- ❑ Painful but repeatable releases

## Continuous Delivery Pipeline Maturity Level 3: Continuous Flow

- Figure shows key characteristics of People, Process and Technology evident at this level of maturity.
- At this level continuous delivery processes extend from end-to-end across the pipeline.
- Dev and QA teams cooperate to ensure a good level of test coverage is automated.
- Release standards are using automated testing metrics.
- Typical outcomes include repeatable quality releases with some bottlenecks and inefficiencies.

### People

- DevOps leadership
- Collaboration between cross-functional teams
- DevOps training program

### Process

- End-to-end CI/CD pipeline automated
- Standards across the org for applications, releases processes and infrastructure

### Tech

- Toolchain orchestrates and automates builds, tests and packaging deliverables
- Infrastructure as code
- Metrics and analysis for release acceptance and deployment

## Continuous Delivery Pipeline Maturity Level 4: Continuous Feedback

- Figure shows key characteristics of People, Process and Technology evident at this level of maturity.
- At this level more advanced knowledge of continuous delivery pipelines is apparent.
- Goals and metrics are set for each stage in the pipeline.
- The culture includes training and mentoring for continuous delivery.
- There is a focus on end-to-end performance trends rather than spot results.
- Automation is applied to test environment orchestration and analytics.
- Typical outcomes at this level reflect a general confidence to obtain repeatable quality releases, with a metrics-driven culture.

### People

- ❑ Collaboration based on shared metrics to remove bottlenecks
- ❑ SLIs, SLOs and SLAs to ensure stakeholder alignment
- ❑ DevOps Mentors and Guilds

### Process

- ❑ Proactive monitoring
- ❑ Metrics collected and analyzed against business goals
- ❑ Visibility and repeatability

### Tech

- ❑ Applications, pipelines and infrastructure fully instrumented
- ❑ Metrics and analytics dashboards
- ❑ Orchestrated deployments with automated rollbacks

## Continuous Delivery Pipeline Maturity Level 5: Continuous Improvement

- Figure shows key characteristics of People, Process and Technology evident at this level of maturity.
- At this level there is a high level of knowledge and confidence regarding continuous delivery pipelines.
- Dev and QA teams are tightly integrated to optimize knowledge and efficiency.
- End-to-end processes focus on the end customer experience and more sophisticated risk-based strategies.
- Typical Outcomes include an extremely high confidence and satisfaction by all stakeholders, with an innovation-driven culture.
- Achievement at this level provides a platform for autonomous continuous improvement strategies in which automation and intelligence drives innovation.

### People

- Culture of continuous experimentation and improvement

### Process

- Self-service automation
- Risk and cost optimization
- High degree of experimentation

### Tech

- Zero downtime deployments
- Immutable infrastructure
- Actively enforce resiliency by forcing failures

# DevOps for AI/ML

- DevOps and AI/ML development are two independent methodologies with a common goal: to put an AI application into production.
- AI/ML projects need to incorporate some of the operational and deployment practices that make DevOps effective and DevOps projects need to accommodate the AI/ML development process to automate the deployment and release process for AI/ML models.
- The AI/ML process relies on experimentation and iteration of models and it can take hours or days for a model to train and test.
- Carve out a separate workflow to accommodate the timelines and artifacts for a model build and test cycle.
- Avoid gating time-sensitive application builds on AM/ML model builds.
- For AI/ML teams, think about models as having an expectation to deliver value over time rather than a one-time construction of the model.
- Adopt practices and processes that plan for and allow a model lifecycle and evolution.
- DevOps is often characterized as bringing together business, development, release, and operational expertise to deliver a solution.
- Ensure that AI/ML is represented on feature teams and is included throughout the design, development, and operational sessions.









# The Trends and Future of DevOps

## 1. DevOps in the Security Field

- The field of security is peculiar because the more you automate, the higher chances of automating problems too.
- Implementation of DevOps should ensure the security of the product being developed in production and even in the test environments.
- This stands as the governance and codes of ethics of DevOps philosophy.
- DevOps must ensure security protocols that ensure the application's integrity and conformance with the security policies of the company.

## 2. AI/ML in the DevOps Framework

- 
- 
- 
- 
- The software development life cycle is revolutionized with the DevOps methodology, cloud-native approach, and microservices architecture.
  - DevOps integrates testing and production environments, and developers get to see the problems before applications go live.
  - Applying AI and ML to the DevOps pipelines can help you run builds and automation in a much better way with closer insights and control.
  - People are moving from DevOps to DataOps and AIOps, which focus on the use of artificial intelligence and machine learning to learn from logs and monitoring metrics to drive DevOps in a controlled fashion.







# The Trends and Future of DevOps

## 3. Automation for Every Company

- In today's world, everything happens over the internet.
- Most companies are changing to be like an IT company that provides some particular services. For example, booking.com was a travel company which now functions as an IT company that provides travel services.
- For every company, their software is the critical element that brings in sales and business.
- Hence automation of software deployment and infrastructure provisioning key to all modern businesses.
- DevOps methodologies thus play a significant role in all modern companies today.

## 4. Container Technology

- 
- 
- 
- 
- Container technology is evolving and emerging faster than before.
  - Containers can be used in various ways to provide different benefits.
  - Containers can be used to sandbox applications for security and resource constraints.
  - Research is going around using containers per user or user session.
  - This idea brings a limitless array of opportunities for improving user security, system security, and performing user analytics.
  - As containerization technology improves, containers will become more cost-effective to deploy.

# The Trends and Future of DevOps

## 5. Platform as a Service (PaaS)

- Platform as a service (PaaS) is a growing field with a lot of applications for DevOps concepts.
- Gone are those days when people worried about building an entire infrastructure for the application.
- Today people only ask for the platform on which their applications can be hosted.
- DevOps has a lot of applications in providing PaaS solutions in terms of configurations management, continuous security, and containerization.

## 6. DevOps and Focus on Integration Between Edge Services

- The traditional model of on-premises is clearly changing. In the last few years, companies have moved to Infrastructure as a service (IaaS), Database as a Service (DBaaS), and Platform as a Service (PaaS) solution.
- With cloud technologies pitching in heavily and containerization technologies going widespread, DevOps has to play a significant role in the integration of all these services that are hosted on different platforms.



Thank You