DevOps Fundamentals

.NET CORE

DevOps is the union of people, process, and products to enable continuous delivery of value to end users.

[HTTPS://DOCS.MICROSOFT.COM/EN-US/AZURE/DEVOPS/LEARN/WHAT-IS-DEVOPS](https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops)

What is DevOps?

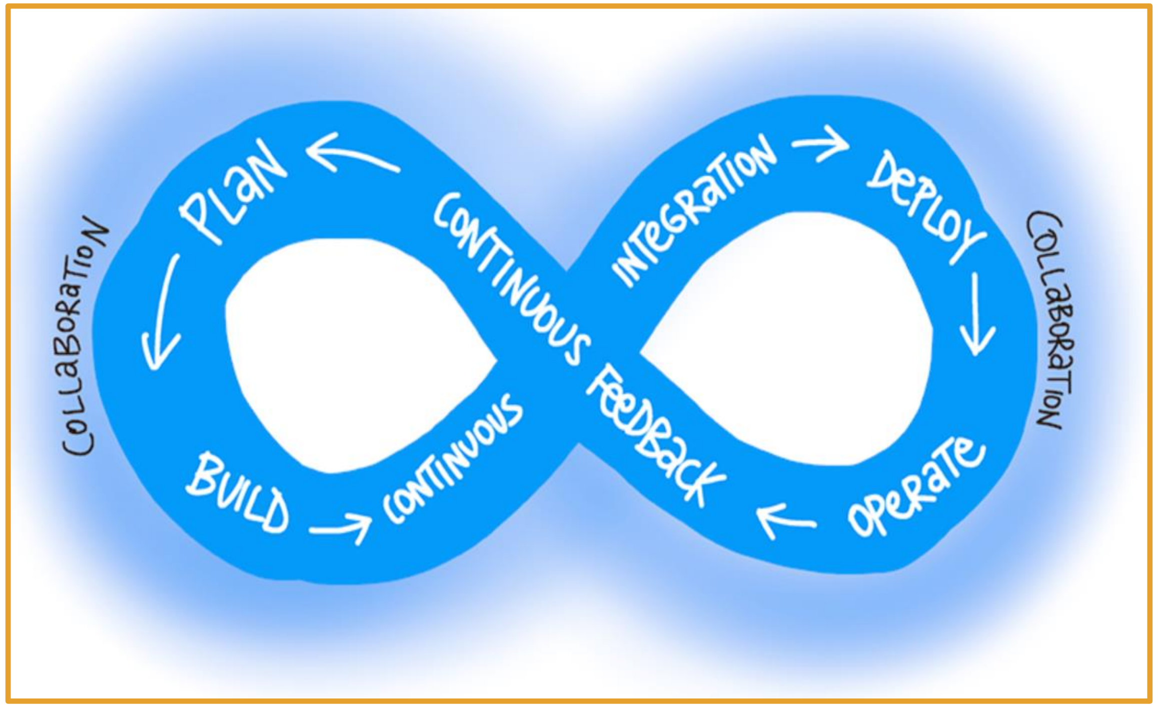
https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops

The contraction of “Dev” and “Ops” refers to replacing “Siloed” DEV-elopment and OP-erations Teams.

Instead, multidisciplinary teams that work together with shared, more efficient practices and tools are created.

Essential DevOps practices include:

- Agile planning,
- Continuous Integration,
- Continuous Delivery, and
- monitoring of applications.



Who is DevOps?

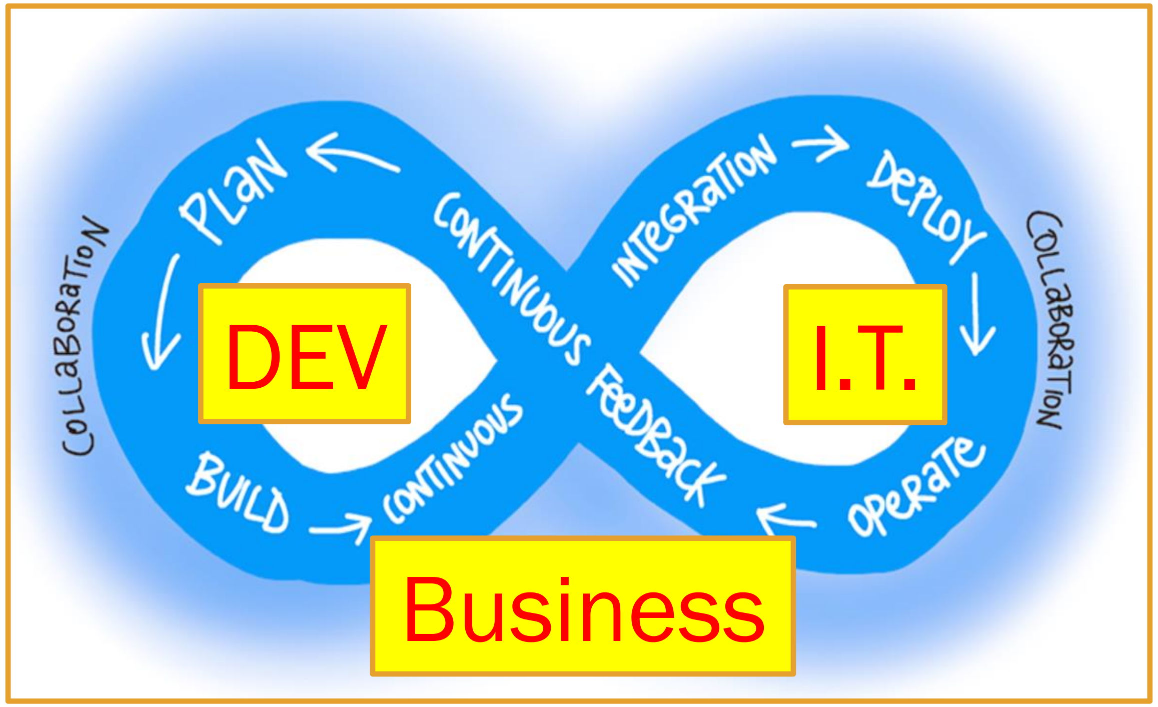
https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops

DevOps is the combination of the processes of the:

- Business team,
- IT team,
- Development team

In *DevOps*, these teams form a feedback loop that has a singular goal.

- The Dev team plans and builds the app.
- The IT team deploys and maintains the app.
- The Business Team verifies that the correct product is created and delivered.



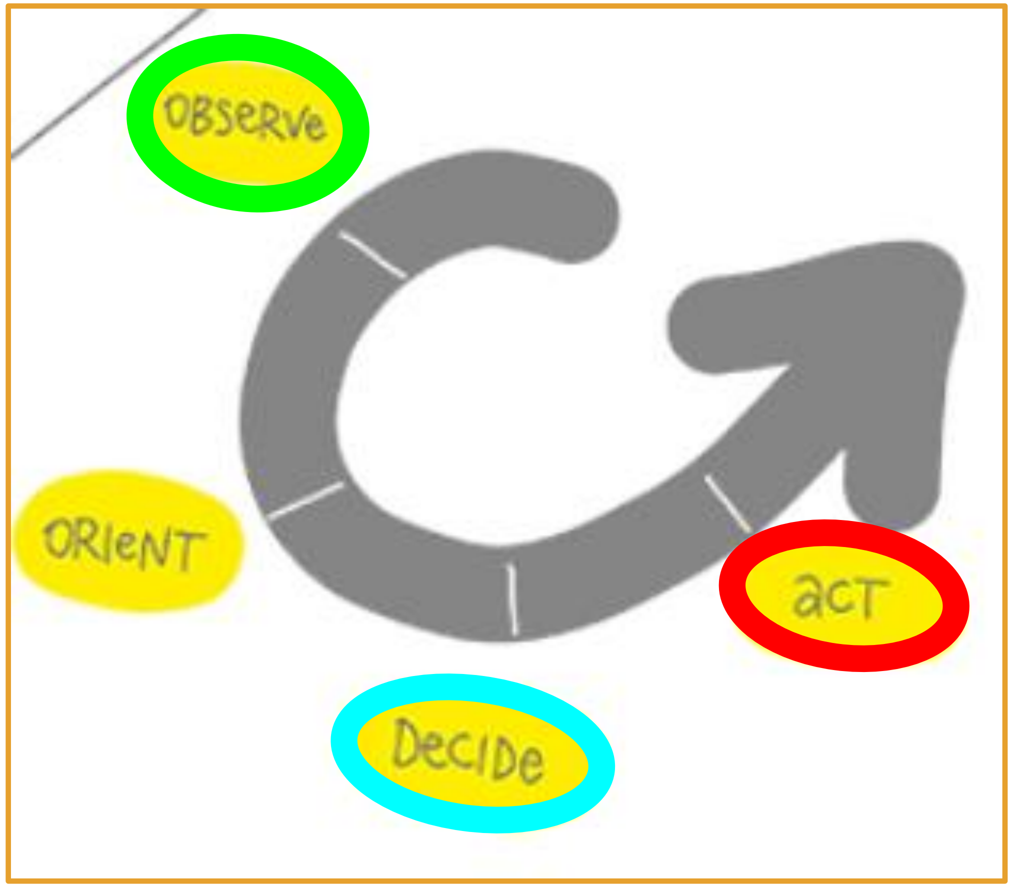
DevOps and The O.O.D.A. Loop

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops#understand-your-cycle-time
http://www.slideshare.net/adriancockcroft/speeding-up-31799721

The OODA loop:

1. O - observe business and market needs and current user behavior.
2. O - orient with the options for what you can deliver.
3. D - decide what goals to pursue.
4. A - act by delivering working software to real users.

The four OODA Loop steps occur in a ***Cycle Time***. The Cycle repeats until a project is complete.

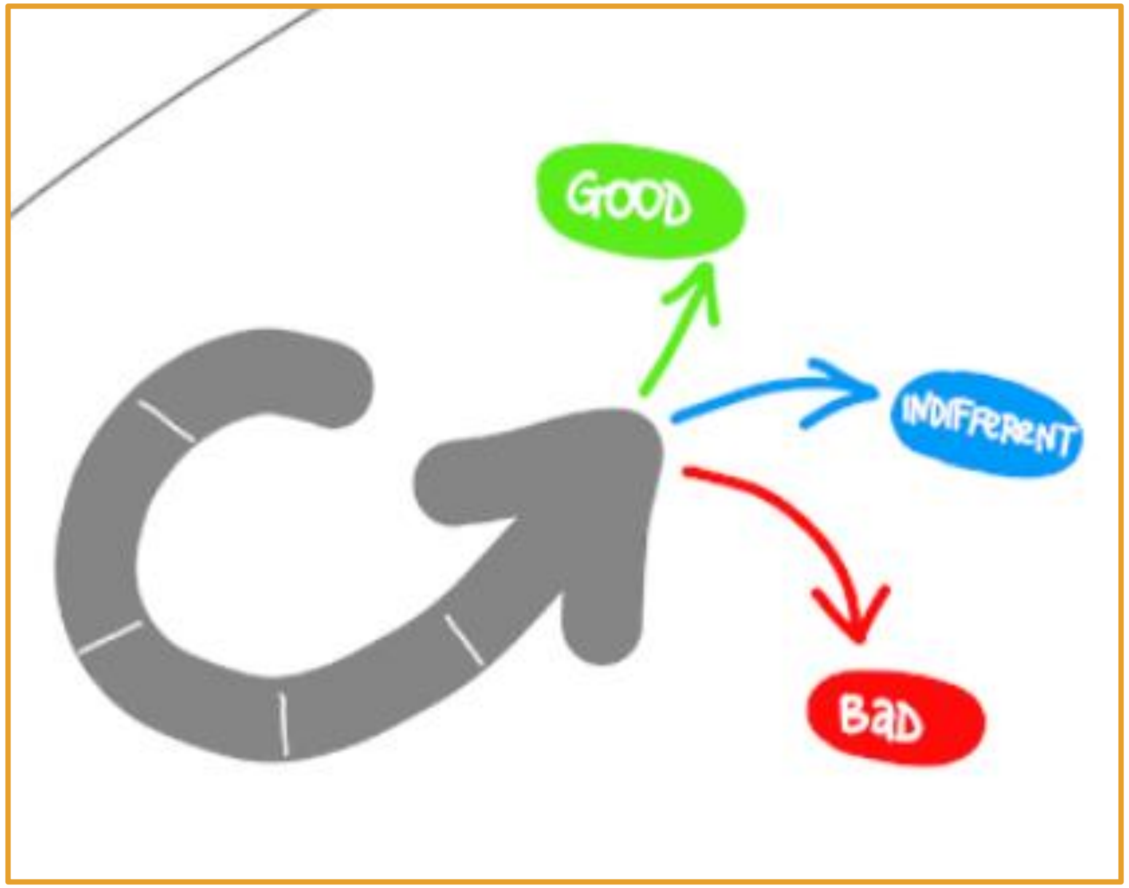


The OODA Loop - Cycle Time

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops#understand-your-cycle-time
http://www.slideshare.net/adriancockcroft/speeding-up-31799721

Your ***Cycle Time*** is determined by how quickly you can complete the four steps.

The ***feedback*** that you gather with each cycle should be real, actionable data. Something should be learned from each cycle. This is called ***Validated Learning***.



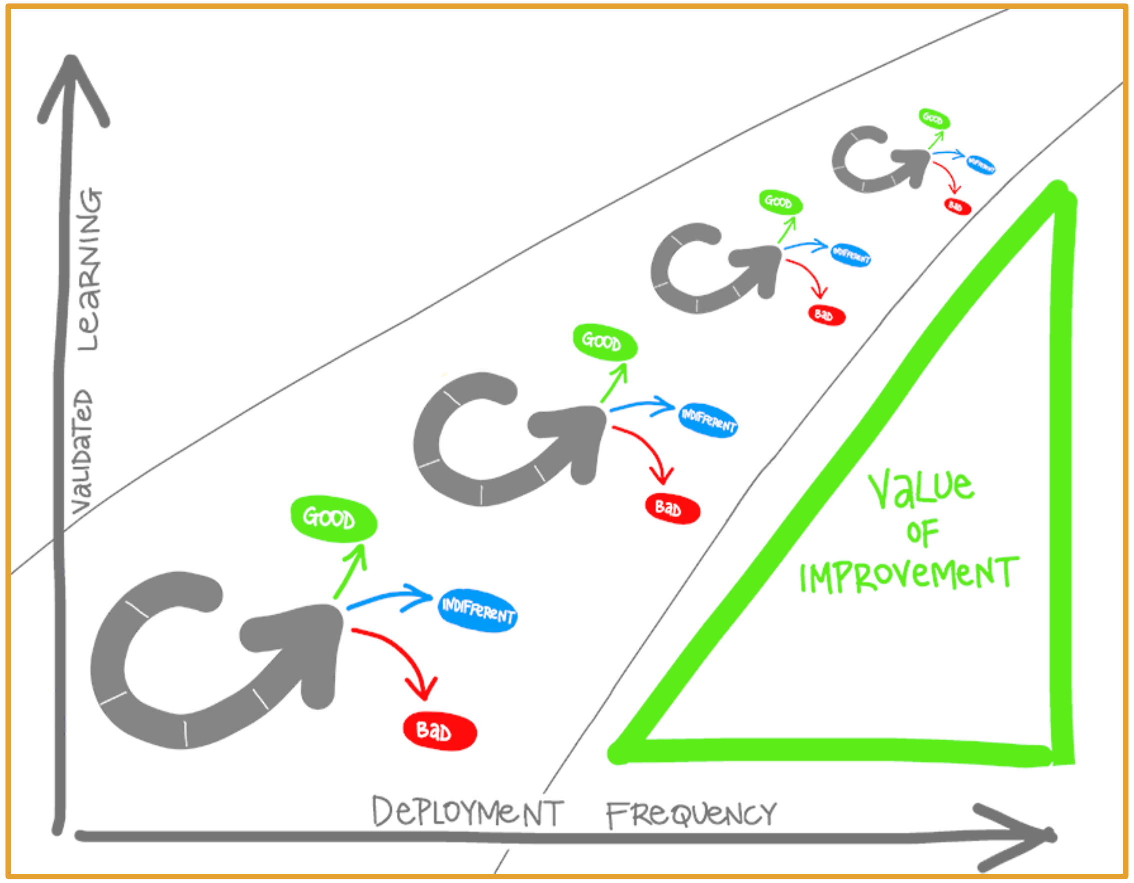
DevOps shortens Cycle Time

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops#shorten-your-cycle-time

When ***DevOps*** practices are adopted, smaller, more focused teams will:

- use more automation,
- improve the release pipeline, and
- deploy more frequently.

The more frequent the deployment, the more experimentation can be done, and the more opportunity there is to gain ***Validated Learning*** after each cycle.

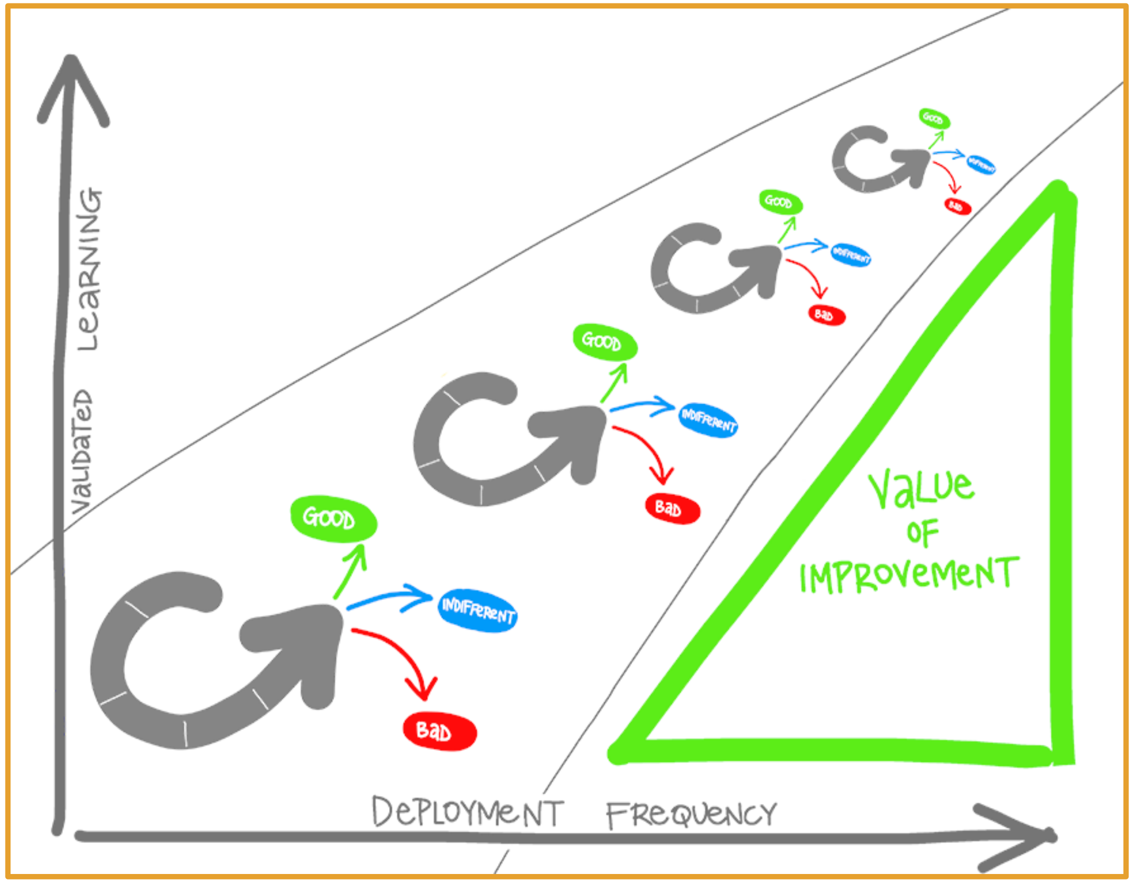


Achieving Devops

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops#how-to-achieve-devops

The overall goal is to eventually shorten the project ***Cycle Time*** to zero.

This is achieved through ***Continuous Integration and Continuous Delivery (CI/CD)***.



CI- Continuous Integration

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops#how-to-achieve-devops

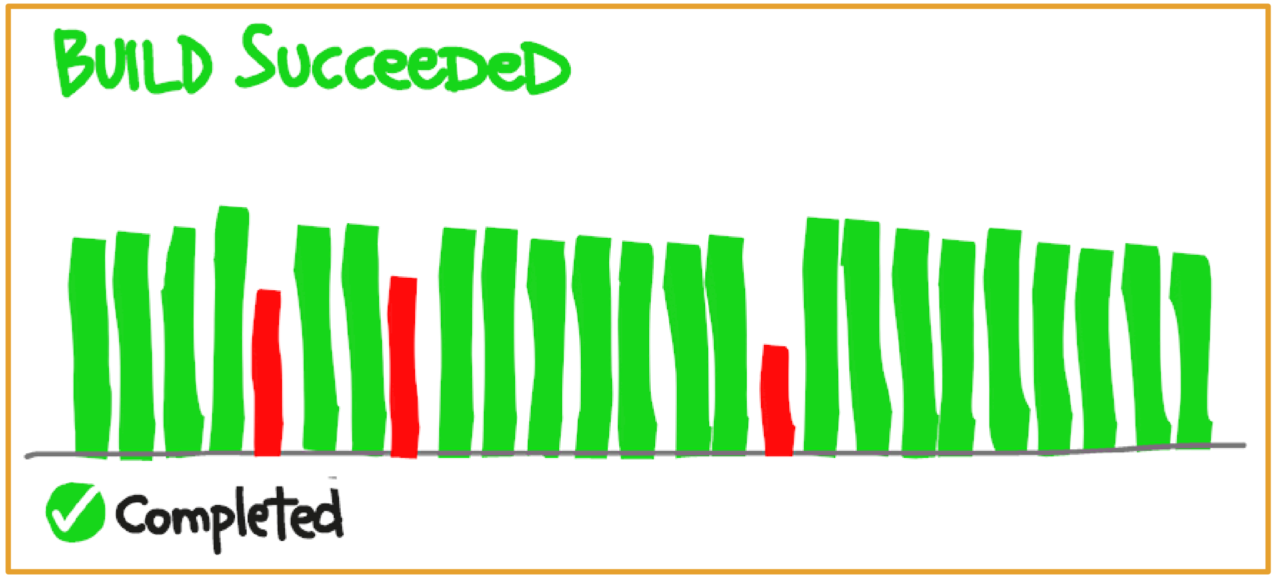
Continuous Integration (CI) is the process of automating the build and testing of code every time a team member commits changes to version control. Ideally, changes are committed multiple times per day. Developers merge even small changes to version control.

To achieve Continuous Integration, the commit of new code triggers an automated build system to grab the new code from the shared repository and build, test, and validate the full master branch.

Constantly merging code avoids

- merge conflicts,
- duplicated efforts, and
- divergant strategies.

A developer submits a “pull request” when a feature or change is complete. The changes are accepted and merged into the master branch. Then the feature branch is deleted.



CD – Continuous Delivery

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-continuous-delivery

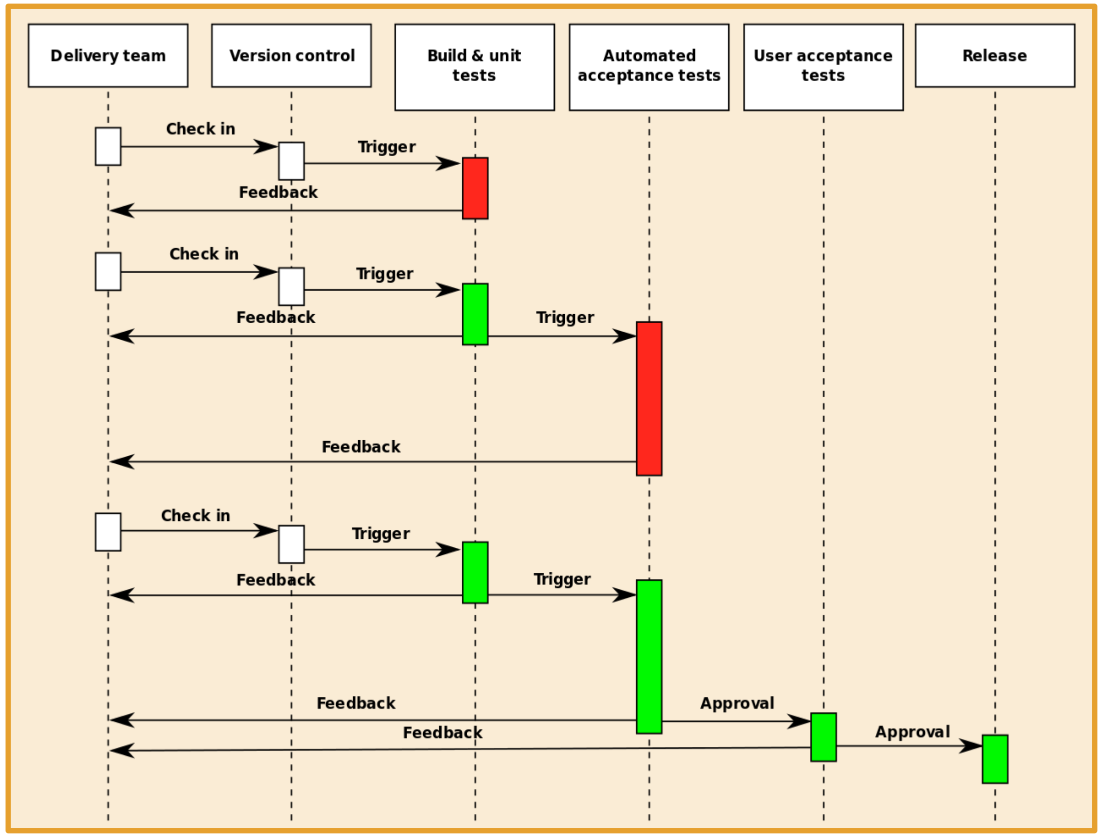
Continuous Delivery (CD) has been shown to achieve the shortest path from new code to final deployment.

CD is the process of building, testing, configuring, and deploying code to a production environment.

A ***Release Pipeline*** is made up of multiple build, test, or staging environments which are used to automate the deployment. Automation is preferred because manual processes are unreliable and produce delays and errors.

Without ***Continuous Delivery***, software release cycles become a bottleneck for dev teams.

An automated ***Release Pipeline*** allows a “fail fast” approach to validation, where tests fail quickly so code can be immediately refactored.



Pipeline Monitoring and Logging

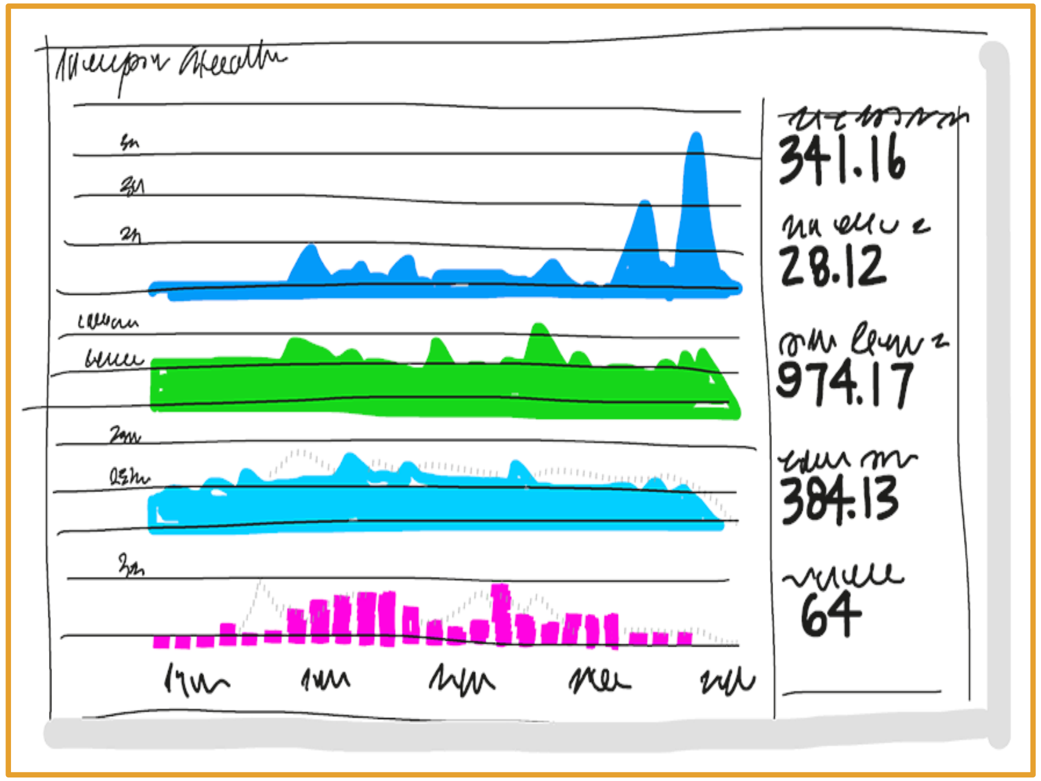
https://docs.microsoft.com/en-us/azure/devops/learn/what-is-monitoring

Monitoring should be built into the Pipeline to allow “test in production”.

Monitoring enables ***Validated Learning*** by immediately delivering details about an application’s performance and usage patterns.

Issues are immediately fed back to development teams via the automated build, test, and report phases in the process. The team can quickly pivot their strategy if needed.

There are various third-party sites to which the pipeline can report testing code coverage and code quality analysis.


341.16
28.12
974.17
384.13
64