DevOps Fundamentals

.NET CORE

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

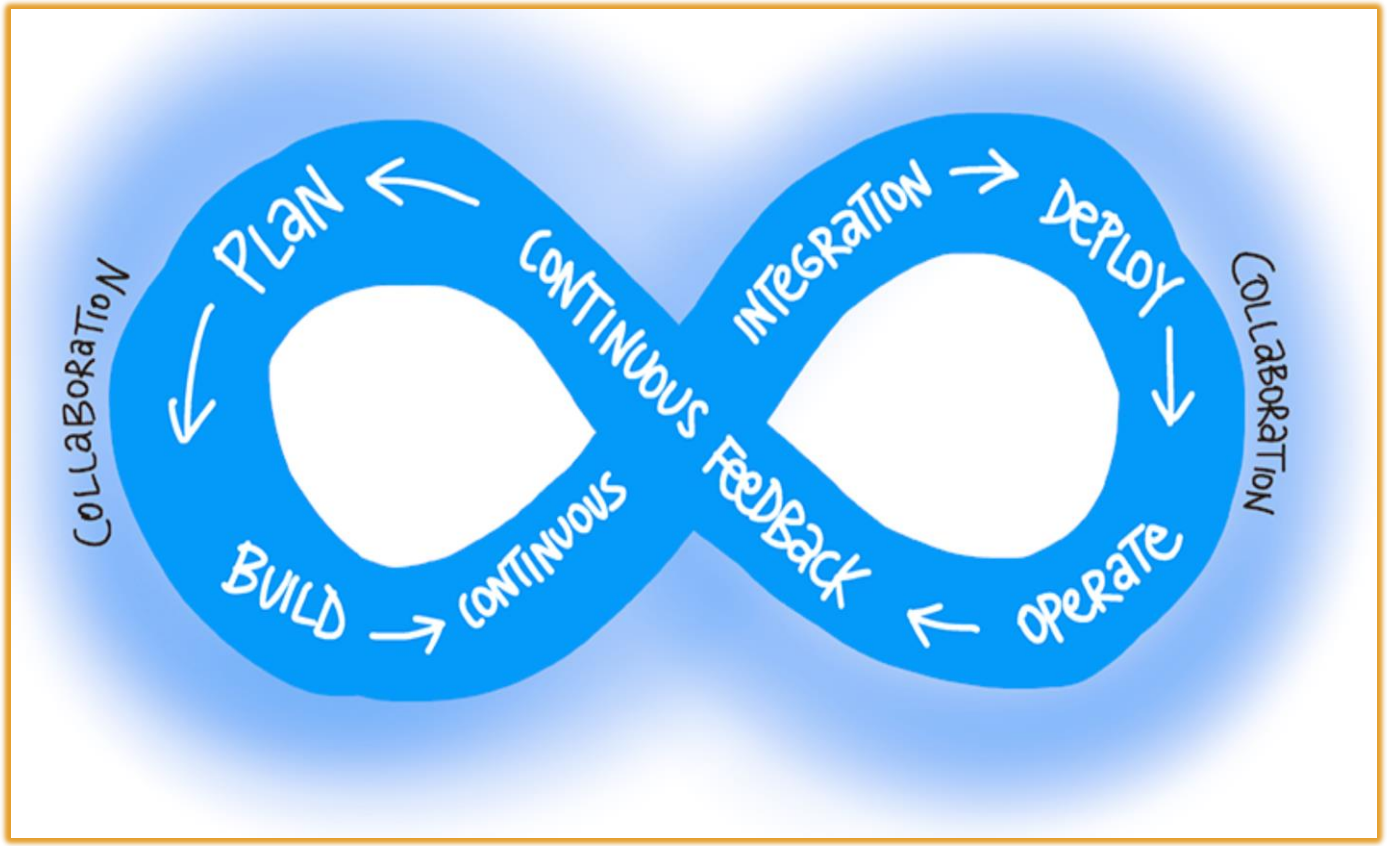
https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops

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 Development and Operations to create multidisciplinary teams that work together with shared, efficient practices and tools.

Essential DevOps practices include Agile planning, Continuous Integration, Continuous Delivery, and monitoring of applications.



What is DevOps?

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

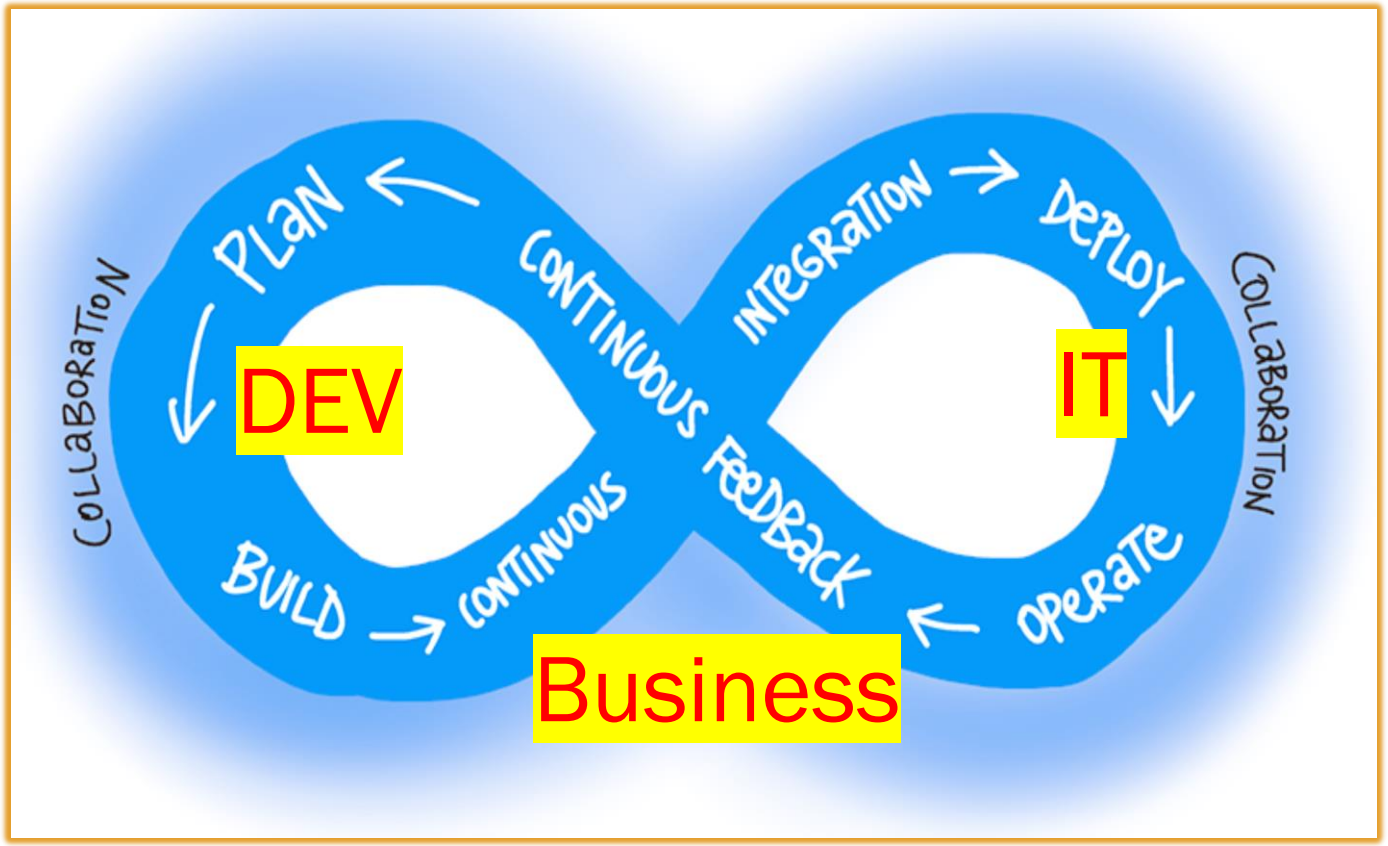
What is this and why is it important? Why was this shape chosen for this structure?

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, the IT team, and the Development team into a loop that has a singular goal.

The Dev team is on the left of the Infinite loop (plan and build). The IT team in on the Right (deploy and maintain). The Business Team is at the intersection where the they verify that the correct product is delivered.



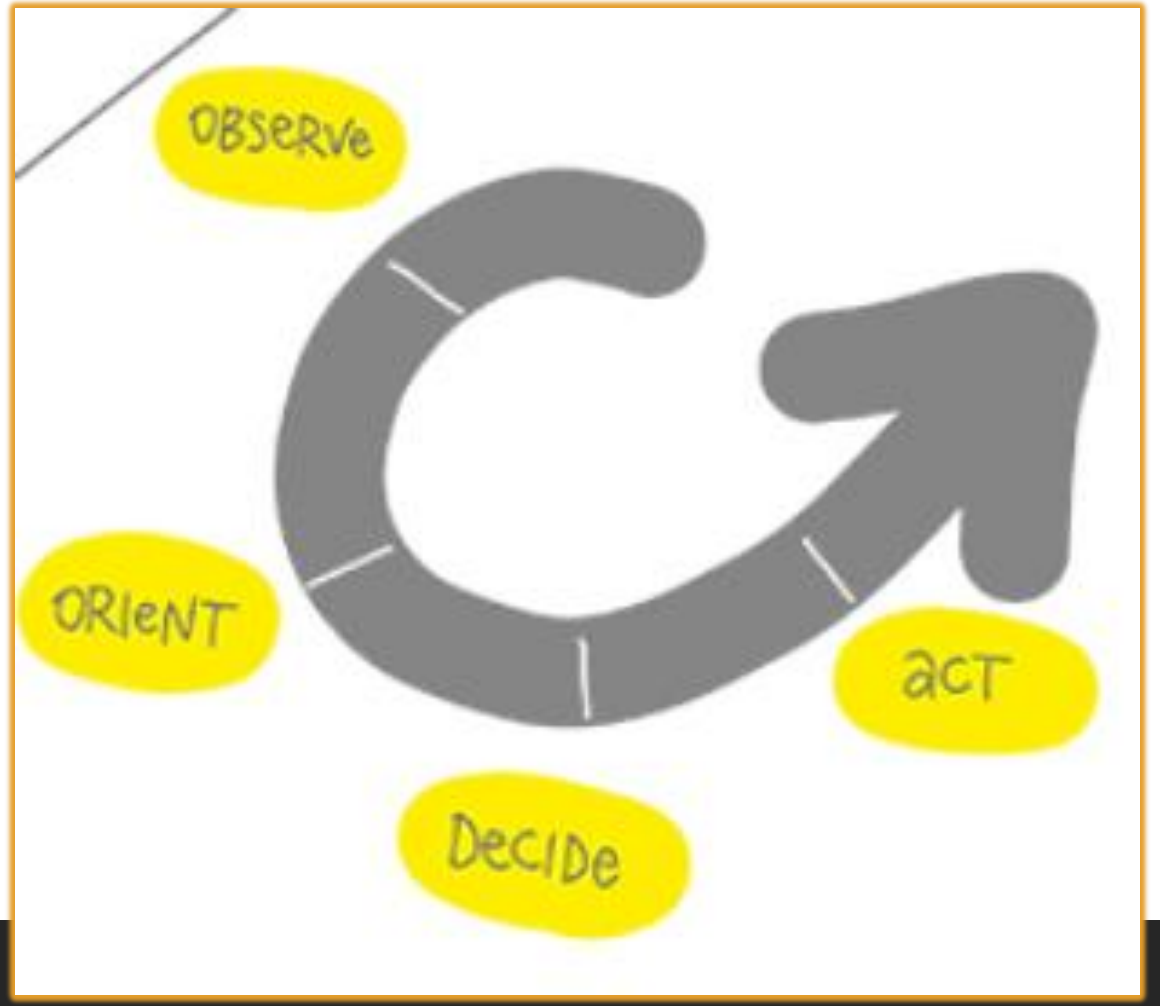
DevOps Cycle Time – The OODA Loop

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

With the OODA loop:

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

All of this occurs in a ***Cycle Time***.

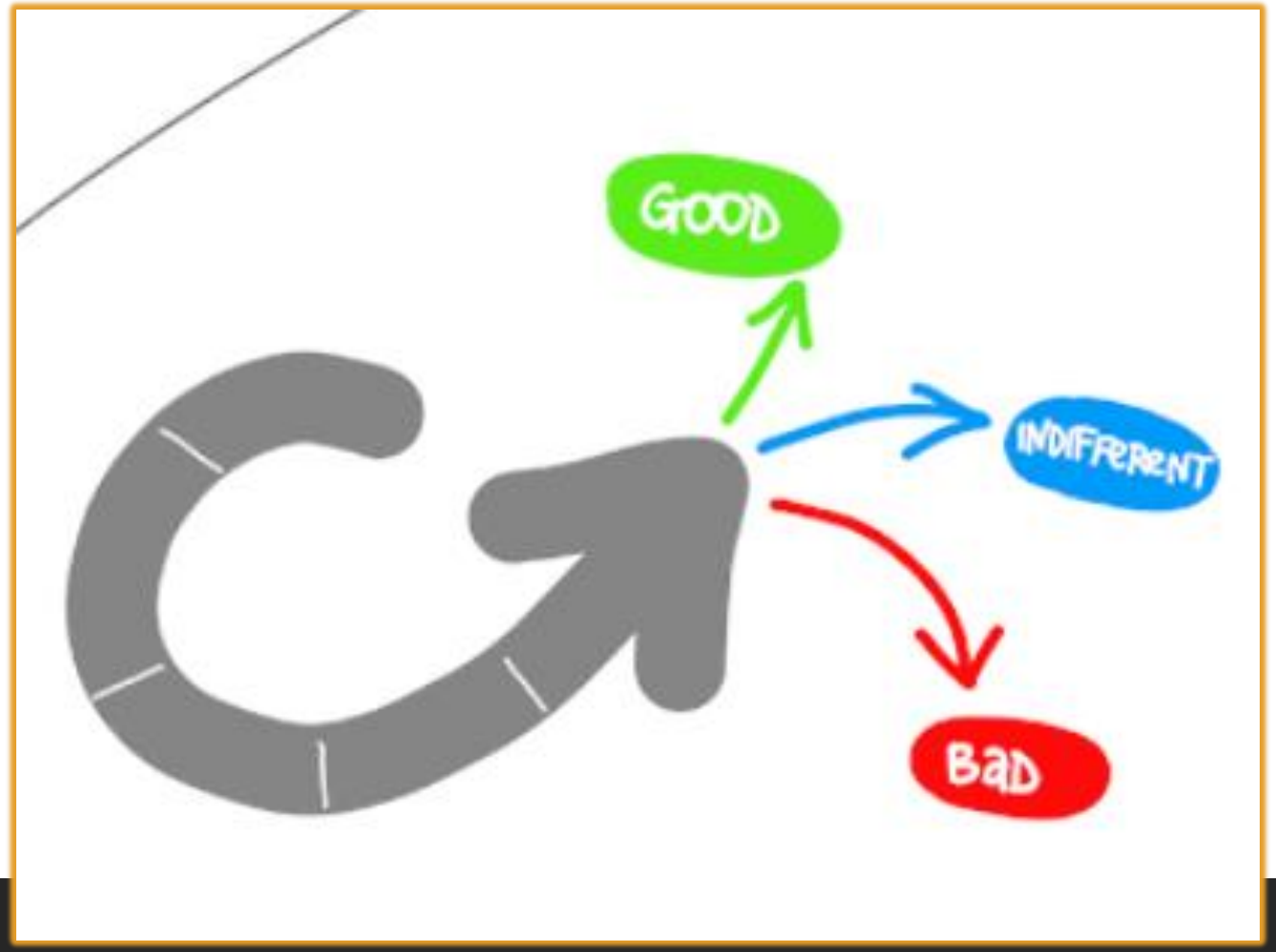


DevOps Cycle Time – The OODA Loop

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

Your ***Cycle Time*** determines how quickly you can gather ***feedback*** to determine what happens in the next loop.

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

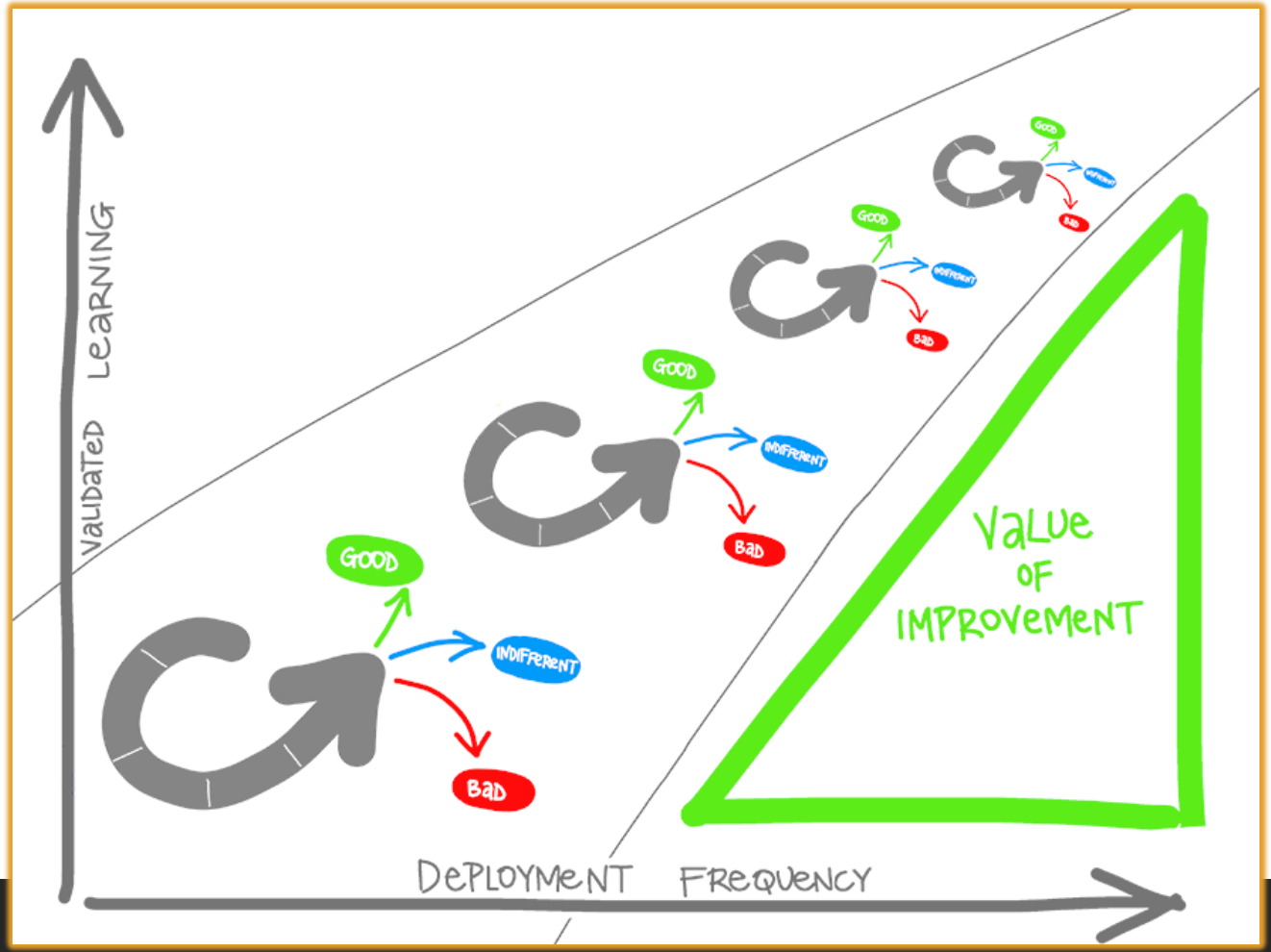


DevOps shortens Cycle Time

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

When ***DevOps*** practices are adopted, ***Validated Learning*** helps shorten the ***Cycle Time*** by using more automation, hardening the ***release pipeline***, improving telemetry, and deploying more frequently.

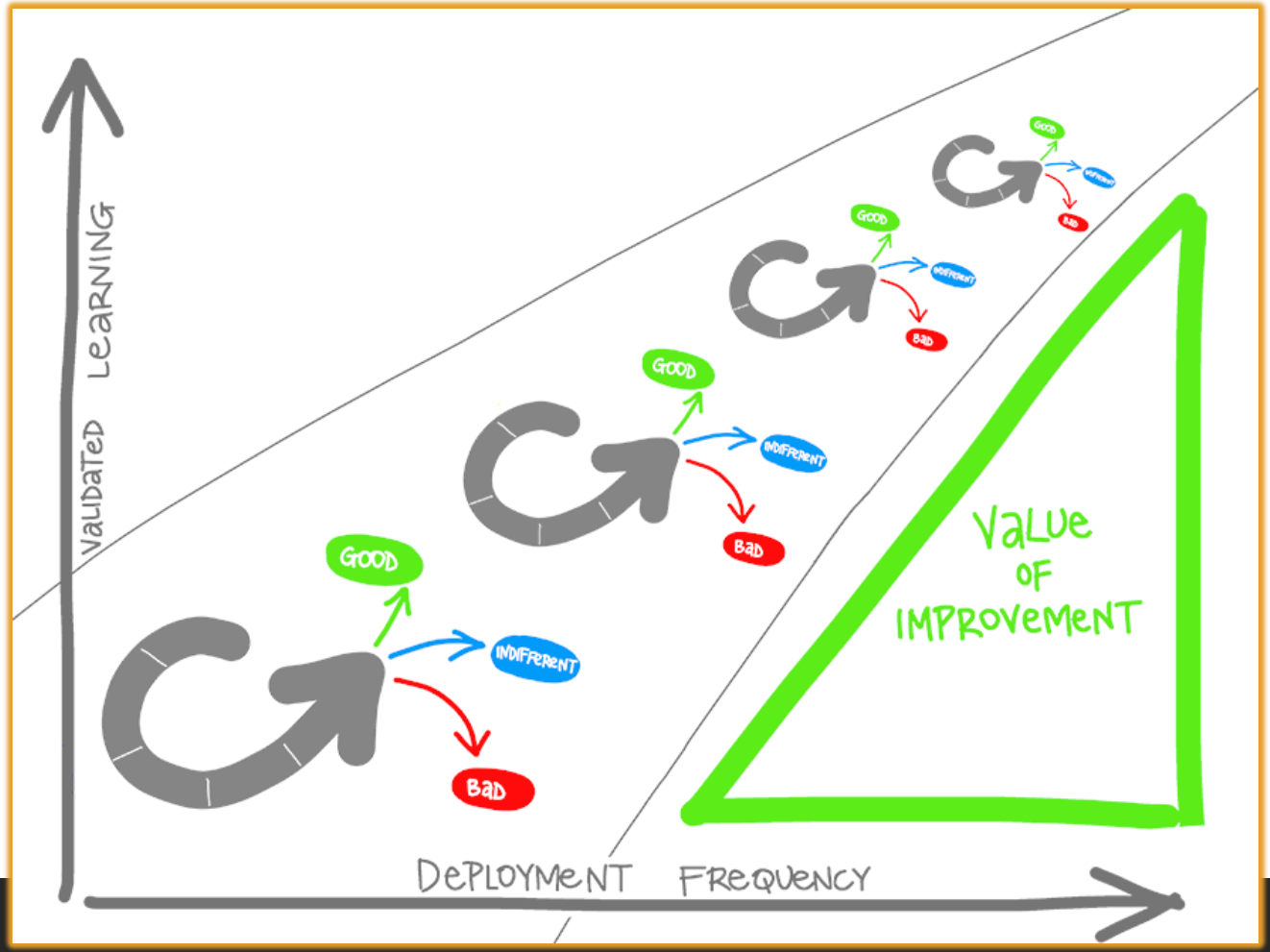
The more frequent the deployment, the more experimentation can occur and the more opportunity there is to gain ***Validated Learning*** for each cycle.



Achieving Devops

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

The ultimate goal is to shorten ***Cycle Time***. 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. ***CI*** encourages developers to share their code and unit tests by merging their changes after every small task completion. Even multiple times per day.

Committing code triggers an automated build system to grab the latest code from the shared repository and to build, test, and validate the full master branch.

CI requires the development team's code be merged to a shared version control branch continuously to avoid merge conflicts, duplicated efforts, and diverging strategies.

A developer submits a "pull request" when a feature is complete. On approval, the changes are merged into the master branch and the feature branch is deleted.

CD – Continuous Delivery

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

Continuous Delivery (CD) is the process to build, test, configure and deploy to a production environment. Multiple testing or staging environments create a *Release Pipeline* to automate the deployment of a new build.

Without *Continuous Delivery*, software release cycles are a bottleneck for application and operations teams. Manual processes lead to unreliability that produces delays and errors. The automated *Release Pipeline* allows a “fail fast” approach to validation, where tests fail quickly and refactoring can be immediate.

CD achieves the shortest path from new code in version control to deployment. Automation allows *CD to* optimize process time and eliminates idle time.