DevOps

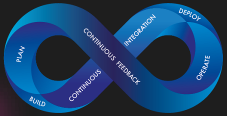
Ahmed Khalil

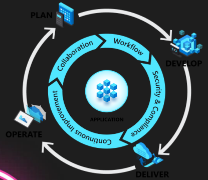
BDAYA DEVELOPMENT

WHOA!

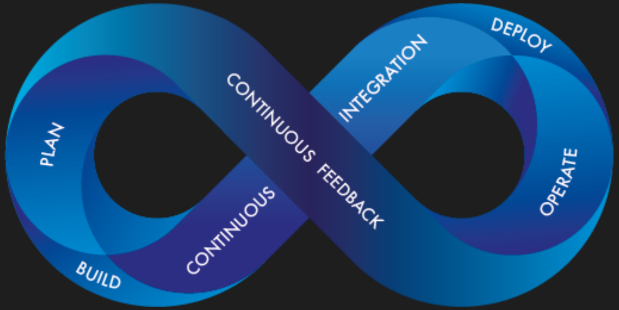
Ahmed Khalil
Microservices Developer

TABLE OF CONTENTS





Is It Complex



01

Introduction

- DevOps as a buzz word
- DevOps for teams
- Benefits of DevOps
- Organizations tell the story

New Enterprise View

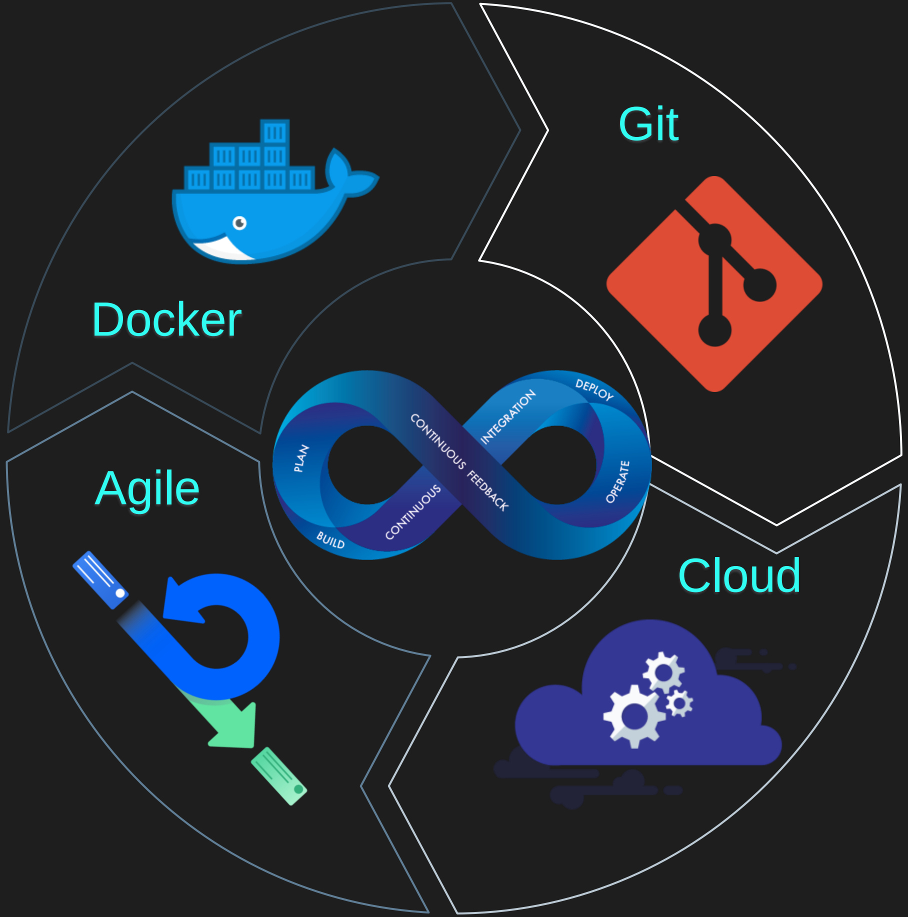
Enterprises are in war of

- saving costs
- solving deployment problems
- work for make it fast to release its software into production

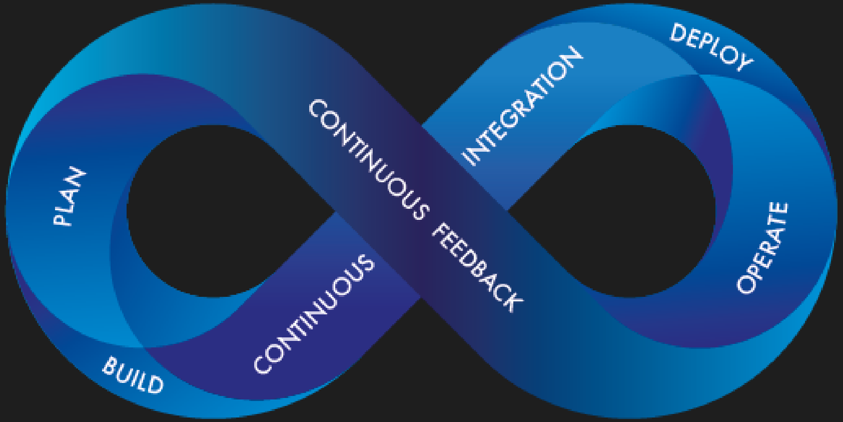
Improving these concepts come in more fuzzy changes that can get the enterprise to the top or its knees

SDLC Ages

- DevOps Age
- Agile Development Age
- RAD & JAD
- Structured Development Age



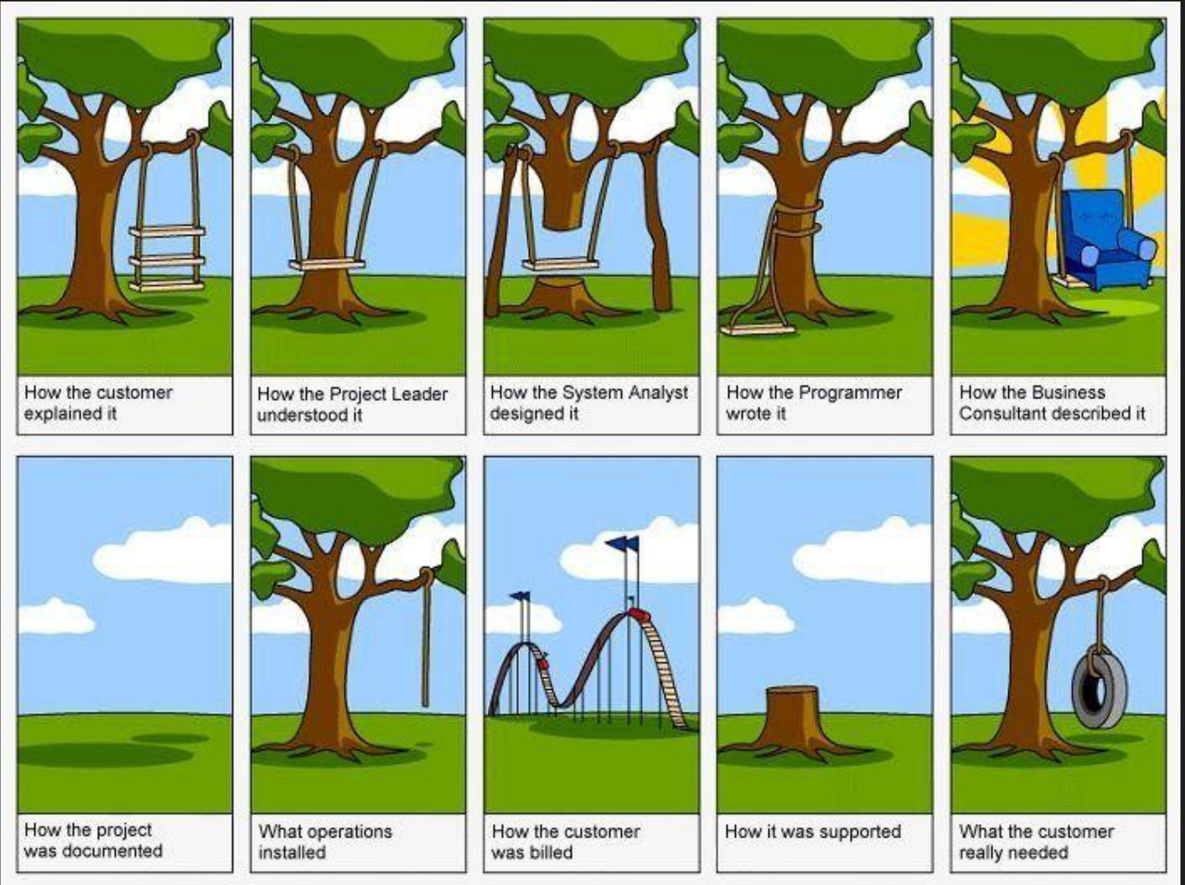
The Meaning of DevOps



DevOps is a set of practices that merge tow concepts software development and operations, It aims to shorten the systems development life cycle and provide continuous delivery with high software quality.

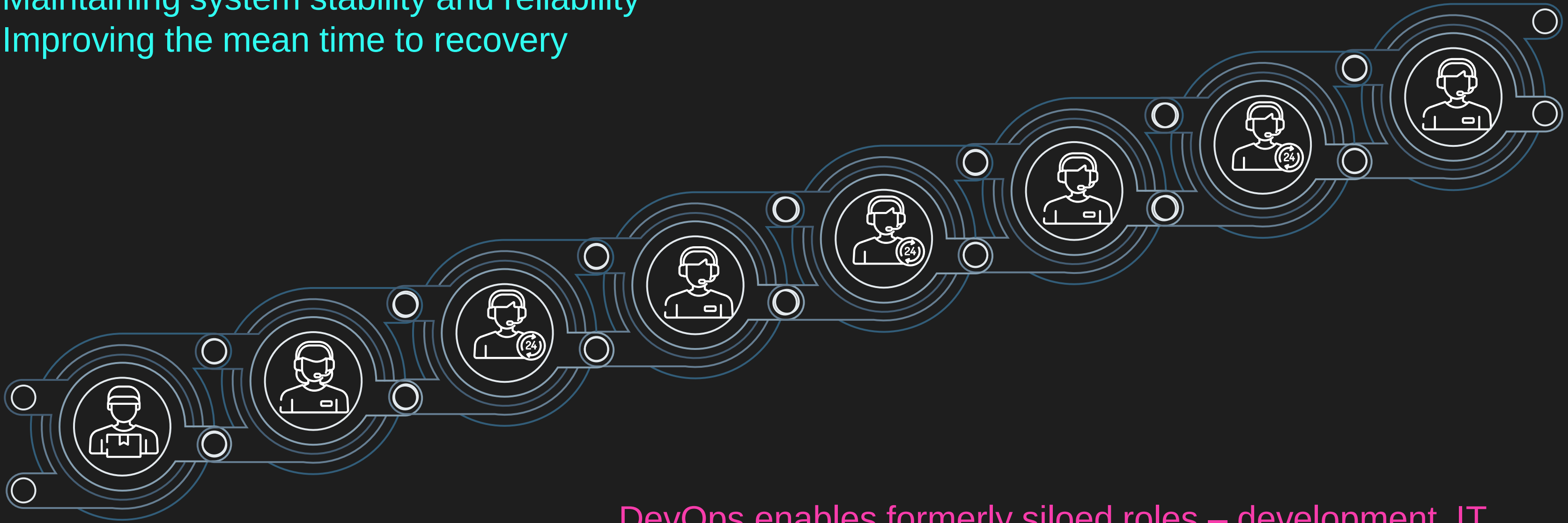
The cruise taking you in infinity mode, this mode is DEVOPS.

Is Your Team In Soils?


How the customer explained it
How the Project Leader understood it
How the System Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

DevOps Solution

- Accelerating time to market
- Adapting to the market and competition
- Maintaining system stability and reliability
- Improving the mean time to recovery

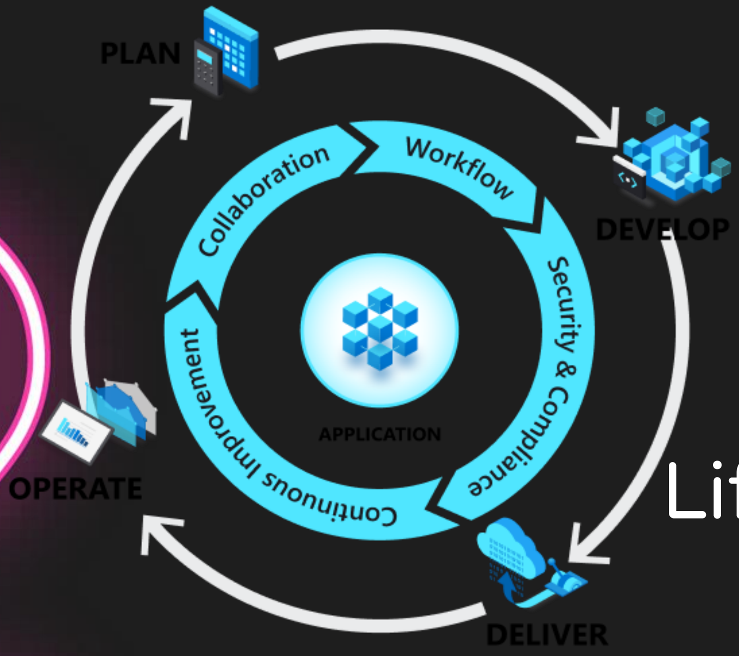


DevOps enables formerly siloed roles – development, IT operations, quality engineering and security – to coordinate and collaborate to produce better, more reliable products.

- 200 X more frequent deploys
- 24 X faster recovery times
- 3 X Lower change failure rates

The numbers tell the story.
In their “State of DevOps Report,” Puppet details that high performing DevOps organizations.

أى سؤال قبل ما نقلب لحوار جديد ؟



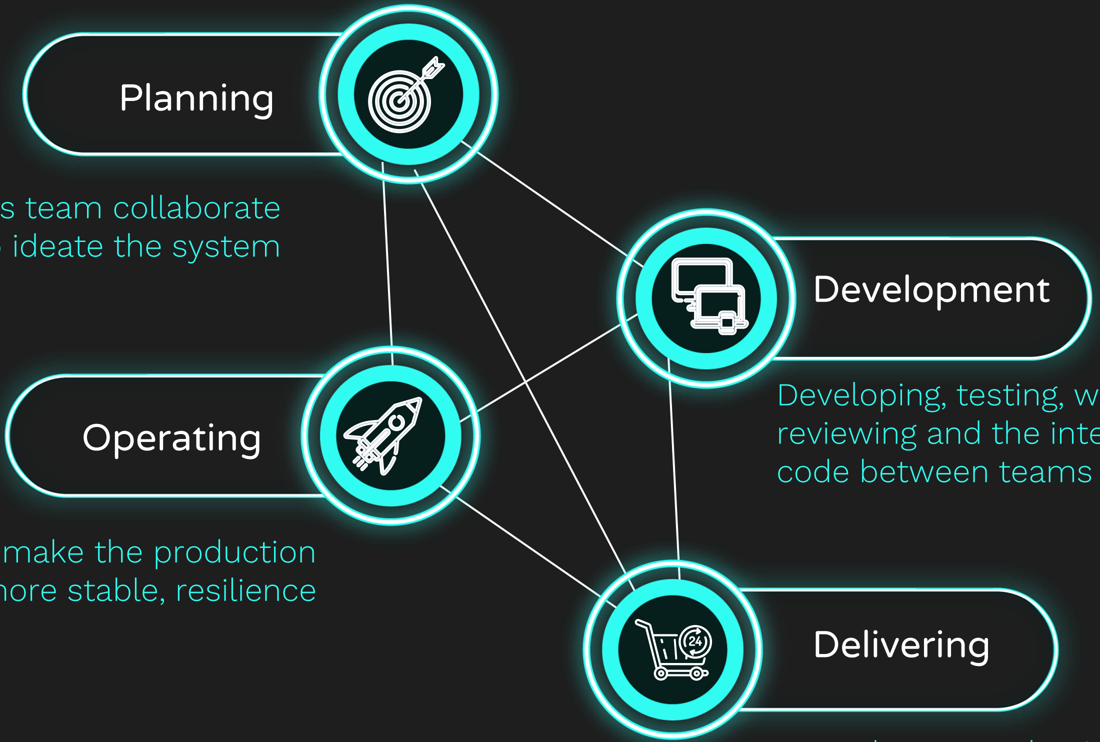
02
Life Cycle

- Plan
- Develop
- Deliver
- Operate

DevOps Lifecycle

DevOps Lifecycle

Each phase is not role-specific but each role is evolved in each phase for some extent at the true environment of DevOps.



DevOps team collaborate to ideate the system

Developing, testing, writing, reviewing and the integration of code between teams

To make the production more stable, resilience

Deploy to production

Planning

DevOps team collaborate to ideate, define and describe features and the capabilities of the system they are building , tracking the progress in high and low granularity levels, from single-product tasks to tasks that span portfolios of multiple products.

An enterprise-wide Agile-DevOps transformation typically starts with a Vision followed by many items such as articulation of expected business outcome.

it's agile-driven in planning it shares the agile practices of Creating backlogs, tracking bugs, managing agile software development with Scrum, using Kanban boards and visualizing progress with dashboards are some of the ways DevOps teams plan with agility and visibility.

Development

This phase include everything about developing, testing, writing, reviewing and the integration of code between teams, it’s relative more about using git and other VCSs to fork, merge, commit, pull, push, for code and project components.

As well as building that code into build artifacts that can be deployed into various environments. These techniques is managed from many frameworks like GitHub and other sources like in our organization GitLab.

DevOps teams seek to innovate rapidly without sacrificing quality, stability and productivity.

To do that as mentioned tools they use highly productive tools, automate mundane and manual steps, and iterate in small increments through automated testing and continuous integration.

Delivering

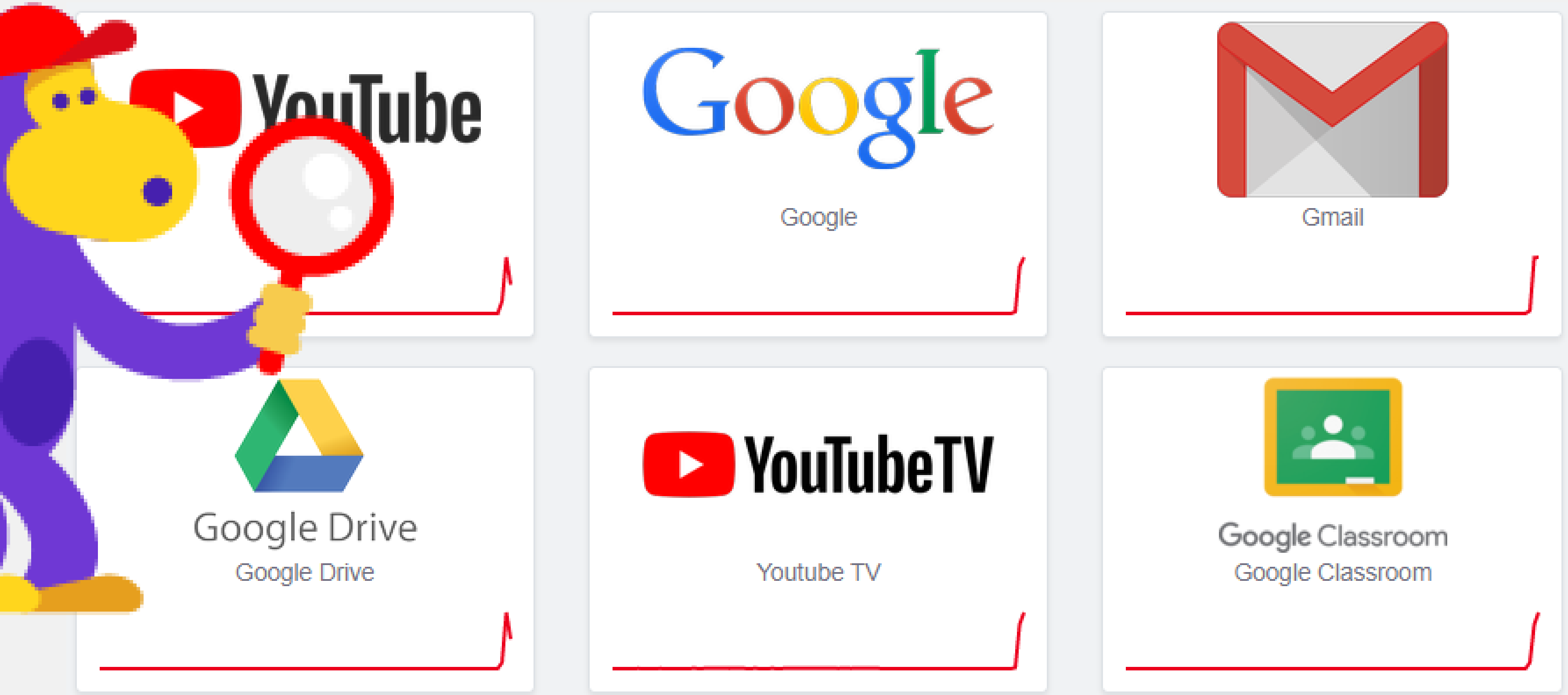
In other concepts of life like to deliver orders to customers in their homes, also the deliver phase is about sending the tested code as running app in production, and it ready to consume from users.

Deploy to production environment means that app is stable enough and reliable to consume from customers.

The deliver phase also includes deploying and configuring the fully governed foundational infrastructure that makes up those environments.

These days the biggest tool use and become a de facto for most of DevOps base system is Docker environment, based on containerization, virtualization approach.

Automating these processes makes them scalable, repeatable, controlled. This way, teams who practice DevOps can deliver frequently with ease, confidence and peace of mind.


YouTube
Google
Google
Gmail
Google Drive
Google Drive
YouTubeTV
Youtube TV
Google Classroom
Google Classroom

Operating

To make the production more stable, resilience. The Operate phase work for maintaining, monitoring, troubleshooting application in production environment. This phase aims to zero downtime while reinforcing security and governance.

DevOps teams seek to identify issues before they affect the customer experience and mitigate issues quickly when they do occur.

Maintaining this vigilance requires rich telemetry, actionable alerting and full visibility into applications and the underlying system.

أى سؤال قبل ما نقلب لحوار جديد ؟

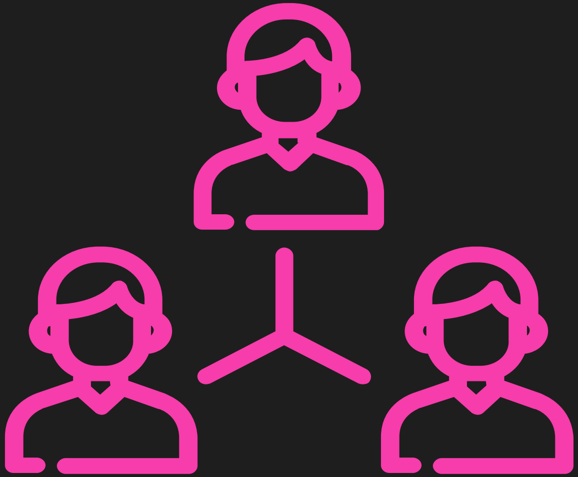
03
Culture and Practices

- CSSC
- CI/CD, VCS, Agile, IaC, Configuration Management, Continuous
Monitoring

DevOps Culture

While adopting DevOps practices automates and optimizes processes through technology, it all starts with the culture inside the organization - and the people who play a part in it.

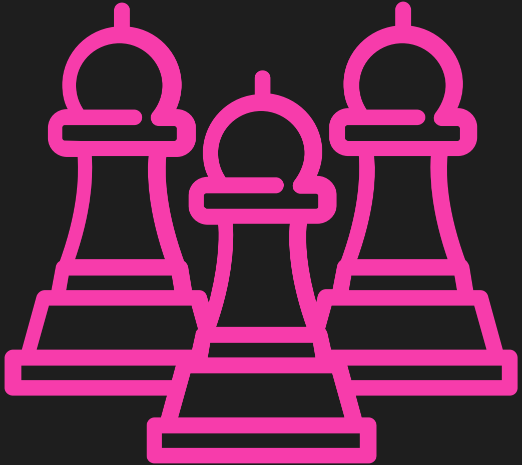
Collaboration, Visibility and Alignment



The healthy DevOps environment come from visibility, The IT Operation team and the development must share their DevOps processes, priorities and concerns to make the whole environment more stable.

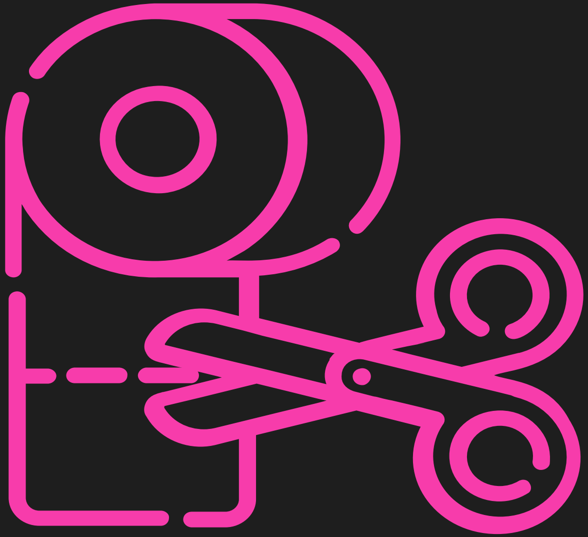
These teams must also plan work together as well as align on goals and measures of success as they relate to the business.

Shifts in scope and accountability



As mentioned the DevOps is not a man job, but the whole team job, the environment is shared, the team is align so take the ownership and become involved in additional lifecycle phases – not just the ones central to their roles, for example the developers become accountable in all phases not only the plan and develop but also the deploy and operate because the performance and stability is related to the code and the entire production environment.

Shorter release cycles



DevOps is based on agile techniques, which are based on cycles like SCRUM sprints and many other frameworks in agile adapt these cycles, these cycles have incremental behavior, so the progress is incremental, these cycles make it easier to risk management and planning, which is also reduce system stability.

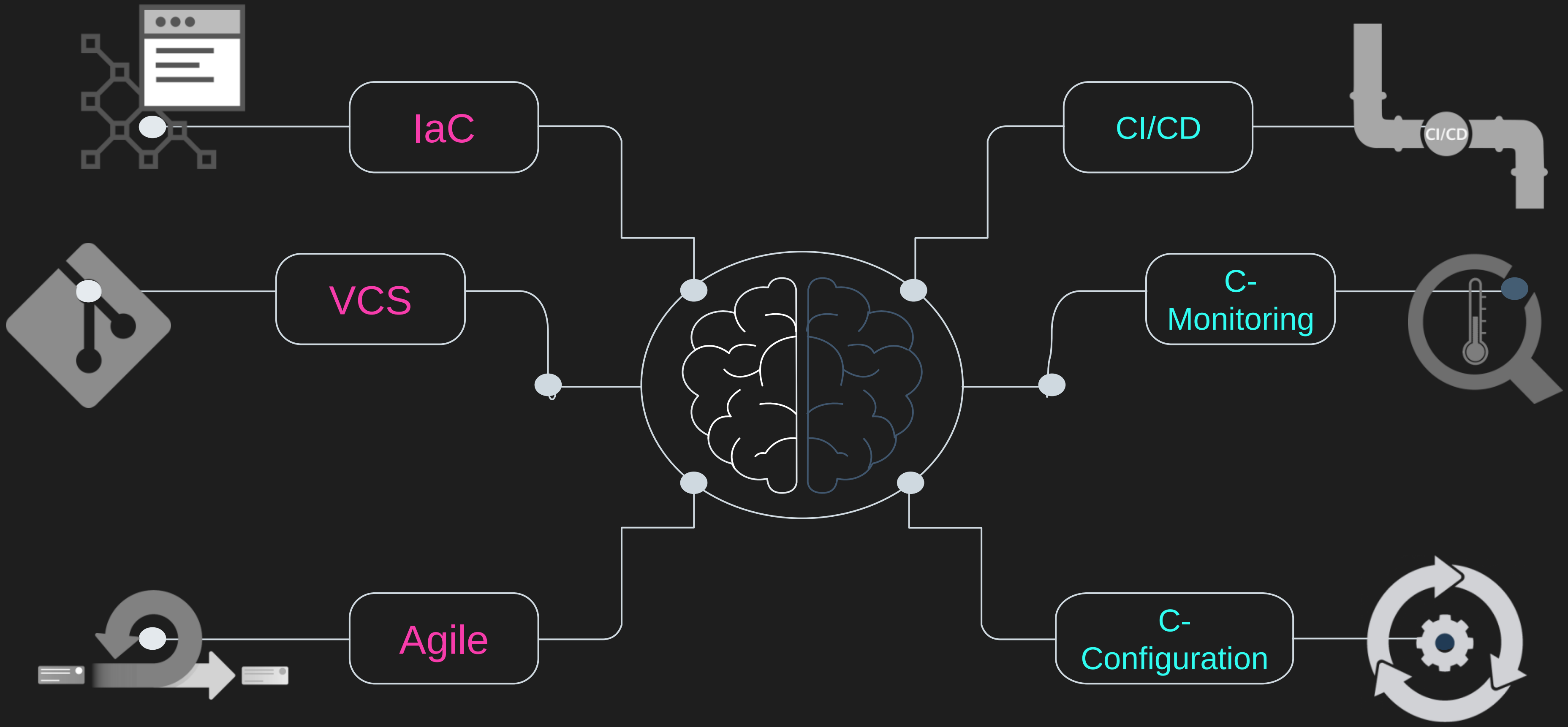
Shortening the release cycle also allows organizations to adapt and react to evolving customer needs and competitive pressure.

Continuous learning

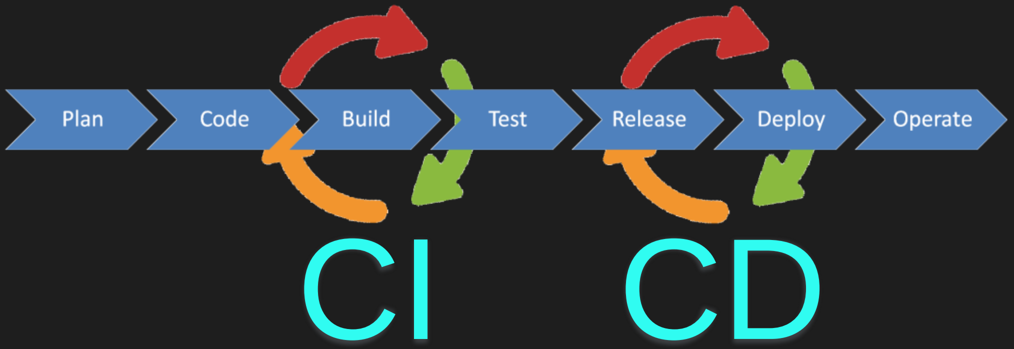
DevOps is like a journey, it's like a room to grow, when the high performance DevOps team establish their growth mindset, the fail fast and incorporate learnings into their process.

This behavior can continually improving, increasing customer satisfaction, and accelerating innovation and market adaptability.

DevOps Practices


IaC
VCS
Agile
CI/CD
CI/CD
C-Monitoring
C-Configuration

Continuous Integration/Continuous Delivery



Continuous integration is a software development practice in which developer merge code changes frequently into the main code branch. And it employ the continuous testing or the automated testing using some configurations, which run every time the code committed so the code in the main branch is stable.

Continuous delivery is the frequent, automated deployment of application into production environment, this accelerate the deployment strategies and reduce issues, as it frequent pattern of deployment to the same production environment.

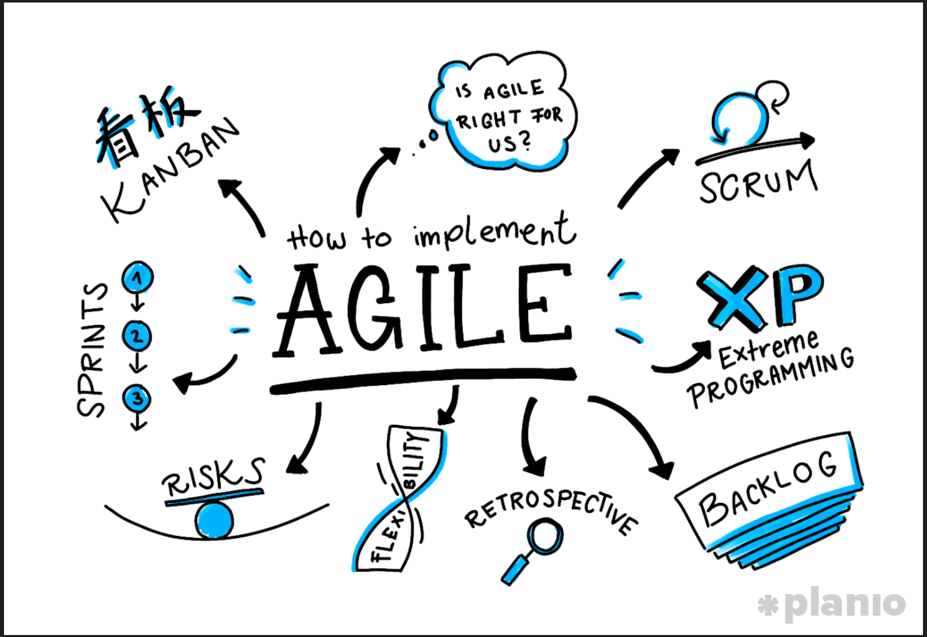
Version Control

Version control is a practice to manage the code into series of versions, as a practice, tracking revisions and change history to make code easy to review and recover. Like any system the user consume, it have many version of release and each number has its own features.

The practice is usually implemented using Version Control System like Git which allows multiple developers to collaborate and authoring code.

The VCSs usually contain many tools to merge, pull, and manage the process in a same file of development, manage conflicts and roll back changes in early state of failure.

Agile

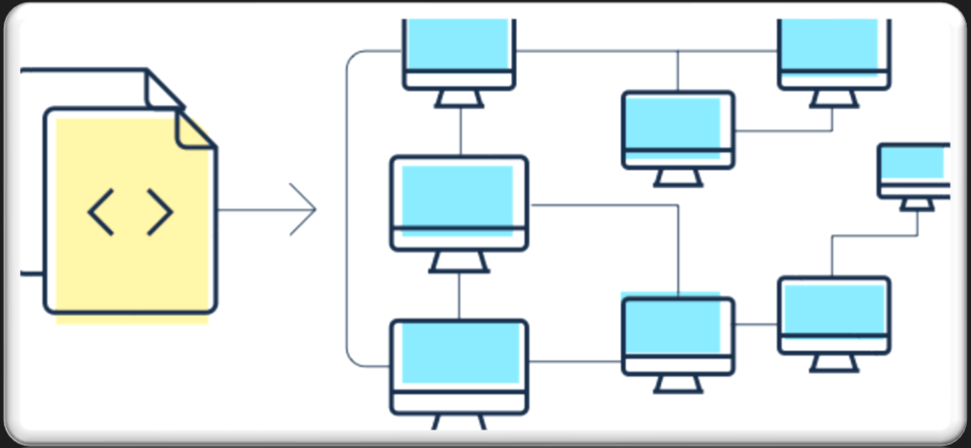


Agile is a modern approach for software development that emphasizes team collaboration, customer and user continuous feedback, and high adaptability to change through short release cycles. The feedback from users is the main change director adjusted from the (need and want) of business today engines.

Agile is so different from other traditional frameworks like waterfall or RAD, they have a long release cycles defined by sequential phases.

Kanban and SCRUM are two most popular frameworks associated with agile.

Infrastructure as Code



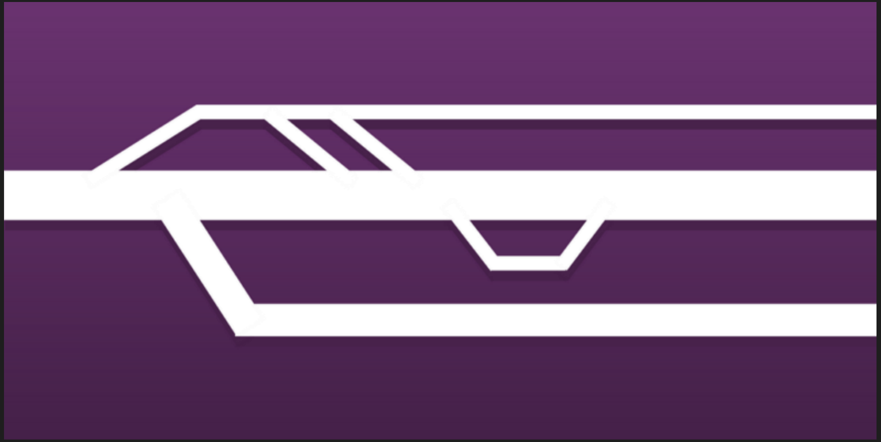
Infrastructure management means to configure and manage the software running on its process over its hardware.

In the past, the managing of IT infrastructure is so hard job, system administrator manually manage the whole process of managing the infrastructure, by writing commands of shell and monitoring every process repletely.

Today, IaC defines system resources and topologies in a descriptive manner that allows the team to manage these resources as they code.

That mean to write just little files to manage the infrastructure within your project.

Configuration Management



The CM concept is working to find a stable way and tools to manage resources state, including servers, virtual machines and databases, using Configuration Management help the teams to roll out the changes in the environment once it happened.

Teams use the Configuration management tools to track system state and help to avoid the configuration drift, which is how a system resource’s configuration deviates over time from the desired state defined for it. It is practiced in conjunction of IaC, both system definition and configurations are easy to templatize and automate, helping teams to operate the system in a massive scale.

Continuous Monitoring

Continuous Monitoring mans to have the real time visibility into the performance and health of the application stack or pool, that delivered to users from the infrastructure running to serve the application to the higher level software component.

The visibility is containing the telemetry and metadata and some predefined alerts as a predefined conditions which warrant attention from an operator, telemetry comprises the events and logs of the system that collected from its components, stored and collected and monitored when the operate.

أى سؤال قبل ما نقلب لحوار جديد ؟

04

DevOps a New Controller

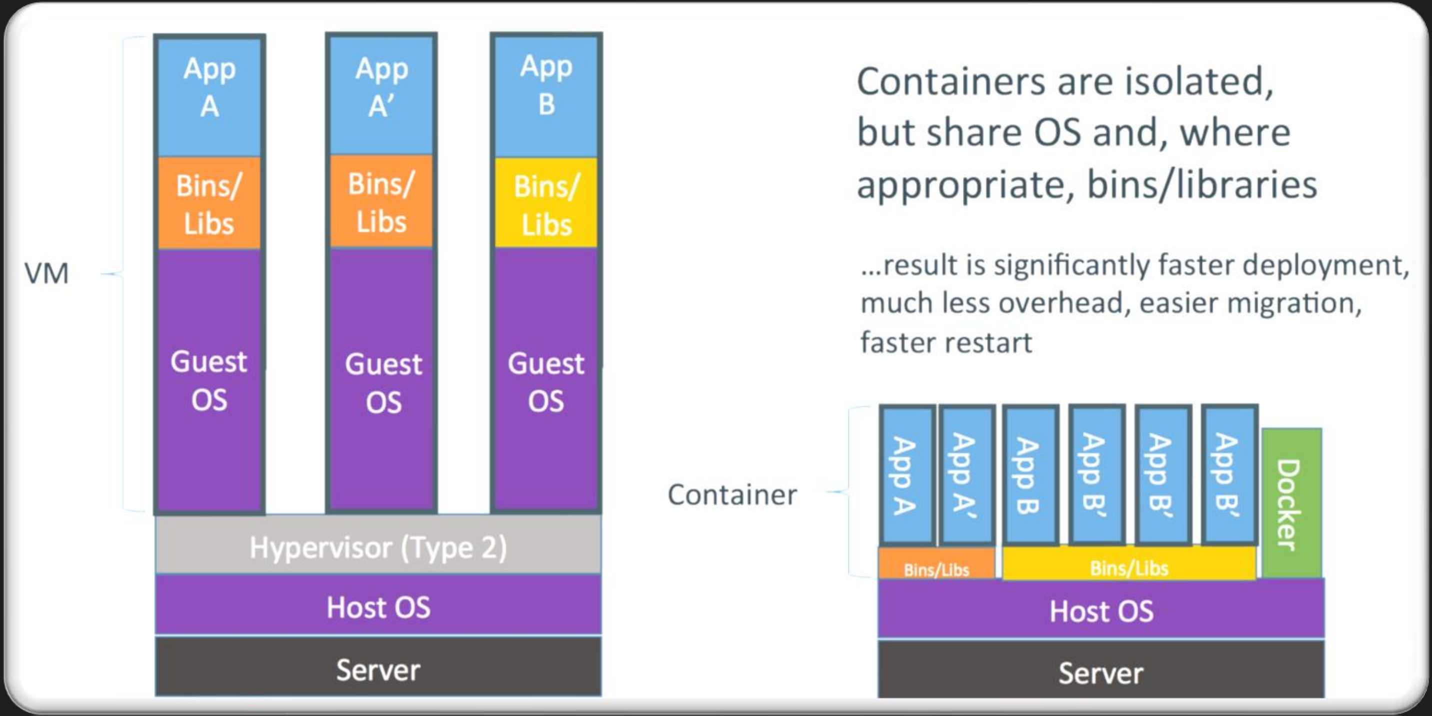
- Microservices
- Mobile
- Web
- IOT
- Data Science

Docker

Developing apps today is so challenging, the challenge is how to write code faster, reusable, secure for environments, shippable to its containers and can be changed with its domain.

The coders can work with multiple framework with complex architectures like Microservices, docker is the right tool that helps the developer to achieve their complex apps of innovation, docker simplifies and accelerate the workflows, it give the developer all his/her needs to innovate with code, no matter this code written, docker is the environment that can ship the developer code to production environment, with simple configuration in code.

Virtualization vs Containerization



Containers are isolated, but share OS and, where appropriate, bins/libraries

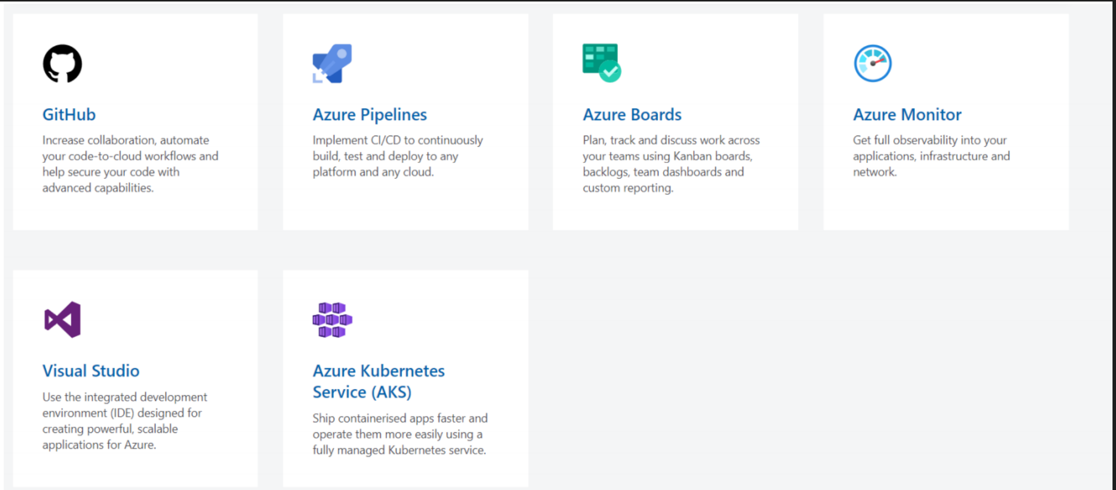
...result is significantly faster deployment, much less overhead, easier migration, faster restart

Large Organizations Toolchains

Microsoft

The teams use the DevOps tools to facilitate the culture of DevOps environment in their organizations, not a one tool they use, but various of solutions had been founded and they can either customize their own tools, these tools are a chain called a toolchain because it processes the delivery from a previous tool and deliver its product to another tool like automated toolchain.

Using these toolchains can predefined the whole process of DevOps, this is a streamline of events and milestones, getting these tools to put DevOps into Practice.



GitHub
Increase collaboration, automate your code-to-cloud workflows and help secure your code with advanced capabilities.

Azure Pipelines
Implement CI/CD to continuously build, test and deploy to any platform and any cloud.

Azure Boards
Plan, track and discuss work across your teams using Kanban boards, backlogs, team dashboards and custom reporting.

Azure Monitor
Get full observability into your applications, infrastructure and network.

Visual Studio
Use the integrated development environment (IDE) designed for creating powerful, scalable applications for Azure.

Azure Kubernetes Service (AKS)
Ship containerised apps faster and operate them more easily using a fully managed Kubernetes service.

Integrations

Cloud

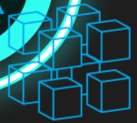
People think that DevOps was born from cloud to help cloud, The big answer is the team turn cloud functionality from serving only users to serve them for their working.

DevOps is a progression of Agile development, but it really need cloud, and the cloud is vital to accomplishment with DevOps.

In many ways, what seems to be support of a perfect system in DevOps goes out to be nobody more than an explanation of ordinary realism in the cloud. If DevOps hadn't full-grown out of the Agile drive then it, or somewhat very DevOps-like, would nearly surely have industrialized from cloud deployment performs as a practical response to the competences and tasks of the cloud.

DevOps and the cloud are a usual contest for each other. But this isn’t merely a ‘nice-to-have’ - originalities can save masses by using cloud stages and DevOps ways of working in grouping.

Microservices



In Versus of Monolithic architecture, Microservices are characterized by applications that are independently scalable and decentralized.

They are applications intended to be readily replaceable thanks to their independent and autonomous nature.

Microservices are an extension of containerized applications and are functional components with standardized interfaces to enable the disparate services to function as a singular whole from the user's perspective.

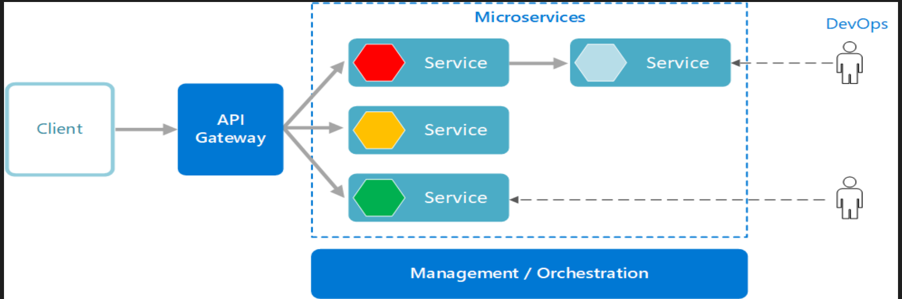
Instead of creating a massive, complex beast of a program, the microservice architectural style promotes the creation of a single application in the form of a suite of smaller services that each run their own processes.

Microservices

DevOps practices reinforce the idea of breaking large problems into smaller pieces and tackling them one at a time as a team.

Microservices fits perfectly into the DevOps ideals of utilizing small teams to create functional changes to the enterprise's services one step at a time. Microservices empower the implementation of small teams collaborating together in an environment of increased developer freedom.

Furthermore, microservices can readily scale up or down without impacting resource allocations for the rest of the system.



IOT

Most don't understand the connections. If IoT systems managers asked where DevOps actually fits in, most don't understand. It's actually pretty straight forward, and not at all complex.

First of all, IoT is something that comes into play with automated security testing (continuous testing), integration testing (continuous integration), and, finally, deployment (continuous deployment).

When the IoT application is tested, a DevOps system needs to consider security around the information coming from sensors.

These are points-of-entry that are easily compromised, such as somebody hacking into your thermostat and sending misleading data.

DevOps security testing tools need to regress through the IoT application to ensure that exposure is minimized.

IOT

While we’re still playing around with DevOps and IoT, it’s clear that the connections can be made, and we can indeed benefit from the marriage of IoT and DevOps.

Those who build IoT applications now, such as IoT software providers, already understand the connections and continue their progression toward synergy. Enterprises are new to DevOps and have yet to get onboard. It’s just a matter of time until one won’t exist without the other.

Microsoft
Azure IoT Edge development with Azure DevOps
http://aka.ms/iotedgedevops
For instructions, source code, examples, Etc.
Contributions encouraged!
Follow me on Twitter
@pjdecarlo
The IoT Show

THANKS!

Do you have any questions?

Progeng_Ahmed_Khalil@outlook.com
+20 1 989 43 5 43