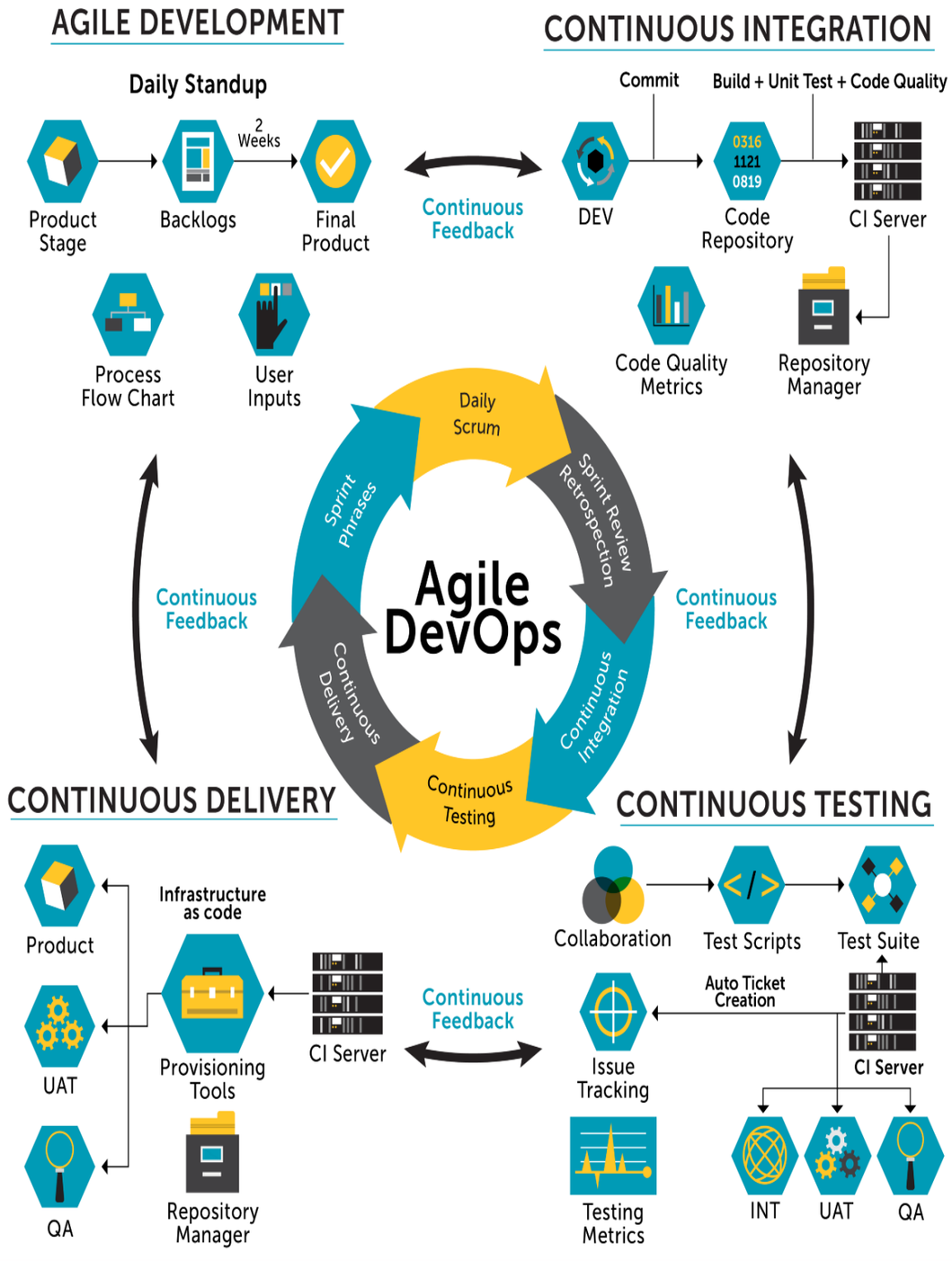
Introduction to DevOps

DevOps is a software development approach that combines development (Dev) with IT operations (Ops).
It is a culture to promote the development and operation process collectively.
It aims to shorten the systems development life cycle and deliver high-quality software.

- Bhakti Menkar
Batch: 2023-10427
Enrollment No: EBEON0923756263



What is DevOps?

Collaboration & Integration

DevOps is a culture and practice emphasizing collaboration and integration between software developers and IT operations teams.

Automation & Tooling

It involves the use of automation and specialized tools to streamline the software development and delivery process.

Continuous Delivery

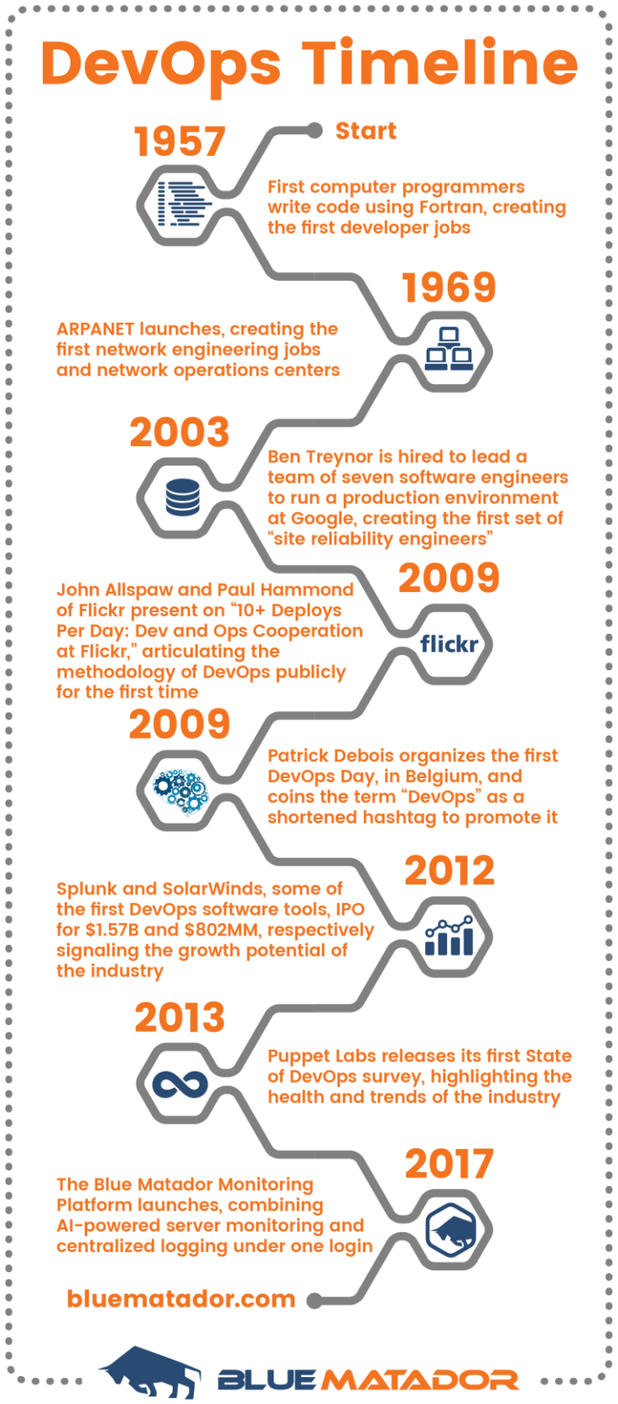
DevOps aims for continuous delivery, where code changes are released to production frequently and reliably.

Reduces Disconnection

DevOps helps you to reduce the disconnection between software developers, QA engineers, and system administrators.

Why DevOps?

- The operation and development team worked in complete isolation.
- Team members are spending a large amount of time on designing, testing, and deploying instead of building the project.
- Manual code deployment leads to human errors in production.
- Coding and operation teams have their separate timelines and are not in sync, causing further delays.
- Fosters a culture of innovation as teams are empowered to experiment and release new features quickly.
- Software can be delivered faster and more frequently, with improved quality due to increased focus on testing and feedback.
- Automation reduces manual work and errors, leading to cost savings.
- Improved collaboration and a shared sense of ownership boost team morale and motivation.



History

1

Early Beginnings

DevOps can trace its roots back to the Agile Manifesto in the early 2000s. It emerged as a response to the traditional siloed approach to software development and IT operations.

2

Collaborative Practices

In 2009, the concept of DevOps gained traction with the "10 Deploys Per Day" presentation by John Allspaw and Paul Hammond, showcasing the benefits of collaboration between development and operations teams.

3

Growth and Evolution

As organizations recognized the need for speed, efficiency, and innovation, DevOps principles continued to evolve, leading to the establishment of best practices and a cultural shift in the industry.

DevOps Architecture Features

Automation

- Reduce time consumption
- The productivity increases
- Releases are made quicker
- Catching bugs quickly so that it can be fixed easily.

Collaboration

- Improves the cultural model
- Teams become more productive
- Shared responsibilities and work closely in sync, which in turn makes the deployment to production faster.

Integration

- continuous integration and delivery are implemented to deliver in a quicker, safer, and reliable manner.
- There are significant operational challenges while integration and testing

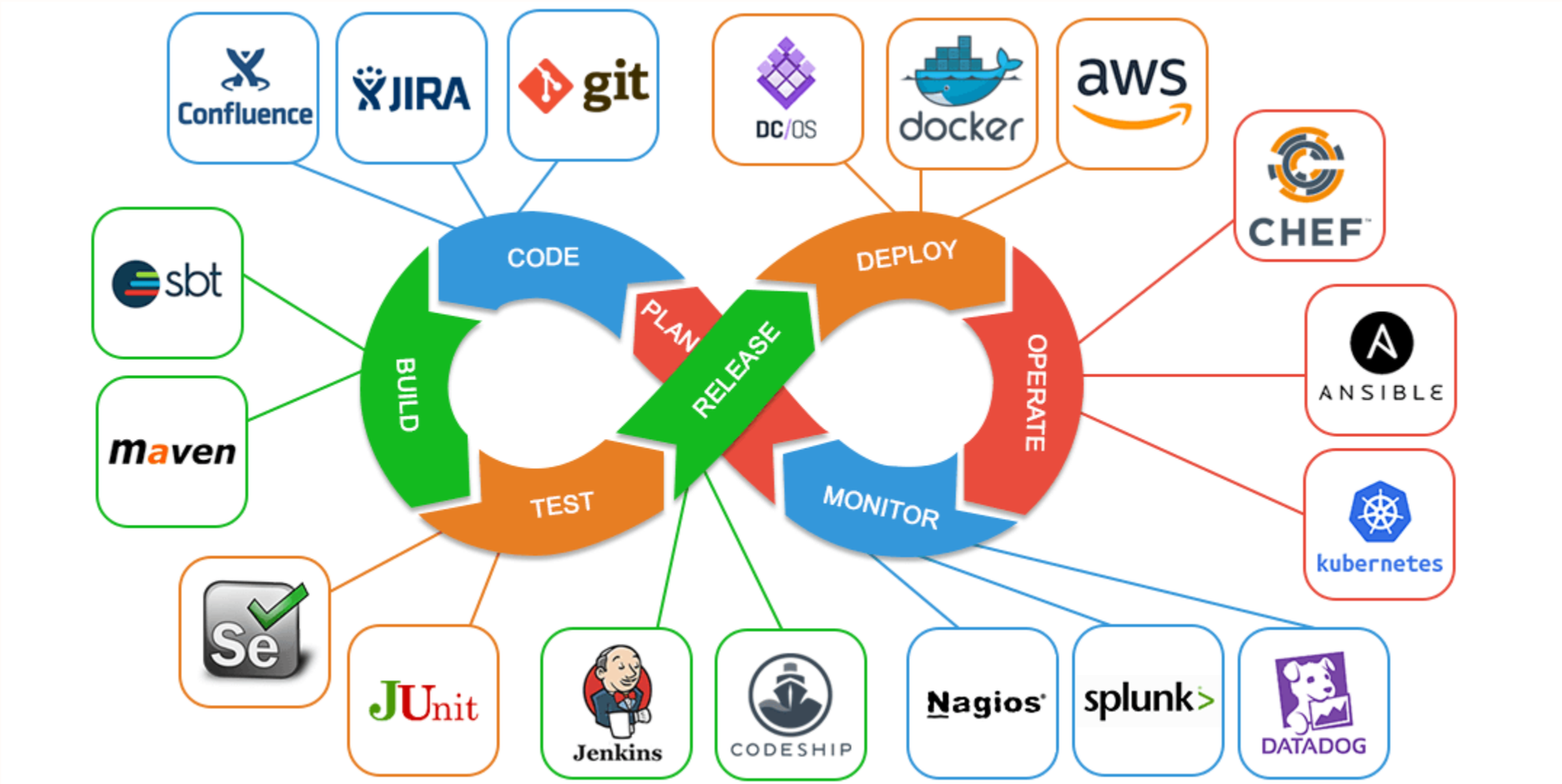
Configuration Management

- Application interacts with only those resources that are concerned with environment in which it runs.
- The configuration file can be written during deployment, or they can be loaded at the run time, depending on the environment in which it is running.

DevOps Architecture

DevOps Tools and Technologies

DevOps tools and technologies encompass a wide range of software, platforms, and infrastructure-as-code solutions designed to streamline software development, testing, deployment, and monitoring. These tools enable automation, collaboration, and the implementation of DevOps practices, fostering faster delivery, improved quality, and more efficient operations across the entire development lifecycle.



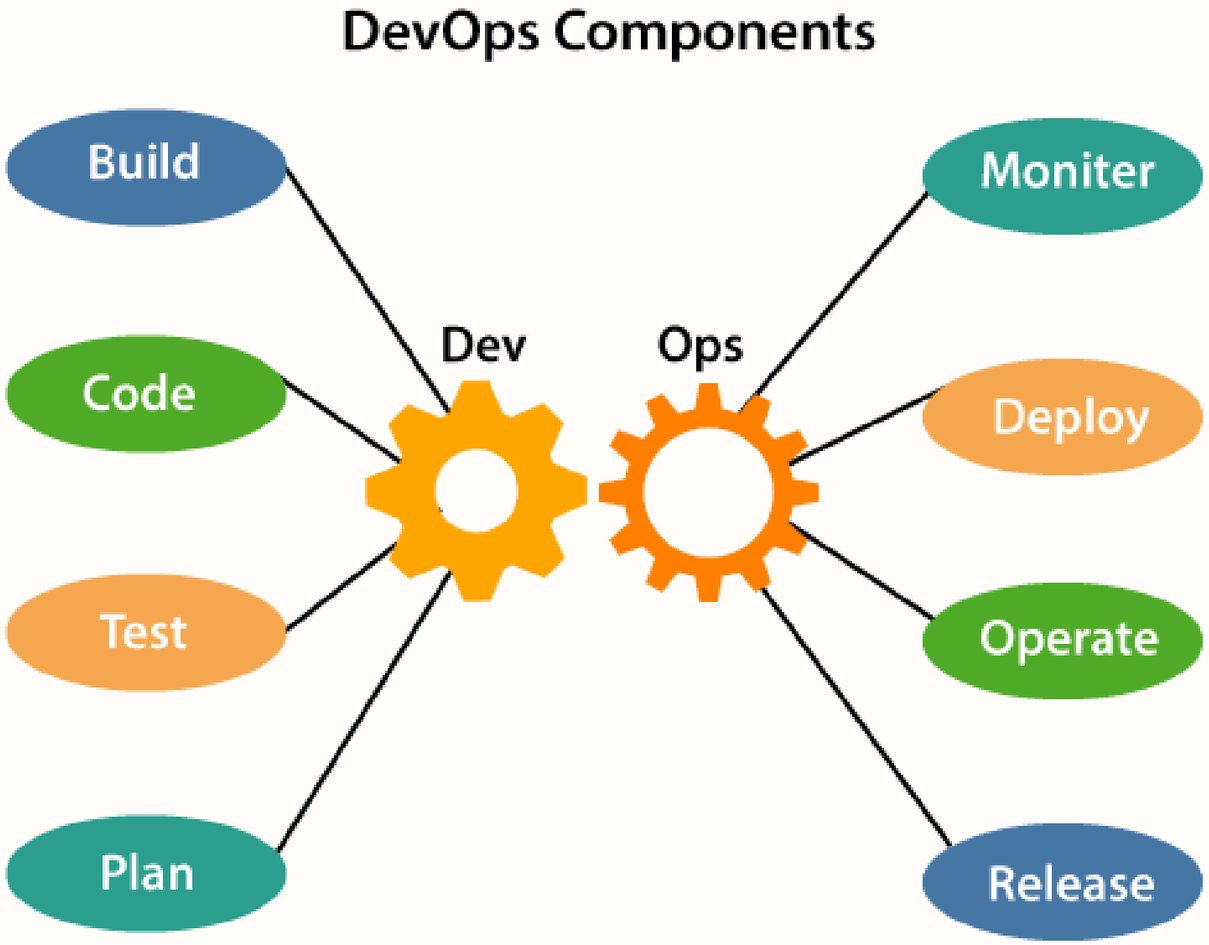
DevOps Components

Compile and assemble code into executable artifacts

Developers write code to implement the planes features and fixes

Insure code quality, security and functionality through testing

Define the requirements and objectives for software development and deployment



Continuously monitor the application's performance and user experience

Roll out the application to various environments

Ensure that the application performs smoothly in production

Prepare and package the application for deployment

DevOps Lifecycle

- Planning and coding of the software
- Vision of the project is decided
- No DevOps tool required for planning, but required for maintaining the code

- Based on the continuity with complete automation
- Critical factor in DevOps
- More efficient software product

Development

Operations

Integration

- Heart of the lifecycle
- Frequent commits to the source code
- Daily or weekly basis
- Allows early detection of problems if any

- Code is deployed to production server
- Ensures that the code is correctly used on all servers
- Vagrant and Docker are popular tools that are used in the process

Deployment

Testing

- Continuous testing for bugs
- Automation testing tools such as TestNG, JUnit, Selenium etc are used.
- No flaw in functionality is ensured

Feedback

Monitering

- Constant feedback about the next versions between operations and development
- Analysis for software working results
- Application development consistently improved

- Involves all operational factors of process
- Important information about the use of software is recorded
- It is integrated within the operational capabilities of software

Advantages

1 **Excellent Approach**

Faster response to market changes and improves business growth

2 **Enhanced Quality Decreased Costs**

Decreased failure rate, delivery and transportation costs of new releases and faster resolution of issues.

3 **Collaborative Work Environment**

Enhanced collaboration and communication between development and operations teams.

4 **Profit Growth**

Clarity in product development and delivery results into escalated business profits

Disadvantages

1 **Talent Gap**

DevOps professional or expert's developers are less available.

2 **Expensive Approach**

Developing with DevOps is so expensive.

3 **Difficult to Manage**

Adopting new DevOps technology into the industries is hard to manage in short time.

4 **Lack of knowledge**

Lack of DevOps knowledge can be a problem in the continuous integration of automation projects.