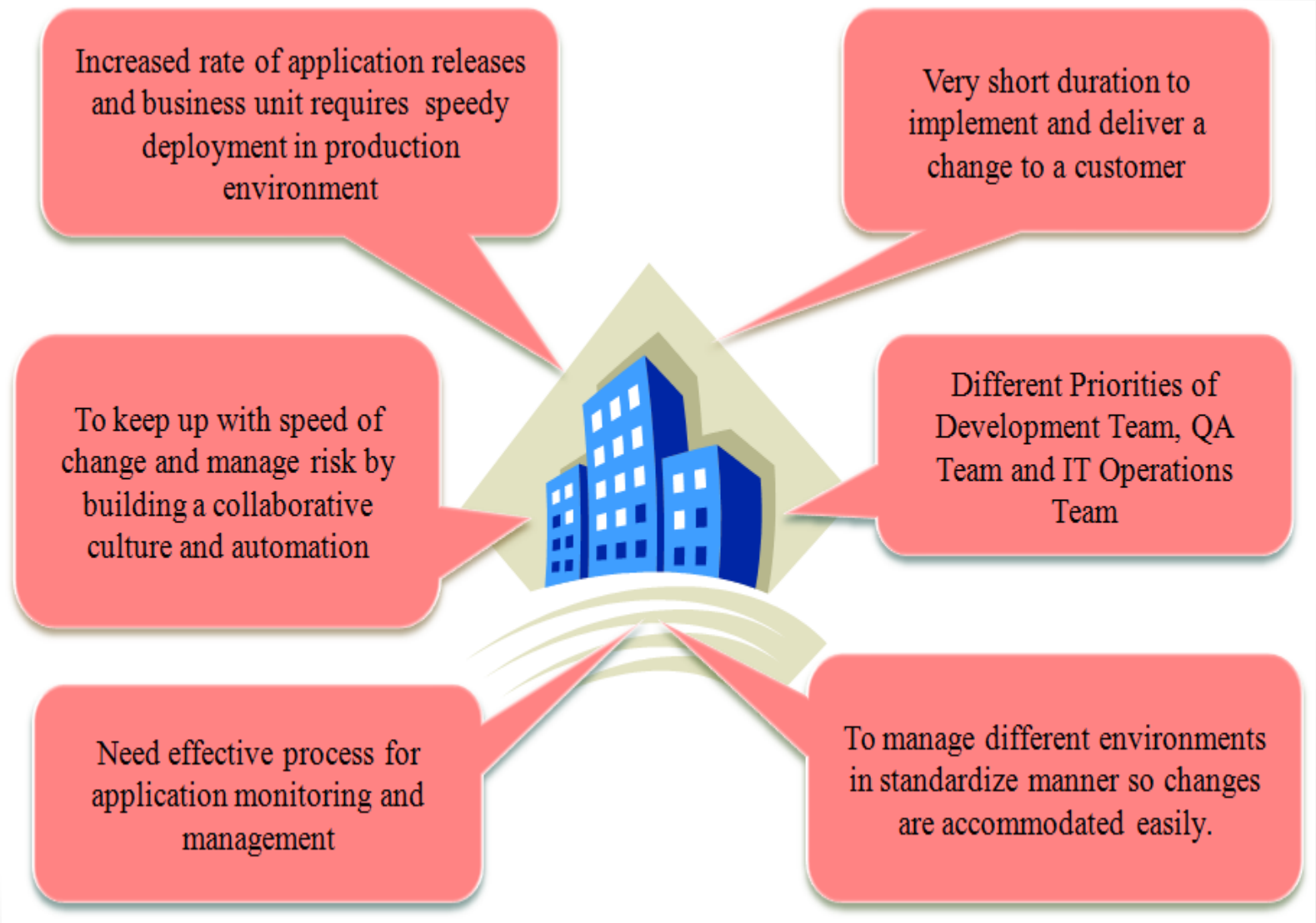
DevOps

DevOps

- Need for DevOps
- How DevOps culture can evolve
- Importance of PPT—people, process, and technology
- Why DevOps is not all about tools
- DevOps assessment questions

Need for DevOps



Need For DevOps

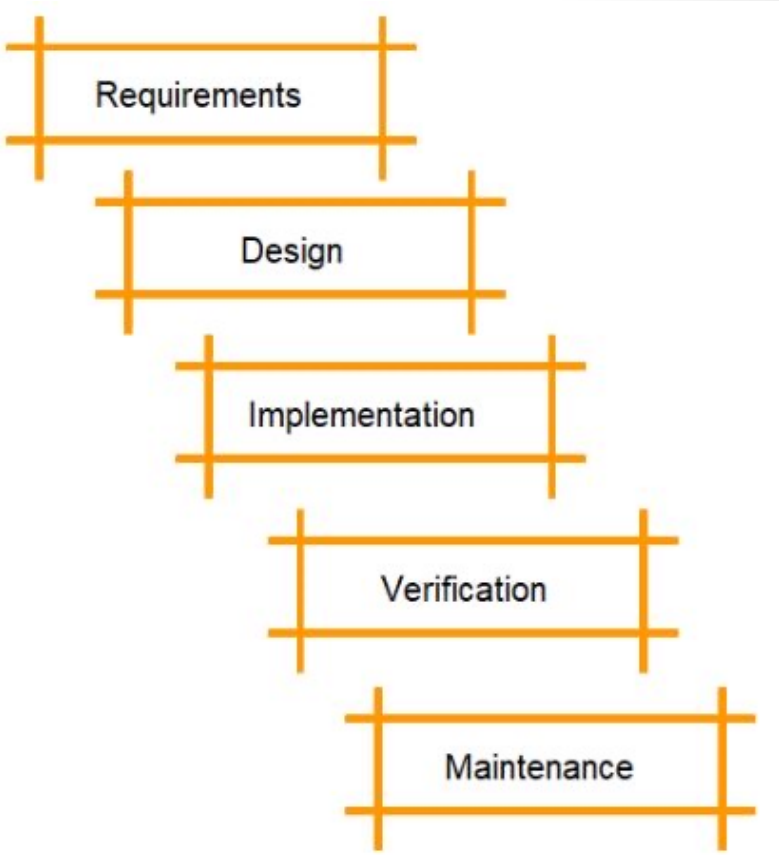
- Considering the changing patterns and competitive environment in business,
- it is the need of the hour to improve application life cycle management.
- Are there any factors that can be helpful in these modern times which can help us to improve application life cycle management?
- Yes. Cloud computing has changed the game. It has opened doors for many path-breaking solutions and innovations.
- Let’s understand what cloud computing really means and how
- terms like DevOps and automation play an important role for enterprise companies.

Why DevOps?

- Before DevOps, there were two development models: Waterfall and Agile Method.

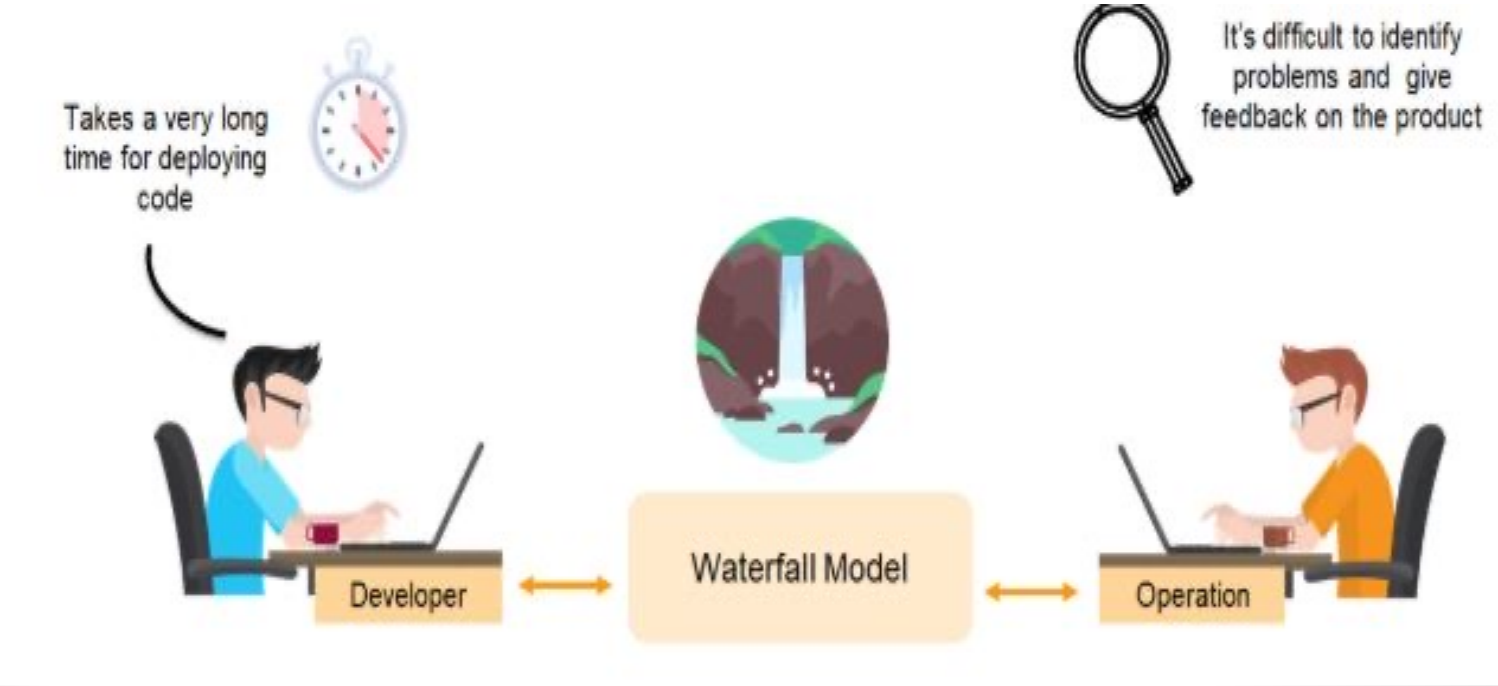
Waterfall Model

- The waterfall model is the first model to be introduced in software development.
- It is a sequential process and very easy to understand.
- In this approach, software development is divided into several phases, and the output of one phase becomes the input for the next phase.
- This model is similar to a waterfall when the water flows off from the cliff; it cannot go back to its previous state.



Drawbacks of the waterfall model:

- It’s difficult to make changes to the previous stage
- Not recommended for large-sized projects
- Developers and testers don’t work together (which can result in a lot of bugs at the end)
- Not recommended for projects that will likely have changing requirements.

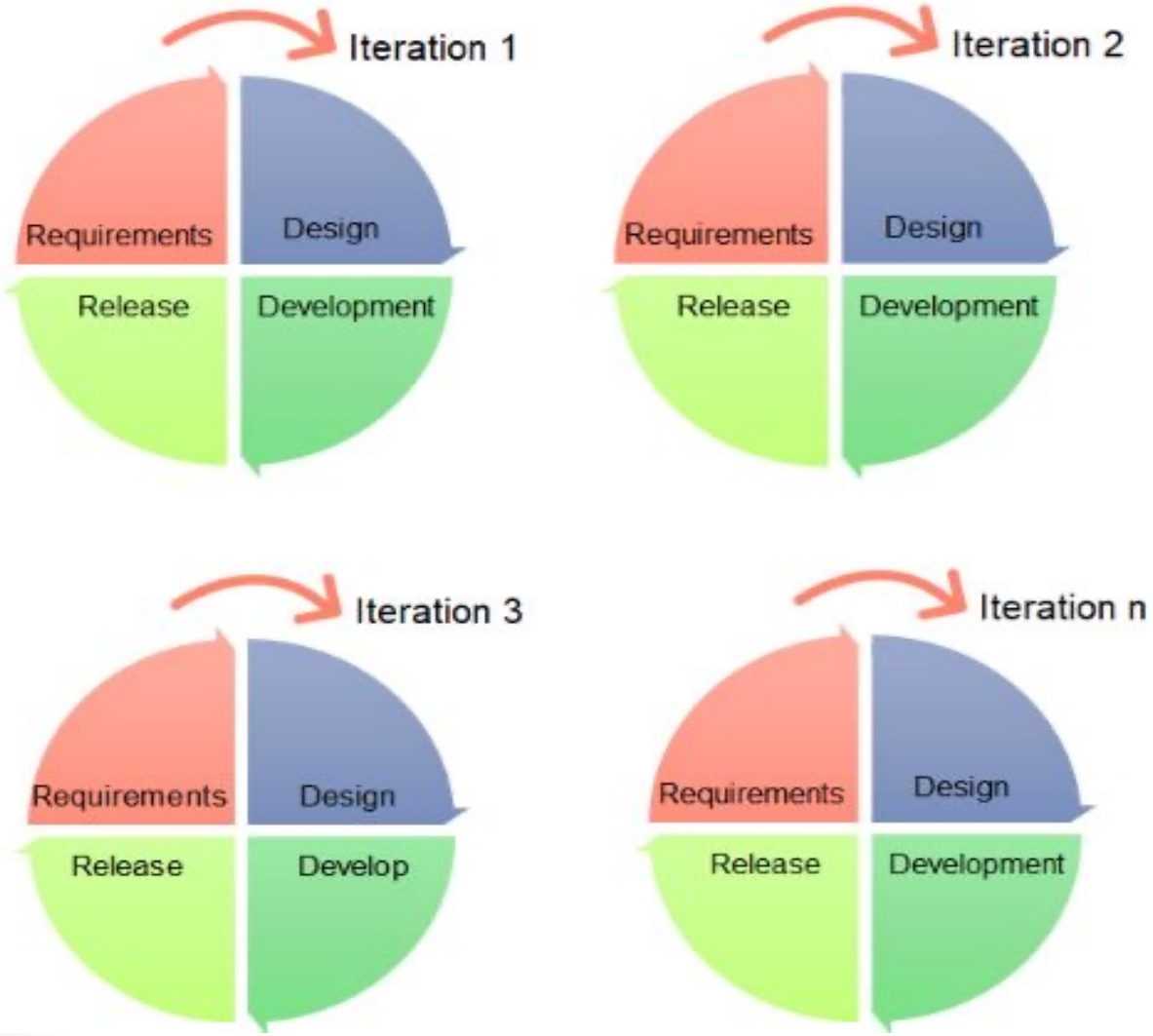


Agile Model

- Agile is an approach in software development where each project splits into multiple iterations.
- As a result, at the end of each iteration, a software product is delivered.
- Each iteration lasts about one to three weeks.
- Every iteration involves functional teams working simultaneously on various areas, such as:
    - Requirements
    - Design
    - Development
    - Release

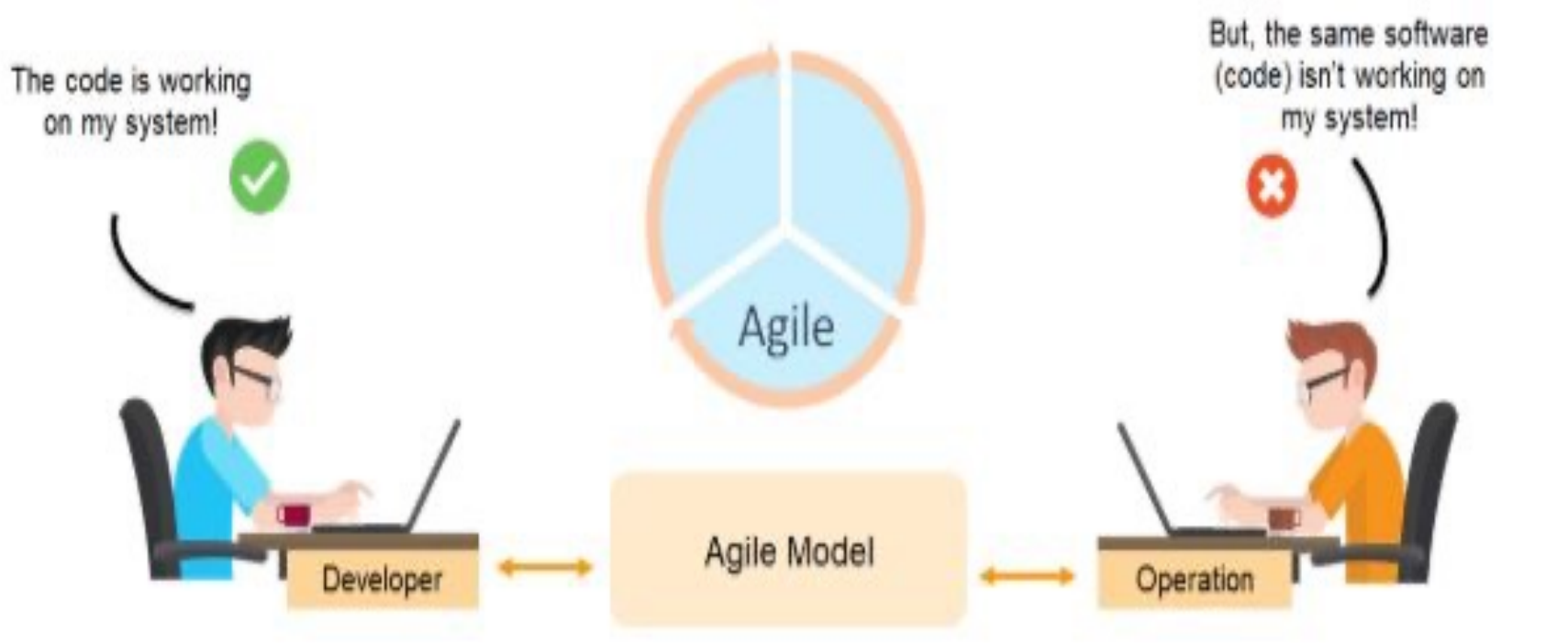
Agile Model

The figure below indicates that there can be a number of iterations needed to deliver a final product in agile method.



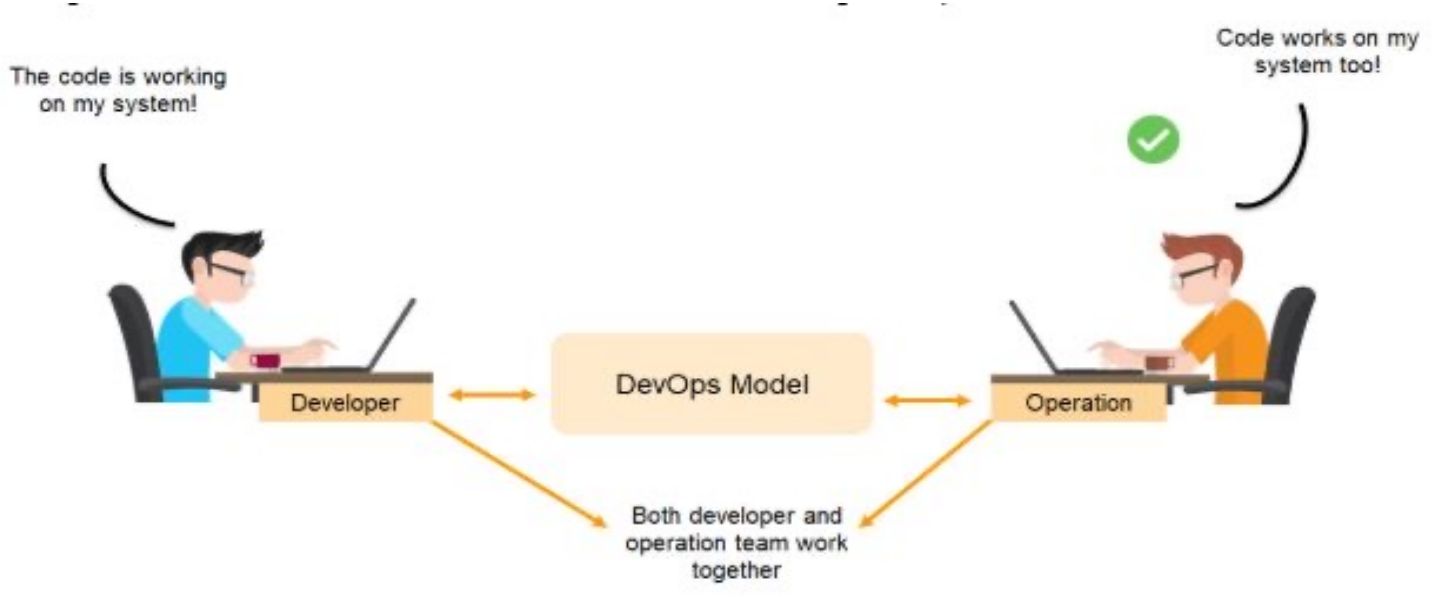
Agile Model

- Using the agile method, the code that works for the developer may not work for the operations team.



DevOps Over Agile Model and Waterfall model

- With DevOps, there is continuous integration between deployment of code and the testing of it.
- Near real-time monitoring and immediate feedback through a DevOps continuous monitoring tool enables both the developer and operations team work together.



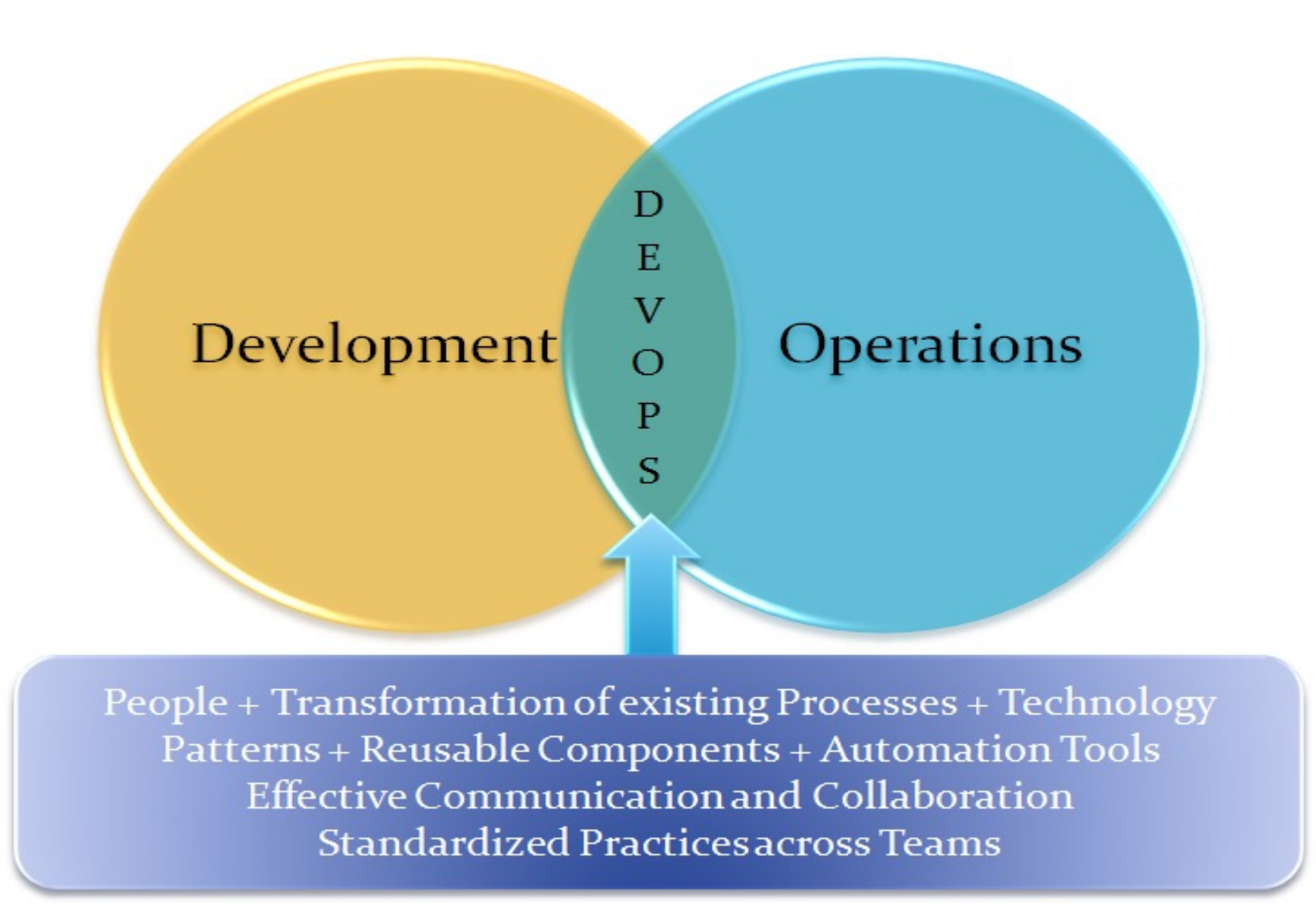
Overview of DevOps

- DevOps is all about the culture of an organization, processes, and technology to develop communication and collaboration between development and IT operations teams to manage the application life cycle more effectively than the existing ways of doing it.

It helps to define practices for

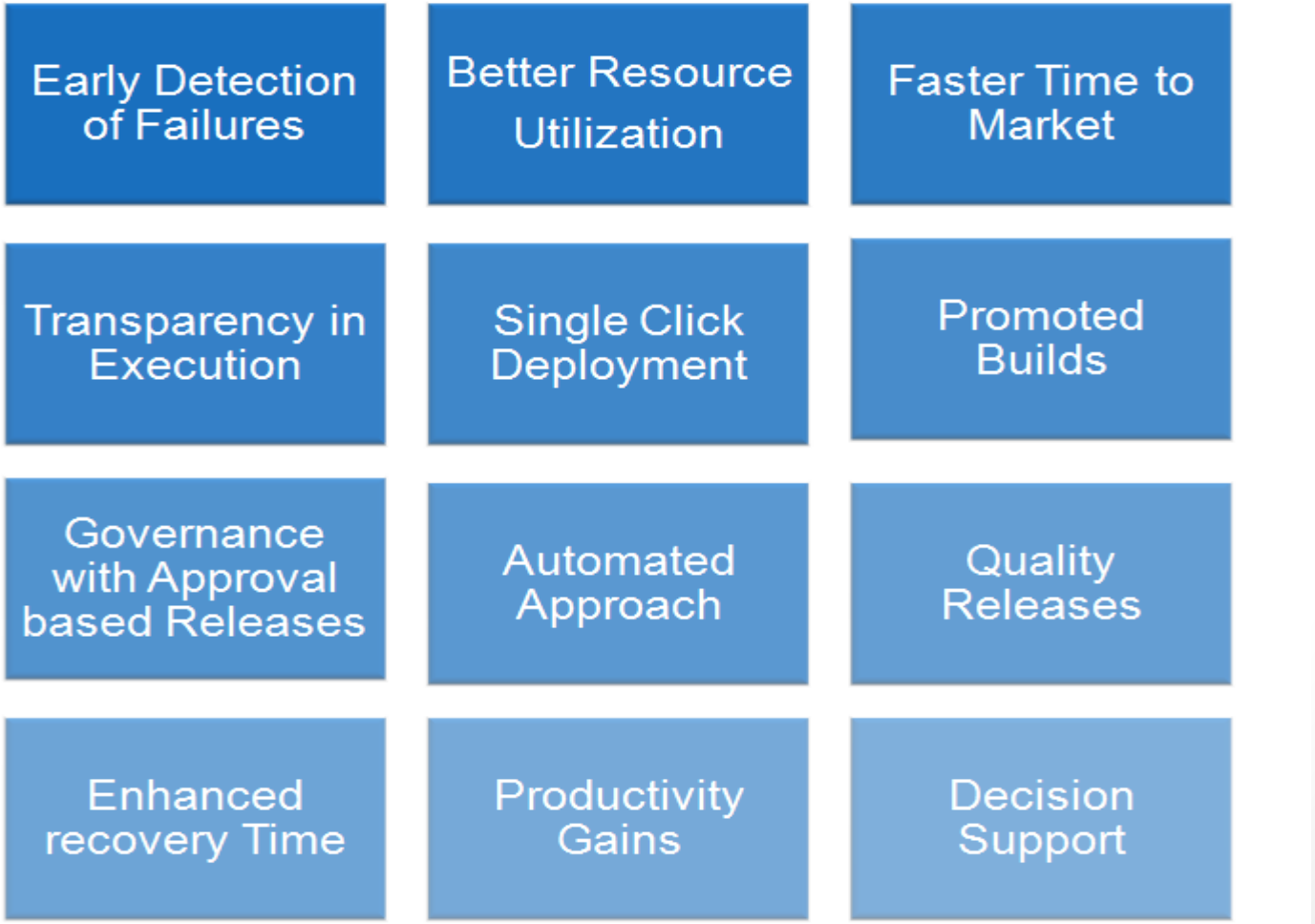
- a way of designing,
- a way of developing,
- a way of testing,
- a way of setting up resources,
- a way of managing environments,
- a way of configuration management,
- a way of deploying an application,
- a way of gathering feedback,
- a way of code improvements,
- a way of doing innovations.

Overview of DevOps



Overview of DevOps

The following are some of the visible benefits that can be achieved by implementing DevOps practices



What is DevOps?

- DevOps is a combination of the two words “development” and “operations.” Patrick Debois, a DevOps expert, came up with the term “DevOps” in 2009 and it stuck ever since.
- Some people say that it was around this time that there was a shift in IT culture, and DevOps represents this shift.
- DevOps is an umbrella term that describes the operation of a team collaborating throughout an entire programming production process - from the design through the development stages

- DevOps programmers typically use conventional infrastructure management and software development processes.

- When it comes to software development, DevOps tends to take an Agile approach.

DevOps

- DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability **to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations** using traditional software development and infrastructure management processes.

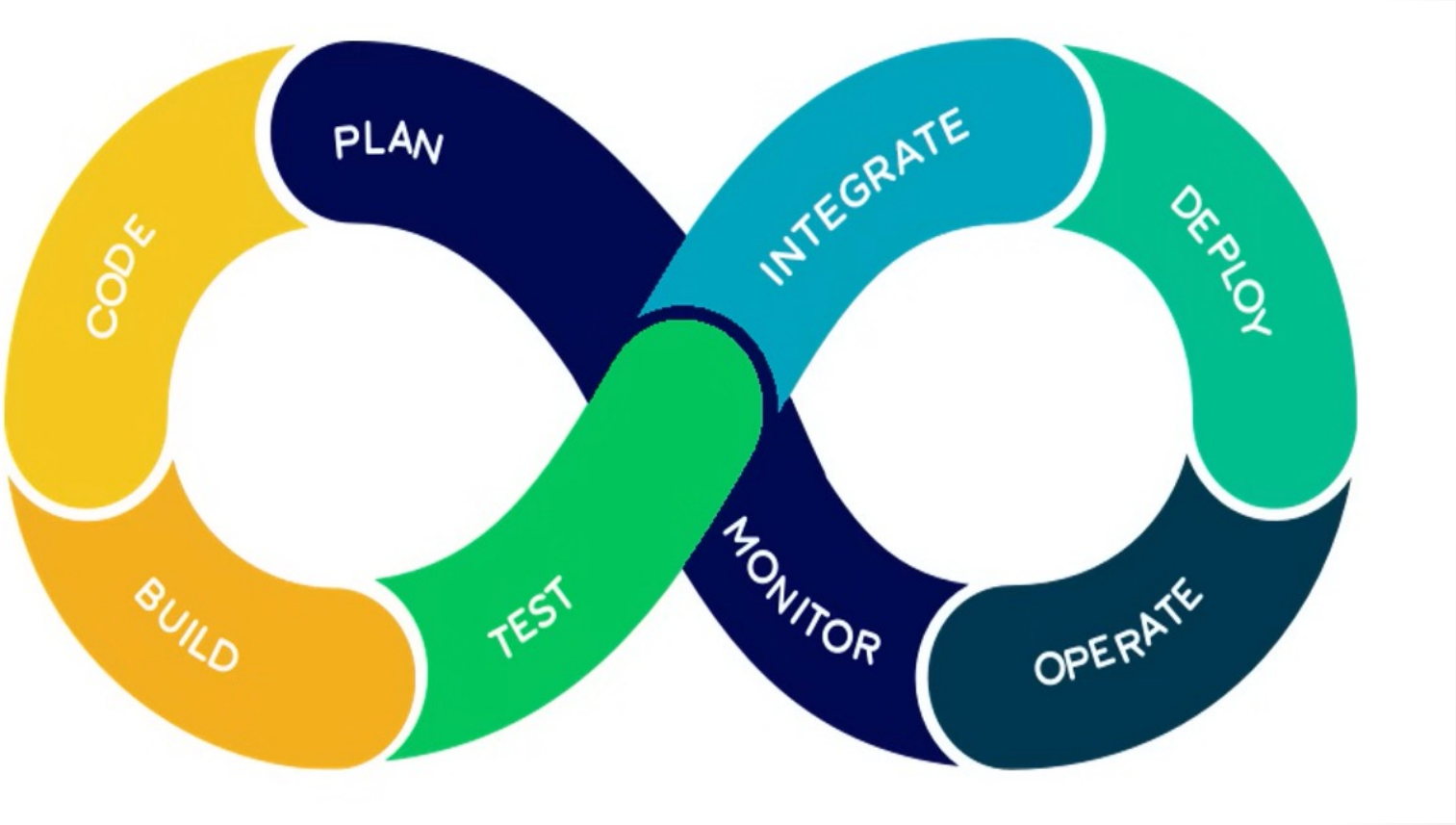
DevOps in Depth

- Moreover, at the core of any successful strategy, is what is known as the “DevOps Trinity”:
- **People and Culture –** This means breaking down the traditional silos between teams in the organization and working together towards a common goal.
- The goal is to get quality software to the customer as quickly as possible.
- **Processes and Practices –** Agile and DevOps go hand in hand. By adopting Agile, Scrum or Kanban, plus automation, organizations can streamline processes in predictable and repeatable ways.
- **Tools and Technologies** – Without the right tools and technologies in place, DevOps is not a sustainable model. These enable automation, continuous integration, configuration management, testing, packaging, releasing, and monitoring.

Benefits of DevOps

- Continuous delivery of software
- Better collaboration between teams
- Easy deployment
- Better efficiency and scalability
- Errors are fixed at the initial stage
- More security
- Less manual intervention (which means fewer chances of error)

DevOps Phases



Plan

- The planning phase is exactly what it sounds like: planning the project's lifecycle.
- In contrast to conventional methods to the development lifecycle, this model assumes that each stage will be repeated as necessary.
- In this manner, the DevOps workflow is planned with the likelihood of future iterations and likely prior versions in mind.
- This implies that we will likely have information from past iterations that will better inform the next iteration, and that the present iteration will likewise inform the next iteration.

Code

- The developers will write the code and prepare it for the next phase during the coding stage.
- Developers will write code in accordance with the specifications outlined in the planning phase and will ensure that the code is created with the project's operations in mind

Build

- Code will be introduced to the project during the construction phase, and if necessary, the project will be rebuilt to accommodate the new code.
- This can be accomplished in a variety of ways, although GitHub or a comparable version control site is frequently used.

Test

- Throughout the testing phase, teams will do any necessary testing to ensure the project performs as planned.
- Teams will also test for edge and corner case issues at this stage.
- An “edge case” is a bug or issue that only manifests during an extreme operating event, whereas a “corner case” occurs when many circumstances are met.

Deploy

- In the deploy phase, the project is prepared for the production environment and is operating as planned in that environment.
- This would be the responsibility of the operations team; in DevOps, it is a shared responsibility.
- This shared duty pushes team members to collaborate to guarantee a successful deployment.

Operate

- In the operating phase, teams test the project in a production environment, and end users utilize the product.
- This crucial stage is by no means the final step.
- Rather, it informs future development cycles and manages the configuration of the production environment and the implementation of any runtime requirements.

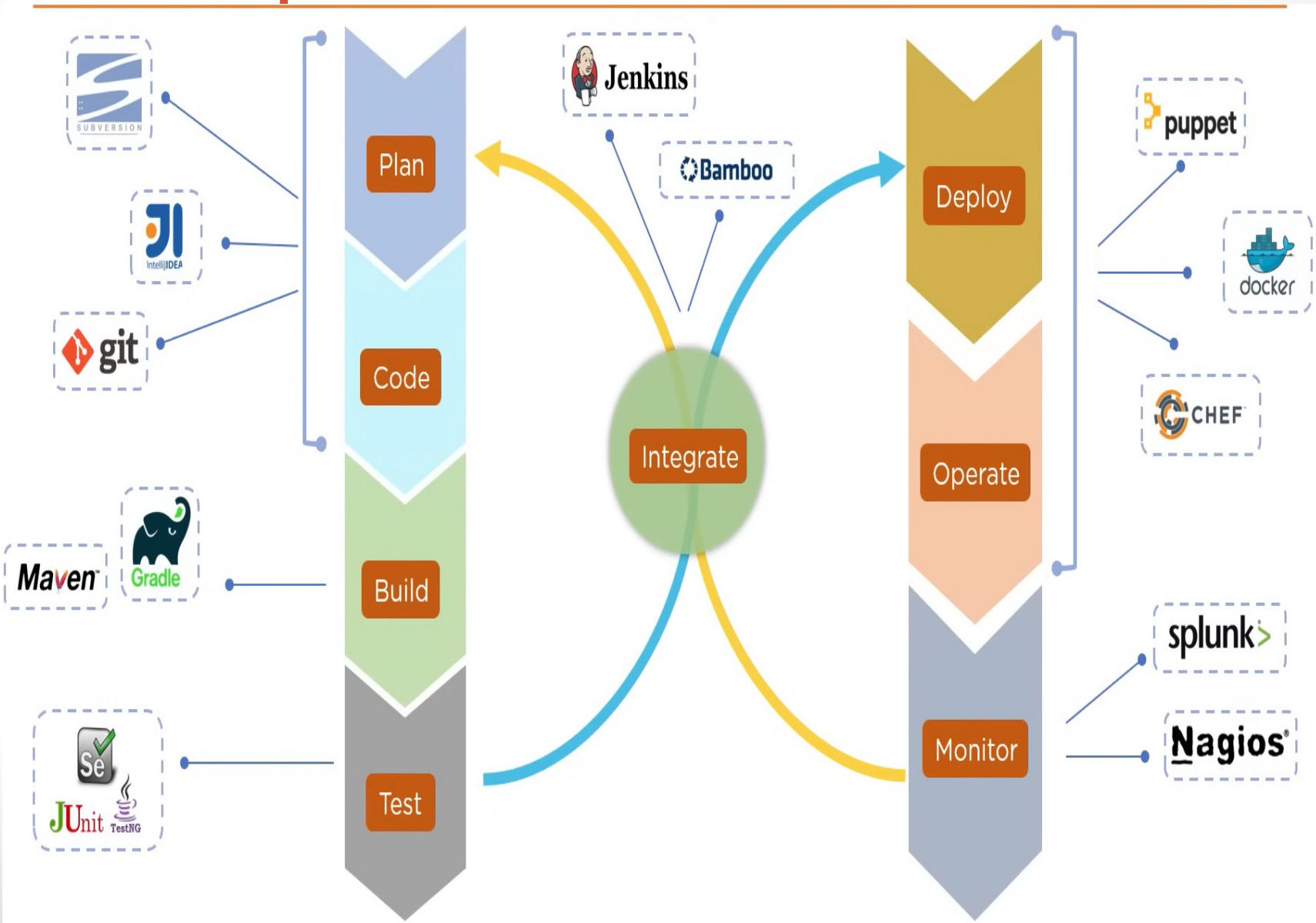
Monitor

- During the monitoring phase, product usage, as well as any feedback, issues, or possibilities for improvement, are recognized and documented.
- This information is then conveyed to the subsequent iteration to aid in the development process.
- This phase is essential for planning the next iteration and streamlines the pipeline’s development process.

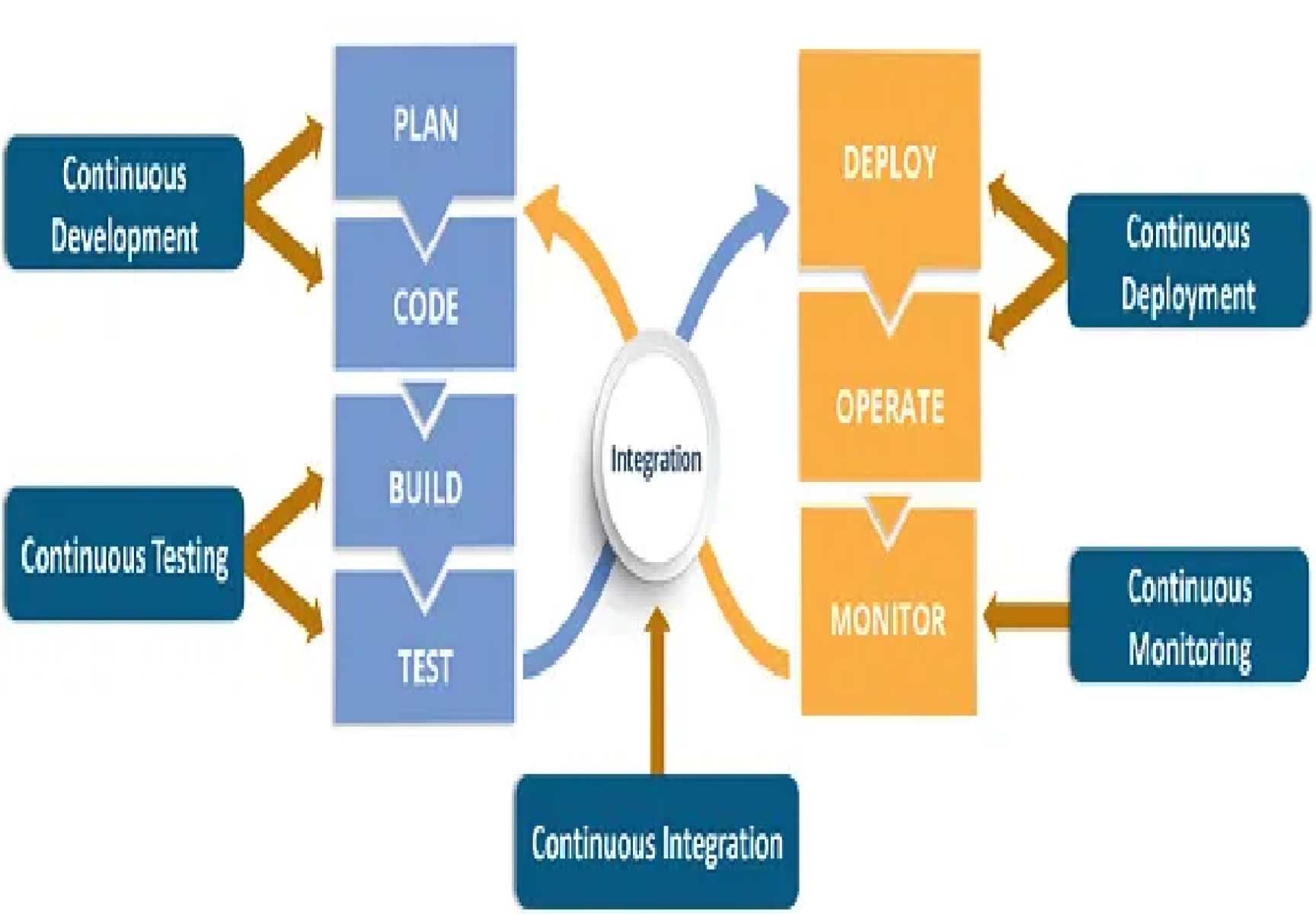
Integrate

- The integrate phase occurs when the code has been verified as ready for deployment and a last check for production readiness has been performed.
- The project will subsequently enter the deployment phase if it satisfies all requirements and has been thoroughly inspected for bugs and other problems.

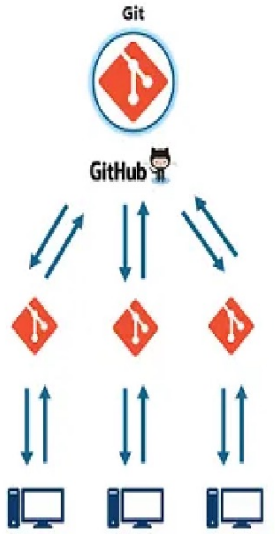
DevOps Tools

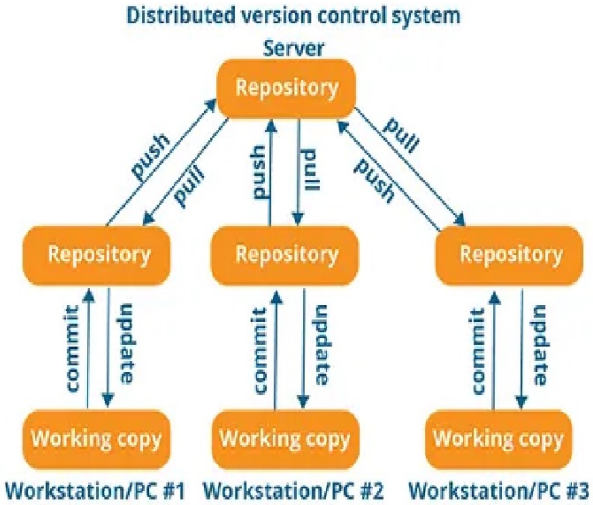


7 Cs of DevOps



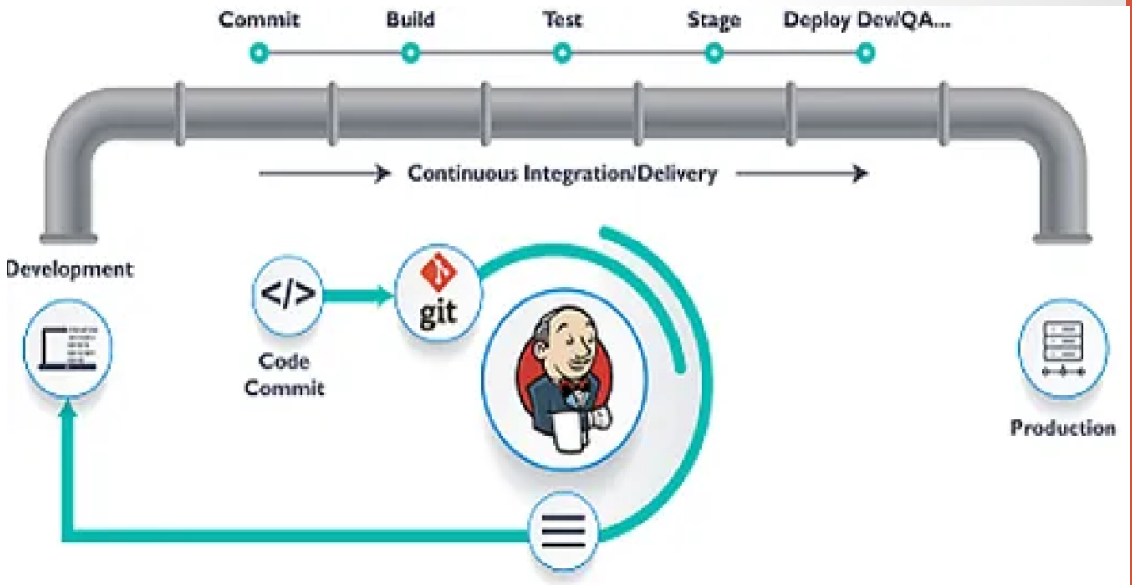
Continuous Development





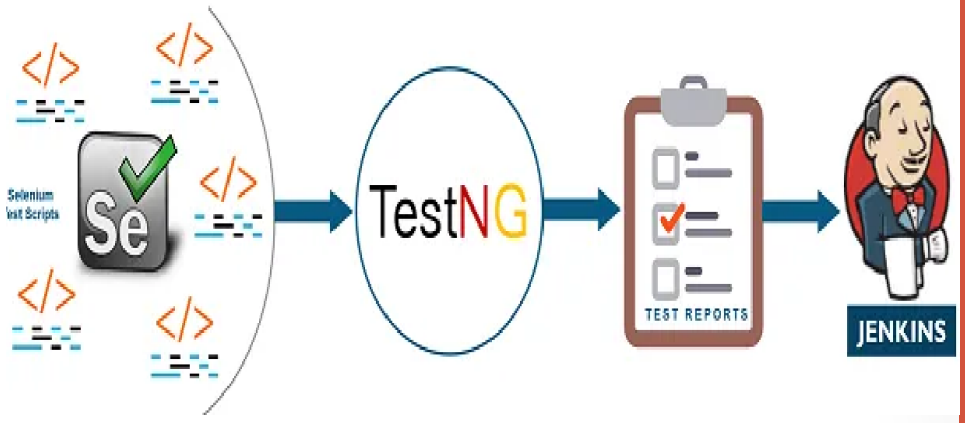
- This step is crucial in defining the vision for the entire software development process.
- It focuses mostly on project planning and coding.
- At this phase, stakeholders and project needs are gathered and discussed.
- In addition, the product backlog is maintained based on customer feedback and is divided down into smaller releases and milestones to facilitate continuous software development.

Continuous Integration



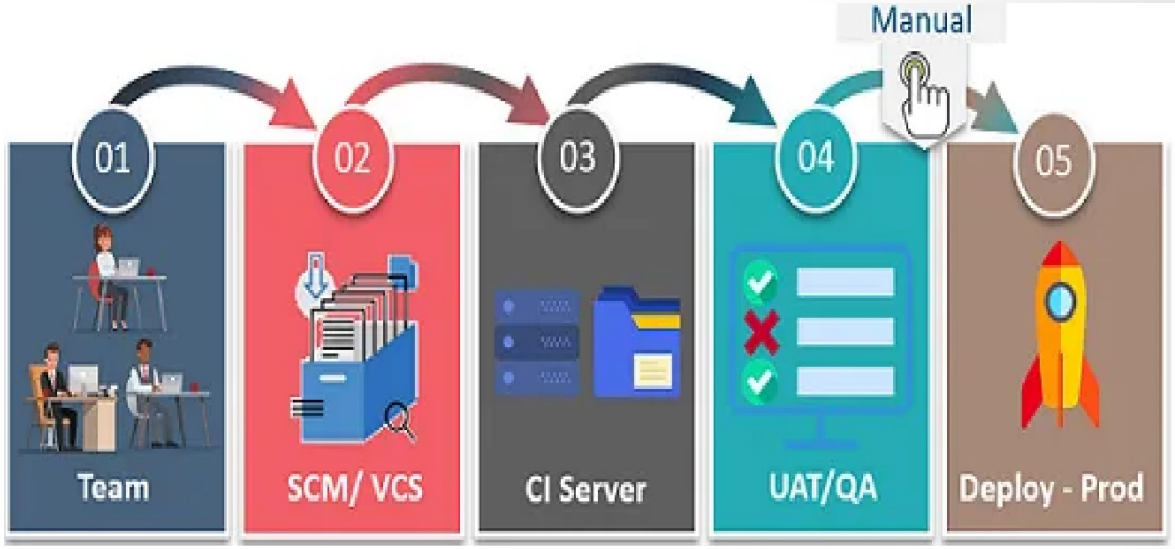
- Continuous integration is the most important stage of the DevOps lifecycle.
- At this phase, updated code or new functionality and features are developed and incorporated into the existing code.
- In addition, defects are spotted and recognized in the code at each level of unit testing during this phase, and the source code is updated accordingly.
- This stage transforms integration into a continuous process in which code is tested before each commit.
- In addition, the necessary tests are planned during this period.

Continuous Testing



- Some teams conduct the continuous testing phase prior to integration, whereas others conduct it after integration.
- Using Docker containers, quality analysts regularly test the software for defects and issues during this phase.
- In the event of a bug or error, the code is returned to the integration phase for correction.
- Moreover, automation testing minimizes the time and effort required to get reliable findings.
- During this stage, teams use technologies like as Selenium.
- In addition, continuous testing improves the test assessment report and reduces the cost of delivering and maintaining test environments.

Continuous Deployment



- This is the most important and active step of the DevOps lifecycle, during which the finished code is released to production servers.
- Continuous deployment involves configuration management to ensure the proper and smooth deployment of code on servers.
- Throughout the production phase, development teams deliver code to servers and schedule upgrades for servers, maintaining consistent configurations

Continuous Feedback

- Constant feedback was implemented to assess and enhance the application's source code.
- During this phase, client behavior is routinely examined for each release in an effort to enhance future releases and deployments.
- Companies can collect feedback using either a structured or unstructured strategy.
- Under the structural method, input is gathered using questionnaires and surveys.
- In contrast, feedback is received in an unstructured manner via social media platforms.

Continuous Monitoring

- During this phase, the functioning and features of the application are regularly monitored to detect system faults such as low memory or a non-reachable server.
- This procedure enables the IT staff to swiftly detect app performance issues and their underlying causes.
- Whenever IT teams discover a serious issue, the application goes through the complete DevOps cycle again to determine a solution.