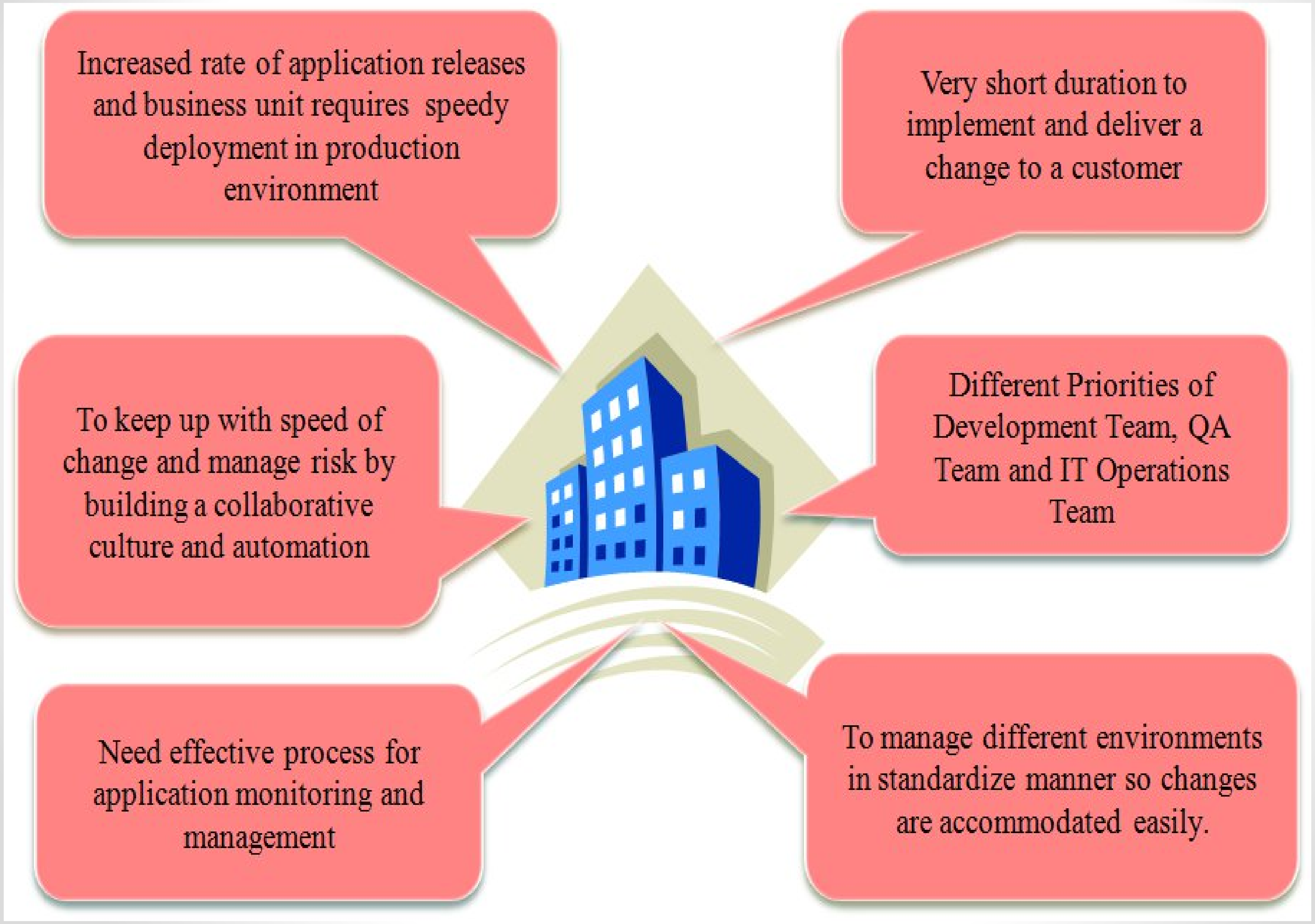
DevOps

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



Increased rate of application releases and business unit requires speedy deployment in production environment

Very short duration to implement and deliver a change to a customer

To keep up with speed of change and manage risk by building a collaborative culture and automation

Different Priorities of Development Team, QA Team and IT Operations Team

Need effective process for application monitoring and management

To manage different environments in standardize manner so changes are accommodated easily.

Need For DevOps

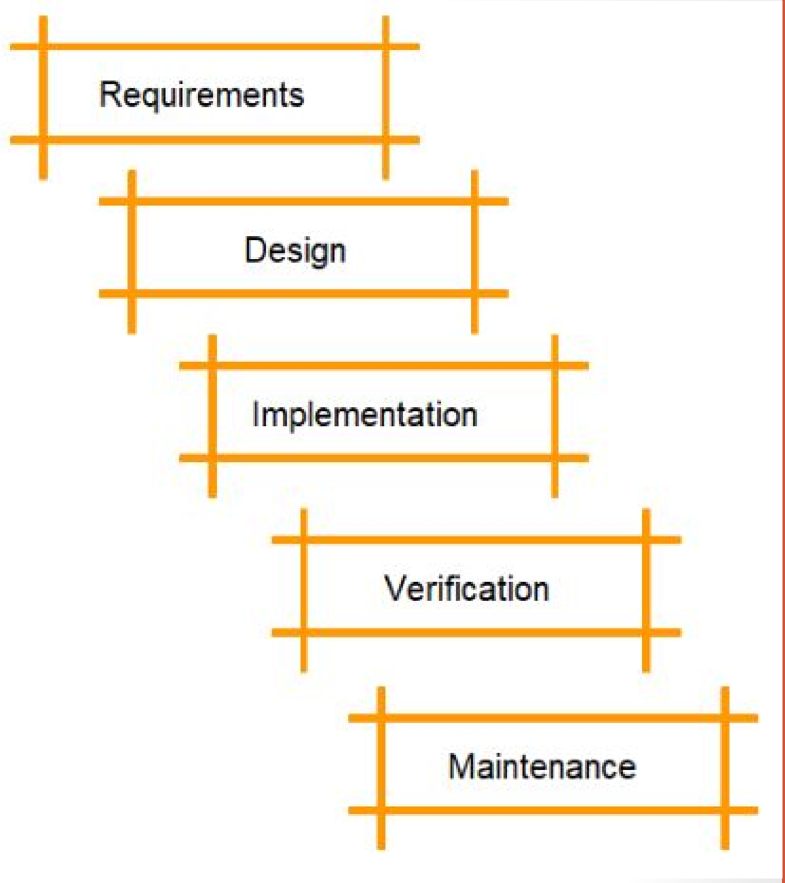
- Considering the changing patterns and competitive environment in business,
- it is the need of the hour to improve application life cycle management.
- Are there any factors that can be helpful in these modern times which can help us to improve application life cycle management?
- Yes. Cloud computing 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.

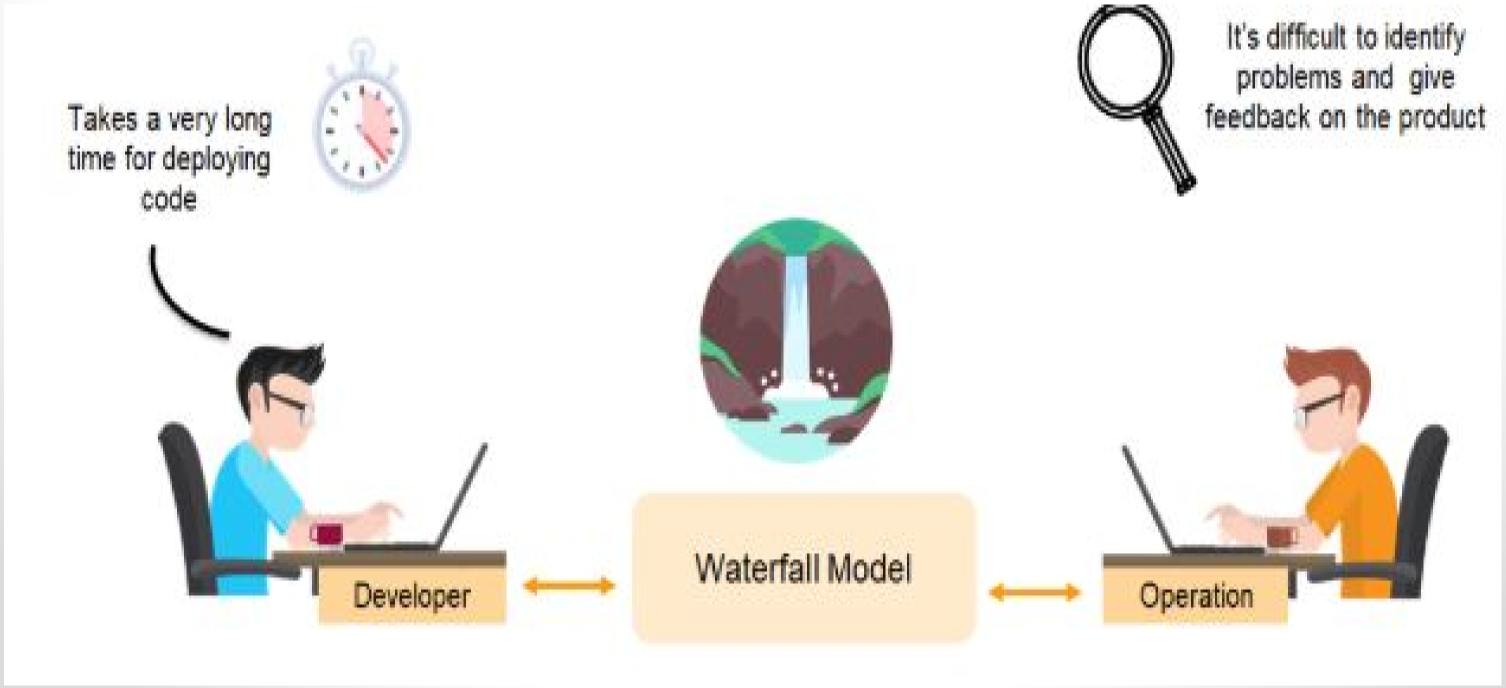
Waterfall Model

- 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.

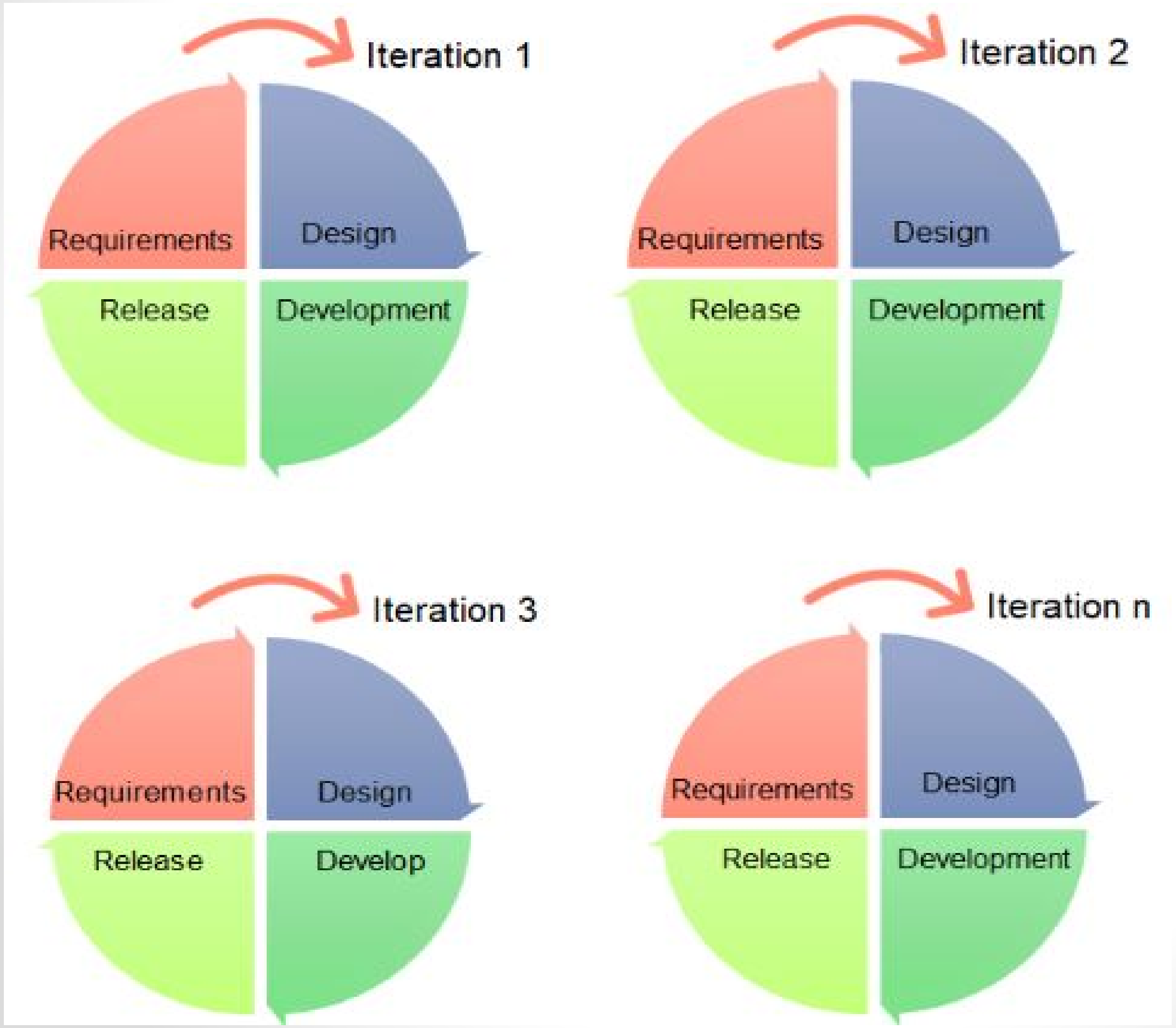

Takes a very long time for deploying code
It's difficult to identify problems and give feedback on the product
Developer
Waterfall Model
Operation

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

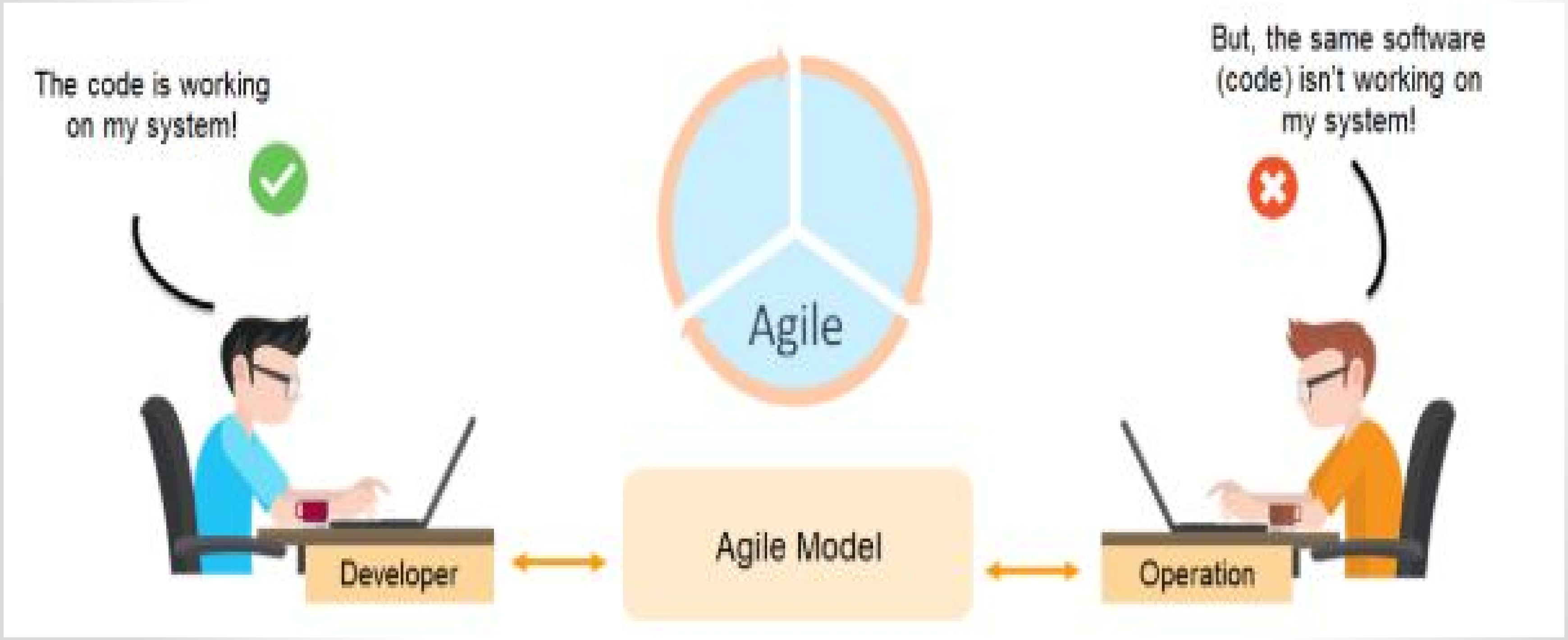
Agile Model

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.


The code is working on my system!
Agile
But, the same software (code) isn't working on my system!
Developer
Agile Model
Operation

DevOps

- DevOps is a software development methodology that accelerates the delivery of high-performance applications and services by combining and automating the work of software development (Dev) and IT operations (Ops) teams.

-IBM

DevOps Over Agile Model and Waterfall model

- DevOps is an evolution of the agile software development methodology, which emerged as an alternative to the waterfall methodology.
- In the waterfall approach, software development teams spent months developing large bodies of code, which then underwent months of testing before release. In contrast, agile development takes an iterative approach to the software delivery lifecycle.

- DevOps adds new processes and tools to the agile methodology, notably the automation of much of the CI/CD pipeline.

https://www.ibm.com/think/topics/devops

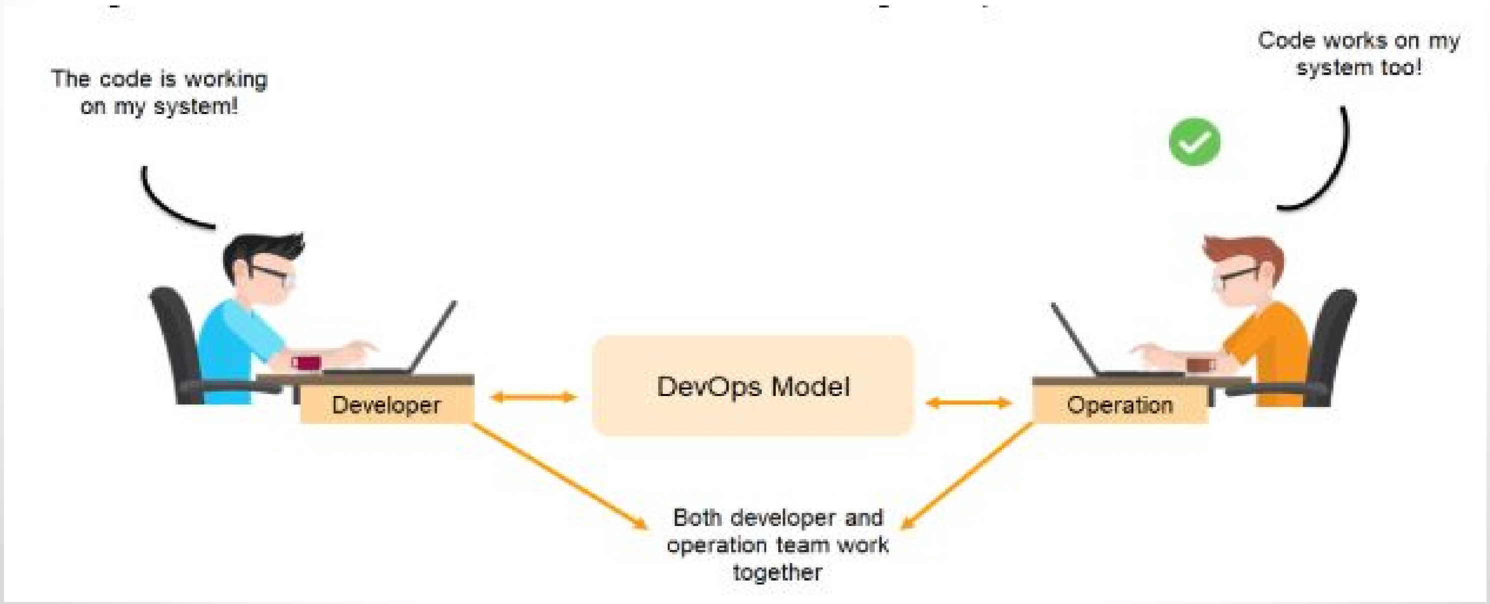
DevOps-CI/CD Pipeline

- The hallmarks of DevOps are continuous integration and continuous delivery (CI/CD), which support smaller, faster software updates.
- **With CI/CD, small chunks of new code are merged into the code base at frequent intervals, and then automatically integrated, tested and prepared for deployment to the production environment.**

https://www.ibm.com/think/topics/devops

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.


The code is working on my system!
Code works on my system too!
Developer
DevOps Model
Operation
Both developer and operation team work together

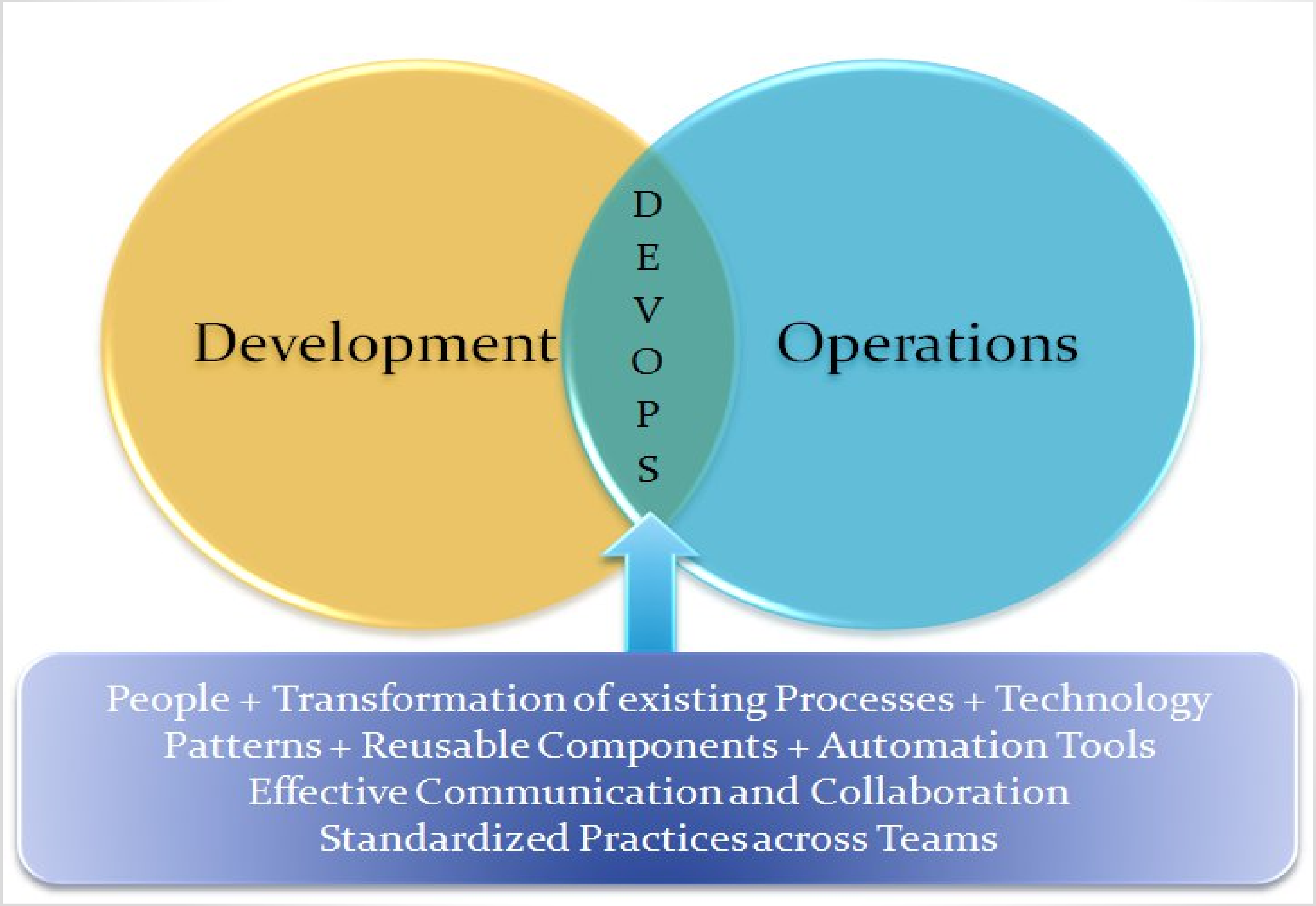
Overview of DevOps

- 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

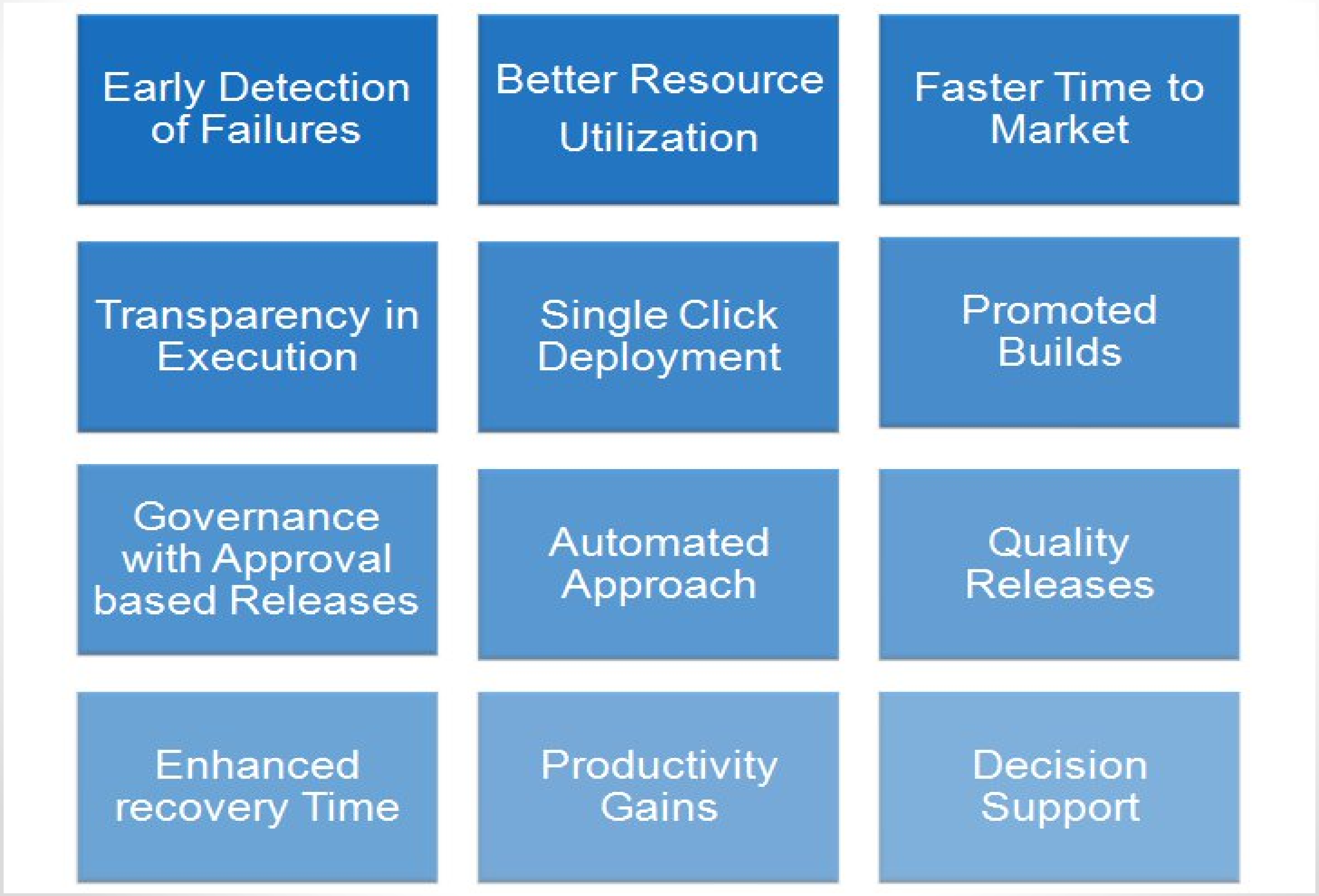
- a way of designing,
- a way of developing,
- a way of testing,
- a way of setting up resources,
- a way of managing environments,
- a way of configuration management,
- a way of deploying an application,
- a way of gathering feedback,
- a way of code improvements,
- a way of doing innovations.

Overview of DevOps


Development
D E V O P S
Operations
People + Transformation of existing Processes + Technology
Patterns + Reusable Components + Automation Tools
Effective Communication and Collaboration
Standardized Practices across Teams

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.

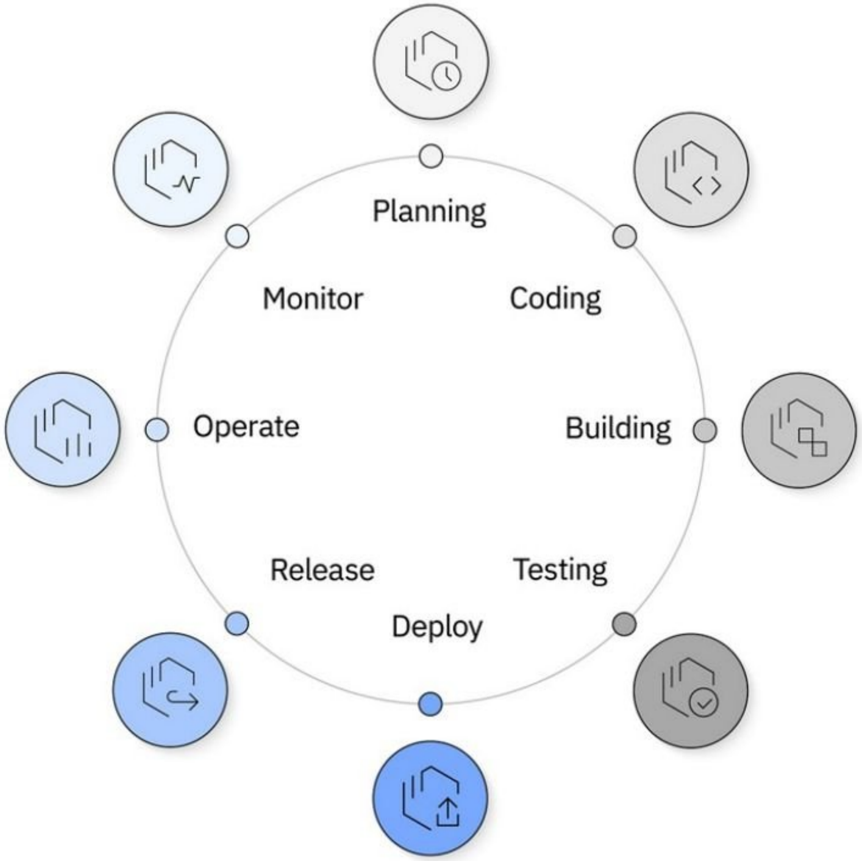
DevOps in Depth

- 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.
- **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 coding, building, 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/The DevOps lifecycle



Plan

- First, teams scope out **new features and functions for the next release**. During this workflow, they draw on user feedback, **case studies and inputs from internal stakeholders such as platform and infrastructure engineers, security, compliance, governance, risk management and line-of-business teams.**

- The goal of the planning stage is to create **a backlog document. The backlog is a prioritized list of new features, improvements and bug fixes that will be added to the product over time.**

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 DevOps team codes the new and enhanced features identified in the backlog. Common coding practices in DevOps include:
- **Test-driven development (TDD):** Developers first create tests that the code must pass before they write the code itself.

- **Pair programming:** Two programmers collaborate at the same workstation, with one programmer writing the code and the other evaluating the code.

- **Peer code reviews**: Members of the development team review each other's code for bugs, errors or areas for improvement.
- Developers often use their local workstations to write and test code before sending it down to the next stage of the continuous delivery pipeline.

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

- **New code is integrated into the existing code base, then tested and packaged for release and deployment**. Activities that are often automated at this stage include **merging code changes into a master copy; placing the updated code into a repository; and compiling, testing and packaging code into an executable file.**

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.**

Test

- **DevOps teams use testing, typically automated testing, to make sure that the updated application meets appropriate standards and requirements.**
- **Continuous testing helps implement the principle of shift-left testing,** a software development approach that emphasizes moving testing activities earlier in the development process. This approach helps organizations identify problems sooner and remediate them more effectively.

Release

- **The release stage is the last workflow before users access the application.** This stage includes a series of final tests to ensure that the software meets quality, compliance and security standards and is ready for external use.
- **If errors or defects are found, the team has a chance to intercept and remediate any problems before users see them. When all issues are fixed and the application meets all requirements, it can be released to the production environment**

Deploy

- **At this stage, the project moves to a production environment where users can access the updated application.**

- **Many organizations deploy first to a subset of end users to ensure that the application works properly.** When stability is established, the application can be deployed to everyone.

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

- **DevOps teams check that new features are running smoothly and are available to users with no interruptions in service.**
- They use **automated observability and management tools to continuously monitor and optimize operations** to make sure that network, storage, platform, compute and security postures are all working properly.

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.**

Monitor

- **Teams collect and analyze feedback from users and lessons from previous workflows** to help improve processes and products going forward.

DevOps Tools

- DevOps implementations require specialized toolchains that enable **asynchronous collaboration, seamless integration of DevOps workflows and as much automation as possible** throughout the DevOps lifecycle.

- Categories of DevOps tools include:

1) Version control systems
2) CI/CD pipelines
3) Containerization platforms and tools

DevOps Tools

CI/CD pipelines

- The CI/CD pipeline helps automate core software development tasks such as **integrating code, testing code quality, compiling and packaging code and deploying software.**

- **Popular tools in this category include Jenkins, CircleCI and TeamCity.**

DevOps Tools

Containerization platforms and tools

- Containerization encapsulates apps in streamlined, portable packages called “containers” that can run on any platform.
- This capability makes containerization useful for the rapid release and management cycles of DevOps.
- **Organizations often use open-source tools such as Docker and Kubernetes** to help build, orchestrate and automate the deployment of containerized apps.

DevOps Tools

Version control systems

- Version-controlled coding environments enable **multiple developers to manage code changes, track changes and work collaboratively on the same code base.**
- These code repositories typically integrate with CI/CD, testing and security tools through application programming interfaces (APIs), so when code is committed to the repository it can automatically move to the next step.
- **Popular version control systems include Git (often used on GitHub), Apache Subversion and Mercurial.**

DevOps Tools

Configuration management tools

- Configuration management tools help DevOps teams **configure infrastructure, software and applications in various IT environments.**
- **These tools automate configuration tasks such as the setup and deployment of hardware or applying software patches** to ensure consistency, reduce errors and improve reliability. **Popular configuration management tools include Puppet, Chef and SaltStack.**

DevOps Tools

Infrastructure as code (IaC) tools

- IaC uses a **high-level descriptive coding language to automate the provisioning of IT infrastructure. Instead of instructing a system how to provision infrastructure, developers describe the desired end state, and the IaC software handles the rest.**

DevOps Tools

Monitoring and observability tools

- Monitoring and observability tools help DevOps teams **identify and resolve system issues, such as slow response times or excessive resource consumption.** They also gather and analyze data in real time to reveal how code changes impact application performance.

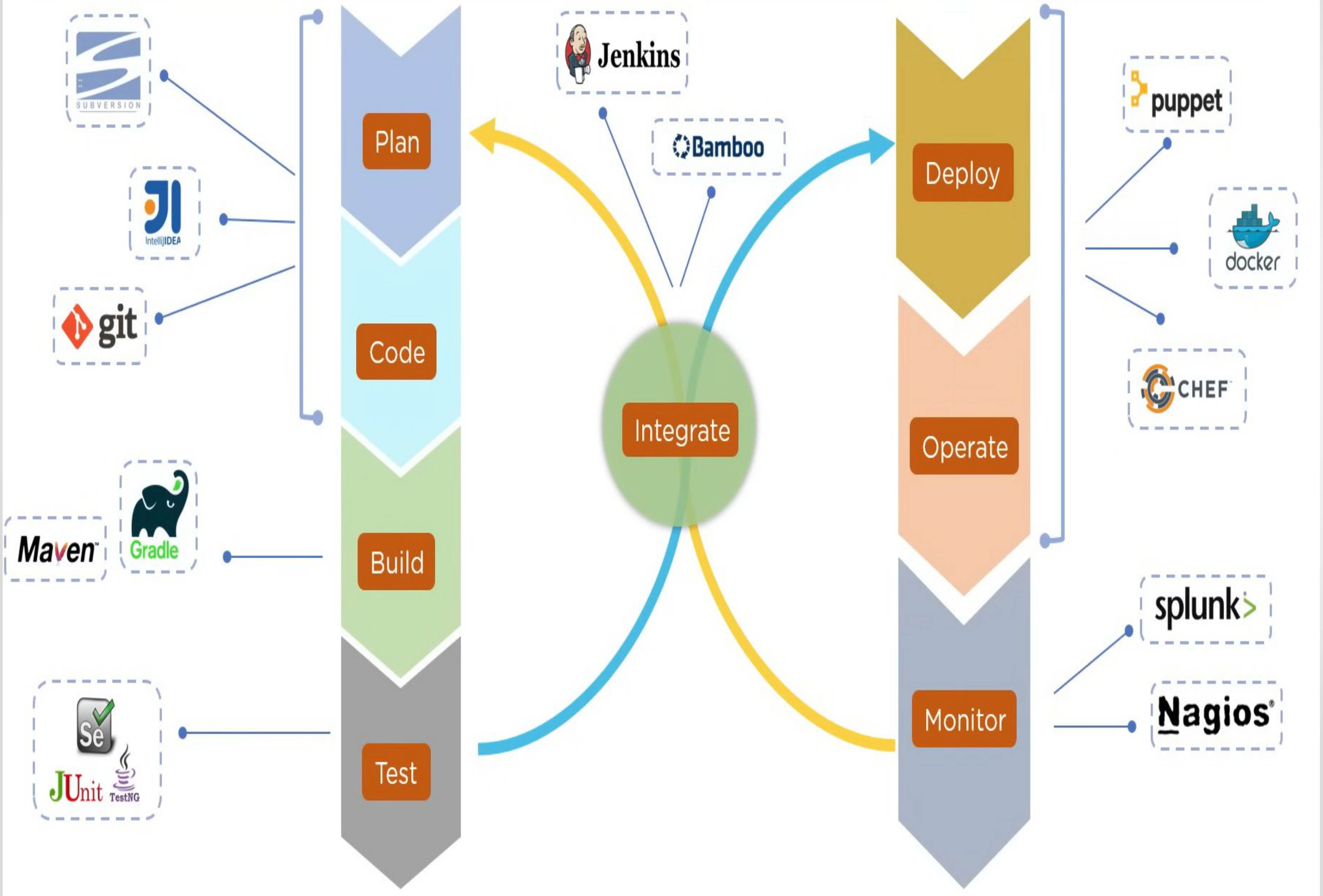
- **Popular observability and monitoring tools include Prometheus, Datadog, IBM Instana®, New Relic and Splunk.**

DevOps Tools

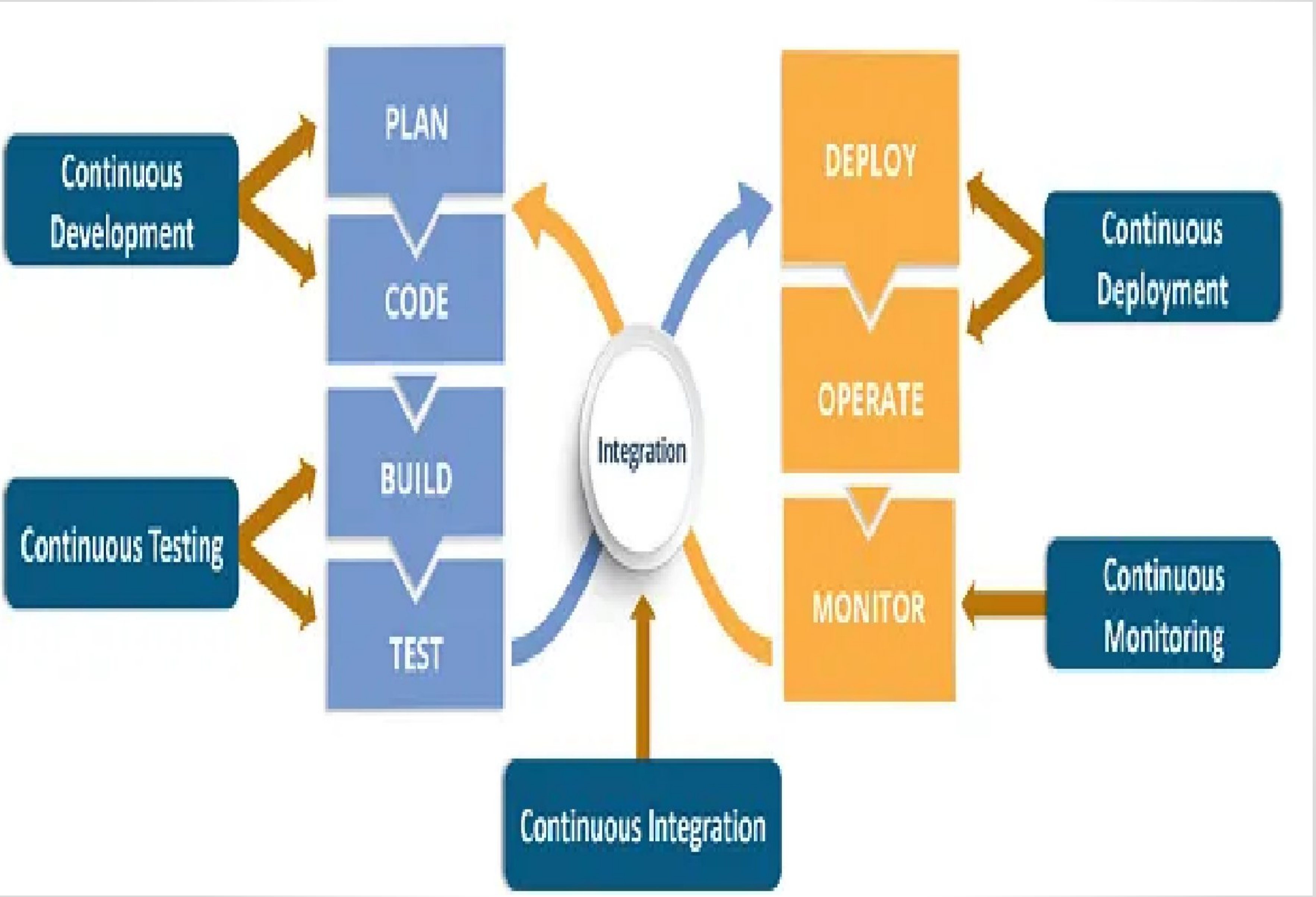
Continuous feedback tools

- **These tools gather feedback from users, either through heat mapping (recording users' actions on the screen), surveys, polls or self-service issue ticketing.**
- Some tools also monitor social media to **collect user feedback and measure satisfaction with application updates.**

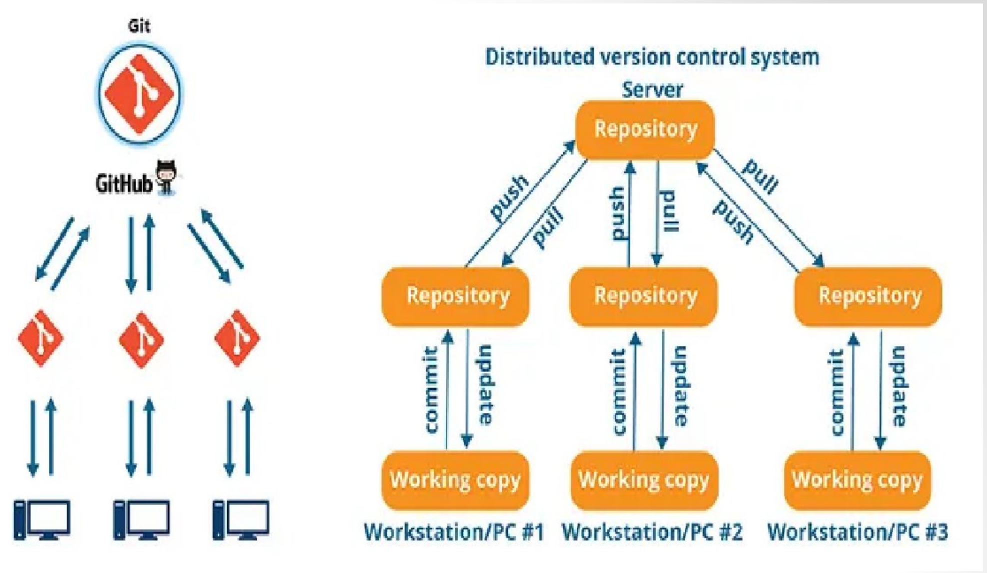
DevOps Tools



Cs of DevOps

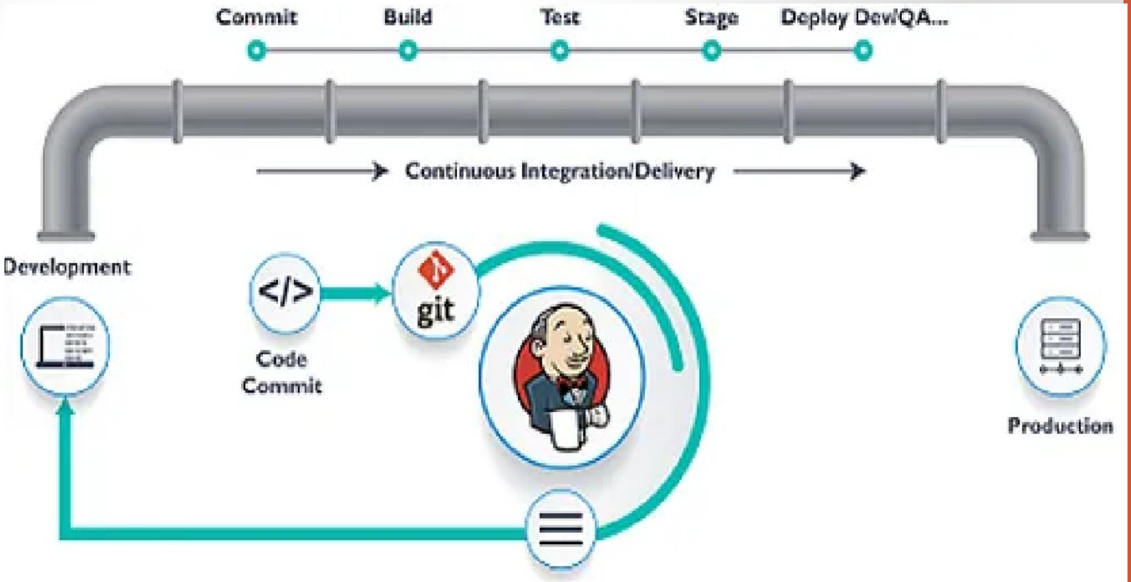

Continuous Development
PLAN
CODE
BUILD
TEST
Continuous Testing
Integration
DEPLOY
OPERATE
MONITOR
Continuous Deployment
Continuous Monitoring
Continuous Integration

Continuous Development



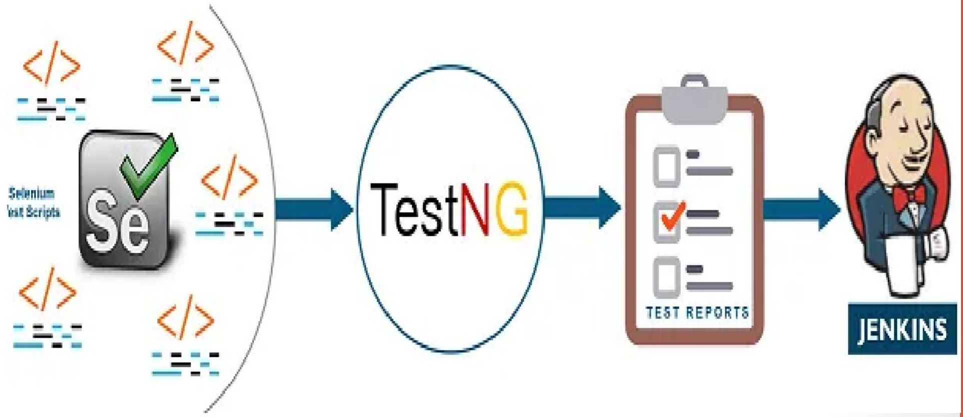
- This step is crucial in defining the vision for the entire software development process.
- It focuses mostly on project planning and coding.
- At this phase, stakeholders and project needs are gathered and discussed.
- In addition, the product backlog is maintained based on customer feedback and is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

Continuous Integration



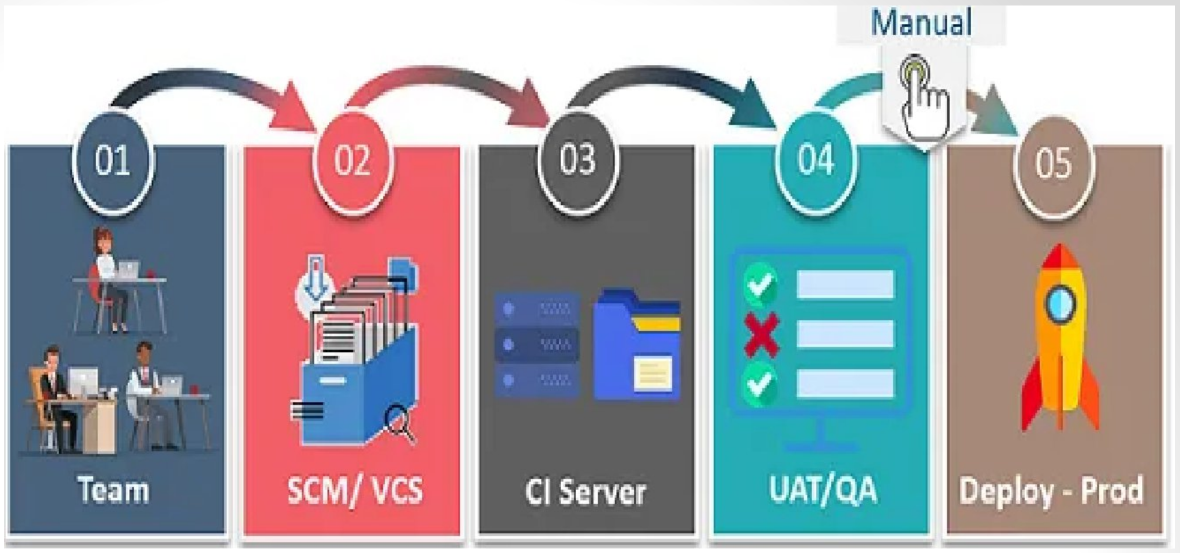
- Continuous integration is the most important stage of the DevOps lifecycle.
- At this phase, updated code or new functionality and features are developed and incorporated into the existing code.
- In addition, defects are spotted and recognized in the code at each level of unit testing during this 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.