3. If additional changes need to be made or not ?
4. Whether the software engineering standards are properly followed ?
5. Do the attributes of configuration objects reflect the change ?
6. Whether all the SCI are updated properly ?
7. Whether the SCM process(object identification, change and version control, configuration audit and status reporting) are properly followed ?

- The above questions can be asked as a part of formal technical review.

## 8.7.5 Status Reporting

The status reporting focuses on communication of changes to all people in an organization that involve with changes.

During status reporting following type of questions were asked.

1. **What happened ? :** What are the changes that are required ?
2. **Who did it ? :** Who will be handling these changes ?
3. **When did it happen ? :** The time at which these changes are arised.
4. **What else will be affected ? :** The objects or part of the software that might be reflected due to these changes.

**Review Questions**

1. *Define software configuration management. Explain change control management and version control management.*
2. *Explain software configuration management and change control management in detail.*
3. *Explain "How to manage the different versions that get created and how to maintain the quality of code under changing conditions ?".* **GTU : Summer-2013, Marks 7**
4. *Explain version and change control management.* **GTU : Summer-2018, 2019, Winter-2019, Marks 4**
5. *Explain SCM process in details.* **GTU : Summer-2019, Marks 7**

❑❑❑

# 9 DevOps

***Syllabus***

*Product lifetime : Independent product Vs. continues, Improvement, Software as a service, SaaS architecture.*

***Contents***

## 9.1 Overview

- DevOps encourages the development, IT operations, quality engineering and security activities to be performed in coordination and collaboration to produce better, more reliable products. By adopting a DevOps culture along with DevOps practices and tools, teams gain the ability to better respond to customer needs, increase confidence in the applications they build and achieve business goals faster.
- DevOps is a culture which promotes collaboration between Development and Operations Team to deploy code to production faster in an automated and repeatable way.
- Teams that adopt DevOps culture, practices and tools become high-performing, building better products faster for greater customer satisfaction.

### 9.1.1 Difference between DevOps and Agile

| Sr. No. | Agile | Devops |
|---|---|---|
| 1. | The idea in agile is to develop software in small iterations and be thus able to adapt to the changing customer needs. | Devops is to deliver technology to business units in a timely fashion and ensure the technology runs without interruption or disruption. |
| 2. | It adopts rapid development approach | This is not a rapid development approach. |
| 3. | The focus of agile development is merely on software development and release. | The focus of Devops is not only on software development, its release but on it safest deployment in working environment. |
| 4. | In agile development, every team member has a skill of design, development and coding. Any available team member should be able to do what's required for progress. | Devops, on the other hand, assumes there will be development teams and operational teams, the two will be separate. These teams can communicate between themselves on frequent and regular basis. |
| 5. | Communication in agile development is informal and in the form of daily meetings. | In Devops communication, specifications, documents are involved and it is formal. It does not occur on daily basis. |
| 6. | The team is small in nature. | Large team size and multiple teams are required in Devops. |
| 7. | Agile is about software development. | Devops is about software development and management. |
| 8. | Documentation is not much important in agile development. | Documentation is very much important in devops. |
| 9. | Agile development teams, may choose to use certain automation tools. But there are no specific tools required for an agile team. | Devops absolutely depends on automated development to make everything happen smoothly and reliably. Certain tools are an integral part of devops. |
| 10. | Agile is less fexible. | Devops is more flexible. |
| 11. | Agile has limited scope. | Devops has broader scope. |

**Review Question**

1. *Differentiate between DevOps and agile Development.*

## 9.2 Problem Case Definition

In this section we will consider some real world problem for application development. Suppose we want to design online share trading application. This application can be developed using following stages -

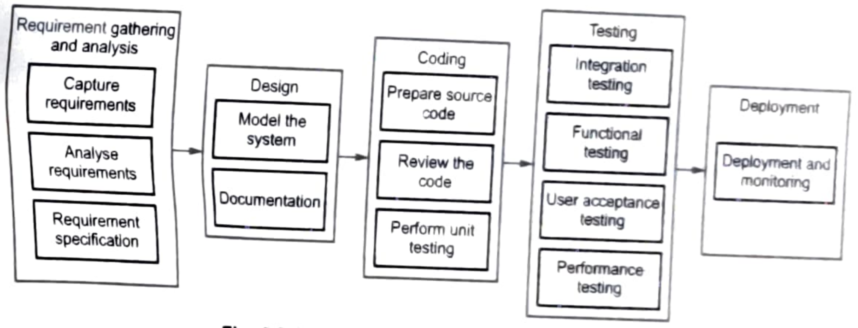

**Fig. 9.2.1 Application development Life Cycle**

Consider that the agile process is adopted as a delivery process by the development tram.

1. **Requirement analysis** : During this phase, the requirements of the system are identified. Both the functional and non functional requirements are gathered, analyzed and presented to the development team in specific format.
2. **Design** : Design is an activity to outline the processes to be conducted. The using diagrammatic representation, the activities are modeled.
3. **Coding** : Once the design of the system is done, the coding process starts. With the help of suitable programming language, the codes are written, and reviewed by the development team. Some testing is also performed with the coding. This type of testing is usually unit testing.

4. **Testing :** Before deploying the product in the working system, four types of testing are carried out -

1) Integration testing  2) Functional testing
3) User acceptance testing  4) Performance testing

Multiple teams perform testing. The operational team ensures that product is working accurately in working environment.

5. **Deployment :** This is a stage in which the product is packaged, deployed and monitored for correct functioning.

## 9.2.1 Challenges in Application Development

The challenges during application development life cycle are -

1) For execution of certain activities, during application development there exists total dependency on individuals in the team.
2) Quality is an important concern during development stages.
3) For testing the application, the provision of working environment must be made. It may cause delay in testing process.
4) Tracing the requirements during the development stage is a tedious job.
5) Testing can be costly, if repeated testing is required.

## 9.3 Benefits of Fixing Application Development Challenges

Following are the key benefits of fixing application development challenges -

1) **Time :** User requirements are changing many times. The ability of IT system to align with the business creates huge impact.

2) **Cost :** Cost of the software is an important factor. The software cost can be reduced by reducing waste in the development process.

3) **Quality :** By fixing the application development challenge the product quality can be improved. The delivery team should be able to quantify the product quality in early stage of development.

**Review Question**

1. *What are the challenges in application development ?*

## 9.4 DevOps Adoption Approach through Assessment

**Definition of Assessment :** The process of reviewing the application and related processes to identify the status of DevOps challenges is called assessment or maturity assessment.

There are five steps to be followed to identify and fix the application challenges. These steps are -

**Step 1 : Assess to Identify Gaps :** This step identifies the gap in process, technology, tools and automation in software development process.

**Step 2 : Define Solution :** In this step, the solution are created for the issues that are identified in the step 1. The solutions are prepared as per the DevOps strategy.

**Step 3 : Implement Solution :** The solution is implemented by taking maximum benefit.

**Step 4 : Measure Benefit :** The benefit is measured understand the effectiveness of the adopted strategy.

**Step 5 : Feedback Benefits and Challenges :** The feedback is important to refine the solution the solution with better quality and in few iterations.

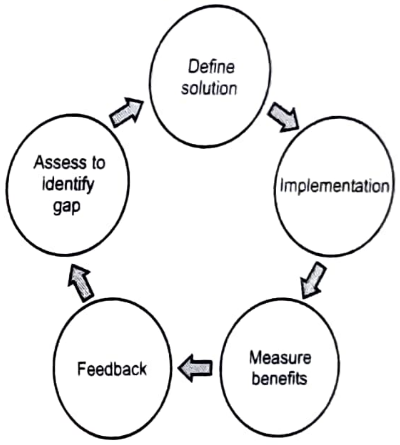

Fig. 9.4.1

## 9.5 Solution Dimensions

There are three key dimensions that can be used to solve issues in the application development in an integrated way.

1) **Process :** After requirements gathering the methodology of development is decided. This method can be waterfall model, spiral model, agile development process model or any other suitable process model. These process methods must be reviewed to fix the issues in application development.

2) **Tools :** The tools are used in application development process to automate the manual activities. The solution must focus on effectiveness of the tools and the manner in which the tool can be used. The solution must identify if additional toolset is required or not.

**3) Architecture and technology** : The solution must evaluate underlying architecture. It must be checed if parallel processing is possible or not. It must also be evaluated if the automated processes can be executed on this architecture or not.

**Review Question**

1. *What is solution Dimension in solving the issues of typical application development process.*

## 9.6 What is DevOps?

**Definition** : Devops is a practice in which development and operation engineers participate together in entire lifecycle activities of system development from design, implementation to product support.

- The term Devops is derived from "Software **DEV**elopment " and "information technology **OP**erationS".
- Devops promotes a set of processes and methods from the three department **Development**, IT **operations** and **Quality assurance** that communicate and collaborate together for development of software system.

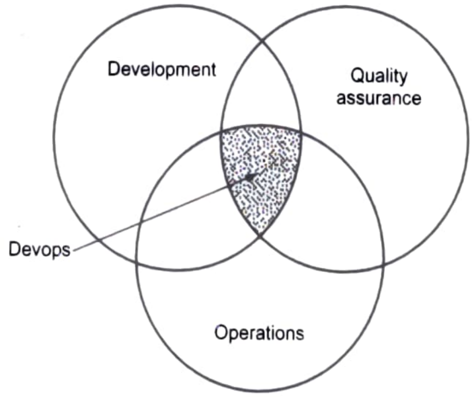

**Fig. 9.6.1**

**Review Question**

1. *What is DevOps ?*

## 9.7 DevOps Importance and Benefits

**Importance**

- Devops enhances the organization's performance, improves the productivity and efficiency of development and operations teams.

- Bringing the two teams together centralizes the responsibility on the entire team and not specific individuals working.
- Devops is more than just a tool or a process change. It inherently requires an organizational culture shift.
- This cultural change is especially difficult, because of the conflicting nature of departmental roles :

1. Operations - seeks organizational stability;
2. Developers - seek change;
3. Testers - seek risk reduction.

- Adoption of Devops is driven by various factors. These factors are -

1. Demand for an increased rate of production releases - from application and business unit stakeholders.
2. Increased usage of data center automation and configuration management tools.
3. Use of agile and other development processes and methods.
4. Increased focus on test automation and continuous integration methods.
5. Wide availability of virtualized and cloud infrastructure.

**Benefits**

Various benefits of Devops are -

- **Technical benefits**

1. Continuous software delivery is possible
2. There is less complexity to manage the project.
3. The problems in the project gets resolved faster.

- **Cultural benefits**

1. The productivity of teams get increased.
2. There is higher employee engagement.
3. There arise greater professional development opportunities.

- **Business benefits**

1. The faster delivery of the product is possible.
2. The operating environment becomes stable.
3. The communication and collaboration gets improved among the teams members and customer.
4. More time is available for innovation rather than fixing and maintaining.

**Review Question**

1. *Explain the importance and benefits of DevOps.*

## 9.8 DevOps Principles and Practices

DevOps principles and practices are enlisted as given below -

**Principles**

1) **Customer centric action** : Continuous feedback of end user or real customer should get reflected in all the development activities. The product and services must be developed in such a manner that the customer satisfaction is always protected.

2) **Focus on end result** : The developers and engineers must work by keeping the complete picture of the end product in mind.

3) **End-to-End responsibility** : All the services required to develop, maintain and monitor must be done by the same team. This helps in improving the quality of end product.

4) **Cross functional autonomous teams** : If same team works on all the phases such as development, maintenance and monitoring then it helps in improving the quality of end product. The members of team having multiple skill set on various technologies and use of variety of testing tools help to make the team self-sufficient.

5) **Continuous improvement** : The continuous improvement is normally done for optimizing the speed of processing, reduced cost and minimized waste. The objective of continuous improvement is to bring best quality product.

6) **Use of automated tools and techniques** : The use of automated tools help in improving the speed and reducing the development efforts. Hence automate the things wherever possible.

**Practices**

1) **Perspective consideration** : The difference between the developers and participants perspective must be considered.

2) **Flexibility** : The organization must have flexibility to switch between the delivery of the product using DevOps values and without DevOps values.

3) **Integrate changes** : It must be possible to accommodate the required changes in the system.

4) **Agility** : The process as well as tool selection needs to be agile according the project needs.

5) **Transparency** : There must be transparency about the goals, delivery plan and activities between developers and operational engineers.

## 9.9 The 7 C's of DevOps Lifecycle for Business Agility

The 7Cs of DevOps are as follows -

1) **Continuous business planning** : During this phase the planning for identifying skills, resources required and outcome is made.

2) **Continuous development** : In this phase, development sketch plan is prepared and programming techniques are identified. Before continuous integration, development teams would write a bunch of code for three to four months. Then those teams would merge their code in order to release it.

3) **Continuous integration** : Continuous integration is the practice of quickly integrating newly developed code with the main body of code that is to be released. Continuous integration saves a lot of time when the team is ready to release the code.

The continuous integration process from a DevOps perspective involves checking your code in, compiling it into usable (often binary executable) code and running some basic validation testing.

4) **Continuous deployment** : It is the practice of deploying all the way into production without any human intervention. Teams that utilize continuous delivery don't deploy untested code; instead, newly created code runs through automated testing before it gets pushed out to production. The code release typically only goes to a small percentage of users and there's an automated feedback loop that monitors quality and usage before the code is propagated further.

5) **Continuous testing** : Continuous testing (CT) is the process of executing automated test cases as part of the software delivery pipeline. Its goal is to obtain immediate and continuous feedback on the business risks associated with a software release candidate.

6) **Continuous delivery and monitoring** : With continuous delivery, every code change is built, tested, and then pushed to a non-production testing or staging environment. There can be multiple, parallel test stages before a production deployment.

Continuous monitoring is a process of monitoring the application continuously to check its service, performance and security. The continuous monitoring can be done using different tools. This monitoring process involves generation of reports.

7) **Continuous feedback** : This is a process which allows for an immediate response from your customers for your product and its features and helps you modify accordingly.

**Review Question**

1. *Explain 7 C's of DevOps lifecycle for business agility.*

## 9.10 DevOps and Continuous Testing

- Continuous testing is an important aspect in DevOps.
- **Definition of continuous testing** : Continuous testing is defined as software testing that involves the process of testing early, testing frequently and automate the test.
- Test automation is a mandatory step in continuous testing.
- Continuous testing is a vital aspect that bridges continuous integration and continuous delivery.

### 9.10.1 Continuous Testing Process in DevOps

- In DevOps processes the a software change is continuously moving from development to testing to deployment. That means the code is continuously developed, delivered, tested and then deployed.
- During continuous testing, there is an execution of various types of tests, continuously on the code base and on different environments that it gets deployed on to, as predefined and designed in the continuous delivery pipeline.

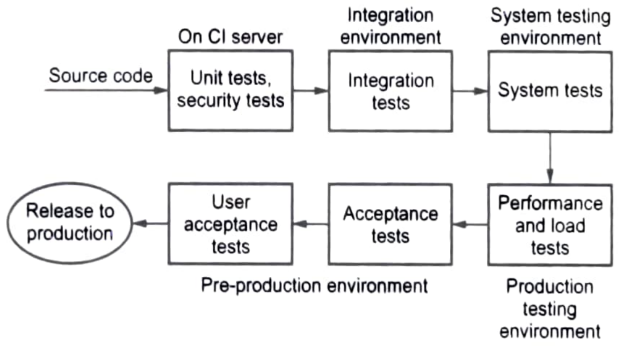
On CI server
Integration environment
System testing environment
Source code
Unit tests, security tests
Integration tests
System tests
Release to production
User acceptance tests
Acceptance tests
Performance and load tests
Pre-production environment
Production testing environment

**Fig. 9.10.1 Continuous testing architecture in DevOps**

- The unit tests happen on the Continuous Integration(CI) server itself, which tests each unit of the system in isolation.
- Integration tests happen on Integration environment which basically verifies the components integrated together.
- System tests in the system testing environment where the BIG system with all the integrated components and interfaces are tested through system-level scenarios in a system testing environment and so on.
- And the depth of testing often progresses as the simulation of environment gets closer to production.

- Then goes to the 'Acceptance testing' which basically includes the automated site acceptance test cases and then finally on to the 'User Acceptance Testing' which could be a manual execution and includes end-user participation to carry out the tests and a this will be a kind of final sign off on the product or a feature.
- Here the continuous testing would be mainly running the automated test cases automatically with a trigger.

### 9.10.2 Advantages of Continuous Testing

1) Quality of the product gets improved.

2) Speed of execution gets increased by involvement of automated testing techniques

3) Faster feedback on code.

4) The continuous testing process boosts the confidence of the team and encourages them to improve continuously.

### 9.10.3 Challenges in Continuous Testing

**1) Tool early stage testing** : Early testing is normally done at requirements analysis phase. In this phase requirements can get changed. There may be lack of test case coverage or there can be multiple overlapping test cases. Poorly defined or incomplete requirements can not be tested completely in this phase.

**2) Test data management** : Another challenge in continuous delivery is locating proper test data. Hence proper test data management must be done consistently. Following are the difficulties in test data management -

(i) extraction of test data spread across multiple databases

(ii) limited access to production systems

(iii) maintenance of multiple dataset versions for different tests

(iv) creating test data without copying the production data

**3) Availability of testing environment** : Within continuous delivery, tests run in parallel in the development phase and integration phase. Because of that, the team needs to have many environments available for different purposes. The lack of test environments is one of the biggest challenges to achieve continuous testing.

**4) Lack of availability of proper automated testing tool** : Most of the legacy testing tools are unfit for continuous testing in DevOps, hence use of appropriate automated tools at every stage of development life cycle is a big challenge.

**Review Question**

1. *Explain continuous integration process in DevOps.*

## 9.11 How to Choose Right DevOps Tools

Following are the 7 steps to be followed to choose right DevOps tools -

**Step 1 : Understand the collaboration and shared tools strategy**

DevOps teams need to come up with a common tools strategy that lets them collaborate across development, testing, and deployment. The common strategy is based on -

- Processes
- Communications and collaboration planning
- Continuous development tools
- Continuous integration tools
- Continuous testing tools
- Continuous deployment tools

**Step 2 : Use tools to capture all the requests**

DevOps tooling should capture every request for new or changed software. DevOps provides the ability to automate the acceptance of change requests that come in either from the business or from other parts of the DevOps teams.

**Step 3 : Use agile Kanban project management for automation**

Kanban is a framework used to implement agile development that matches the amount of work in progress to the team's capacity. It gives teams more flexible planning options, faster output, clear focus, and transparency throughout the development cycle. Kanban tools provide the ability to see all the items in context with each other.

**Step 4 : Use tools to log metrics on both manual and automated processes**

Select tools that can help you understand the productivity of your DevOps processes, both automated and manual.

**Step 5 : Implement test automation and test data provisioning tooling**

With DevOps, testing must be continuous. There can be thousands of test cases that need to be executed during application development. Thus test automation and test data provision tools are essential.

**Step 6 : Perform acceptance tests for each deployment tooling**

For the tool set selected, those charged with DevOps testing processes should to spend time defining the acceptance tests, and ensuring that the tests meet with the acceptance criteria selected. These tests may be changed at any time by development or operations.

**Step 7 : Ensure continuous feedback between the teams**

There must be feedback loops to automate communication between tests that spot issues, and tests that process needs to be supported by your chosen tool. Hence tools that monitor software in production, identify issues in software in an automated or manual way, link the issues to deployable components.

## 9.12 Challenges with DevOps Implementation

**1) Cultural change** : Any organization must promote the collaborative culture for effective implementation of DevOps. The leaders should bring transparency in the work process. There must be positive atmosphere in an organization.

**2) Bringing silos and sectors together** : There is a tendency of maintaining silos and sectors within the team. While developers are constantly writing pieces of code in order to build a system, testers perform a thorough analysis to ensure product stability before the final delivery to the customer. Due to this practice, there lies a big gap between teams. Both Dev and Ops work in silos, leading to a lack of transparency and poor teamwork.

**3) Giving up legacy systems** : Organizations must give up the old or outdated systems and must adopt modern and efficient systems. Handling new systems along with the old existing systems in the organization can be challenging many times.

**4) Tool selection confusion** : There are number of tools available in the market which tempt the DevOp developers to choose different tools. This leads to changing and updating the tools frequently. Changing and updating the tools becomes difficult with change in strategy. Hence instead of using ever changing tools and increasing the cost, the team should adopt well organized long term strategy to use specific tool.

**5) Different metrics** : During product development process each team measure their performance using different metrics. When the projects is to be implemented using DevOps technology, there must be a common metric to be used to measure the performance. Accepting a common measure of performance is sometimes challenging for different teams.

**6) Resistance to change** : The teams are normally unwilling to accept the changes. They are resistance to change their preferred working style. They do not open up the silos to other teams. This makes it difficult to other teams to work and may create an unhealthy environment.

**7) Process challenges** : DevOps strategy does not define specific rules to implement the process or to use the tools. The only restriction is to follow the project goal. Although this gives lot of flexibility and chances to use innovative methodologies , it may lead to challenging situations like confusion and disputes among the team members of some strategies.

## 9.13 Must Do Things for DevOps

**Following is a list of things that are most required for DevOps to implement.**

**1) Find DevOps driver :** A DevOps driver is a resource, process or condition that will help to stay focused in software development activities, security and data management.

**2) Adopt with DevOps culture :** Adopting DevOps culture means instead of having traditional software development approach,the team should work in collaboration with the other teams.It should have an involvement in all the activities from development to deployment of the product and chase for the quality in work.

**3) Move in right direction :** For moving the team in right direction coaching in both Agile and DevOps help the team members to embrace the changes in culture and practices. Team members should also be trained to use new tools and techniques.

**4) Make the customer happy :** By creating user friendly product and high quality service within the stipulated time- makes your customer happy.

**5) Follow certain principles :** Some of the commonly followed principles are -

(i) Automate repetitive tasks.

(ii) Keep it simple

(iii) Every must be responsible.

(iv) Get continuous feedback and work accordingly.

**6) Ensure security :** Adopting DevOps practices introduce complications for implementing standard security practices. Team must think about making the product and service more secure.

**7) Make use of right tools :** No tool is perfect. But the team should adopt well organized long term strategy to use specific tool.

**8) Use key technologies :** DevOps implementation require frequent deployment and continuous feedback. Due to which new features are getting added frequently. Hence make use of key technologies and practices instead of making use of traditional, monolithic development.

## 9.14 Mapping My App to DevOps

Mapping My Application to DevOps is called adoption of DevOps. The DevOps adoption is a continuous iterative process. It consists of following steps -

1) Assessment

2) Definition

3) Implementation

4) Measurement

5) Feedback

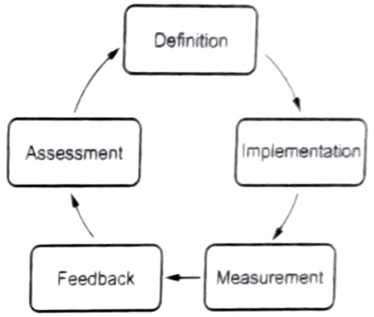

Fig. 9.14.1

DevOps strategy can be adopted to the entire end-to-end development process or it can be applied to particular life cycle stage (For instance - coding).

## 9.15 Assessment, Definition, Implementation, Measure and Feedback

**1) Assessment :**

- This is the first step of mapping an application to DevOps.
- This step is basically called as **DevOps maturity assessment.**
- There are a set of parameters to be measured at every stage of DevOps Maturity Model to confirm an organization's level of DevOps maturity.
- Assessment the Devops maturity is based on the various parameters such as process, people, tools, culture, measurement and reporting.
- The DevOps capability is measured in terms of Crawl, Walk, Run and Sprint or equivalent levels measuring maturity in increasing order.
- Method of assessment is arranging brain-storming sessions with different stakeholders. This is an iterative activity in which stakeholders put their views and ideas and the application is refined further.
- Typically set of predefined questionnaire is prepared for measuring the maturity of application in activity of development lifecycle.
- Finally a detailed report about DevOps maturity is prepared.

**2) Definition :**

- Definition is the second phase in which solution is defined. That means the actions are identified which help the application to reach to next level of maturity.
- The input to definition phase are - 1) Objectives of the project 2) findings from assessment phase and 3) Enterprise DevOps strategy.

- Enterprise DevOps strategy consists of set of guidelines for individual project about adopting DevOps strategy. For instance - the guideline may consists of list of approved tools for the project.
- Enterprise DevOps strategy must be flexible enough by accepting contributions from individual project, even if it is not implemented earlier in the organization.
- Hence Enterprise DevOps strategy must be evolving for betterment.
- At the end of definition phase, a blue print of project of DevOps is prepared with action items that are obtained in assessment report.

**3) Implementation :**

- This is a phase in which the items which are identified during the definition phase are implemented.
- Due to adoption of DevOps strategy, use of automated tools saves lot of resources during each phase of development life cycle. This enhances the efficiency of application development process.
- Following Fig. 9.15.1 and 9.15.2 illustrates the difference between pre DevOps implementation and post DevOps implementation

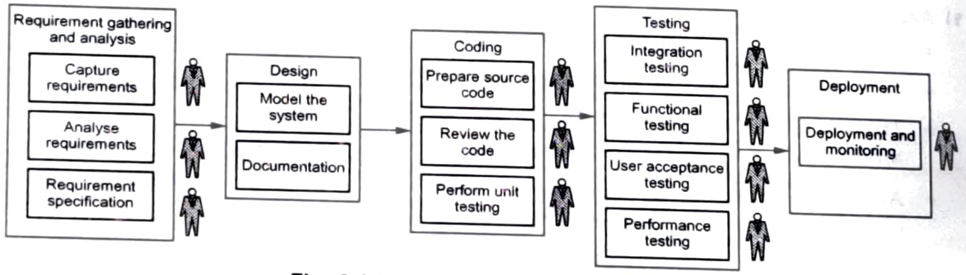

**Fig. 9.15.1 Pre DevOps development**

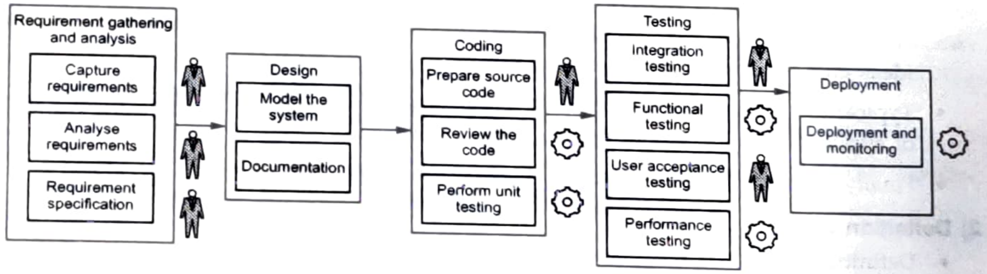

**Fig. 9.15.2 Post DevOps implementation**

Note that during the use of automated tools at various stages of development life cycle makes the things simplified. For instance at the coding phase, code review can be possible by the tool 'SonarQube', and for unit testing one can use the tool such as Junit. Similarly different tools can be used at testing and deployment phases.

**4) Measure and Feedback :**

- This is a final important step in which the success of the project is measured.
- After implementation and deployment of the application in the working environment, this step must appear so that the quality of implemented application can be judged.
- Following are the **Key Performer Indicators(KPI)** that measures the benefits of tools and process implementation -
  - **Failure rate :** What is the frequency of occurring failures in the operations ?
  - **Deployment time :** Does the duration of deployment time get shorten ?
  - **Mean time to recovery :** How quickly the application can recover from failure ?
- In addition to these KPIs, the **dashboards** can be provided to measure the success of the application implementation. These dashboards are provided by the tools to monitor the individual tools parameters. Hence it is possible to measure testing time, productivity and so on.
- Thus the ultimate goal of mapping the application to DevOps is to reduce the overall cost of the implementation and to enhance the quality of project.

**Review Question**

1. *Explain Assessment, Definition, Implementation, Measure and Feedback in DevOps.*

❑❑❑