**Notes for MCA-II (Semester- IV)**

**Subject :- DevOps**

**(Subject Code:- IT-41**

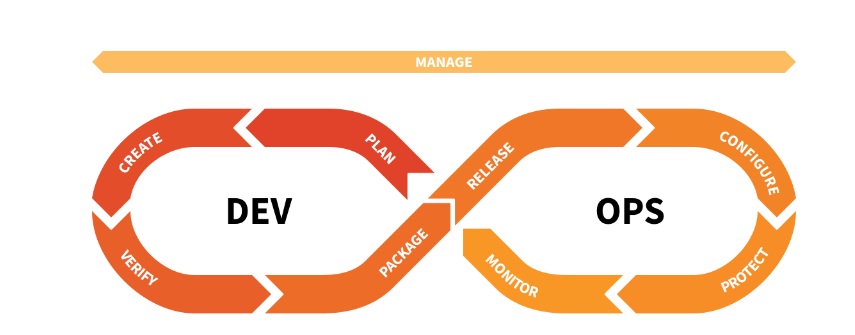
**Chapter: 1]** **Introduction to DevOps**

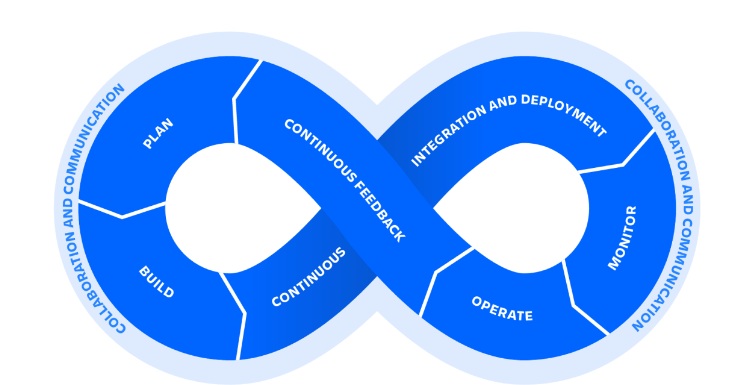
* **1.1 Define DevOps :-**

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. This speed enables organizations to better serve their customers and compete more effectively in the market.



DevOps combines development and operations to increase the efficiency, speed, and security of software development and delivery compared to traditional processes. A more nimble software development lifecycle results in a competitive advantage for businesses and their customers.

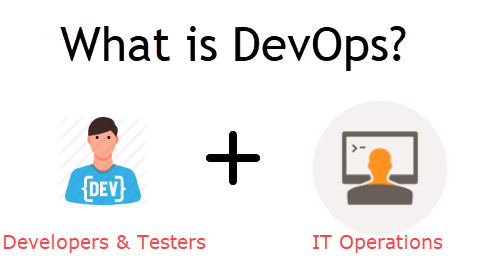




* **1.2 What is DevOps ?**

DevOps is a collaboration between Development and IT Operations to make software production and Deployment in an automated & repeatable way. DevOps helps increase the organization’s speed to deliver software applications and services. The full form of ‘DevOps’ is a combination of ‘Development’ and ‘Operations.’

It allows organizations to serve their customers better and compete more strongly in the market. In simple words, DevOps can be defined as an alignment of development and IT operations with better communication and collaboration.



**Why is DevOps Needed?**

* Before DevOps, the development and operation team worked in complete isolation.
* Testing and Deployment were isolated activities done after design-build. Hence they consumed more time than actual build cycles.
* Without using DevOps, team members spend a large amount of their time testing, deploying, and designing instead of building the project.
* Manual code deployment leads to human errors in production.
* Coding & operation teams have separate timelines and are not synch, causing further delays.
* There is a demand to increase the rate of software delivery by business stakeholders. As per Forrester Consulting Study, Only 17% of teams can use delivery software quickly, proving the pain point.

**How is DevOps different from traditional IT**

* In this DevOps training, let’s compare the traditional software waterfall model with DevOps to understand the changes DevOps brings.
* We assume the application is scheduled to go live in 2 weeks, and coding is 80% done. We assume the application is a fresh launch, and the process of buying servers to ship the code has just begun-

|  |  |
| --- | --- |
| **Old Process** | **DevOps** |
| After placing an order for new servers, the Development team works on testing. The  Operations team works on extensive paperwork as required in enterprises to deploy the infrastructure. | After placing an order for new servers Development  and Operations team work together on the paperwork  to set up the new servers. This results in better visibility of infrastructure requirements. |
| Projections about failover, redundancy, data centre locations, and storage requirements are skewed as no inputs are available from developers who have deep knowledge of the application. | Projections about failover, redundancy, disaster recovery, data center locations, and storage requirements are pretty accurate  due to the inputs from the developers. |
| The operations team has no clue about the progress of the Development team. The operations team develops a monitoring plan as per their understanding. | In DevOps, the Operations team is completely aware of the developers’ progress. Operations teams interact with developers and jointly develop a monitoring plan that caters to IT and business needs. They also use advanced [Application Performance Monitoring (APM) Tools](https://www.guru99.com/apm-tools.html). |
| Before going go-live, the load testing crashes the application, and the release is delayed. | Before going go-live, the [load testing](https://www.guru99.com/load-testing-tutorial.html) makes the application a bit slow. The development team quickly fixes the bottlenecks, and the application is released on time. |

* **1.3 SDLC models, Lean, ITIL, Agile:-**

SDLC stands for Software Development Life Cycle. SDLC is a process that consists of a series of planned activities to develop or alter the Software Products. This tutorial will give you an overview of the SDLC basics SDLC models available and their application in the industry. This tutorial also elaborates on other related methodologies like Agile, RAD and Prototyping.

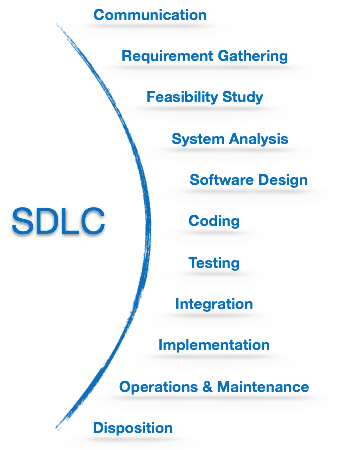
**What is SDLC?**

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



A typical Software Development Life Cycle consists of the following stages:



* **Lean :-**

A lean organization understands customer value and focuses its key processes to continuously increase it. The ultimate goal is to provide perfect value to the customer through a perfect value creation process that has zero waste.

To accomplish this, lean thinking changes the focus of management from optimizing separate technologies, assets, and vertical departments to optimizing the flow of products and services through entire value streams that flow horizontally across technologies, assets, and departments to customers.

Eliminating waste along entire value streams, instead of at isolated points, creates processes that need less human effort, less space, less capital, and less time to make products and services at far less costs and with much fewer defects, compared with traditional business systems. Companies are able to respond to changing customer desires with high variety, high quality, low cost, and with very fast throughput times. Also, information management becomes much simpler and more accurate.

A popular misconception is that lean is suited only for manufacturing. Not true. Lean applies in every business and every process. It is not a tactic or a cost reduction program, but a way of thinking and acting for an entire organization.

Businesses in all industries and services, including healthcare and governments, are using lean principles as the way they think and do. Many organizations choose not to use the word lean, but to label what they do as their own system, such as the Toyota Production System or the Danaher Business System. Why? To drive home the point that lean is not a program or short term cost reduction program, but the way the company operates. The word **transformation or lean transformation** is often used to *characterize a company moving from an old way of thinking to lean thinking. It requires a complete transformation on how a company conducts business*. This takes a long-term perspective and perseverance.

The term "lean" was coined to describe Toyota's business during the late 1980s by a research team headed by Jim Womack, Ph.D., at MIT's International Motor Vehicle Program.

The characteristics of a lean organization and supply chain are described in *Lean Thinking*, by Womack and Dan Jones, founders of the Lean Enterprise Institute and the Lean Enterprise Academy (UK), respectively. While there are many very good books about lean techniques, *Lean Thinking* remains one of the best resources for understanding "what is lean" because it describes the *thought process,* the overarching key principles that must guide your actions when applying lean techniques and tools.

**Purpose, Process, People:-**   
Womack and Jones recommend that managers and executives embarked on lean transformations think about three fundamental business issues that should guide the transformation of the *entire organization:*

* **Purpose**: What customer problems will the enterprise solve to achieve its own purpose of prospering?
* **Process**: How will the organization assess each major value stream to make sure each step is valuable, capable, available, adequate, flexible, and that all the steps are linked by flow, pull, and leveling?
* **People**: How can the organization ensure that every important process has someone responsible for continually evaluating that value stream in terms of business purpose and lean process? How can everyone touching the value stream be actively engaged in operating it correctly and continually improving it?

Lean is a business methodology that promotes the flow of value to the customer through two guiding tenets: continuous improvement and respect for people. Jim Benson of Modus Cooperandi defines Lean methodology in this way: “Lean is both a philosophy and a discipline which, at its core, increases access to information to ensure responsible decision making in the service of creating customer value.”

Lean methodology is not a new concept, but its modern application to business is constantly evolving. Before Lean was known as a business methodology, it was an approach to the manufacturing process. Keep reading to learn more about the history and application of Lean, as well as key Lean methodology principles.

**Roots in Manufacturing**

[Lean methodology originated with the Toyota Production System](https://www.planview.com/resources/guide/lean-principles-101/what-is-lean/), or TPS, which revolutionized the manufacture of physical goods in the 1950s, ‘60s, and beyond. Lean maintains its hold in manufacturing, but has also found new applications in knowledge work, helping businesses in all industries eliminate waste, improve processes, and boost innovation.

**Expansion into Software Development**

Lean methodology’s first applications outside of manufacturing appeared in software development, in a discipline known as Agile methodology. Conceptually, Agile software development is a Lean development methodology for [optimizing the software development cycle](https://www.planview.com/resources/articles/lkdc-agile-software-development/).

Software development is a natural application of Lean methodology because, much like manufacturing, it:

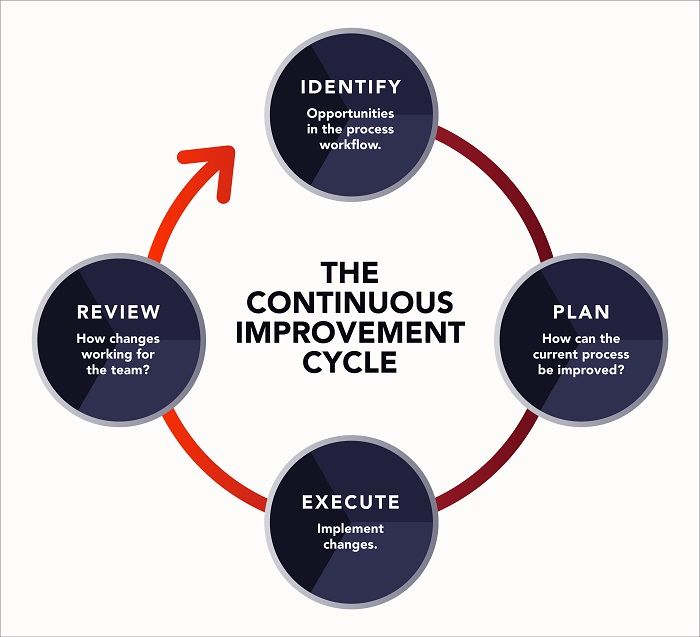
* Generally follows a defined process
* Has some defined conditions of acceptance
* Results in the delivery of tangible value

Over time, the success of applying Agile and Lean principles to software development piqued the interest of other departments and other industries. Today, Lean development methodology is being applied to knowledge work that follows a process – which is essentially all knowledge work.

**Pillars of Lean Methodology: Continuous Improvement and Respect for People**

There are two primary concepts that guide all practice of Lean methodology, which we call the Pillars of Lean. They are: continuous improvement and respect for people.

**Continuous improvement**



Continuous improvement is one of the Pillars of Lean, which guide all Lean methodology practice.

When some people think of Lean methodology, they equate it with the elimination of waste. While it’s true that Lean organizations aim to eliminate waste (defined as anything that does not deliver value to the customer), the goal is not elimination – it’s value creation.

So how do we create value? We become learning organizations. We set out to learn what our customers want and need, and how to eliminate what they don’t. We work to continuously improve so that our value stream, from end to end, is continuously optimizing to create more value for the customer.

How do we learn what is valuable? We deliver quickly. When we deliver quickly, based on what we know about the customer, we are able to get feedback quickly. And whether what we deliver is a failure or a success (or somewhere in between), we gain valuable insight into how to improve. This is how we achieve business agility; this is how we, through the process of creating value, eliminate waste.

**Respect for frontline workers**

Often, [the best ideas come from the people with their hands on the product](https://blog.planview.com/respect-for-people-leans-neglected-pillar/). In most organizations, decisions are made at the top of the organization and trickled down to the frontline. Lean thinking encourages allowing everyone, especially those closest to the product and the customer, to have an equal voice, to ensure that the voice of the customer, and those doing the work, is heard. This is the Lean concept of going to the gemba – going to the place where the work is done – to get ideas for improving work and creating value. Lean thinking says that good people want to do their best work and are motivated to make decisions that optimize their time and talent to create the most value for the customer. Going to the gemba allows the organization to capture the best ideas and bring them to fruition.

* **ITIL (Information Technology Infrastructure Library) :**

During the late 1980s the Central Computer and Telecommunication Agency (CCTA) in the United Kingdom started to work on what is now known as ITIL, the Information Technology Infrastructure Library.

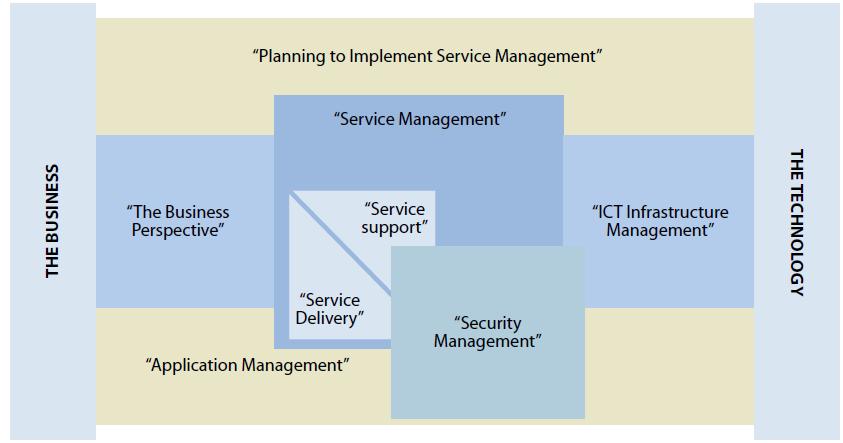
The Information Technology Infrastructure Library (ITIL) is a set of concepts and techniques for managing information technology (IT) infrastructure, development, and operations.

The Information Technology Infrastructure Library (ITIL) is a set of practices for IT service management (ITSM) that focuses on aligning IT services with the needs of business.

ITIL describes processes, procedures, tasks and checklists that are not organization-specific, used by an organization for establishing integration with the organization's strategy, delivering value and maintaining a minimum level of competency. It allows the organization to establish a baseline from which it can plan, implement and measure. It is used to demonstrate compliance and to measure improvement.

ITIL provides a comprehensive, consistent and coherent set of best practices for IT Service Management processes, promoting a quality approach to achieving business effectiveness and efficiency in the use of information systems.

***ITIL is focused on identifying best practices in regards to managing IT service levels and is particularly process-oriented:***

****

**Figure : ITIL**

* **“Planning to Implement Service Management.**”

It deals explicitly with the question of where to start with ITIL. It outlines the steps necessary to identify how the organization would benefit from ITIL. It helps identify current strengths and weaknesses and gives practical guidance on the evaluation of the current maturity levels of service management within the

current organization.

* **“The Business Perspective.”**

The Business Perspective is designed to familiarize business management with the architecture and components of information and communications technology (ICT) —infrastructure required to support the business processes. It helps business leaders better understand the benefits of best practices in IT service management.

* **“Software Asset Management.”**

It encompasses the entire infrastructure and processes necessary for the effective management, control, and protection of the software assets within an organization, throughout all stages of their life cycle.

* **“Service Support.”**

Service Support focuses on ensuring that the customer has access to appropriate services to support their business functions. It covers configuration management and other support management issues including incident, problem, change, and release management.

* **“Service Delivery.”**

Service Delivery covers the service the business requires of IT to enable adequate support to the business users. This includes processes for service-level management, availability management, capacity management, financial management for IT services, and continuity management.

* **“Security Management.”**

It looks at security from the service provider perspective, identifying the relationship between security management and the IT security officer, as well as outlining how it provides the level of security necessary for the entire organization. It further focuses on the process of implementing security requirements identified in the IT service level agreement.

* **“ICT Infrastructure Management.”**

This covers all aspects of infrastructure management from identification of business requirements to acquiring, testing, installation, and deployment of infrastructure components. It includes the design and planning processes, deployment processes, operations processes, and technical support processes.

* **“Application Management.”**

Application Management addresses the complex subject of managing applications from initial business requirements through the application management lifecycle, up to and including retirement. A strong emphasis is placed on ensuring that IT projects and strategies are tightly aligned with those of the business throughout the applications life cycle. Once an application is approved and funded, it is tracked throughout its life cycle by the software asset management function of ITIL.

* **1.4 Why is DevOps used?**

DevOps allows Agile Development Teams to implement [Continuous Integration and Continuous Delivery](https://www.guru99.com/continuous-integration.html), which helps them launch products faster into the market.

Other Important reasons are:

**1. Predictability:**DevOps offers a significantly lower failure rate of new releases.

**2. Reproducibility:**Version everything so that earlier versions can be restored anytime.

**3. Maintainability:**Effortless recovery process in the event of a new release crashing or disabling the current system.

**4. Time to market:**DevOps reduces the time to market up to 50% through streamlined software delivery. It is particularly the case for digital and mobile applications.

**5. Greater Quality:**DevOps helps the team improve application development quality by incorporating infrastructure issues.

**6. Reduced Risk:**DevOps incorporates security aspects in the software delivery lifecycle, and it helps reduce defects across the lifecycle.

**7. Resiliency:**The Operational state of the software system is more stable, secure, and changes are auditable.

**8. Cost Efficiency:**DevOps offers cost efficiency in the software development process, which is always an aspiration of IT management.

**9. Breaks larger code base into small pieces:**DevOps is based on the agile programming method. Therefore, it allows breaking larger codebases into smaller and manageable chunks.

### When to adopt DevOps?

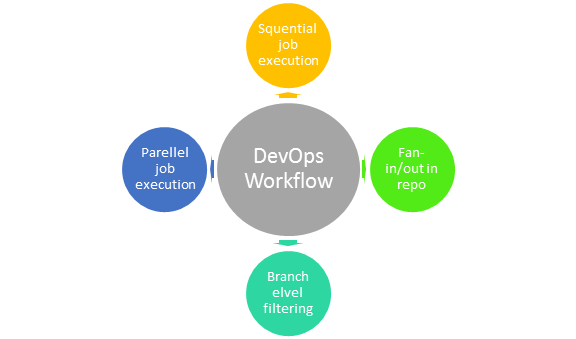
DevOps should be used for large distributed applications such as [eCommerce sites](https://www.guru99.com/best-ecommerce-platform.html) or applications hosted on a cloud platform.

### When not to adopt DevOps?

It should not be used in mission-critical applications like banks, power and other sensitive data sites. Such applications need strict access controls on the production environment, a detailed change management policy, and access control policy to the data centres.

## DevOps Workflow

Workflows provide a visual overview of the sequence in which input is provided. It also tells about performed actions, and output is generated for an operations process.

DevOps WorkFlow

Workflow allows the ability to separate and arrange jobs that the users top request. It also can mirror their ideal process in the configuration jobs.

## 1.5 History of DevOps :-

## DevOps is a set of practices that combines [software development](https://en.wikipedia.org/wiki/Software_development) (Dev) and [IT operations](https://en.wikipedia.org/wiki/IT_operations) (Ops). It aims to shorten the [systems development life cycle](https://en.wikipedia.org/wiki/Systems_development_life_cycle) and provide [continuous delivery](https://en.wikipedia.org/wiki/Continuous_delivery) with high [software quality](https://en.wikipedia.org/wiki/Software_quality). DevOps is complementary with [Agile software development](https://en.wikipedia.org/wiki/Agile_software_development); several DevOps aspects came from the Agile methodology.

## Other than it being a cross-functional combination of the terms and concepts for "development" and "operations", academics and practitioners have not developed a universal definition for the term "DevOps"

Most often, DevOps is characterized by key principles: shared ownership, workflow automation, and rapid feedback.

From an academic perspective, [Len Bass](https://en.wikipedia.org/wiki/Len_Bass), [Ingo Weber](https://en.wikipedia.org/w/index.php?title=Ingo_Weber&action=edit&redlink=1), and [Liming Zhu](https://en.wikipedia.org/w/index.php?title=Liming_Zhu&action=edit&redlink=1)—three computer science researchers from the [CSIRO](https://en.wikipedia.org/wiki/CSIRO) and the [Software Engineering Institute](https://en.wikipedia.org/wiki/Software_Engineering_Institute)—suggested defining DevOps as "a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality"

The DevOps movement started to coalesce some time between 2007 and 2008, when IT operations and software development communities raised concerns what they felt was a fatal level of dysfunction in the industry.

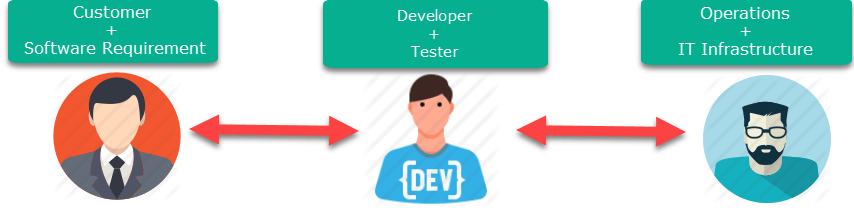
They railed against the traditional software development model, which called for those who write code to be organizationally and functionally apart from those who deploy and support that code.

Developers and [IT/Ops professionals](https://www.atlassian.com/it-unplugged/devops) had separate (and often competing) objectives, separate department leadership, separate key performance indicators by which they were judged, and often worked on separate floors or even separate buildings. The result was teams concerned only with their own fiefdoms, long hours, botched releases, and unhappy customers. Surely there’s a better way, they said. So, the two communities came together and started talking

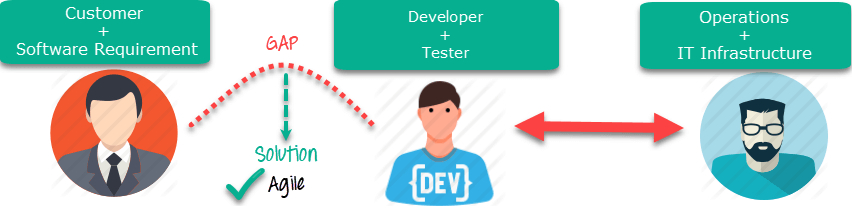
* In 2009, the first conference named DevOpsdays was held in Ghent Belgium. Belgian consultant and Patrick Debois founded the conference.
* In 2012, the state of DevOps report was launched and conceived by Alanna Brown at Puppet.
* In 2014, the annual State of DevOps report was published by Nicole Forsgren, Jez Humble, Gene Kim, and others. They found DevOps adoption was accelerating in 2014 also.
* In 2015, Nicole Forsgren, Gene Kim, and Jez Humble founded DORA (DevOps Research and Assignment).
* In 2017, Nicole Forsgren, Gene Kim, and Jez Humble published "Accelerate: Building and Scaling High Performing Technology Organizations".

## 1.10 How is DevOps different from Agile? DevOps Vs Agile

Stakeholders and communication chain a typical IT process.

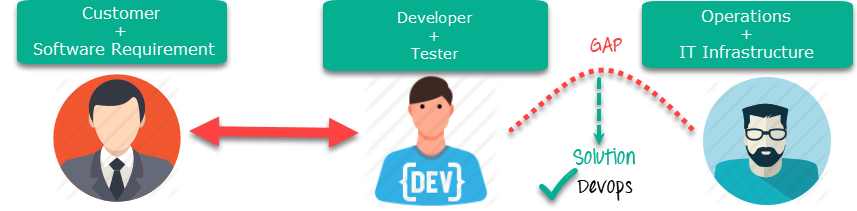


Agile addresses gaps in Customer and Developer communications



**Fig. Agile Process**

DevOps addresses gaps in Developer and IT Operations communications

**Fig. DevOps Process**

## Difference between DevOps and Agile

|  |  |
| --- | --- |
| **Agile** | **DevOps** |
| **Emphasize breaking down barriers between developers and management.** | **DevOps is about software deployment and operation teams.** |
| **Addresses gaps between customer requirements and development teams.** | **Addresses the gap between the development and Operation team** |
| **Focuses more on functional and non-functional readiness** | **It focuses on operational and business readiness.** |
| **Agile development pertains mainly to the company’s way development is thought out.** | **DevOps emphasises deploying software in the most reliable and safest ways that aren’t always the fastest.** |
| **Agile development emphasises training all team members to have varieties of similar and equal skills. So that, when something goes wrong, any team member can get assistance from any member in the absence of the team leader.** | **DevOps likes to divide and conquer, spreading the skill set between the development and operation teams. It also maintains consistent communication.** |
| **Agile development manages on “sprints”. It means that the timetable is much shorter (less than a month), and several features are to be produced and released in that period.** | **DevOps strives for consolidated deadlines and benchmarks with significant releases rather than smaller and more frequent ones.** |

## DevOps Principles

Here are six principles that are essential when adopting DevOps:

**1. Customer-Centric Action:**The DevOps team must constantly take customer-centric action to invest in products and services.

**2. End-To-End Responsibility:**The DevOps team needs to provide performance support until they become end-of-life. This enhances the level of responsibility and the quality of the products engineered.

**3. Continuous Improvement:**DevOps culture focuses on continuous improvement to minimize waste, and it continuously speeds up the improvement of products or services offered.

**4. Automate everything:**Automation is a vital principle of the DevOps process, and this is not only for software development but also for the entire infrastructure landscape.

**5. Work as one team:**In the DevOps culture, the designer, developer, and tester are already defined, and all they need to do is work as one team with complete collaboration.

**6. Monitor and test everything:**Monitor and test everything: The DevOps team needs robust monitoring and testing procedures.

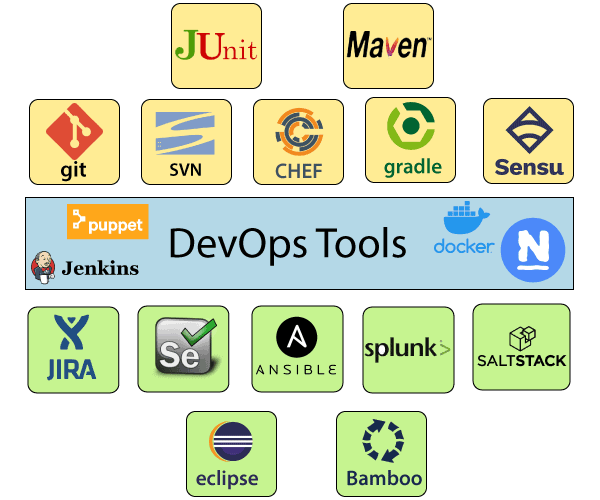
## Who is a DevOps Engineer?

A DevOps Engineer is an IT professional who works with software developers, system operators, and other production IT staff to administer code releases. DevOps should have hard and soft skills to communicate and collaborate with development, testing, and operations teams.

The DevOps approach needs frequent, incremental changes to code versions, requiring frequent deployment and testing regimens. Although DevOps engineers need to code occasionally from scratch, they must have the basics of software development languages.

A DevOps engineer will work with development team staff to tackle the coding and scripting needed to connect code elements, like libraries or software development kits.

* **1.11 DevOps Tool** :-



**1) Puppet**

Puppet is the most widely used DevOps tool. It allows the delivery and release of the technology changes quickly and frequently. It has features of versioning, automated testing, and continuous delivery. It enables to manage entire infrastructure as code without expanding the size of the team.

**Features**

* Real-time context-aware reporting.
* Model and manage the entire environment.
* Defined and continually enforce infrastructure.
* Desired state conflict detection and remediation.
* It inspects and reports on packages running across the infrastructure.
* It eliminates manual work for the software delivery process.
* It helps the developer to deliver great software quickly.

**2) Ansible**

Ansible is a leading DevOps tool. Ansible is an open-source IT engine that automates application deployment, cloud provisioning, intra service orchestration, and other IT tools. It makes it easier for DevOps teams to scale automation and speed up productivity.

Ansible is easy to deploy because it does not use any **agents** or **custom** **security** infrastructure on the client-side, and by pushing modules to the clients. These modules are executed locally on the client-side, and the output is pushed back to the Ansible server.

**Features**

* It is easy to use to open source deploy applications.
* It helps in avoiding complexity in the software development process.
* It eliminates repetitive tasks.
* It manages complex deployments and speeds up the development process.

**3) Docker**

Docker is a high-end DevOps tool that allows building, ship, and run distributed applications on multiple systems. It also helps to assemble the apps quickly from the components, and it is typically suitable for container management.

**Features**

* It configures the system more comfortable and faster.
* It increases productivity.
* It provides containers that are used to run the application in an isolated environment.
* It routes the incoming request for published ports on available nodes to an active container. This feature enables the connection even if there is no task running on the node.
* It allows saving secrets into the swarm itself.

**4) Nagios**

Nagios is one of the more useful tools for DevOps. It can determine the errors and rectify them with the help of network, infrastructure, server, and log monitoring systems.

**Features**

* It provides complete monitoring of desktop and server operating systems.
* The network analyzer helps to identify bottlenecks and optimize bandwidth utilization.
* It helps to monitor components such as services, application, OS, and network protocol.
* It also provides to complete monitoring of Java Management Extensions.

**5) CHEF**

A chef is a useful tool for achieving scale, speed, and consistency. The chef is a cloud-based system and open source technology. This technology uses Ruby encoding to develop essential building blocks such as recipes and cookbooks. The chef is used in infrastructure automation and helps in reducing manual and repetitive tasks for infrastructure management.

Chef has got its convention for different building blocks, which are required to manage and automate infrastructure.

**Features**

* It maintains high availability.
* It can manage multiple cloud environments.
* It uses popular Ruby language to create a domain-specific language.
* The chef does not make any assumptions about the current status of the node. It uses its mechanism to get the current state of the machine.

**6) Jenkins**

Jenkins is a DevOps tool for monitoring the execution of repeated tasks. Jenkins is software that allows continuous integration. Jenkins will be installed on a server where the central build will take place. It helps to integrate project changes more efficiently by finding the issues quickly.

**Features**

* Jenkins increases the scale of automation.
* It can easily set up and configure via a web interface.
* It can distribute the tasks across multiple machines, thereby increasing concurrency.
* It supports continuous integration and continuous delivery.
* It offers 400 plugins to support the building and testing any project virtually.
* It requires little maintenance and has a built-in GUI tool for easy updates.

**7) Git**

Git is an open-source distributed version control system that is freely available for everyone. It is designed to handle minor to major projects with speed and efficiency. It is developed to co-ordinate the work among programmers. The version control allows you to track and work together with your team members at the same workspace. It is used as a critical distributed version-control for the DevOps tool.

**Features**

* It is a free open source tool.
* It allows distributed development.
* It supports the pull request.
* It enables a faster release cycle.
* Git is very scalable.
* It is very secure and completes the tasks very fast.

**8) SALTSTACK**

Saltstack is a lightweight DevOps tool. It shows real-time error queries, logs, and more directly into the workstation. SALTSTACK is an ideal solution for intelligent orchestration for the software-defined data center.

**Features**

* It eliminates messy configuration or data changes.
* It can trace detail of all the types of the web request.
* It allows us to find and fix the bugs before production.
* It provides secure access and configures image caches.
* It secures multi-tenancy with granular role-based access control.
* Flexible image management with a private registry to store and manage images.

**9) Splunk**

Splunk is a tool to make machine data usable, accessible, and valuable to everyone. It delivers operational intelligence to DevOps teams. It helps companies to be more secure, productive, and competitive.

**Features**

* It has the next-generation monitoring and analytics solution.
* It delivers a single, unified view of different IT services.
* Extend the Splunk platform with purpose-built solutions for security.
* Data drive analytics with actionable insight.

**10) Selenium**

Selenium is a portable software testing framework for web applications. It provides an easy interface for developing automated tests.

**Features**

* It is a free open source tool.
* It supports multiplatform for testing, such as Android and ios.
* It is easy to build a keyword-driven framework for a WebDriver.
* It creates robust browser-based regression automation suites and tests.
* **1.12 Configuration Management :-**

Configuration Management (CM) is a [systems engineering](https://en.wikipedia.org/wiki/Systems_engineering) process for establishing and maintaining consistency of a product's performance, functional, and physical attributes with its requirements, design, and operational information throughout its life. The CM process is widely used by military engineering organizations to manage changes throughout the [system lifecycle](https://en.wikipedia.org/wiki/System_lifecycle) of [complex systems](https://en.wikipedia.org/wiki/Complex_system),

CM applied over the life cycle of a system provides visibility and control of its performance, functional, and physical attributes. CM verifies that a system performs as intended, and is identified and documented in sufficient detail to support its projected life cycle. *The CM process facilitates orderly management of system information and system changes for such beneficial purposes as to revise capability; improve performance, reliability, or maintainability; extend life; reduce cost; reduce risk and liability; or correct defects.* The relatively minimal cost of implementing CM is returned many fold in cost avoidance. The lack of CM, or its ineffectual implementation, can be very expensive and sometimes can have such catastrophic consequences such as failure of equipment or loss of life.

*CM emphasizes the functional relation between parts, subsystems, and systems for effectively controlling system change. It helps to verify that proposed changes are systematically considered to minimize adverse effects. Changes to the system are proposed, evaluated, and implemented using a standardized, systematic approach that ensures consistency, and proposed changes are evaluated in terms of their anticipated impact on the entire system.* CM verifies that changes are carried out as prescribed and that documentation of items and systems reflects their true configuration. A complete CM program includes provisions for the storing, tracking, and updating of all system information on a component, subsystem, and system basis.

A structured CM program ensures that documentation (e.g., requirements, design, test, and acceptance documentation) for items is accurate and consistent with the actual physical design of the item. In many cases, without CM, the documentation exists but is not consistent with the item itself. For this reason, engineers, contractors, and management are frequently forced to develop documentation reflecting the actual status of the item before they can proceed with a change. This [reverse engineering](https://en.wikipedia.org/wiki/Reverse_engineering) process is wasteful in terms of human and other resources and can be minimized or eliminated using CM.

Software configuration management, SCM is an activity which is used at every level and every part of the process of software Engineering. Every improvement takes the shape of better control. This is a discipline which controls betters and according to client need in software Engineering. With the help of this many types are changes which play an important role in software Engineering and development process.

In the simple way if we define the term configuration of management, this is the tool which makes better control, easy maintenance during the whole process of software development. *With the help of software configuration management we can easily find out what modification and controlling required by the developer. SCM have the capacity to control all those effects which comes in software projects. The main objectives of SCM is increase the production by reduce the errors.*

When a software development process start then SCM take change by identification, control, alteration, audit and etc. after that the output of total process provided to our customer. We can clarify the action of SCM as:

1. **Software Configuration Identification** - Normally software is used in various kinds of programs and documentation and data related to each program is called Configuration Identification. With the help of C.I. we can make a guide line which will be helpful in software development process, several time the requirement of guideline for check the document and design of software. Document related to SCM are the useful item, with the help of this we can make better control and take a basic unit for configuration.
2. **Software configuration control**- This is the process of deciding with the help of this we make coordination between the changes which is necessary and apply them as per mentioned in guideline. Configuration control board gives the permission for any kind of change or modification which is necessary for the project. Many times CCB take advice of those members which are the part of software development process.
3. **Accounting status of Software configuration** - The process of maintaining record of all data which is necessary for the software is called accounting status of software. It has all the data related to the old software to new software that what changes are done or required for the fulfilment of the customer need.
4. **Auditing of software configuration** - Auditing of software configuration is may be defined as an art with the help of this we can understand that the required actions or changes are done by the developer or not. Some of the item involved in the process of verifying or auditing.
   * Function is properly performed by the software.
   * The process of documentation, data is completed or not.

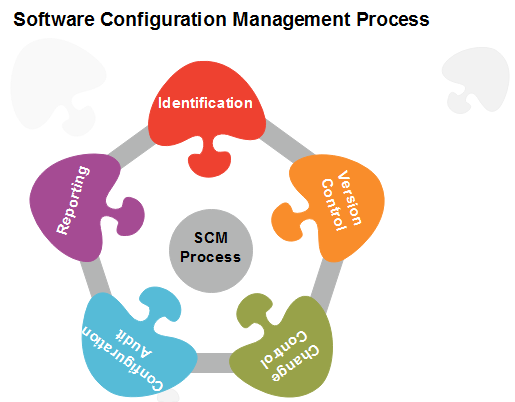
**Benefits**

* + With the help of SCM we can easily control all changes which are done in development process.
  + It gives the surety to check that changes are done on required area.
  + It is helpful to generate the new software with old components.
  + SCM has the capacity to explain everything about the process of software development.

**SCM Process**

It uses the tools which keep that the necessary change has been implemented adequately to the appropriate component. The SCM process defines a number of tasks:

* Identification of objects in the software configuration
* Version Control
* Change Control
* Configuration Audit
* Status Reporting



ChangeManagementis the process by which changes to the Project’s scope,

deliverables, timescales or resources are formally defined, evaluated and approved prior to implementation. This is achieved by understanding the business and system drivers requiring the change, documenting the benefits and costs of adopting the change and formulating a structured plan for implementing the change.

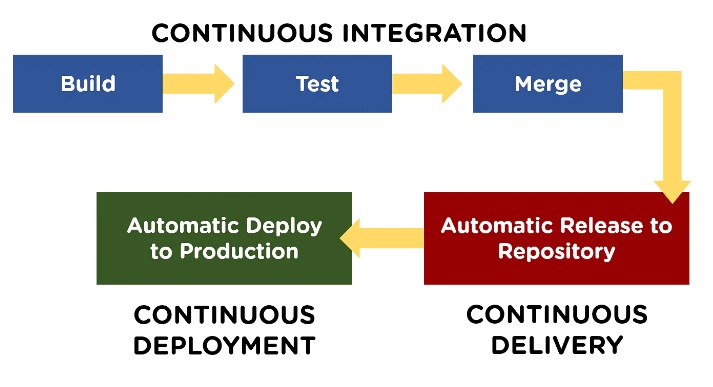
The change management process in systems engineering is the process of requesting, determining attainability, planning, implementing, and evaluating of changes to a system. It has two main goals: supporting the processing of changes and enabling traceability of changes, which should be possible through proper execution of the process described.

* Change management is not a stand-alone process for designing a business solution.
* Change management is the processes, tools and techniques for managing the people-side of change.
* Change management is not a process improvement method.
* Change management is a method for reducing and managing resistance to change when implementing process, technology or organizational change.
* Change management is a necessary component for any organizational performance improvement process to succeed, including programs like: Six Sigma, Business Process Reengineering, Total Quality Management, Organizational Development, Restructuring and continuous process improvement.
* Change management is how we drive the adoption and usage we need to realize business results.

|  |  |
| --- | --- |
| **Table 1: Role descriptions for the change management process** | |
| **Role** | **Description** |
| **Customer** | The [customer](http://en.wikipedia.org/wiki/Customer) is the role that requests a change due to problems encountered or new functionality requirements; this can be a person or an organizational entity and can be in- or external to the company that is asked to implement the change. |
| **Project manager** | The [project manager](http://en.wikipedia.org/wiki/Project_manager) is the owner of the [project](http://en.wikipedia.org/wiki/Project) that the CHANGE REQUEST concerns. In some cases there is a distinct change manager, who in that case takes on this role. |
| **Change committee** | The change [committee](http://en.wikipedia.org/wiki/Committee) decides whether a CHANGE REQUEST will be implemented or not. Sometimes this task is performed by the project manager as well. |
| **Change builder** | The change builder is the person who plans and implements the change; it could be argued that the planning component is (partially) taken on by the project manager. |
|  |  |

|  |  |  |
| --- | --- | --- |
| **Table 2: Activity descriptions for the change management process** | | |
| **Activity** | **Sub-activity** | **Description** |
| **Identify potential change** | Require new functionality | A customer desires new functionality and formulates a REQUIREMENT. |
|  | Encounter problem | A customer encounters a problem (e.g. a [bug](http://en.wikipedia.org/wiki/Software_bug)) in the system and this leads to a PROBLEM REPORT. |
|  | Request change | A customer proposes a change through creation of a CHANGE REQUEST. |
| **Analyze change request** | Determine technical feasibility | The project manager determines the technical feasibility of the proposed CHANGE REQUEST, leading to a CHANGE TECHNICAL FEASIBILITY. |
|  | Determine costs and benefits | The project manager determines the costs and benefits of the proposed CHANGE REQUEST, resulting in CHANGE COSTS AND BENEFITS. This and the above sub-activity can be done in any order and they are independent of each other, hence the modeling as unordered activities. |
| **Evaluate change** |  | Based on the CHANGE REQUEST, its CHANGE TECHNICAL FEASIBILITY and CHANGE COSTS AND BENEFITS, the change committee makes the go/no-go decision. This is modeled as a separate activity because it is an important process step and has another role performing it. It is modeled as a sub-activity (without any activity containing it) as recommended by Remko Helms (personal communication). |
| **Plan change** | Analyze change impact | The extent of the change (i.e. what other items the change effects) is determined in a CHANGE IMPACT ANALYSIS. It could be argued that this activity leads to another go/no-go decision, or that it even forms a part of the Analyze change request activity. It is modeled here as a planning task for the change builder because of its relationship with the activity Propagate change. |
|  | Create planning | A CHANGE PLANNING is created for the [implementation](http://en.wikipedia.org/wiki/Implementation) of the change. Some process descriptions (e.g. Mäkäräinen, 2000) illustrate that is also possible to ‘save’ changes and process them later in a [batch](http://en.wikipedia.org/wiki/Batch_production). This activity could be viewed as a good point to do this. |
| **Implement change** | Execute change | The change is ‘programmed’; this activity has a strong relationship with Propagate change, because sometimes the change has to be adapted to other parts of the system (or even other systems) as well. |
|  | Propagate change | The changes resulting from Execute change have to be propagated to other system parts that are influenced by it. Because this and the above sub-activity are highly dependent on each other, they have been modeled as concurrent activities. |
|  | Test change | The change builder tests whether what (s)he has built actually works and satisfies the CHANGE REQUEST. As depicted in the diagram, this can result in an [iterative](http://en.wikipedia.org/wiki/Iterative) process together with the above two sub-activities. |
|  | Update documentation | The DOCUMENTATION is updated to reflect the applied changes. |
|  | Release change | A new SYSTEM RELEASE, which reflects the applied change, is made public. |
| **Review and close change** | Verify change | The implementation of the change in the new SYSTEM RELEASE is verified for the last time, now by the project manager. Maybe this has to happen before the release, but due to conflicting literature sources and diagram complexity considerations it was chosen to model it this way and include this issue. |
|  | Close change | This change [cycle](http://en.wikipedia.org/wiki/Instruction_cycle) is completed, i.e. the CHANGE LOG ENTRY is wrapped up. |

* **1.13 Continuous Integration and Deployment:-**

****

## What is Continuous Integration?

Continuous Integration (CI) is a DevOps software development practice that enables the developers to merge their code changes in the central repository. That way, automated builds and tests can be run. The amendments by the developers are validated by creating a built and running an automated test against them.

In the case of Continuous Integration, a tremendous amount of emphasis is placed on testing automation to check on the application. This is to know if it is broken whenever new commits are integrated into the main branch.

## The "CI" in CI/CD always refers to continuous integration, which is an automation process for developers. Successful CI means new code changes to an app are regularly built, tested, and merged to a shared repository. It’s a solution to the problem of having too many branches of an app in development at once that might conflict with each other.

## What is Continuous Delivery?

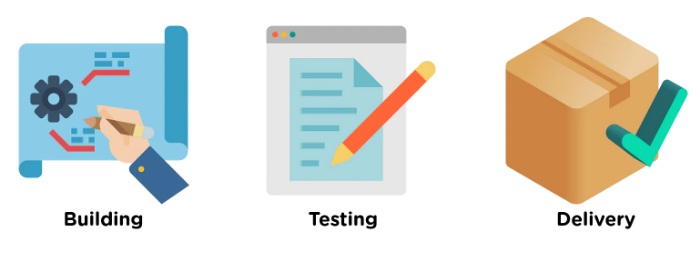
Continuous Delivery (CD) is a DevOps practice that refers to the building, testing, and delivering improvements to the software code. The phase is referred to as the extension of the Continuous Integration phase to make sure that new changes can be released to the customers quickly in a substantial manner.

This can be simplified as, though you have automated testing, the release process is also automated, and any deployment can occur at any time with just one click of a button.

Continuous Delivery gives you the power to decide whether to make the releases daily, weekly, or whenever the business requires it. The maximum benefits of Continuous Delivery can only be yielded if they release small batches, which are easy to troubleshoot if any glitch occurs.

The "CD" in CI/CD refers to continuous delivery and/or continuous deployment, which are related concepts that sometimes get used interchangeably. Both are about automating further stages of the pipeline, but they’re sometimes used separately to illustrate just how much automation is happening.

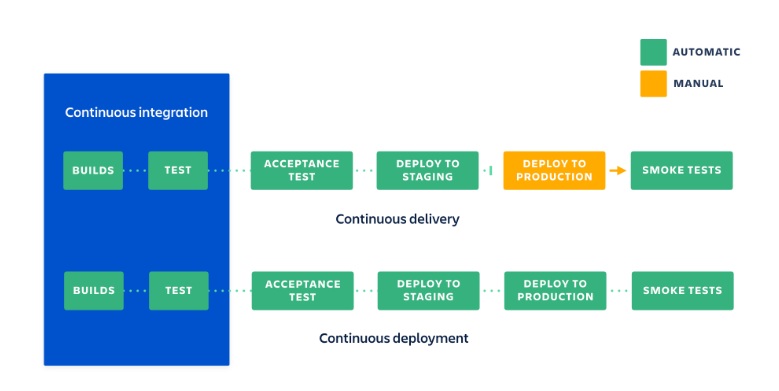
Continuous delivery usually means a developer’s changes to an application are automatically bug tested and uploaded to a repository (like GitHub or a container registry), where they can then be deployed to a live production environment by the operations team



## What is Continuous Deployment?

When the step of Continuous Delivery is extended, it results in the phase of Continuous Deployment. Continuous Deployment (CD) is the final stage in the pipeline that refers to the automatic releasing of any developer changes from the repository to the production.

Continuous Deployment ensures that any change that passes through the stages of production is released to the end-users. There is absolutely no way other than any failure in the test that may stop the deployment of new changes to the output. This step is a great way to speed up the feedback loop with customers and is free from human intervention.



### Continuous integration

#### What you need (cost) :-

* Your team will need to write automated tests for each new feature, improvement or bug fix.
* You need a continuous integration server that can monitor the main repository and run the tests automatically for every new commits pushed.
* Developers need to merge their changes as often as possible, at least once a day.

#### What you gain:-

* Less bugs get shipped to production as regressions are captured early by the automated tests.
* Building the release is easy as all integration issues have been solved early.
* Less context switching as developers are alerted as soon as they break the build and can work on fixing it before they move to another task.
* Testing costs are reduced drastically – your CI server can run hundreds of tests in the matter of seconds.
* Your QA team spends less time testing and can focus on significant improvements to the quality culture.

### Continuous delivery:-

#### What you need (cost)

* You need a strong foundation in continuous integration and your test suite needs to cover enough of your codebase.
* Deployments need to be automated. The trigger is still manual but once a deployment is started there shouldn't be a need for human intervention.
* Your team will most likely need to embrace feature flags so that incomplete features do not affect customers in production.

#### What you gain

* The complexity of deploying software has been taken away. Your team doesn't have to spend days preparing for a release anymore.
* You can release more often, thus accelerating the feedback loop with your customers.
* There is much less pressure on decisions for small changes, hence encouraging iterating faster.

### Continuous deployment

#### What you need (cost)

* Your testing culture needs to be at its best. The quality of your test suite will determine the quality of your releases.
* Your documentation process will need to keep up with the pace of deployments.
* Feature flags become an inherent part of the process of releasing significant changes to make sure you can coordinate with other departments (support, marketing, PR...).

#### What you gain

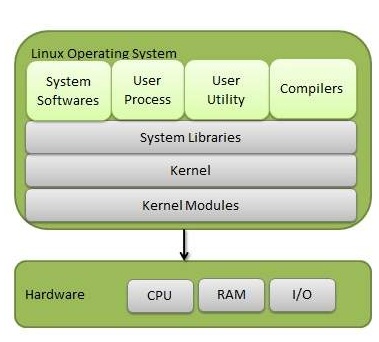
* You can develop faster as there's no need to pause development for releases. Deployments pipelines are triggered automatically for every change.
* Releases are less risky and easier to fix in case of problem as you deploy small batches of changes.
* Customers see a continuous stream of improvements, and quality increases every day, instead of every month, quarter or year.
* **1.14 Linux O.S. Introduction :-**

Linux is one of popular version of UNIX operating System. It is open source as its source code is freely available. It is free to use. Linux was designed considering UNIX compatibility. Its functionality list is quite similar to that of UNIX.

Components of Linux System

Linux Operating System has primarily three components

* **Kernel** − Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.
* **System Library** − System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not requires kernel module's code access rights.
* **System Utility** − System Utility programs are responsible to do specialized, individual level tasks.



**Kernel Mode vs User Mode**

Kernel component code executes in a special privileged mode called **kernel mode** with full access to all resources of the computer. This code represents a single process, executes in single address space and do not require any context switch and hence is very efficient and fast. Kernel runs each processes and provides system services to processes, provides protected access to hardware to processes.

Support code which is not required to run in kernel mode is in System Library. User programs and other system programs works in **User Mode** which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low level tasks.

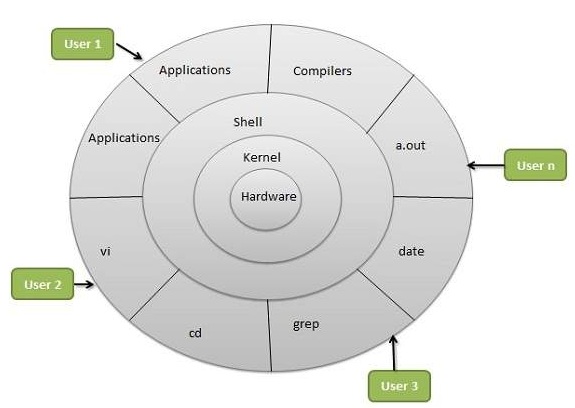
**Basic Features**

Following are some of the important features of Linux Operating System.

* **Portable** − Portability means software can works on different types of hardware in same way. Linux kernel and application programs supports their installation on any kind of hardware platform.
* **Open Source** − Linux source code is freely available and it is community based development project. Multiple teams work in collaboration to enhance the capability of Linux operating system and it is continuously evolving.
* **Multi-User** − Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.
* **Multiprogramming** − Linux is a multiprogramming system means multiple applications can run at same time.
* **Hierarchical File System** − Linux provides a standard file structure in which system files/ user files are arranged.
* **Shell** − Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs. etc.
* **Security** − Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

**Architecture**

The following illustration shows the architecture of a Linux system −



The architecture of a Linux System consists of the following layers −

* **Hardware layer** − Hardware consists of all peripheral devices (RAM/ HDD/ CPU etc).
* **Kernel** − It is the core component of Operating System, interacts directly with hardware, provides low level services to upper layer components.
* **Shell** − An interface to kernel, hiding complexity of kernel's functions from users. The shell takes commands from the user and executes kernel's functions.
* **Utilities** − Utility programs that provide the user most of the functionalities of an operating systems.

**1.15 Importance of Linux in DevOps:-**

Linux and DevOps share a common agenda—scalability. Scalability enables fast software delivery, without forcing developers to compromise the quality of their codebase. Scalability promotes measurable and self-paced growth.

Linux was designed with scalability in mind. It was during the days when UNIX was the operating system in charge. One day, UNIX was closed off for modification, and developers no longer had control over the inner workings of the operating systems. Linux was born out of this need, and today it powers almost all of the technologies you’re using in your everyday life.

Linux is a free open-source Operating System (OS), offered for use under the GNU General Public License (GPL). Linux, like any OS, mediates between the hardware of the machine (CPU, memory, and storage) and its software. The OS manages how the hardware is used to meet the needs of the software.

A Linux-based OS uses a Linux kernel, which is used to manage the hardware resources. A boot loader runs the machine through a start-up sequence, and daemons—background services—work in the background to ensure key functions run smoothly. The OS shell, or command line, receive code instructions from the developer and transmits them to the machine.

Linux offers the DevOps team the flexibility and scalability needed to create a dynamic development process. You can set it up any way that suits your needs. Rather than letting the operating system dictate how you work, you can configure it to work for you. There’s no greater freedom than that, and there is no better asset for DevOps teams then a Linux OS that supports the process at scale.

* **1.16 Linux Basic Command Utilities :**

**1) pwd Command**

The [pwd](https://www.javatpoint.com/linux-pwd) (Present Working Directory ) command is used to display the location of the current working directory.

**2) mkdir Command**

The [mkdir](https://www.javatpoint.com/linux-mkdir) command is used to create a new directory under any directory.

**3) rmdir Command**

The [rmdir](https://www.javatpoint.com/linux-rmdir) command is used to delete a directory.

**4) ls Command**

The [ls](https://www.javatpoint.com/linux-ls) command is used to display a list of content of a directory.

**5) cd Command**

The [cd](https://www.javatpoint.com/linux-cd) command is used to change the current directory.

**6) touch Command**

The [touch](https://www.javatpoint.com/linux-touch) command is used to create empty files. We can create multiple empty files by executing it once.

**7) cat Command**

The [cat](https://www.javatpoint.com/linux-cat) command is a multi-purpose utility in the Linux system. It can be used to create a file, display content of the file, copy the content of one file to another file, and more.

**Syntax:**

cat [OPTION]... [FILE]..

To create a file, execute it as follows:

cat **>** **<file** name**>**

// Enter file content

Press "**CTRL+ D**" keys to save the file. To display the content of the file, execute it as follows:

cat **<file** name**>**

**8) rm Command**

The [rm](https://www.javatpoint.com/linux-rm) command is used to remove a file.

**9)cp Command**

The [cp](https://www.javatpoint.com/linux-cp) command is used to copy a file or directory.

**10) mv Command**

The [mv](https://www.javatpoint.com/linux-mv) command is used to move a file or a directory form one location to another location.

**Syntax:**

mv **<file** name**>** **<directory** path**>**

**11) rename Command**

The [rename](https://www.javatpoint.com/linux-rename) command is used to rename files. It is useful for renaming a large group of files.

**Syntax:**

rename 's/old-name/new-name/' files

For example, to convert all the text files into pdf files, execute the below command:

rename 's/\.txt$/\.pdf/' \*.txt

**12) head Command**

The [head](https://www.javatpoint.com/linux-head) command is used to display the content of a file. It displays the first 10 lines of a file.

**13) tail Command**

The [tail](https://www.javatpoint.com/linux-tail) command is similar to the head command. The difference between both commands is that it displays the last ten lines of the file content. It is useful for reading the error message.

**14) tac Command**

The [tac](https://www.javatpoint.com/linux-tac) command is the reverse of cat command, as its name specified. It displays the file content in reverse order (from the last line).

**15) more command**

The [more](https://www.javatpoint.com/linux-more) command is quite similar to the cat command, as it is used to display the file content in the same way that the cat command does. The only difference between both commands is that, in case of larger files, the more command displays screenful output at a time.

**16) less Command**

The [less](https://www.javatpoint.com/linux-less) command is similar to the more command. It also includes some extra features such as 'adjustment in width and height of the terminal.' Comparatively, the more command cuts the output in the width of the terminal.

**17) su Command**

The [su](https://www.javatpoint.com/linux-su-commands) command provides administrative access to another user. In other words, it allows access of the Linux shell to another user.

**Syntax:**

su **<user** name**>**

**18) cat Command**

The [cat](https://www.javatpoint.com/linux-cat-filters) command is also used as a filter. To filter a file, it is used inside pipes.

**Syntax:** cat **<fileName>** | cat or tac | cat or tac |. . .

**19 )grep Command**

The [grep](https://www.javatpoint.com/linux-grep) is the most powerful and used filter in a Linux system. The 'grep' stands for "**global regular expression print**." It is useful for searching the content from a file. Generally, it is used with the pipe.

**Syntax:** command | grep **<searchWord>**

* **1.17 Linux Administration :-**

Linux is an operating system or a kernel created by *Linus Torvalds with other contributors*. It was first released on September 17, 1991. The main advantage of Linux is that it is distributed under an open-source license means programmers can use the Linux Kernel to design their own custom operating systems. Most of Linux code is written in [C Programming Language](https://www.geeksforgeeks.org/c-programming-language/).

The job of a Linux systems administrator is to manage the operations of a computer system like maintain, enhance, create user account/report, taking backups using Linux tools and command-line interface tools. Most computing devices are powered by Linux because of its high stability, high security, and open-source environment. There are some of the things that a Linux system administrator should know and understand:

* Linux File Systems
* File System Hierarchy
* Managing Root/super User
* Basic Bash Command
* Handling File, Directories and Users

elow are some duties of a Linux Administrator:

* Maintain all internet requests inclusive to DNS, RADIUS, Apache, MySQL, [PHP](https://www.geeksforgeeks.org/php/).
* Taking regular back up of data, create new stored procedures and listing back-up is one of the duties.
* Analyzing all error logs and fixing along with providing excellent customer support for Webhosting, ISP and LAN Customers on troubleshooting increased support troubles.
* Communicating with the staff, vendors, and customers in a cultivated, professional manner at all times has to be one of his characteristics.
* Enhance, maintain and creating the tools for the Linux environment and its users.
* Detecting and solving the service problems ranging from disaster recovery to login problems.
* Installing the necessary systems and security tools. Working with the Data Network Engineer and other personnel/departments to analyze hardware requirements and makes acquiring recommendations.
* Troubleshoot, when the problem occurs in the server.
* **1.18 Linux Environment Variable :**

**Environment variables**or **ENVs**basically define the behavior of the environment. They can affect the processes ongoing or the programs that are executed in the environment.

### Scope of an environment variable

Scope of any variable is the region from which it can be accessed or over which it is defined. An environment variable in Linux can have **global** or **local** scope.

**Global**

A globally scoped ENV that is defined in a terminal can be accessed from anywhere in that particular environment which exists in the terminal. That means it can be used in all kind of scripts, programs or processes running in the environment bound by that terminal.

**Local**

A locally scoped ENV that is defined in a terminal cannot be accessed by any program or process running in the terminal. It can only be accessed by the terminal( in which it was defined) itself.

### How to access ENVs?

**SYNTAX:**   $NAME

**(NOTE:** Both local and global environment variables are accessed in the same way.)

### How to display ENVs?

### To display any ENV

**SYNTAX:**   $ echo $NAME

### To display all the Linux ENVs

**SYNTAX:**

$ printenv //displays all the global ENVs

or

$ set //display all the ENVs(global as well as local)

or

$ env //display all the global ENVs

### How to set environment variables?

**To set a global ENV**

$ export NAME=Value

or

$ set NAME=Value

### To set a local ENV

**SYNTAX:**

$ NAME=Value

### Some commonly used ENVs in Linux

**$USER:** Gives current user's name.

**$PATH:** Gives search path for commands.

**$PWD:** Gives the path of present working directory.

**$HOME:** Gives path of home directory.

**$HOSTNAME:** Gives name of the host.

**$LANG:** Gives the default system language.

**$EDITOR:** Gives default file editor.

**$UID:** Gives user ID of current user.

**$SHELL:** Gives location of current user's shell program.

* **1.19 Networking:-**

Every computer is connected to some other computer through a network whether internally or externally to exchange some information. This network can be small as some computers connected in your home or office, or can be large or complicated as in large University or the entire Internet.

Maintaining a system's network is a task of System/Network administrator. Their task includes network configuration and troubleshooting.

Here is a list of Networking and Troubleshooting commands:

|  |  |
| --- | --- |
| [ifconfig](https://www.javatpoint.com/linux-ifconfig) | Display and manipulate route and network interfaces. |
| [ip](https://www.javatpoint.com/linux-ip) | It is a replacement of ifconfig command. |
| [traceroute](https://www.javatpoint.com/linux-traceroute) | Network troubleshooting utility. |
| [tracepath](https://www.javatpoint.com/linux-tracepath) | Similar to traceroute but doesn't require root privileges. |
| [ping](https://www.javatpoint.com/linux-ping) | To check connectivity between two nodes. |
| [netstat](https://www.javatpoint.com/linux-netstat) | Display connection information. |
| [ss](https://www.javatpoint.com/linux-ss) | It is a replacement of netstat. |
| [dig](https://www.javatpoint.com/linux-dig) | Query DNS related information. |
| [nslookup](https://www.javatpoint.com/linux-nslookup) | Find DNS related query. |
| [route](https://www.javatpoint.com/linux-route) | Shows and manipulate IP routing table. |
| [host](https://www.javatpoint.com/linux-host) | Performs DNS lookups. |
| [arp](https://www.javatpoint.com/linux-arp) | View or add contents of the kernel's ARP table. |
| [iwconfig](https://www.javatpoint.com/linux-iwconfig) | Used to configure wireless network interface. |
| [hostname](https://www.javatpoint.com/linux-hostname) | To identify a network name. |
| [curl or wget](https://www.javatpoint.com/linux-curl-and-wget) | To download a file from internet. |
| [mtr](https://www.javatpoint.com/linux-mtr) | Combines ping and tracepath into a single command. |
| [whois](https://www.javatpoint.com/linux-whois) | Will tell you about the website's whois. |
| [ifplugstatus](https://www.javatpoint.com/linux-ifplugstatus) | Tells whether a cable is plugged in or not. |

* **1.20 Linux Server Installation :-**

### To create a user account on Linux and assign the required permissions

1. Log on as **root**.
2. Create the BCAN\_USER account.
3. Assign a password to the BCAN\_USER account. Make a note of the group to which the user account belongs. You would be asked for this group name by the installer.  
   (It recommends not using the at sign (@) in the password because some device file transfers might fail because they use the user:password@host/file format. So if the password contains an at sign, the file transfer thinks all characters after the at sign is the host.)
4. Create a home directory that is owned and writable by that user.
5. Ensure the shell associated with the BCAN\_USER account is the Bourne shell (**/bin/sh**). Also ensure the entry in **/etc/passwd** is similar to the following example:  
   bcan:x:800:800:BCA-Networks User:/export/home/bcan:/bin/sh
6. Execute the umask 022 command.  
   This ensures that files and directories created by the installer are usable by the BCAN\_USER account.
7. Endure that the BCAN\_USER account has access to the FTP and SCP directories.

## Checking required disk space on a Linux server

Perform the following steps to check the required disk space on a Linux server:

1. Confirm that the drive where you plan to install the BMC Network Automation application server has at least 1.2 GB of free disk storage. Do not install the software on a network or NFS mounted drive. You must install the software on a local drive.
2. Confirm that the **/tmp** directory has at least 450 MB of free space.
3. If the **/tmp** directory does not have sufficient free space, you can run the following commands before installing to make the installation use an alternate temporary directory. The directory specified in this command must have at least 200 MB of free space available.  
   $ IATEMPDIR=/your/free/space/tmp\_directory  
   $ export IATEMPDIR
4. Verify that the server host computer has sufficient disk space, as discussed in [System requirements](https://docs.bmc.com/docs/NetworkAutomation/89/planning/system-requirements).  
   The installation uses **/var/bca-networks-data** as the default for the location of the dynamic data for the application.  
   As an alternative, you can install the **BCA-Networks-Data** directory into a separate partition that meets the disk space requirements.
5. Configure the following permissions:

| **Directory** | **Permissions** | **Octal Value** |
| --- | --- | --- |
| Installation directory | rwx rwx rwx | 777 |
| Temp or alternate directory | rwx rwx rwx | 777 |
| Parent directories of the temp or alternate directory | rwx rwx r-x | 775 |

## Confirming installation of FTP or SCP

Ensure that the Trivial File Transfer Protocol (TFTP), FTP, and Secure Shell (SSH) or Secure Copy (SCP) services are installed, configured, and enabled (depending on which file transfer protocols you plan to use between the server and your network devices).  
  
You can configure the location of the TFTP, FTP, and SCP directories later in the Device Agent Editor, **Admin > Device Agents**.  
  
For TFTP, verify that the directory (usually **/tftpboot** or **/var/lib/tftpboot**) has user, group, and world read, write, and execute permissions.

Configuring the hosts file

Confirm that the **/etc/hosts** file has at least two entries.

* Assign 127.0.0.1 to localhost.

**(Warning** If 127.0.0.1 is not assigned to localhost, installation would fail.)

* Assign the static IP address of the server to the server hostname.
* 127.0.0.1 localhost

10.1.2.3 ena-server-01

## Library requirements

This section discusses the library requirements.

### Confirming the availability of libXtst.so and libXrender.so

If you are installing on a 64-bit version of Linux, confirm that both 32-bit and 64-bit version of the following libraries are available:

* **libXtst.so**: Confirm that the library is available by executing the following command:

*locate libXtst.so*

* If the library is not available and **yum** is configured on a Red Hat Linux system, you can install 32-bit and 64-bit version of the library by using the following command:

*yum install libXtst libXtst.i686*

* **libXrender.so**: Confirm that the library is available by executing the following command:  
    
  locate libXrender.so
* If the library is not available and **yum** is configured on a Red Hat Linux system, you can install 32-bit and 64-bit version of the library by using the following command:
* yum install libXrender libXrender.i686

### Other library requirements

Ensure that the **nslookup** library is available by executing the following command:

*locate nslookup*

If not installed, and **yum** is configured on a Red Hat Linux system, then install the library by executing the following command:

*yum install nslookup*

* **1.21 RPM and YUM Installation:-**

**RPM** Package Manager (also known as RPM), originally called the **Red-hat Package Manager**, is an open source program for installing, uninstalling, and managing software packages in Linux. RPM was developed on the basis of the Linux Standard Base (LSB).

**YUM** is the primary package management tool for installing, updating, removing, and managing software packages in Red Hat Enterprise Linux. YUM performs dependency resolution when installing, updating, and removing software packages. **YUM** can manage packages from installed repositories in the system or from

**Yum** is a free and open-source command-line package-management application for Linux operating systems that uses the RPM Package Manager.

**Yum** is a front-end tool for rpm that automatically solves dependencies for packages.

**Yum** allows you to install, update, search and remove packages from your system.

If you want to keep your system up-to-date, you can enable automatic updates via yum-cron.

**RPM** keeps an internal database of the installed packages and allows you to later manage the installed packages using the package name. On the other hand, installing a package with YUM only requires the package name, and doesn't require the package location.

Package manager or package management system is a collection of software tools that automate the process of installing, upgrading, configuring, and removing software in the Linux system.

RPM refers to RPM Package Manager (formerly known as Red Hat Package Manager) is a powerful, command-line package management tool developed for the Red Hat operating system.

RPM files comes with the ***.rpm*** extension. The RPM package consists of an archive file, that contains libraries and dependencies for a specific package, that do not conflict with other packages installed on your system.

### Install RPM File Using RPM Command

Use the command ***yum localinstall /path/to/file.rpm***. This command will install the local rpm file as well as searching for required rpms (dependencies, etc) on RHN or other repositories that are configured and install it for the user.

To install a .rpm package in CentOS Linux, enter the following:

sudo rpm -i sample\_file.rpm

The -i switch tells the package manager you want to install the file.

### Install RPM File with Yum

Alternately, you can use the yum package manager to install .rpm files.

Enter the following:

sudo yum localinstall sample\_file.rpm

The localinstall option instructions yum to look at your current working directory for the installation file.