Devops:

The Devops is the combination of two words, one is Development and other is operations.

what is Devops?

The Devops is a combination of two words, one is Software Development, and second is operations. This allows a single team to handle the entire application lifecycle, from development to testing, development, and operations.

Devops helps you to reduce the disconnection between software developers, quality assurance (QA) engineers, and system administrators.

Devops promotes collaboration between development and operations team to deploy code to production faster in an automated & repeatable way.

* Devops helps to increases organization speed to deliver applications and services. It also allows organizations to serve their customers better and compete more strongly in the marked.
* Devops can also be defined as a sequence of development and IT operations with better communication and collaboration
* Devops has become one of the most valuable business disciplines for enterprises or organizations

with the help of Devops. quality, and speed of the application delivery has improved to great extent.

* faster delivery
* Higher quality
* Lesser spending
* Always available.

Development
Developers
Build
Test
QA

operations
Deployment
Maintenance
Monitoring

Why Devops?

* The operation and development team worked in complete isolation
* after the design - build, the testing and deployment are performed respectively. That's why they consumed more time than actual build cycles.
* without the use of Devops, the team members are spending a large amount of time on designing, testing, and deploying instead of building the project.
* Manual code deployment leads to human errors in production
* coding and operation teams have their separate timelines and are not in synch, causing further delays.

History:

In 2009, the first conference named. Devops days 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 forgen, Genekim, and Jez Humble founded (Devops Research and Assignment). DORA

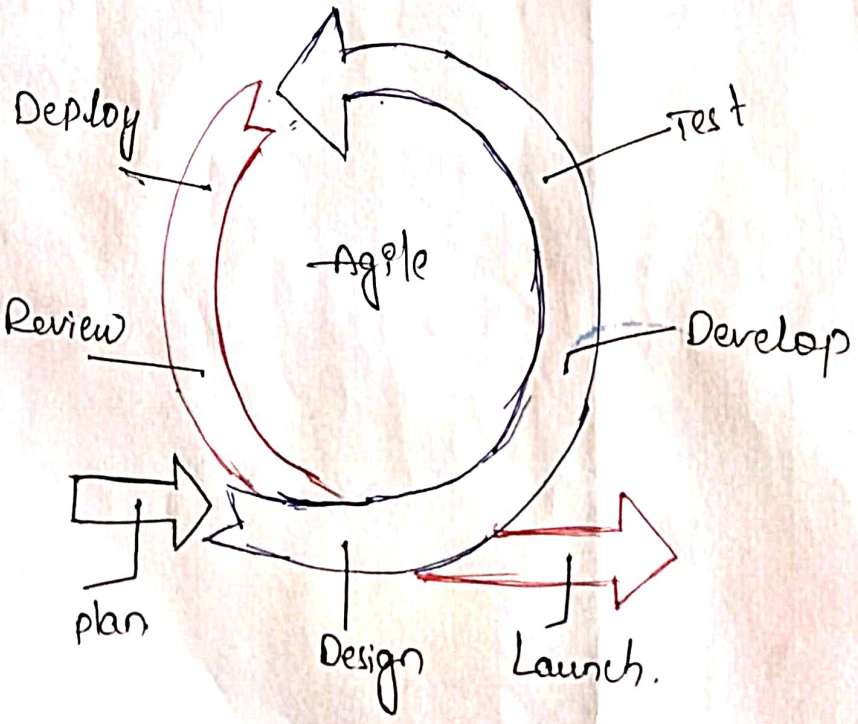
* In 2017, Nicole forgen, Genekim, and Jez Humble published. "Accelerate: Building and scaling High Performing Technology Organizations".

* Agile development model:

-Agile development model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.

-Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing and acceptance testing.

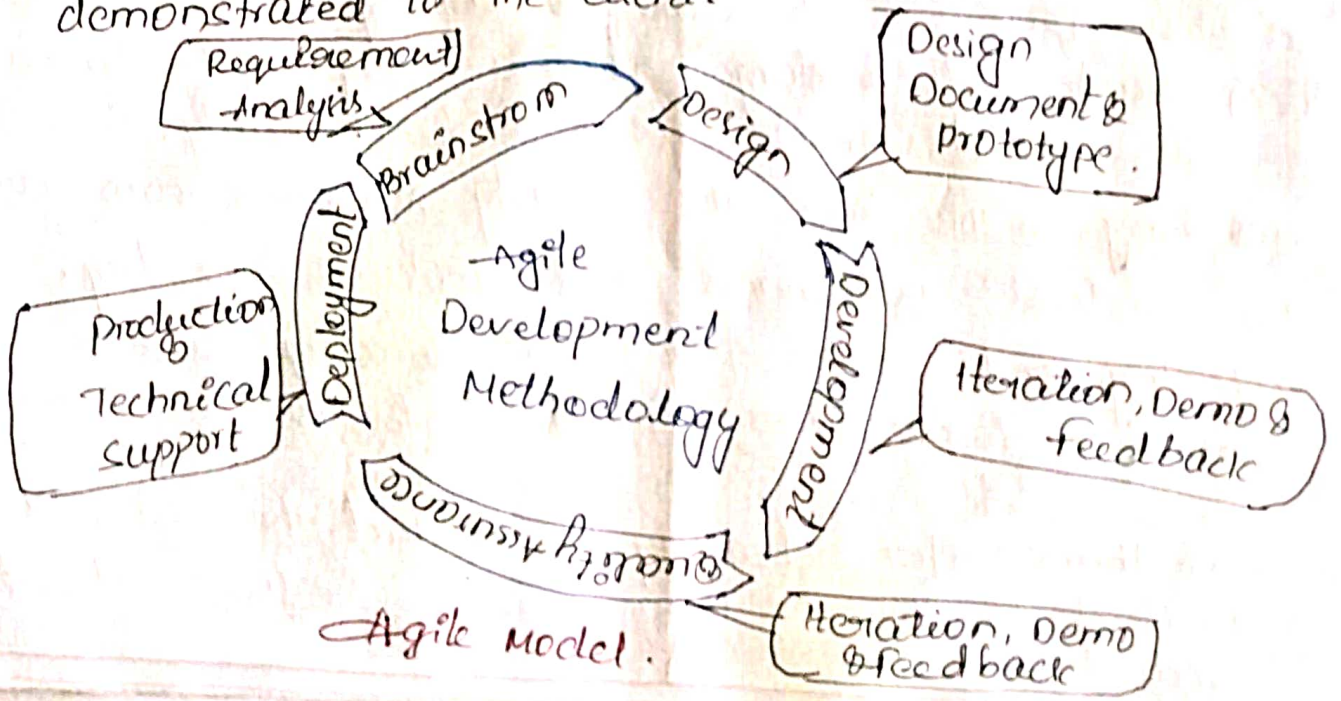
At the end of the iteration a working product is displayed to the customer and important stakeholders.



Agile development model:

The meaning of Agile is swift or versatile. "Agile process model" refers to a software development approach based on iterative development. Agile methods breaks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

Each iteration is considered as a short time "frame" in the agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and reduce the overall project delivery time requirem-ents. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.



Agile Model.

phases of Agile model

following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirement
3. construction / iteration
4. Testing / quality assurance.
5. Deployment
6. feedback.

1. Requirements gathering: in this phase, you must define the requirements. you should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.
2. Design the requirements: when you have identified the project, work with stakeholders to define requirements. you can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.
3. construction/iteration: when the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement so it includes simple, minimal functionality.
4. Testing: In this phase, the Quality Assurance team examines the product's performance and looks for the bug.
5. Deployment: in this phase, the ~~quality/assurance~~ team examines team issues a product for the user's work environment.
6. feedback: After releasing the product, the last step is feedback. in this, the team receives feedback about the product and work through the feedback.

Agile Testing Methods:

- Scrum
- Crystal
- Dynamic software Development Method (DSDM).
- feature Driven Development (FDD)
- Lean software Development
- extreme programming (xp)

Scrum

SCRUM is an agile development process focused primarily on ways to manage tasks in team-based development conditions.

There are three roles in it, and their responsibilities are:

Scrum Master: the Scrum can set up the master team, arrange the meeting and remove obstacles for the process.

Product owner: The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.

Scrum Team: the team manges its work and organizes the work to complete the sprint or cycle.

extreme programming:

This type of methodology is used when customers are constantly changing demands or requirements, or when they are not sure about the system's performance.

Crystal:

These are three concepts of this method:

1 charging: Multi activities are involved in this phase such as making a development team, performing.

-feasibility analysis, developing plans, etc.

2. cyclic delivery: under this, two more cycles consist these are
   A. Team updates the release plan.
   B. Integrated product delivers to the users.

3. wrap up: According to the user environment, this phase performs deployment, post-deployment.

Dynamic Software Development Method (DSDM):

DSDM is a rapid application development strategy for software development and gives an agile project distribution structure, the essential features of DSDM are that users must be actively connected and teams have been given the right to make decisions. The techniques used in DSDM are:

1. Time Boxing
2. Moscow Rules → must-have, should-have, could-have, won't-have.
3. prototyping

The DSDM project contains seven stages:

1. Pre-project 2. Feasibility study 3. Business study 4. functional model Iteration 5. Design and build Iteration 6. Implementation 7. post-project.

Feature Driven Development (FDD)

This method focuses on "Designing and Building" features. In contrast to other smart methods, FDD describes the small of the work that should be obtained separately per function.

Lean Software Development:

Lean software development methodology follows the principle "just in time production." the lean method indicates the increasing speed of software development and reducing cost. Lean development can be summarized in seven phases.

1. Eliminating waste 2. Amplifying learning 3. Defer commitment (deciding as late as possible) 4. Early delivery 5. Empowering the team

feedback.

6. Building integrity
7. optimize the whole

The twelve agile Manifesto principles

The Twelve principles are the guiding principles for the methodologies that are included under the title "The agile Movement" they describe a culture in which change is welcome, and the customer is the focus of the work. They also demonstrate the movement's intent as described by Alistair Cockburn, one of the signatories to the agile Manifesto, which is to bring development into alignment with business needs.

* the twelve principles of agile development model:

1. Customer satisfaction through early and continuous software delivery - Customers are happier when they receive working software at regular intervals, rather than waiting extended periods of time between releases.
2. Accommodate changing requirements throught the development process - The ability to avoid delays when a requirement or features request changes.
3. frequent delivery of working software - Scrum accommo-dates this principle since the team operates in software sprints or iterations that ensure regular delivery of working software.
4. collaboration between the business stakeholders and developers throughout the project - Better decisions are made when the business and technical team are aligned
5. Support, trust and motivate the people involved - Motivated teams are more likely to deliver their best work than unhappy teams.
6. Enable face-to-face interactions - communication is more successful when development teams are co-located

7. Working software is the primary measure of progress - Delivering functional software to the customer is the ultimate factor that measures progress.

8. Agile processes to support a consistent development pace - Teams establish a repeatable and maintainable speed at which they can deliver working software, and they repeat it with each release.

9. Attention to technical detail and design enhances agility - The right skills and good design ensure the team can maintain the pace, constantly improve the product, and sustain change.

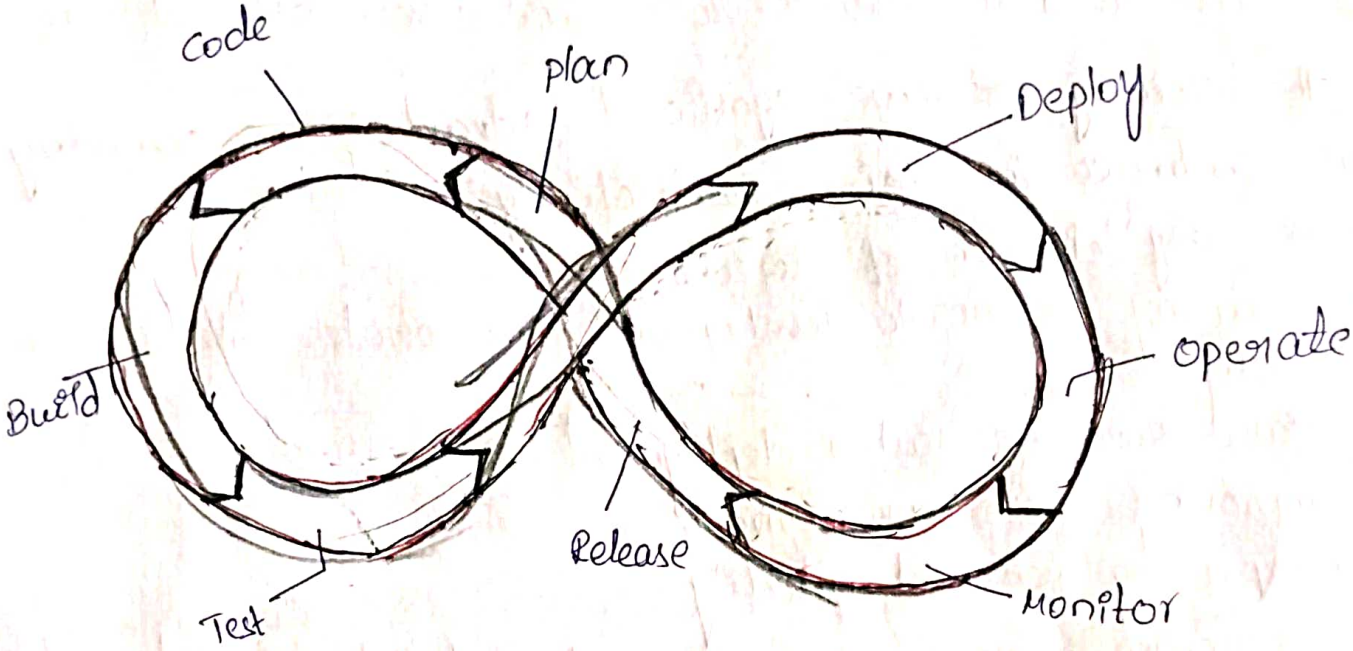
10. Simplicity - Develop just enough to get the job done for right now.

11. Self-organizing teams encourage great architectures, requirements, and designs - skilled and motivated team can members who have decision-making power, take ownership, communicate regularly with other team members, and share ideas that deliver quality products.

12. Regular reflections on how to become more effective - Self-improvement, process improvement, process improvement, advancing skills, and techniques help team members work more efficiently.

* The intention of Agile is to align development with business needs, and the success of Agile is apparent. Agile projects are customer focused and encourage customer guidance and participation. As a result, Agile has grown to be an overarching view of software development throughout the software industry all by itself.

Devops



DevOps is nothing but a combination of development and operation engineers involved together to improve work throughout the SDLC. using Devops tools increase a company's ability to deliver services and applications at high speed. It delivers products faster than companies using infrastructure management processes and traditional software development. This speed allows enterprises to serve their client requirements better and compete efficiently in the current market.

Devops is the solution to build better software in a quick time. Devops practices permit Dev and ops teams to accelerate delivery through fast feedback, automation, and iterative improvement. Devops's goal was to increase the maintainability, predicatability, and efficiency of the operational processes. It was build to address the waterfall method's inefficiencies.

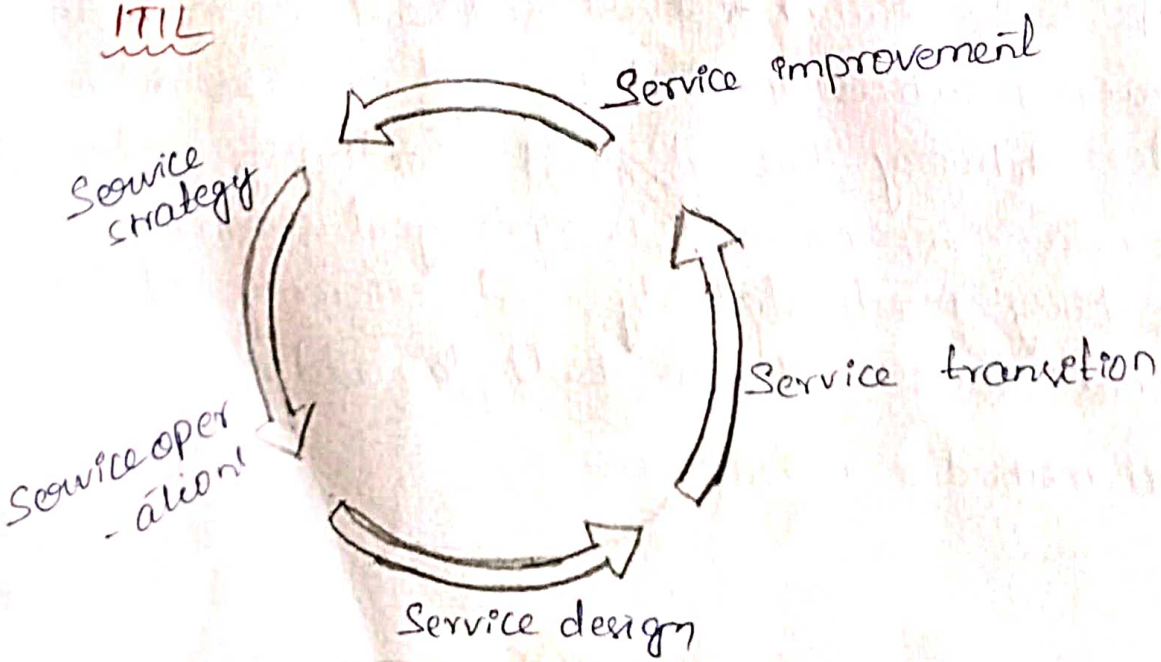
Devops advantages:

How can a business stay ahead in the competitive market and offer the best features to the end users in the set time?

The following are some significant advantages a company can embrace in their productivity:

* The early detection of defections
* Enabling teams to develop and incorporate code continuously.
* Make sure of faster deployment
* Improving work environments.
* Improving product quality
* Allowing more room for innovation due to the automation of repetitive tasks.
* promotes agility in your business to stay ahead in the market
* continuous software delivery
* Speed delivery of features
* Reduce complexity to manage
* Accelerate time to resolution.
* Reduce the cost of production
* Delivers high productivity.

ITIL



- Service Stradegy
  - Strategy Management
  - Service portfolio Management
  - financial Management
  - Demand Management
  - Business Relation-ship Management
- Service Design
  - Design coordination
  - Service catalog Level Management
  - Service catalog Management
  - Capacity Management
  - Availability Management
  - information Security Management
  - Service continuity Management
  - Supplier Management
- Service Organization
  - Transition planning and support
  - change Management
  - Service Asset & configuration Management
  - Release and Deployment Management
  - Knowledge management
- Service operation
  - Event Management
  - Incident Management
  - Request-fulfilment
  - Problem Management
  - Access Management
- Continual service improvement
  - Deming Cycle
  - CSI Model
  - 7-step improvement Process.

DevOps advantages.

1) Service strategy: All managers follow instructions to develop a service strategy that ensures the company can manage all associated costs and risks. There are multiple roles involved in service strategy, and they can define as follows.

A) Business relationship manager
B) Finance manager.
C) IT steering group(ISG)
D) Demand manager
E) Service strategy manager.
F) Service portfolio Manager.

2) Service operation: Service operation includes technical support teams and application management that respond whenan issue has an impact on the business.

3) Service design: It involves when the service's architecture is developed, and the business needs are translated into technical requirements.

4) Service transition: In this stage, all assets are controlled. to deliver a complete service for testing and integration

5) Service improvement: It is a reflective approach that involves four stages to check the services are always. in line with the demands of the business.

ITIL advantages:

Similar to ITSM, the ITIL model boosts effectiveness and efficiency. when IT teams implement this model in their workflow, it allows them into increases the potential of their IT services and fix compliance issues.

* Some advantages of ITIL include:

* Enhancing learning of IT resources and costs
* The improvement of reactions to shift business needs
* Enhancing customer relationships
* Establishing a secure IT environment that promote the development
* Boosting the efficiency of resource use.
* implementing continual procedures
* promoting continuous growth
* Improved alignment between the IT and enterprise.
* Enhanced service delivery and quality of service
* Reduced risk.

ITIL Tools

There are two leading ITIL tools available in the current market.

1. Solarwinds web Help Desk (WHD): Web Help Desk is an excellent piece of software for change management, asset management and ticketing, it allows WHD to create tech tickets from service request emails to save time. It combines WHD with multiple third-party tools such as JAMF casper suite and Microsoft SCCM. SLA reminder alerts to make sure that you never again forget about the Service-level agreements. you may also set up WHD to create user surveys automatically once their service request has been addressed.

2. Solarwinds Service Desk:

It is full coverage of ITSM solutions. It enables ITIL services such as employee service portals, problem management, incident management, and asset management. Solarwinds Service Desk streamlines the ticketing process by permitting seamless employee communication.

Each company uniquely runs ITIL-aligned solutions. it allows you to determine your employees' requirements and fulfill those needs by visualizing the entire issue lifecycle.

Comparision between Devops and ITIL:

Let's have a look at the different parameters between Devopand ITIL

Parameters	Devops	ITIL
Role	the roles involved in Devops are 1) Software developer 2) The release manager 3) The Devops evangelist 4) The automation architect 5) Security professional 6) Assurance professional	There are different roles involved in ITIL, and They are 1) financeal manager 2) Business relationship manager 3) IT steering group (ISG) 4) Demand manager 5) Service strategy manager.
Delivery	Devops aims to provide for better coordination and quicker delivery between developers and the production team	ITIL takes a unique approach to software delivery and aims to combine the delivery process into different business process.
Lifecycle	Devops lifecycle involves 1) Development 2) Testing 3) Integration 4) Deployment and 5) Monitoring	ITIL Service lifecycle involves 1) Service strategy 2) Service Design 3) Service Transition 4) Service Operation and 5) Continual Service improvement.

Parameters	Devops	ITIL
change Management	In Devops, all changes are anticipated at the starting stage of the development life cycle	ITIL provides a set of ITSM best practices, including change management in order to identify service management challenges.
Goal	Devops's goal is to improve coordination and collaboration by developing a better working relationship between the Dev team and ops Team.	ITIL's goal is to standardize a company's ITSM structure for the quick and successful delivery of IT services.
Approach	It uses a methodical approach to reduce conflict between two teams.	ITIL uses a systematic model to manage the IT services capping it off.
Services	CI/CD are hard enough to increase	ITIL services are created, tested, and implemented.

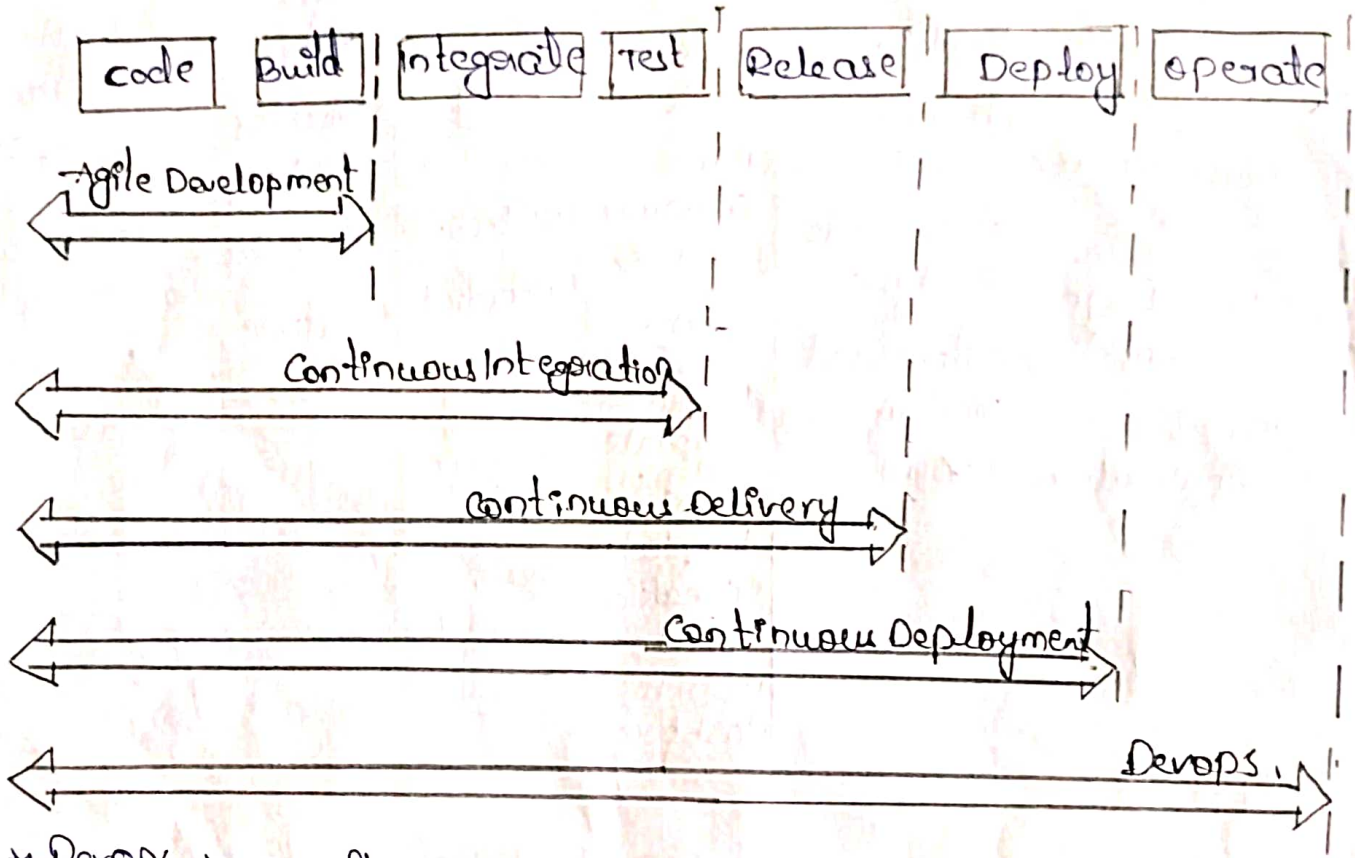
Devops and ITIL can involve in the work when a company encourages a cultural change where every professional is open, collaborative, and work towards the same goal., ITIL use to develop standardizations both inside and outside the company, whereas Devops leverages human resources and promotes creativity.

It guides in pushing the quick release of new updates to the customer. It is better to implement both methodologies that offer something unique and provide

you with a beginning point to improve work and operations faster and better.

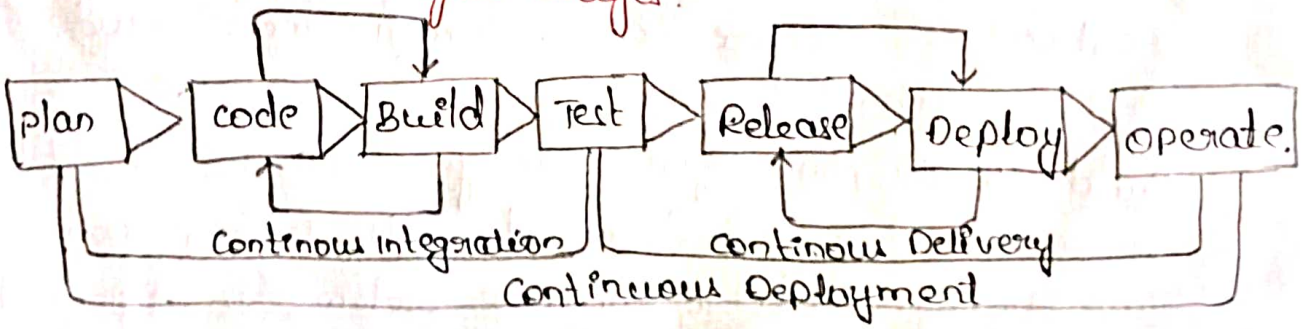
- Adopting Devops and ITIL methodologies is not a big deal, but marking it successful matters.

Devops process and continuous Delivery:



* Devops is a software development approach which involves continuous integration, continuous testing, continuous delivery, continuous deployment, and continuous monitoring of the software throughout its development lifecycle.
* this is the process adopted by all the top companies to develop high-quality software and shorter development lifecycles, resulting in greater customer satisfaction, something that every company wants

Various Devops lifecycle stages:



Planning	Code	Build	test	Release	Deploy
* Requirement finalization * updates & new changes * Architecture Design * task assign -ment * Timeline finalization	* Development * configuration finalization * check-in-source code * static-code analysis * Automated review & peer review	* compile code * unit testing * code-metrics * Build container images or packages * Preparation or update in deploy -ment templated * create or update monitor dashboards	* Integration test with other component * Loads & stress test * UI testing * penetration testing * Requirement testing.	* preparing release notes * version testing * code freeze * feature freeze.	* updating the infrastructure i.e staging, Production * verification on deployment i.e smoke tests.

operate:

* Monitor designed dashboard
* Alarm triggers
* Automatic critical event handler
* Monitor error logs

plan: This phase helps define business value and requirements. Sample tools include Jira or Git to help track known issues and perform project management.

Code: This phase involves software design and the creation of software code. Sample tools include GitHub, GitLab, Bitbucket, or stash.

Build: In this phase, you manage software builds and versions, and use automated tools to help compile and package code for future release to production.

use source code repositories or package repositories that also "package" infrastructure needed for product release. Sample tools include Docker, Ansible, puppet, chef, Gradle, Maven, or Jfrog ~~artifactory~~.

Test: This phase involves continuous testing (manual or automated) to ensure optimal code quality. Sample tools include JUnit, CodeCaption, Selenium, Vagrant, TestNG, or Blaze Meter.

Deploy: This phase can include tools that help manage, coordinate, schedule, and automate product releases into production. Sample tools include puppet, chef, Ansible, Jenkins, kubernetes, openshift, openstack, Docker, or Jira.

Operate: This phase manages software during production. Sample tools include Ansible, puppet, Powershell, chef, salt, or otter.

Monitor: This phase involves identifying and collecting information about issues from a specific software release in production. Sample tools include New Relic, Datadog, Grafana, wireshark, splunk, Nagios, or slack.

Release Management:

Release management is the process of overseeing the planning, scheduling, and controlling of software builds throughteach stage of development and across various environments. Release management typically included the testing and deployment of software releases as well.

Release management has had an important role in the software development lifecycle since before it was known as release management. Deciding when and how to release updates was its own unique problem even when software saw physical disc releases with updates occuring as seldom as every few years.

Now that most software has moved from hard and fast release dates to the software as a service (SaaS) business model, release management has become a constant process that works alongside development. This is especially true for businesses that have converted to utilizing continuous delivery pipelines that see new releases occuring at blistering rates. Devops now plays a large role in many of the duties that were originally considered to be under the preview of release management roles; however, Devops has not resulted in the obsolescence of release management.

Advantages of Release Management for Devops:

With the transition to Devops practices, deployment duties have shifted onto the shoulders of the Devops teams. This doesn't remove the need for release management. Instead, it modifies the data points that matter most to the new role release management performs.

Release management acts as a method for filling the data gap in Devops. The planning of implementation and rollback safety nets is part of Devopsworld. but release management still needs to keep tabs on applications, its components, and the promotion schedule as part of change orders. The key to managing software releases in a way that keeps pace with Devops deployment schedules is through automated management tools.

*Aligning business & IT goals:

The modern business is under more pressure than ever to continuously deliver new features and boost their value to customers. Buyers have come to expect that their software evolves and continues to develop innovative ways to meet their needs. Businesses create an outside perspective to glean insights into their customer needs however, it has to have an inside perspective to develop these features.

Release management provides a critical bridge between these two gaps in perspective. it coordinates between IT work and business goals to maximize the success of each release. Release management balances customer desires with development work to deliver the greatest value to user.

Minimizes organizational risk:

Software products contain millions of interconnected parts that create an enormous risk of failure. users are often affected differently by bugs depending on their other Software applications, and governance tools. plus, faster

deployments to production increase the overall risk that faulty code and bugs slip through the cracks.

Release management minimizes the risk of failure by employing various strategies. Testing and governance can catch critical faulty sections of code before they reach the customers. Deployment plans ensure there are enough team members and resources to address any potential issues before affecting users. All dependencies between the millions of inter connected parts are recognized and understood.

Direct accelerating (speed) change:

Release management is foundational to the discipline and skill of continuously producing enterprise-quality software. The rate of software delivery continues to accelerate and is unlikely to slow down anytime soon. The speed of changes makes release management more necessary than ever.

The move towards CI/CD and increases in automation ensure that the acceleration will only increase. However, it also means increased risk, unmet governance requirements, and potential disorder. Release management helps promote a culture of excellence to scale DevOps to an organizational level.

Release Management best practices:

As Devops increases and changes accelerate, it is critical to have best practices in place to ensure that it moves as quickly as possible. well-refined processes enable Devops teams to more effectively and efficiently. some best practices to improve your processes include.

Define clear criteria for success:

well-defined requirements in releases and testing will create more dependable releases. Everyone should clearly understand when things are actually ready to ship.

well-defined means that the criteria cannot be subjective. Any subjective criteria will keep you from learning from mistakes and refining your release management process to identify what works best. It also needs to be defined for every team member. Release managers, quality supervisors, product vendors, and product owners must all have an agreed-upon set of criteria before starting a project.

Minimize downtime:

Devops is about creating an ideal customer experience. Likewise, the goal of release management is to minimize the amount of disruption that customers feel with updates.

strive more to consistently reduce customer impact and downtime with active monitoring, proactive testing, and real-time collaborative alerts that enable you to quickly notify you of issues during a release.

A good release manager will be able to identify any problems before the customer.

Rel ... it allows deployments to production increase the overall risk that the team can resolve incidents quickly and experience a successful release when proactive efforts are combined with a collaborative response plan.

optimize your staging environment:

The staging environment requires constant upkeep. Maintaining an environment that is as close as possible to your production one ensures smoother and more successful releases. From QA to product owners, the whole team must maintain the staging environment by running tests and combing through staging to find potential issues with deployment. identifying problems in staging before deploying to production is only possible with the right staging environment.

Maintaining a staging environment that is as close as possible to production will enable DevOps teams to confirm that all releases will meet acceptance criteria more quickly.

Strive for immutable:

Whenever possible, aim to create new updates as opposed to modifying new ones. immutable programming drives teams to build entirely new configurations instead of changing existing structures. These new updates reduce the risk of bugs and errors that typically happen when modifying current configurations.

The inherently reliable releases will result in more satisfied customers and employees.

Keep detailed records:

Goods records management on any release/deployment artifacts is critical. form release notes to binaries to compilation of known errors, records are vital for reproducing entire sets of assets. in most cases, tacit knowledge is required.

focus on the team:

well-defined and implemented Devops procedures will usually create a more effective release management structure. they enable best practices for testing and cooperation during the complete delivery lifecycle.

Although automation is a critical aspect of Devops and release management, it aims to enhance team productivity. The more that release management and Devops focus on decreasing human error and improving operational efficiency. the more they'll start to quickly release dependable services.

Automation & release management tools:

Automated release management tools provide end-to-end visibility for tracking application development, quality assurance, and production form a central hub. Release managers can monitor how everything within the system fits together which provides a deeper insight into the changes made and the reasons behind them. This empowers collaboration by providing everyone with detailed updates on the software's position in the current lifecycle which allows for the constant.

Rel deployments to production ... it allows improvement of processes. The strength of automated release management tools is in their visibility and usability - many of which can be accessed through web-based portals.

Powerful release management tools make use of smart automation that ensures continuous integration which enhances the efficiency of continuous delivery pipelines. This allows for the steady deployment of stable and complex applications.

Scrum:

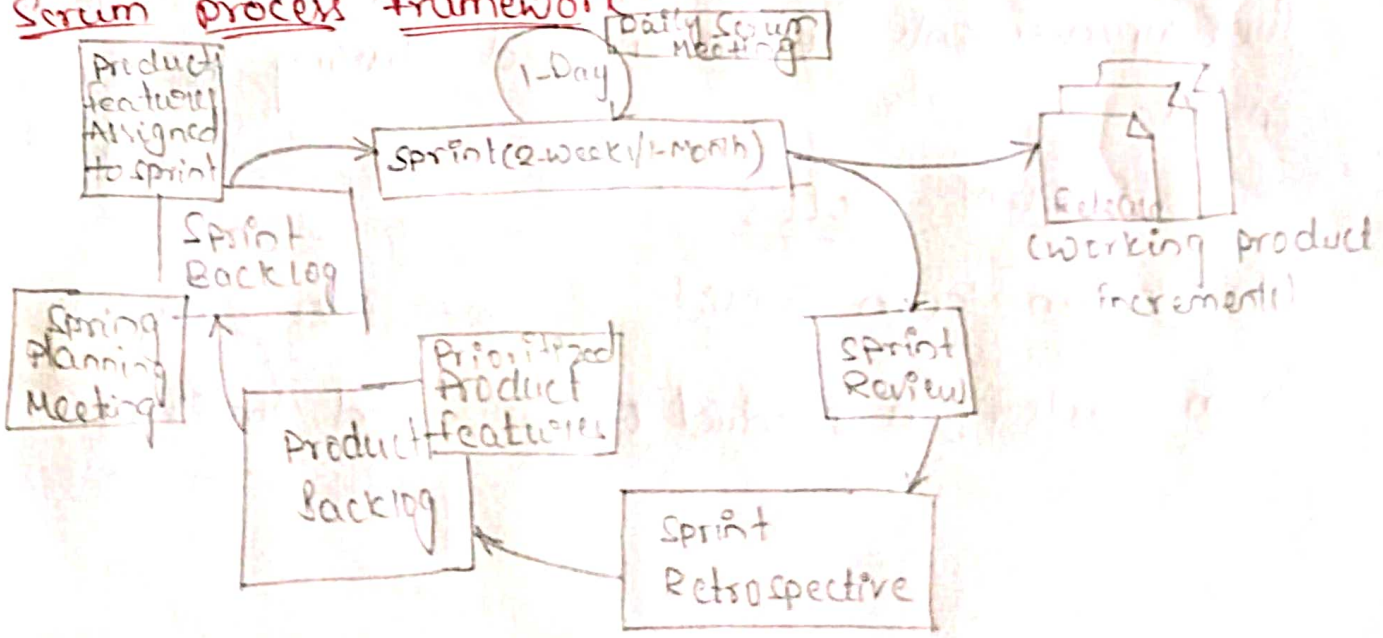
Scrum is a framework within which people can address complex problems, while productively and creatively delivering products of the highest possible value.

Scrum is a process framework that has been used to manage complex product development since the early 1990s. Scrum is not a process or a technique for building products; rather, it is a framework within which you can employ various processes and techniques. Scrum makes clear the relative efficacy of your product management and development practices so that you can improve.

The Scrum framework consists of Scrum Teams and their associated roles, events, artifacts, and rules. Each component within the framework serves a specific purpose and is essential to Scrum's success and usage.

The rules of Scrum bind together the events, roles, and artifacts, governing the relationships and interaction between them. The rules of Scrum are described throughout this topic.

Scrum process framework:



In Scrum the prescribed events are used to create regularity. All events are time-boxed events, such that every event has a maximum duration. The events are described more elaborately in the subsequent chapters.

Sprint: The heart of Scrum is a sprint, a time-box of two weeks or one month during which a potentially releasable product increment is created. A new sprint starts immediately after the conclusion of the previous sprint. Sprints consist of the Sprint planning, daily scrums, the development work, the Sprint review, and the sprint retrospective.

* In sprint planning, the work to be performed in the Sprint is planned collaboratively by the Scrum Team.
* The Daily Scrum Meeting is a 15-minute time-boxed event for the Scrum Team to synchronize the activities and create a plan for that day.
* A Sprint Review is held at the end of the sprint to inspect the increment and make changes to the Product Backlog, if needed.
* The sprint Retrospective occurs after the sprint Review and prior to the next sprint planning. In this meeting, the Scrum Team is to inspect itself and create a plan for improvements to be enacted during the subsequent sprint.

SCRUM - ROLES

The scrum team consists of three roles, namely a Scrum Master, a Product Owner, and the Team.

Scrum Master:

The scrummaster sometimes written as the scrum master, although the official term has no space after "scrum" is the keeper of the scrum process. He/she is responsible for.

* making the process run smoothly
* removing obstacles that impact productivity.
* organizing and facilitating the critical meetings.

Product owner:

The product owner is responsible for maximizing the value of the product and the work of the team. How this is done may vary widely across organizations, scrum teams, and individuals.

The product owner is the sole person responsible for managing the product Backlog. product Backlog management includes:-

* Expressing product Backlog items clearly.
* ordering the product Backlog items to best achieve goals and missions.
* optimizing the value of the work the team performs.
* Ensuring that the team understands items in the product Backlog to the level needed.
* Ensuring that the product Backlog is visible, transparent and clear to all, and shows what the team will work on further.

The Team:

The team is self-organizing and cross-functional. That means the team comprises of analysis, designers, developers, testers, etc. as appropriate and as relevant to the project.

The scrum team works together closely, on a daily basis, to ensure the smooth flow of information and the quick resolution of issues. The scrum team delivers product iteratively and incrementally, maximizing opportunities for feedback. Incremental deliveries of a complete product ensure a potentially useful version of working product is always available.

SCRUM- Events:

Scrum process framework can be viewed by means of a sequence of events and the corresponding artifacts. The scrum events are time-boxed events. That means, in a project, every scrum event has a predefined maximum duration. These events enable transparency on the project progress to all who are involved in the project. The vital events of scrum are:

* The sprint
* Sprint planning
* Daily scrum meetings
* The sprint Review
* The sprint Retrospective.

The sprint:

During a sprint, a working product increment is developed.

It is usually of duration two weeks or one month, and this duration remains constant for all the sprints in the project. we cannot have varying durations for the different sprints in a project. A new sprint starts immediately after the conclusion of the previous sprint.

A sprint should be cancelled if the sprint goal become obsolete. this might occur if the organization changes direction or if market or technology conditions change. A sprint can be cancelled only by product owner, though others have an influence on the same.

If a Sprint is cancelled, and part of the work produced during the sprint is potentially releasable, the product owner typically accepts it. All the incomplete Sprint Backlog items are put back into the Product Backlog.

Sprint planning:

The work to be performed in the sprint is planned in the sprint planning Meeting. Sprint planning Meeting is of duration of maximum of four hours for two weeks sprints and eight hours for one month sprints.

It is the responsibility of the Scrum Master to ensure that the meeting takes place and that all the required attendees are present and understand the purpose of the scheduled meeting.

The Scrum Master moderates the meeting to monitor the sustenance of discussion and closure on time.

The Team:

Sprint planning focuses on the following 2 questions:-

* what needs to be and can be delivered in the sprint increment?
* How will the work needed for the execution of sprint be achieved?

The inputs to this meeting are:-

* the product Backlog
* the latest product increment
* projected capacity of the team during the sprint
* past performance of the team.

The scrum team discusses the functionality that can be developed during the sprint. product owner provides clarifications on the product Backlog items. the team selects the items from the product Backlog for the Sprint, as they are the best to assess what they can accomplish in the product Backlog for sprint. The team comprises of analysts, designers, developers, and testers, the work is carried out in a collaborative fashion, thus minimizing re-work.

Daily Scrum Meetings

The daily scrum meeting is a 15-minute meeting for the team, conducted daily to quickly understand and the work since the last daily scrum meeting and create a plan for the next 24 hours. This meeting is also referred to as daily stand up Meeting.

The daily scrum meeting is held at the same time and same place every day to reduce complexity.

During the meeting, each team member explains:-

* what did he do yesterday the helped the team meet the Sprint Goal?
* what will he do today to help the team meet the the Sprint Goal?
* Does he see any impediments that prevent him or the team from meeting the Sprint Goal?

Following are the benefits of Daily Scrum Meetings -

* Improve communication within the team.
* identify impediments, if any, in order to facilitate an early removal of the same, so as to minimize impact on the Sprint.
* Highlight and promote quick decision-making.
* improve the Team's level of knowledge.

Sprint Review:

A Sprint Review is held at the end of every Sprint. During the Sprint Review a presentation of the increment that is getting released is reviewed. In this meeting, the Scrum Team and the stakeholders collaborate to understand what was done in the Sprint. Based on that, and any changes to the Product Backlog during the Sprint, the attendees arrive at the next steps required that could optimize value. Thus the objective of Sprint Review is to obtain feedback and progress untedly.

The Team:

the sprint Review is normally held for two hours for two week sprints and for four hours for one month sprints.

the scrum master ensures that

* the meeting takes place.
* the participants understand the purpose.
* the meeting is focused on the required agenda and is completed within the required duration.

The sprint Review includes the following aspects:-

* Attendees include the scrum Team and key stakeholders, as invited by the product owner.
* The product owner explains what product Backlog items have been completed during the sprint and what has not been completed.
* the Team discusses what went well during the sprint, what problems it ran into, and how those problems were solved.
* the Team demonstrates the work that it has completed and answers questions, if any, about the increment.
* the entire group then discusses on what to do next.

thus, the sprint Review provides valuable input to sprint planning of the subsequent sprint.

* the scrum Team then reviews the timeline, budget, potential capabilities, and marketplace for the next anticipated release of the product increment.
* the outcome of the sprint Review is an updated product Backlog, which defines the probable product Backlog items for the next sprint.

Sprint Retrospective:

The Sprint Retrospective occurs after the Sprint Review and prior to the next Sprint planning. This is usually a one hour meeting for two-week duration Sprints and a three hour meeting for one month duration Sprints

The purpose of the Sprint Retrospective is to:

- combine the learnings from the last Sprint, with regard to people, relationships, process, and tools
* identify the major items that went well and potential improvements.
* Creation of a plan for implementing improvements to increase product quality.

The Sprint Retrospective is an opportunity for the Scrum Team to introspect and improve within the Scrum process framework so as to make the next Sprint outcome more effective.

Kanban:

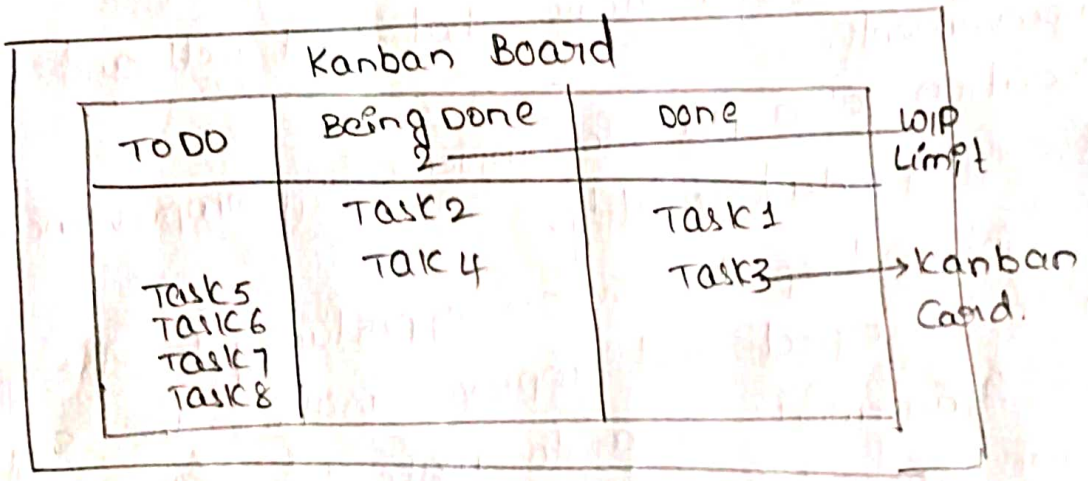
Kanban term came into existence using the flavors of "visual card", "signboard", or "bill board", "signaling System" to indicate a workflow that limits work in progress (WIP). Kanban has been used in lean production for over half century.

The concept of kanban includes:

* visualize workflow
    * split the entire work into defined segments or states, visualized as named columns on a wall
    * write each item on a card and put in a column to indicate where the item is in the workflow

* Limit WIP:
  * Assign explicit limits to how many items can be in progress at each workflow segment / state i.e, work in progress (WIP) is limited in each workflow state.

* Measure the Lead Time
  * Lead time, also known as cycle time is the average time to complete one item. Measure the Lead time and optimize the process to make the lead time as small and predicatable as possible.



Kanban - Lean Practices:

The implementation of kanban, as well as other Lean Manufacturing Methods, such as kaizen, can have significant benefits for almost any type of work. Kanban is more effective because it visually indicates when the production should start and stop. It is faster, more efficient, and saves significant money over most other production models. It is also far more directly responsive to customer demand.

Kanban Benefits:

Kanban has the following commonly observed benefits

* Bottlenecks become clearly visible in real-time.

product

sprint.

this leads people to collaborate to optimize the whole value chain rather than just their part.

* useful for situations where operations and support teams have a high rate of uncertainty and variability.
* Tends to spread throughout the organization naturally, including sales and management. This increases visibility of everything that is going on at the company.
* Reduces inventory in the range of 25% - 75%; thereby reducing company costs.
* since all segments/states in the workflow are visually organized, the required items, reducing the wait times and ensuring speed, continually support all the tasks in the workflow.
* overproduction of inventory is avoided, thereby saving resources and time as well. This is termed as eliminating waste.

Kanban - characteristics:

flexibility in Planning:

Kanban provides improvements in the workflow. with visual representation of the workflow, speed of moving from one task to another is reduced. This is accomplished through the creation of clearly marked flow lanes, kanban cards and clearly marked columns to indicate where each item is in the workflow. if a task needs longer duration, it is allowed to execute without hindrance, and at the same time, the tasks that are completed will flow to the next state.

This allows -

* sufficient duration for longer tasks that cannot be broken down logically.
* preservation of value of such longer tasks.
* effort required by each role to be expended.
* continuous flow of the tasks that are completed without wait time

Hence, planning is flexible and not time-boxed.

Limits work in-progress (WIP)

Explicit limits are assigned to number of items that can be in progress at each workflow state, indicated by a column.

This allows -

* Reducing wait time
* Avoiding stress on resources at a workflow state.
* Identifying bottlenecks causing an item to be in a workflow state than the anticipated time (usually average cycle time) immediately.
* Resolving bottlenecks with collaboration of the entire team.
* Decreasing dependencies in completing a task by splitting it into sub-tasks, so that the sub-task is tracked independently.

Pull Approach:

When you have two teams and the first one is performing better than the second one, it is likely that it pushes more work than the other can actually handle. This often creates friction between the teams. A solution to this is the pull approach.

... become clearly visible ...

In pull approach, the next team pulls work only when it is ready for it, pull approach is implemented by adding a buffer with limited capacity between the two teams.

The benefits of pull approach are-

* avoids piling-up of work.
* Reduces wait time
* facilitates a team to maintain constant pace and focus on quality.
* provides resources balancing.

Minimize Cycle Time:

The cycle time for each task is measured and the process is optimized to reduce the cycle times.

* the bottlenecks are identified immediately and resolved collaboratively by the entire team.
* the correction loops are considered to reduce rework.

Continuous Delivery:

Benefits of continuous delivery are-

* short release cycles result in continuous delive-ry of growing product at regular intervals.
* continuous interactions with customer
  * to understand what customer wants.
  * not to produce anything that the customer does not need.
  * feedback on delivered modules.
* Limited requirements in each release cycle.
  * Developers are not overloaded with requests. this enables them to focus on the delivery.

* there is no partially completed work.

* focus is on finishing work than on starting work.
  * this enables focus on sustaining pace and quality of the product.
  * Deliver before the customer changes mind.
* optimize flow of work from beginning to end.
  * Helps in incremental process improvements.

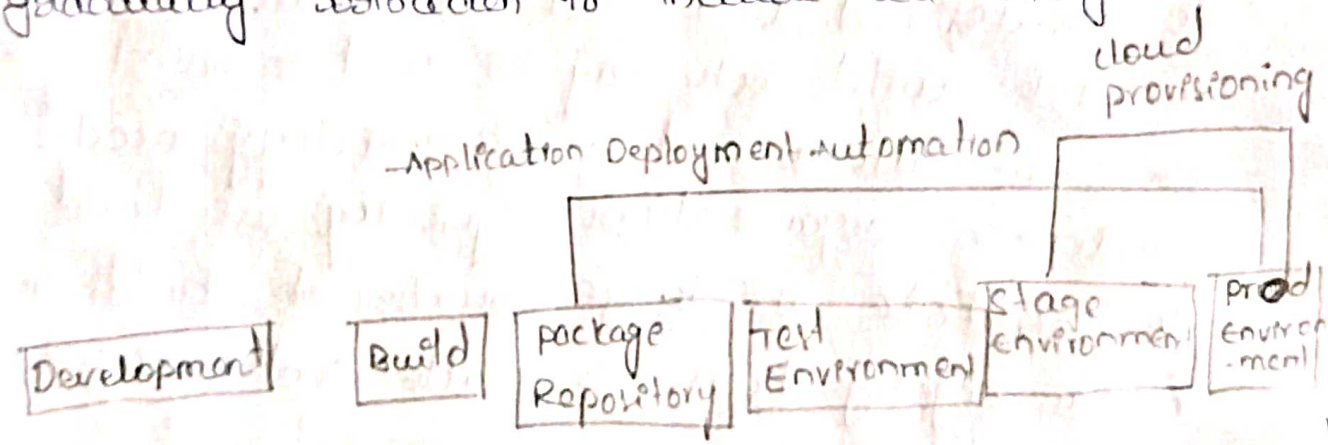
Visual Metrics:

Visually organized workflows (on kanban Boards) facilitate:-

* scheduling as per WIP limits on a workflow state.
* tracking status and progress continually
* Assigning resources dynamically based on the role requirements.

Delivery pipeline :

A delivery pipeline consists of the stages an application goes through from development through to production. figure shows a typical set of stages. these stages may vary from one organization to another, however, and may also vary from one application to another based on the organization's needs software delivery process, and maturity. the level of automation may also vary. Some organizations fully automate their delivery pipelines. others put their software through manual checks and gates. due to regulatory or company requirements. you don't have to address all stages at once start by focusing on the critical part of organization - not everything all at once - and then gradually broaden to include all stages.



Devops delivery pipeline.

A typical delivery pipeline has the stages described in the following sections.

* there is no partially completed work.

Development environment:

An application's development effort takes place in a development environment, which provides multiple tools that enable the developers to write and test code. Beyond the integrated developers development environment (IDE) tools that developers use to write code, this stage includes tools that enable collaborative development, such as tools for source control management, work-item management, collaboration, unit testing and project planning. Tools in this stage typically are cross-platform and cross-technology, based on the type of development being undertaken

Build stage:

The build stage is where the code is compiled to create and unit test the binaries to be deployed. Multiple build tools may be used in this stage, based on cross-platform and cross-technology needs. Development organizations typically use build servers to facilitate the large number of builds required on an ongoing basis to enable continuous integration

package repository:

A package repository (also referred to as an asset repository or artifact repository) is a common storage mechanism for the binaries created during the build stage. These repositories also need to store the assets associated with the binaries to facilitate their

deployment, such as configuration files, infrastructure-as-code files, and deployment scripts.

Test Environment:

→ test environment is where the QA, user acceptance, and development / testing teams do the actual testing. Many flavors of tools are used in this stage, based on QA needs. Here are a few examples:

* Test environment management:

these tools facilitate provisioning and configuring the test environments. they include infrastructure-as-code technologies and (if the environment is in the cloud) cloud provisioning and management tools

Test data management: for any organization that wants to enable continuous testing, managing test data is an essential function. the number of tests that can be run and the frequency with which they're run are limited by the amount of data that's available for testing and the speed at which that data can be refreshed.

Test integration, function, performance, and security

~~Automated~~ tools are available for each of these types of tests. These tools should be integrated with a common test asset management tool or repository where all test scenarios, test scripts, and associated results can be stored and traceability established back to code, requirements, and defects.

Service virtualization: Modern applications aren't simple, monolithic applications. They're complex systems that are dependent on other applications, application servers, databases, and even third-party applications and data sources. unfortunately, at test time, these components may be unavailable or costly. Service virtualization solutions simulate the behavior functionality and performance of select components within an application to enable end-to-end testing of the application as a whole. these tools create stubs (virtual components) of the applications and services that are required for the tests to run. the behavior and performance of the application can be tested as it interacts with these stubs. IBM's Rational Test Virtualization Server provides such test virtualizatron capabilities.

stage and production environments:

Applications are deployed in the staging and production environments. Tools used in these stages include environment management and provisioning tools. Tools for infrastructure as code also play a critical role in these stages, due-to the large scale at which the environment in these stages exist. with the advent of virtualization and cloud technologies, stage and production environments can today be made up of hundreds or even thousands of servers. Monitoring tools allow organizations to monitor the deployed applications in production

Bottlenecks

What Is a Bottleneck?

A bottleneck is a point of congestion in a production system (such as an assembly line or a computer network) that stops or severely slows the system. The inefficiencies brought about by the bottleneck often create delays and higher production costs.

The term "bottleneck" refers to the typical shape of a bottle and the fact that the bottle's neck is the narrowest point, which is the most likely place for congestion to occur, slowing down the flow of liquid from the bottle.

There are two main types of bottlenecks: short-term and long-term. A short-term bottleneck is temporary and typically caused by temporary conditions such as employees on vacation or on sick leave. Long-term bottlenecks are baked into the production process and include such things as inefficient machinery.

Bottlenecking, the process that creates bottlenecks, can have a significant impact on the flow of manufacturing and can sharply increase the time and expense of production. Companies are more at risk for bottlenecks when they start the production process for a new product. This is because there may be flaws in the process that the company must identify and correct; this situation requires more scrutiny and fine-tuning. Operations management is concerned with controlling the production process, identifying potential bottlenecks before they occur, and finding efficient solutions.

KEY TAKEAWAYS

- A bottleneck is a point of congestion in a production system that stops or severely slows the system.
- Short-term bottlenecks are temporary and usually caused by employees on vacation or sick leave.
- Long-term bottlenecks are built into the manufacturing protocol and often related to inefficient equipment or processes.
- Bottlenecking, the process that creates bottlenecks, can have a significant impact on the flow of manufacturing and can sharply increase the time and expense of production.
- Bottlenecks have a negative effect on practical production capacity, keeping it further below theoretical (perfect) capacity than normal.
- Eliminating bottlenecks is key to increasing production efficiency.

Understanding a Bottleneck

As an example, assume that a furniture manufacturer moves wood, metal, and other raw materials into production, then incurs labor and machine costs to produce and assemble furniture. When production is complete, the finished goods are stored in inventory. The inventory cost is often transferred to the cost of goods sold (COGS) when the furniture is sold to a customer.

If there is a bottleneck at the beginning of production, the furniture maker cannot move enough raw materials into the process, which means that machines sit idle and salaried workers don't work productively, creating a situation of underutilization of resources. This increases the cost of production, presents a potentially large opportunity cost, and may mean that completed goods do not ship to customers on time.

Bottlenecks and Production Capacity

A bottleneck affects the level of production capacity that a firm can achieve each month. Theoretical capacity assumes that a company can produce at maximum capacity at all times. This concept assumes no machine breakdowns, bathroom breaks, or employee vacations.

Because theoretical capacity is not realistic, most businesses use practical capacity to manage production. This level of capacity assumes downtime for machine repairs and employee time off. Practical capacity provides a range for which different processes can operate efficiently without breaking down. Go above the optimum range, and the risk increases for a bottleneck due to a breakdown of one or more processes.

If a company finds that its production capacity is inadequate to meet its production goals, it has several options. Company management could decide to lower their production goals to bring them in line with their production capacity. Or, they could work to find solutions that simultaneously prevent bottlenecks and increase production. Companies often use capacity requirements planning (CRP) tools and methods to determine and meet production goals.

Bottlenecks and Production Variances

A variance in the production process is the difference between budgeted and actual results. Managers analyze variances to make changes, including changes to remove bottlenecks. If actual labor costs are much higher than budgeted amounts, the manager may determine that a bottleneck is delaying production and wasting labor hours. If management can remove the bottleneck, labor costs can be reduced.

A bottleneck can also cause a material variance if materials are exposed to spoilage or possible damage as they sit on the factory floor waiting to be used in production. Bottlenecks may be resolved by increasing capacity utilization, finding new suppliers, automating labor processes, and creating better forecasts for consumer demand.

Real-World Example of a Bottleneck

Bottlenecks may also arise when demand spikes unexpectedly and exceeds the production capacity of a firm's factories or suppliers. For instance, when Tesla Inc. (TSLA) first began production of its all-electric vehicles, demand was high for the vehicles, and some analysts were concerned that production would be slowed due to problems in the production line. In fact, Tesla has experienced ongoing production bottlenecks due to the need to manufacture the custom battery packs that supply their vehicles with power.

Tesla founder Elon Musk has said the company's ability to expand its product lineup depends squarely on its ability to produce a large number of batteries.To make that happen, in a joint venture with Panasonic, Tesla opened a massive Gigafactory near Reno, Nev., in 2016, which

makes the company's lithium ion batteries and electric vehicle subassemblies. By mid-2018, the company claimed that its factory was already the highest-volume battery plant in the world in terms of gigawatt-hours (GWh). To make a dent in the waiting list for back-ordered vehicles, Tesla says it will need to continue to invest in and build more Gigafactories worldwide.

Why is it called a bottleneck?

A bottleneck occurs when there is not enough capacity to meet the demand or throughput for a product or service. It is called a bottleneck since the neck of a bottle narrows and tapers, restricting the amount of liquid that can flow out of a bottle at once.

What is a bottleneck in manufacturing?

A bottleneck occurs in manufacturing when there is a stage (or stages) in the process that slows down the overall production of a good. For instance, initial steps may rapidly assemble key parts, but a crucial next step that welds the parts together may not be able to keep pace with the earlier stages. As a result, a backlog occurs and efficiency is reduced. The bottleneck should be solved by expanding that process, investing in better technology to speed up that process, or hiring more workers to help with that process.

What is a bottleneck in the services industry?

Many services are carried out by human beings who have a natural limit on how fast or efficiently they can work. For instance, a barber may only be able to cut the hair of three individuals per hour. If more people want a haircut, they will have to wait, and this can cause a backlog. Ways to reduce a bottleneck are to hire additional barbers, or to increase the efficiency of the barber using technology or skills training (so that they can accommodate four customers per hour).

The Bottom Line

A bottleneck is a point of congestion in a production system that slows or stops progress. Short-term bottlenecks are temporary and often caused by a labor shortage. Long-term bottlenecks are more incorporated into the system itself and characterized by inefficient machinery or processes.

Since bottlenecking is counterproductive and leads to a reduction in production efficiency, eliminating bottlenecks is key to increasing profitability. The best way to eliminate bottlenecks is to increase system capacity by restructuring the process or investing in people and machinery.