innovate achieve lead

BITS Pilani

Pilani Campus

BITS Pilani

Pilani | Dubai | Goa | Hyderabad

By Prof A R Rahman

CSIS Group WILP

innovate achieve lead

BITS Pilani
Pilani Campus

Course Name: Introduction to DevOps
Course Code : CSI ZG514/SE ZG514

By Prof A R Rahman
CSIS Group WILP

Introduction to the course

Handout of Introduction to DevOps

Note: The copy is available in your Teams-Files section along with other learning materials and web sources

Birla Institute of Technology & Science, Pilani
Work Integrated Learning Programmes Division
First Semester 2025-2026
Digital Learning Handout

Part A: Content Design

Course Title	Introduction to DevOps
Course No(s)	CSI ZG514/SE ZG514
Credit Units	4
Credit Model	3-1-0
Course Author	Prof. Sonika Rathi
Lead Instructor	A. ABDUL RAHMAN
Version No:	1.0
Date:	25/02/2025

Course Description:
Continual Service - continuous integration and continuous delivery; Scaling: automating infrastructure and infrastructure as code; DevOps and Cloud: platform-as-a service and DevOps, use of virtual machines and containers for deployment, Micro-services; application lifecycle management: deployment pipeline and application deployment, continuous deployment pipeline; stack management - life cycle of stack and events, resource and event monitoring, auto healing; Security: security of deployment pipeline, policy-as-code.

Course Objectives

No	Course Objective
CO1	To learn the key ideas and techniques to bring development and operations together to produce higher-quality software and deliver it more quickly
CO2	To learn the core principles, business and technical terms used in DevOps from perspective of business and IT teams
CO3	To gain knowledge of the Principles and practices of the DevOps Lifecycle including Continuous Integration, Continuous Inspection, Continuous delivery, Continuous deployment and Continuous monitoring
CO4	To understand the usage of tools and technologies used for implementing DevOps

Text Book(s):

T1	DevOps: A Software Architect's Perspective (SEI Series in Software Engineering) by Len Bass, Ingo Weber, Liming Zhu , Publisher: Addison Wesley (18 May 2015)
T2	Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation by Jez Humble, David Farley. Publisher: Addison Wesley, 2011

Reference Book(s) & other resources:

R1	Effective DevOps: Building A Culture of Collaboration, Affinity, and Tooling at Scale by Jennifer

Introduction to DevOps

Coverage

- Software development lifecycle
- The Waterfall approach
- Agile Methodology
- Operational Methodologies: ITIL
- Development, Testing, Release, and Deployment Concepts
- Provisioning, Version Control
- Test Driven Development, Feature Driven Development
- Behavior-driven development

Contact Session Wise Coverage

CS	Coverage
CS01	Foundational Terminology and Concepts
CS02	Why and What is DevOps
CS03	DevOps Dimensions
CS04	DevOps Dimensions
CS05	Source Code Management
CS06	Continuous build and code quality
CS07	Continuous Integration and Continuous Delivery
CS08	Continuous Integration and Continuous Delivery
CS09	Continuous Deployment
CS10	Continuous Deployment
CS11	Continuous monitoring
CS12	Configuration Management
CS13	Configuration Management
CS14	Virtualization and Containerization
CS15	Virtualization and Containerization
CS16	REVISION – Sessions 1 – 15 contents

S.No	Self Study Lab Exercise
1	Exercises to demonstrate the use of GIT operations and commands (Push, pull, clone etc.,). Creating branches and merging branches using GIT – Reference CS-5
2	Installation of Jenkins and Configuration of Jenkins to work with different version control, build and testing tools – Reference CS-7
3	Create jobs and projects in Jenkins – Reference CS-7
4	Demonstration of continuous integration with Jenkins through source code polling and build triggers – Reference CS-8
5	Demonstrate continuous inspection with Jenkins using sonarqube to ensure code quality – Reference CS-9
6	Demonstration of continuous deployment/delivery to staging/production environment with Jenkins – Reference CS-10

Assignments

Skill Area	Activity	Activity
GIT	Install GIT	Create an account in github
Jenkins	Install Jenkins	Experiment with Jenkins as required in the Assignment.
Docker	Install Docker	Experiment with Docker as required in the Assignment.

Evaluation of Assignment

You are expected to execute the Labs

You are expected to submit the Assignments.

Evaluation will be based on

Continuous Interaction

And Quizzes

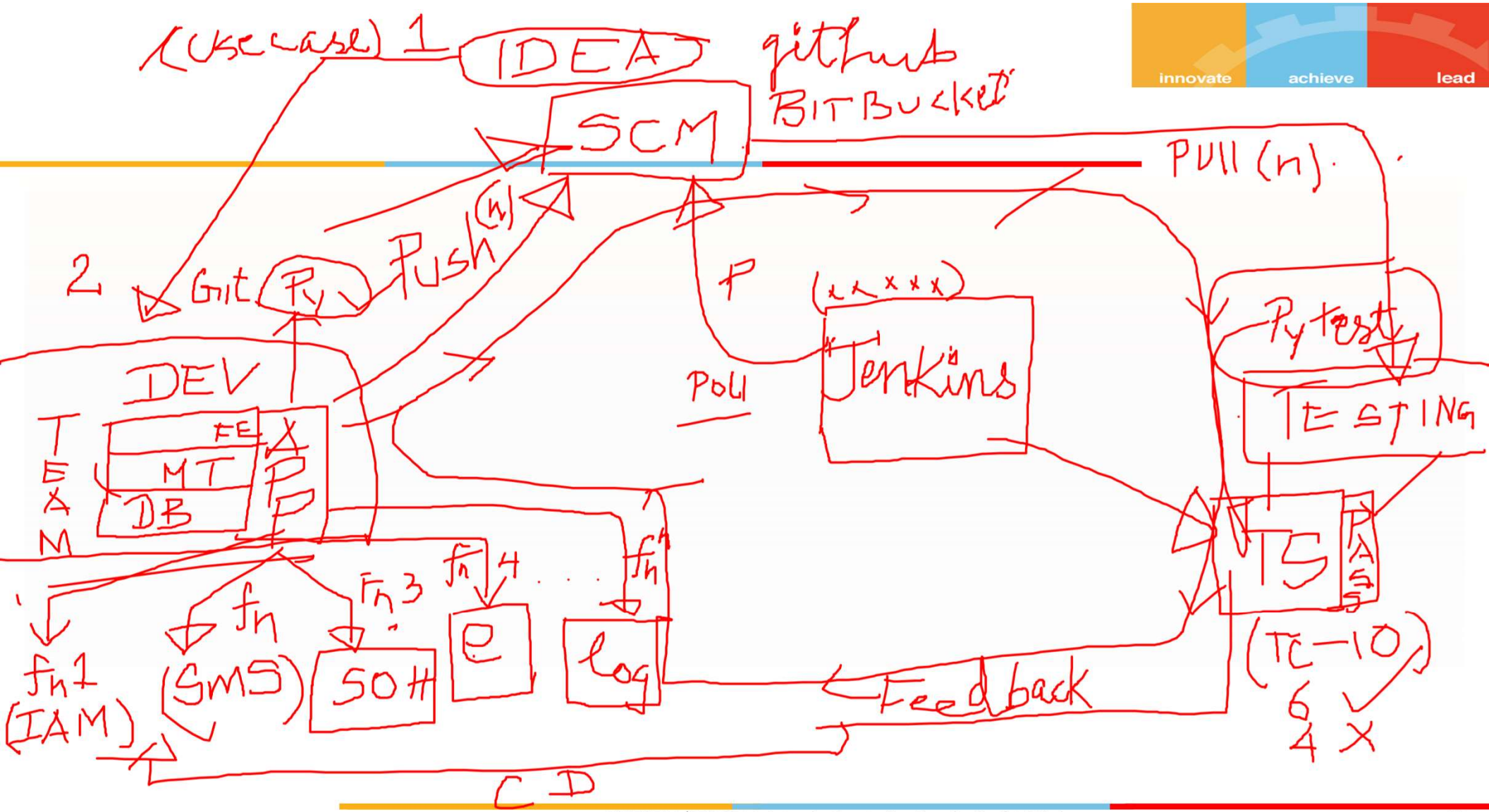
During the Contact Sessions.

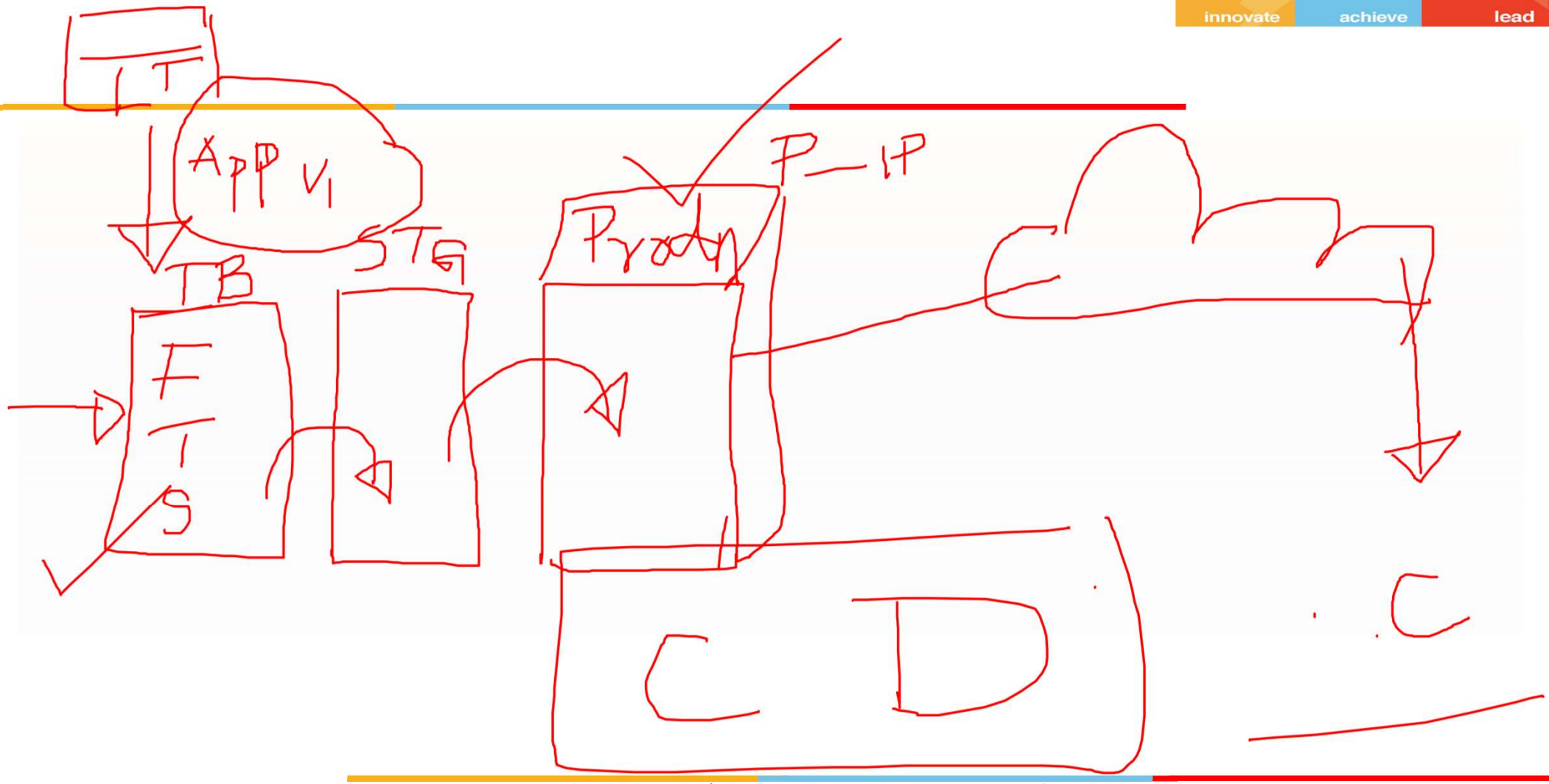
You must share you experience by requesting

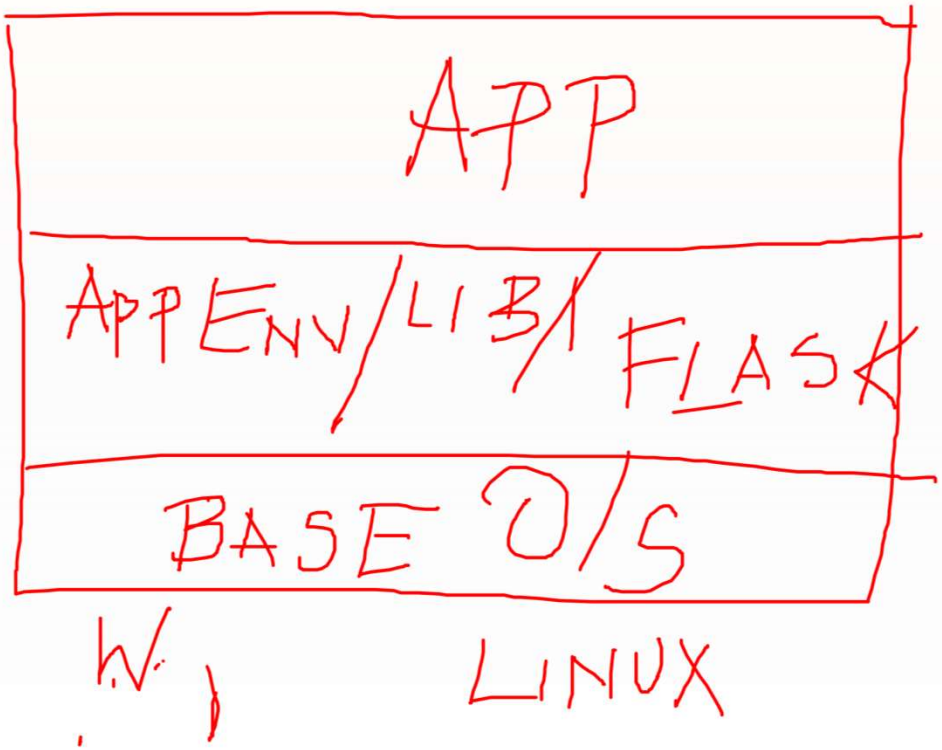
Audio/Video and Presentation Access from faculty.

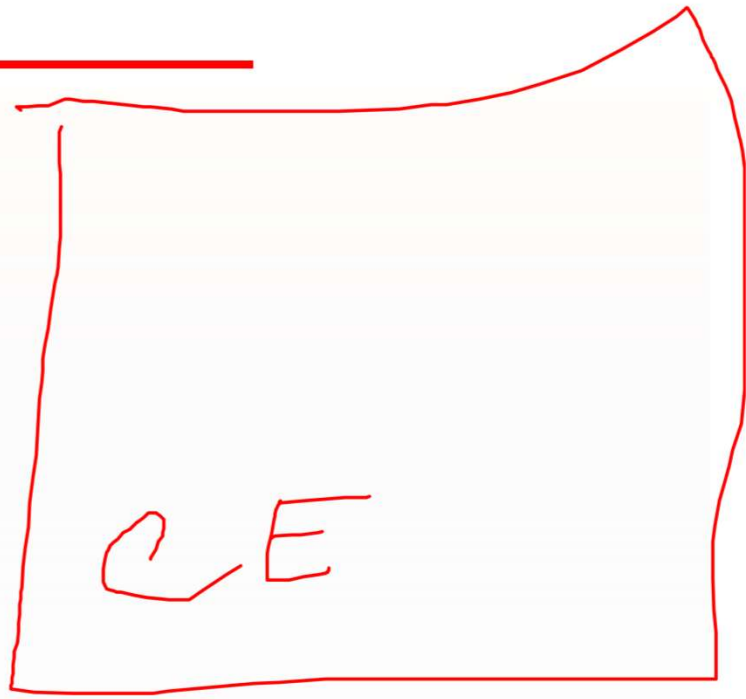
Faculty will not give opportunity to new participants

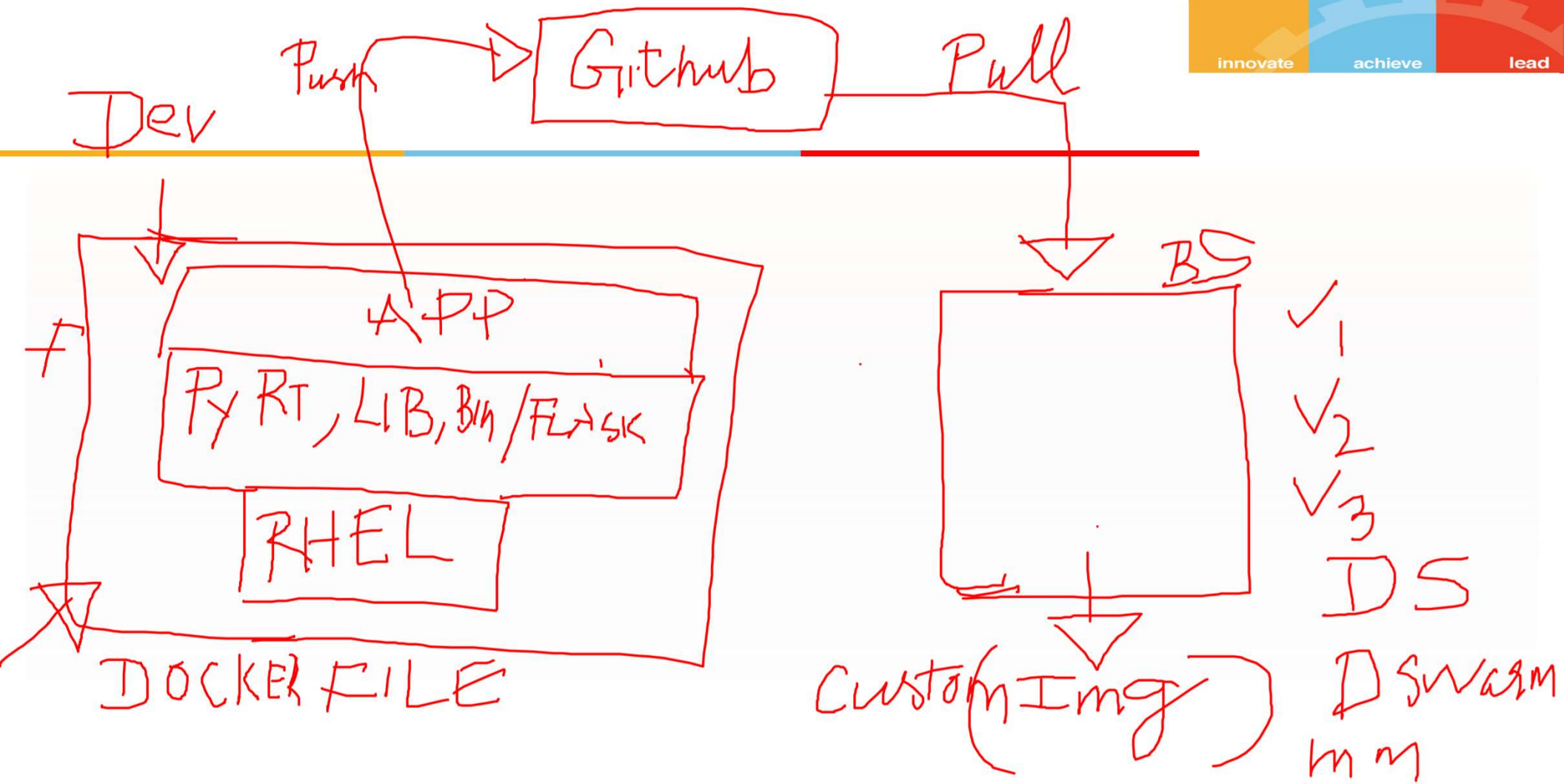
At the end of the course.

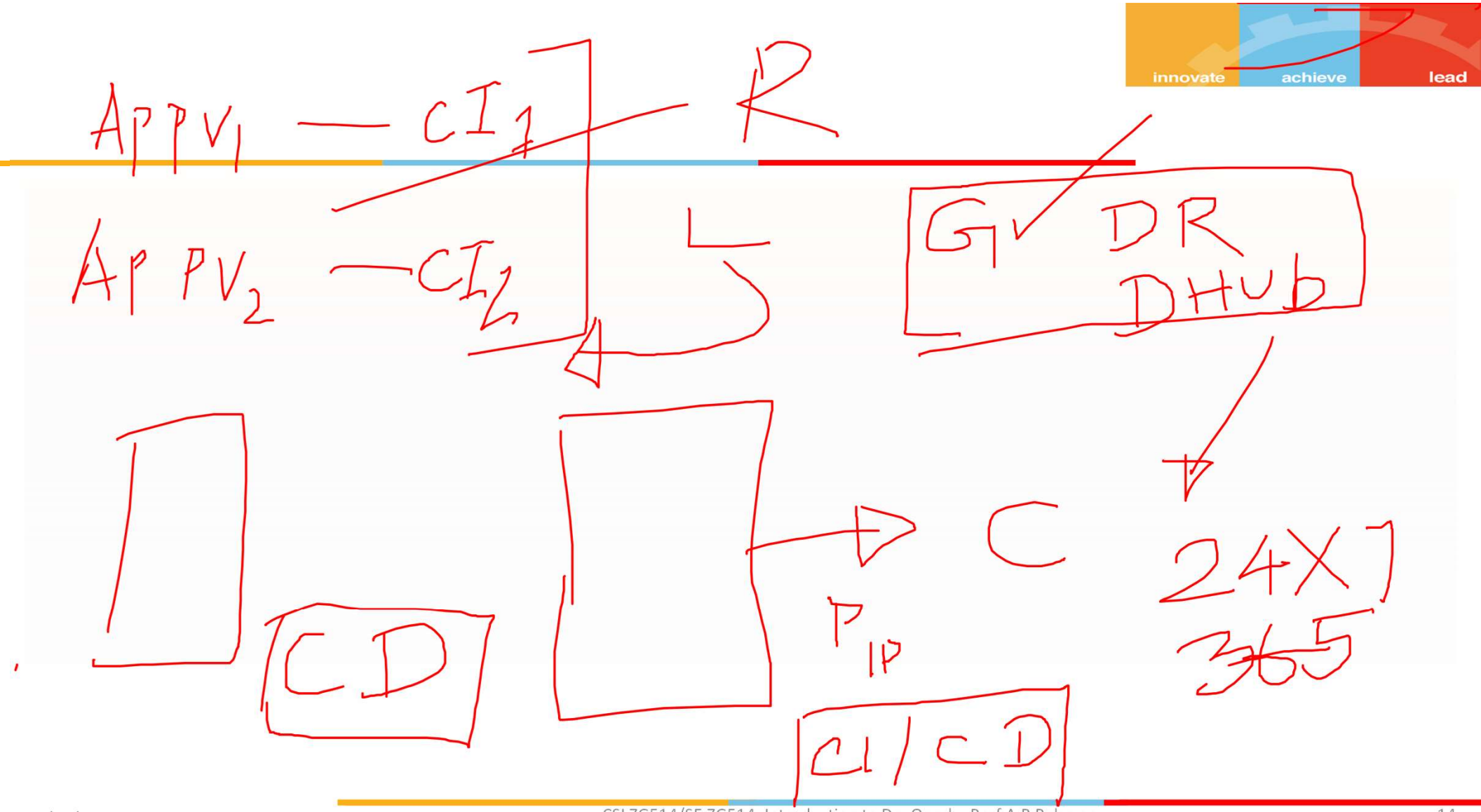

(Use case) 1
IDEA
github
BitBucket
SCM
Pull (n)
2
Git
Push (n)
DEV
TEAM
FE
MT
DB
API
Poll
Jenkins
Pytest
TESTING
TS
PASS
(TC-10)
6
4 X
fn1
(IAM)
fn
(SMS)
fn3
SOH
fn4
e
log
fn
Feedback
CD


T
App V1
TB
STG
F
T
S
Prod
P-IP
C
D
C


APP
APP ENV/LIB/
FLASK
BASE O/S
W
LINUX


CE

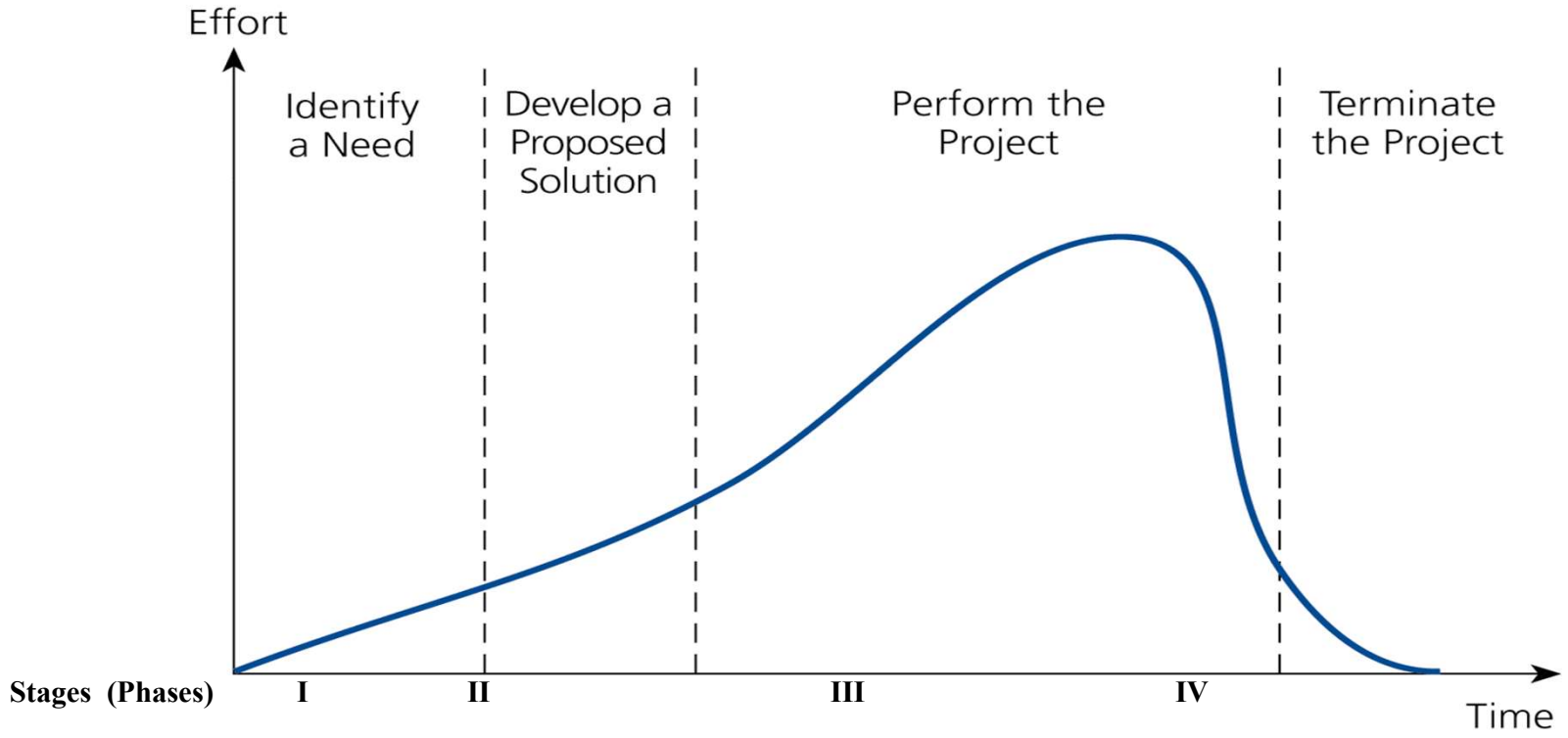

Dev
Push
Github
Pull
APP
PY RT, LIB, BIN/FLASK
RHEL
DOCKERFILE
BS
V1
V2
V3
DS
Custom Img
D Swarm
m m


App V1
CI1
R
App V2
CI2
GitHub
DR
DHub
CD
PIP
CI/CD
24X7
365

DevOps:
Bridging the Gap Between Development and Operations

- Fosters collaboration between Development and Operations teams.
- Traditional siloed approach between Dev and Ops, leading to slow delivery and bottlenecks.
- DevOps bridges this gap, enabling faster and more reliable software delivery.

Typical Project Life-Cycle (4 Phases)



Software development lifecycle (Unified Process)

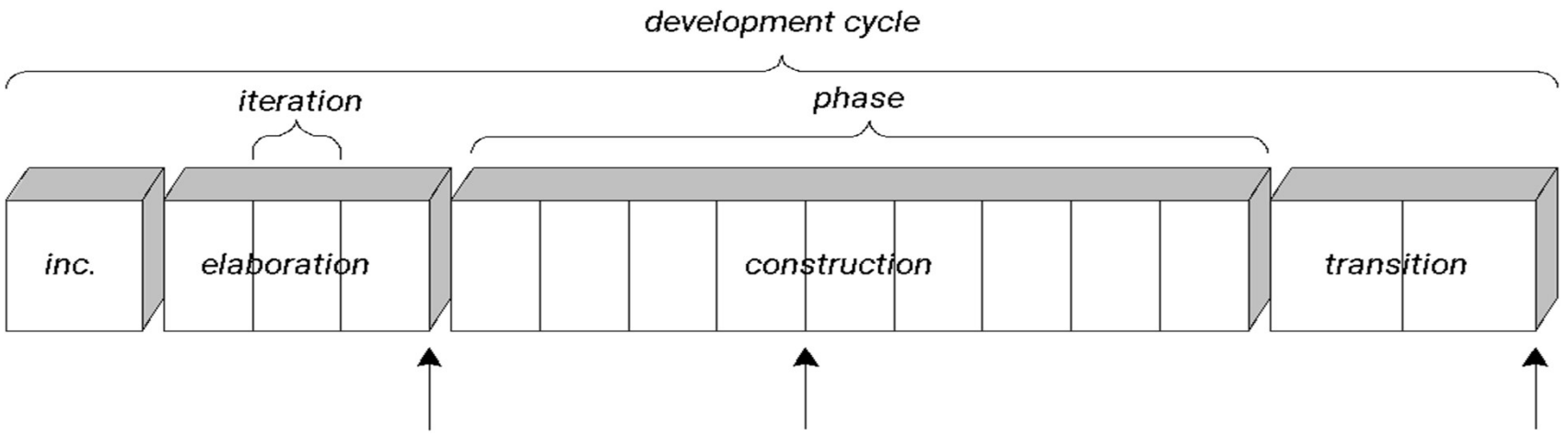
Inception

Elaboration

Construction

Transition

UNIFIED PROCESS Phases



milestone

An iteration end-point when some significant decision or evaluation occurs.

release

A stable executable subset of the final product. The end of each iteration is a minor release.

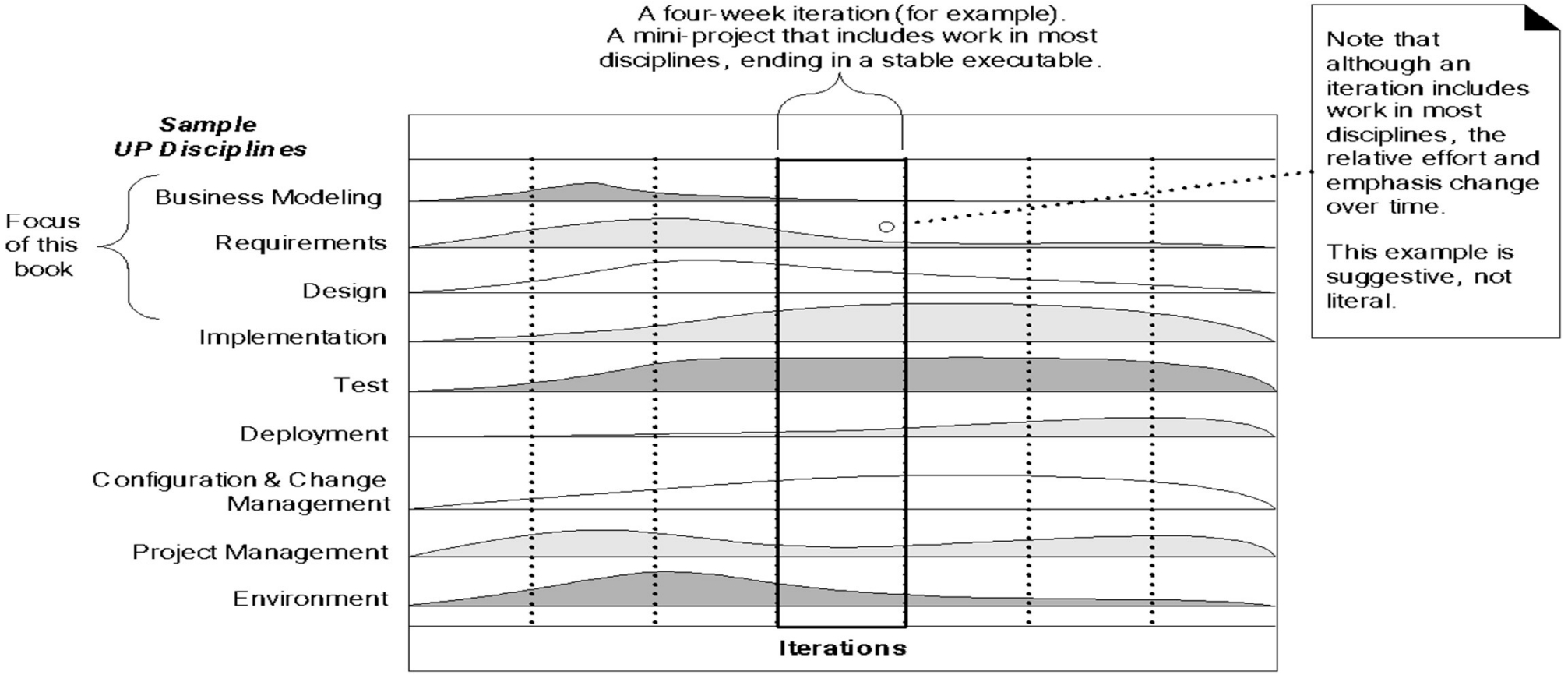
increment

The difference (delta) between the releases of 2 subsequent iterations.

final production release

At this point, the system is released for production use.

UNIFIED PROCESS disciplines



Delivering Software

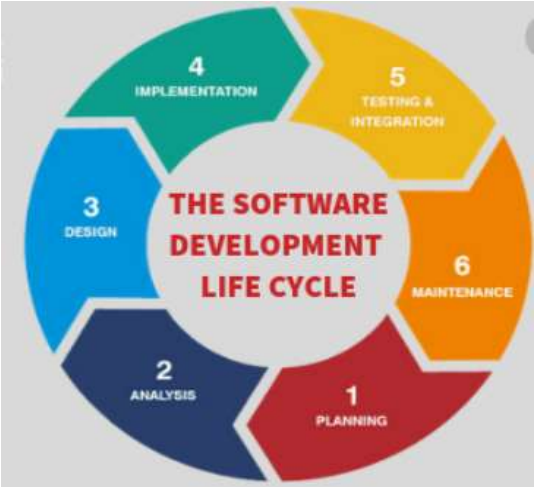
- Typical software delivery involves Requirement gathering, design, build, test, deploy and operations
- Customer ->Requirement->Design->build and test ->Deploy->Operations
- Processes/Approaches to be followed in software development
- How to join and coordinate together to make the process as efficient and reliable
- How to enable developers, testers, build and operations team to work together effectively?

Waterfall approach – Software Development

- Sequential process in the development of a system or software.
- Process often involves phases or activities which are totally dependent on each other, particularly the preceding phases.
- The number of phases in a waterfall model could vary depending on the software or system to be developed.

Advantages:

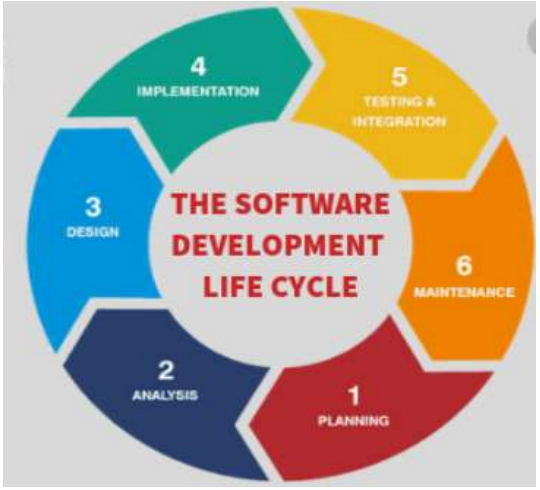
- Well defined requirements
- Each phase has specific deliverables and a review process
- Works well for smaller projects
- Well defined activities



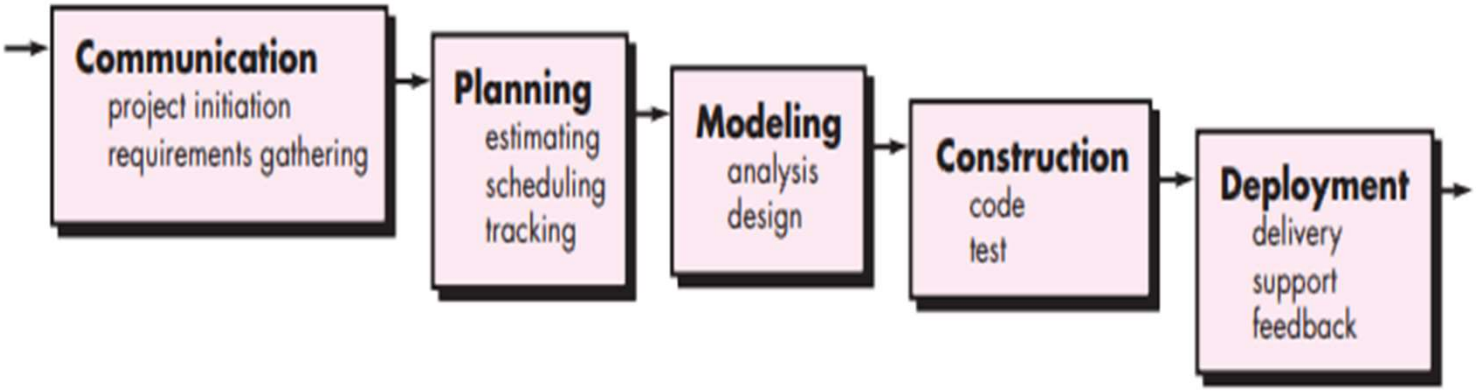
Waterfall approach – Software Development

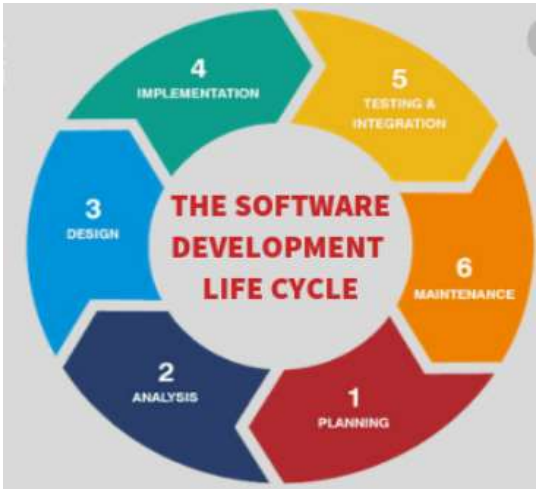
Limitations:

- Spend lot of time to release the final product
- deployment and delivery generally slow
- Doesn’t support a sudden change in development cycle

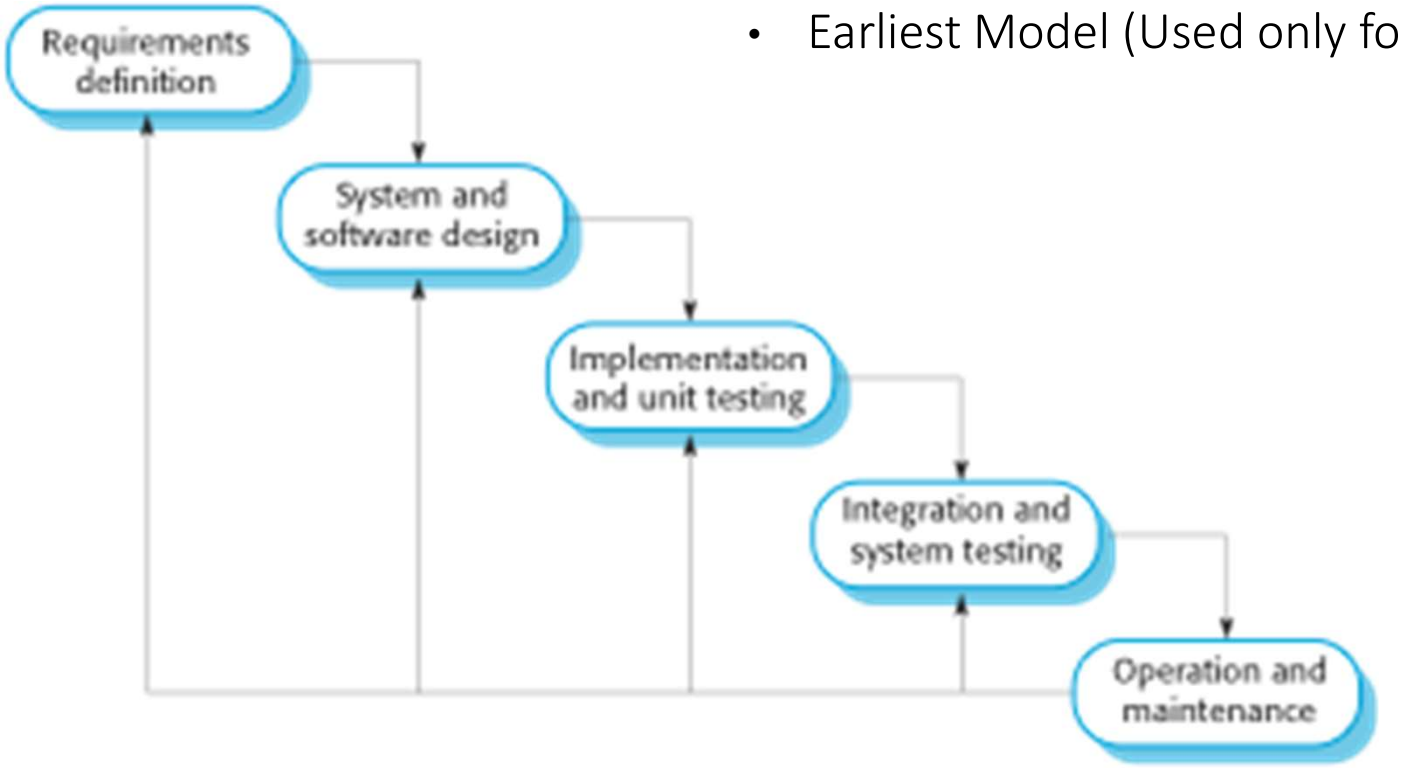


Waterfall approach – Software Development





Waterfall Model (Plan-Driven)



- Earliest Model (Used only for Large Well-Defined Projects)

The Manifesto for Agile Software Development

"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- *Individuals and interactions* over processes and tools
- *Working software* over comprehensive documentation
- *Customer collaboration* over contract negotiation
- *Responding to change* over following a plan

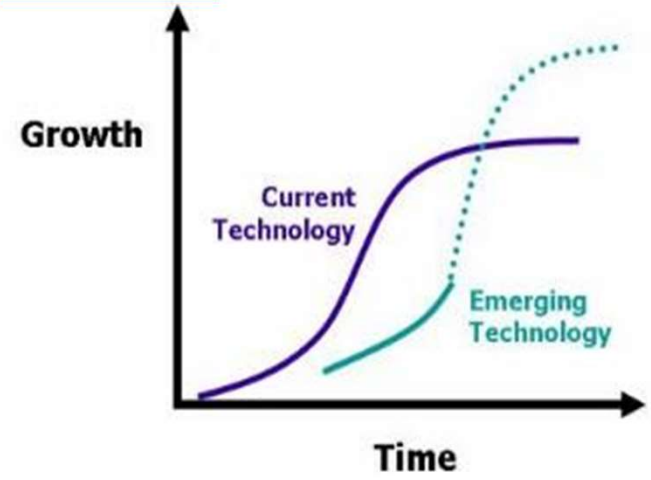
That is, while there is value in the items on the right, we value the items on the left more."

Agile Methodology

- Effective (rapid and adaptive) response to change
- Effective communication among all stakeholders
- Drawing the customer onto the team
- Organizing a team so that it is in control of the work performed
- *Yielding …*
- Rapid, incremental delivery of software

Agile Methods (Flexible, Adaptive)

- Adaptive to Changing or Evolving Customer Requirements



- *Agility*

  The ability to both create and respond to change in the business environment

  - Rigid Processes vs. Agile Frameworks
  - Being Agile = Competitive, Responsive, Flexible,…

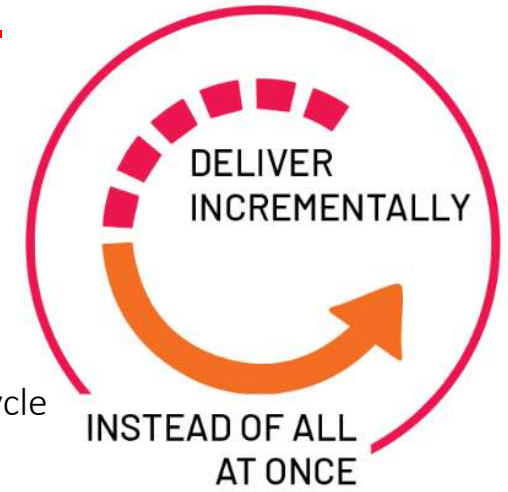
Agile – Methodology

- In contrast to traditional approaches of project management, Agile planning organizes work in short iterations to increase the number of releases.
- It is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project.
- Both development and testing activities are concurrent.

Agile – Methodology

- Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software.
- Scrum is a subset of Agile.
- It is a lightweight process framework for agile development, and the most widely-used one
- Kanban- Another agile development framework, used for Defect/CR workflow

Scrum



- Scrum believes in empowering the development team and advocates working in small teams
- Customer
- Product owner -creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration
- Scrum master -responsible for setting up the team, sprint meeting and removes obstacles to progress
- Scrum Team- Team manages its own work and organizes the work to complete the sprint or cycle
- Product backlog - Repository where requirements are tracked with details on the no of requirements(user stories) to be completed for each release
- Sprint planning
- Epics and User stories
- Daily standup
- Sprint closure
- Sprint retrospective
- Burnt down chart - Amount of work that has been completed in an epic or sprint, and the total work remaining

Operational Methodologies: ITIL
ITSM (IT Service Management) vs Service Delivery

SM: *Strategic* approach to design, deliver, manage and improve the way businesses use information technology (IT).

SM includes all the individual activities and processes that support a service throughout its lifecycle: from service management to change management, problem and incident management, asset management, and knowledge management.

ITSM and IT service delivery sometimes are used interchangeably

- ITSM emphasizes IT service operation and improvement
- IT service delivery focuses on the quality of the work and meeting customer expectations.
- ITIL (Information Technology Infrastructure Library) is a popular ITSM Framework (the terms ITSM and ITIL are used synonymously)

ITSM as a Business Enabler - ITIL

ITSM

Ensure the organization's business goals are supported by high-quality, cost-effective, value-adding IT services

Improve the quality of IT service provisioning

Reduce the long-term cost of IT service infrastructure

IT Infrastructure Library (ITIL) is a Framework described in the following volumes:

- Service Support
- Service Delivery
- Security Management
- Application Management
- ICT Infrastructure Management
- Planning to Implement Service Management
- The Business Perspective

"The implementation and management of quality IT services that meet the needs of the business. IT service management is performed by IT service providers through an appropriate mix of people, process and information technology." - ITIL

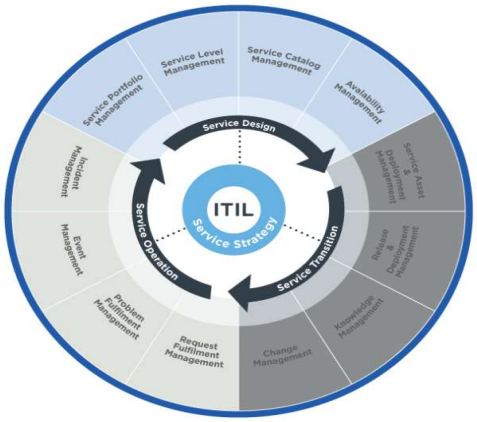
ITIL – Operations

- Information Technology Infrastructure Library, is a set of detailed practices for IT service management (ITSM) that focuses on aligning IT services with the needs of business.
- Set of processes, procedures, tasks, and checklists which are not organization-specific nor technology-specific, but can be applied by an organization toward strategy and delivering value.
- Highly structured methodology designed to increase efficiency and provide statistics for IT operations
- ITIL has many processes which have been segregated into five process areas service strategy, service design, service transition, service operations, continual service improvement.
- Large, medium, and small organizations all over the world use ITIL to help them improve the value of their services. ITIL helps organizations in all industries and sectors solve business issues as well as improving IT capability. Organizations use ITIL as a guide to improve or implement a capability that provides business value.

ITIL – Operations

Advantages

- Faster and more flexible service delivery practices to support digital transformation
- Better strategic alignment between IT and the business
- Smoother integration between evolving software delivery practices and the enterprise customer support framework
- Improved service delivery and customer satisfaction
- Reduced costs through improved use of resources
- Greater visibility of IT costs and assets
- Better management of business risk and service disruption or failure



Development, Testing, Release, and Deployment Concepts

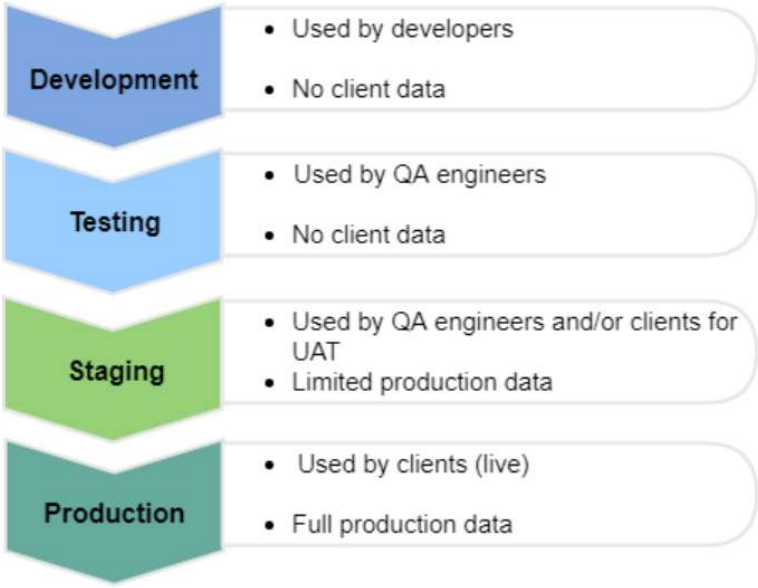
Concept of Service Life Cycle

ITIL V3: Five Distinct Stages in ITSM

Service strategy
Service design
Service transition
Service operation
Continual service improvement.

Delivering Software- Environment

- Development Environment -Developers deploy their code and test
- Testing Environment – Run different tests over the code taken from the development environment.
- Staging Environment – Production like environment for new release and test acceptance test.
- Production Environment - Once the code has been thoroughly tested, it is then pushed to production where it is made available to end-users



Development, Release, and Deployment Concepts

- Version Control - Records changes to files or sets of files stored within the system
- Test-Driven Development – code developer starts by writing a failing test for the new code functionality, then writes the code itself, and finally ensures that the test passes when the code is complete
- Application Deployment – process of planning, maintaining, and executing on the delivery of a software release.

Delivering Software- Common Issues

- Deploying Software Manually
  - Extensive manual to refer and deploy
  - Prone to human error
  - Reliance on manual testing
  - Frequent calls to the development team to explain why a deployment is going wrong on a release day
  - Frequent corrections to the release process
  - Releases that take more than a few minutes to perform
  - Releases and roll back incase of unforeseen problems
  - Extensive work to the team and solve issues

Delivering Software- Common Issues

- Deploying to a Production-like Environment Only after Development Is Complete
  - First deployment to staging is likely to be the most troublesome
  - longer the release cycle, the longer the development team has to make incorrect assumptions before the deployment occurs, and the longer it will take to fix them
  - Developers, testers, and operations personnel are constantly raising tickets (or sending emails) to each other to perform any given deployment
  - Environment incompatibility and bring surprises.
  - May not have much control over target environments, especially outside of a corporate setting.
  - In this case, a great deal of extra testing will be required.

Delivering Software- Common Issues

- Manual Configuration Management of Production Environments
  - Operations team take a long time to prepare an environment for a Release
  - Cannot step back to an earlier configuration of your system, which may include operating system, application server, web server, RDBMS, or other infrastructural settings.
  - Servers in clusters have, unintentionally, different versions of operating systems, third-party infrastructure, libraries, or patch levels
  - Configuration of the system is carried out by modifying the configuration directly on production systems

Delivering Software- Common Issues

- Deployments should tend towards being fully automated
  - Automated deployments encourage collaboration, because everything is explicit in a script.
- Integrate the testing, deployment, and release activities into the development process
- Testing, staging, and production environments, specifically the configuration of any third-party elements of your system, should be applied from version control through an automated process.
- Process should be Automated, support Frequent release, Speed deployment, Trigger Feedback for any change.
- How to Achieve ?

Provisioning, Version Control

Reproducibility.

Traceability.

These capabilities give teams several important benefits:

Disaster recovery.

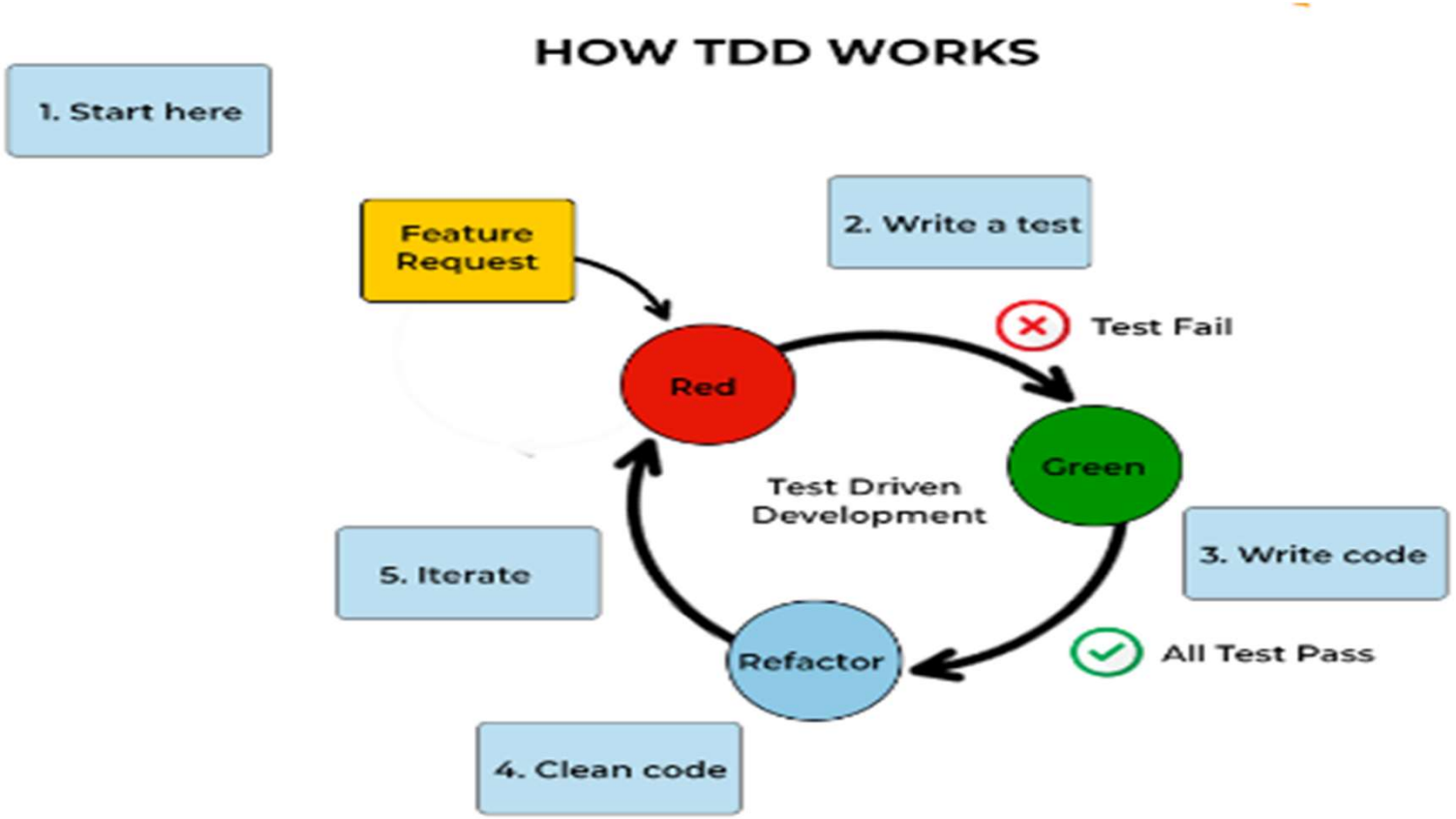
Auditability.

Higher quality.

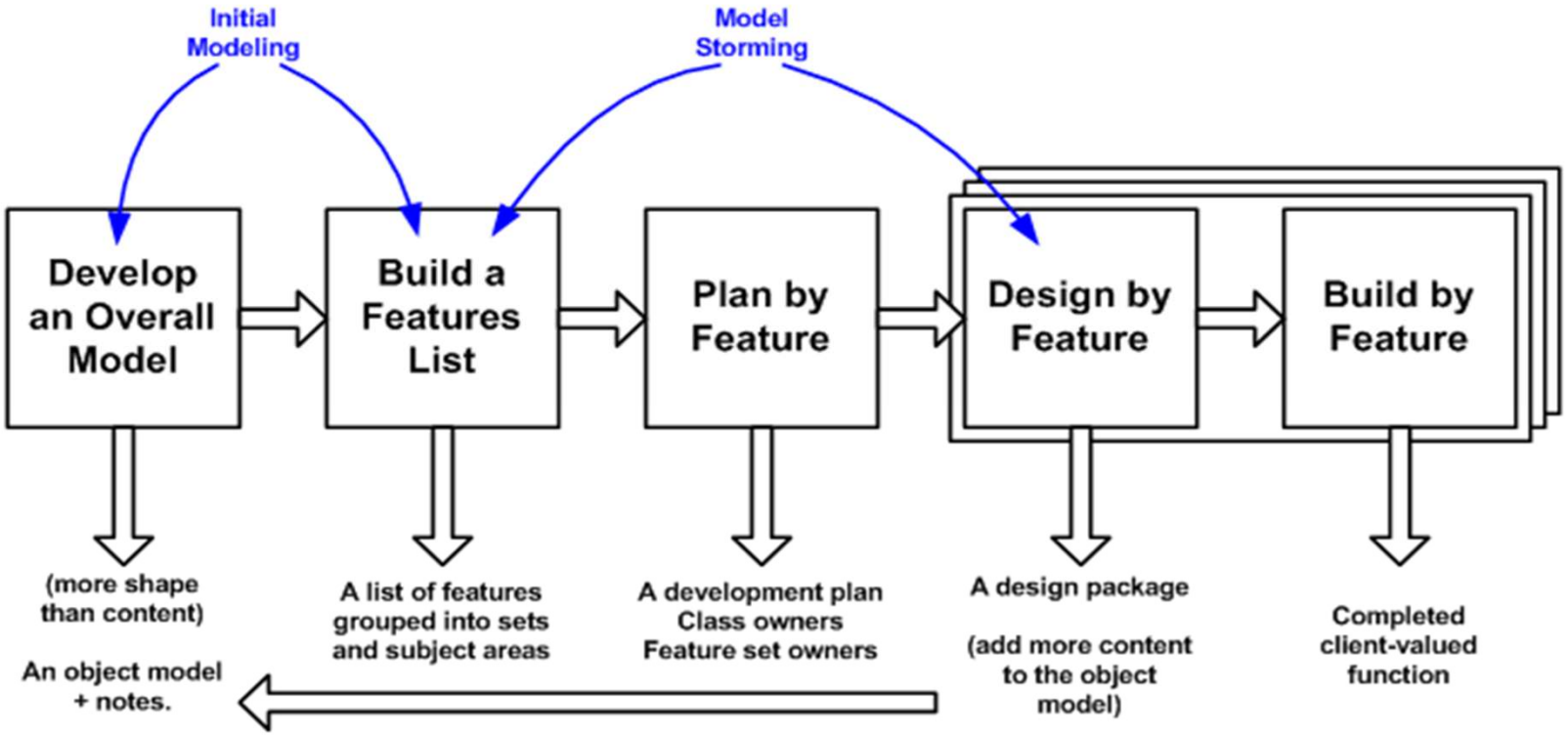
Capacity management.

Response to defects.

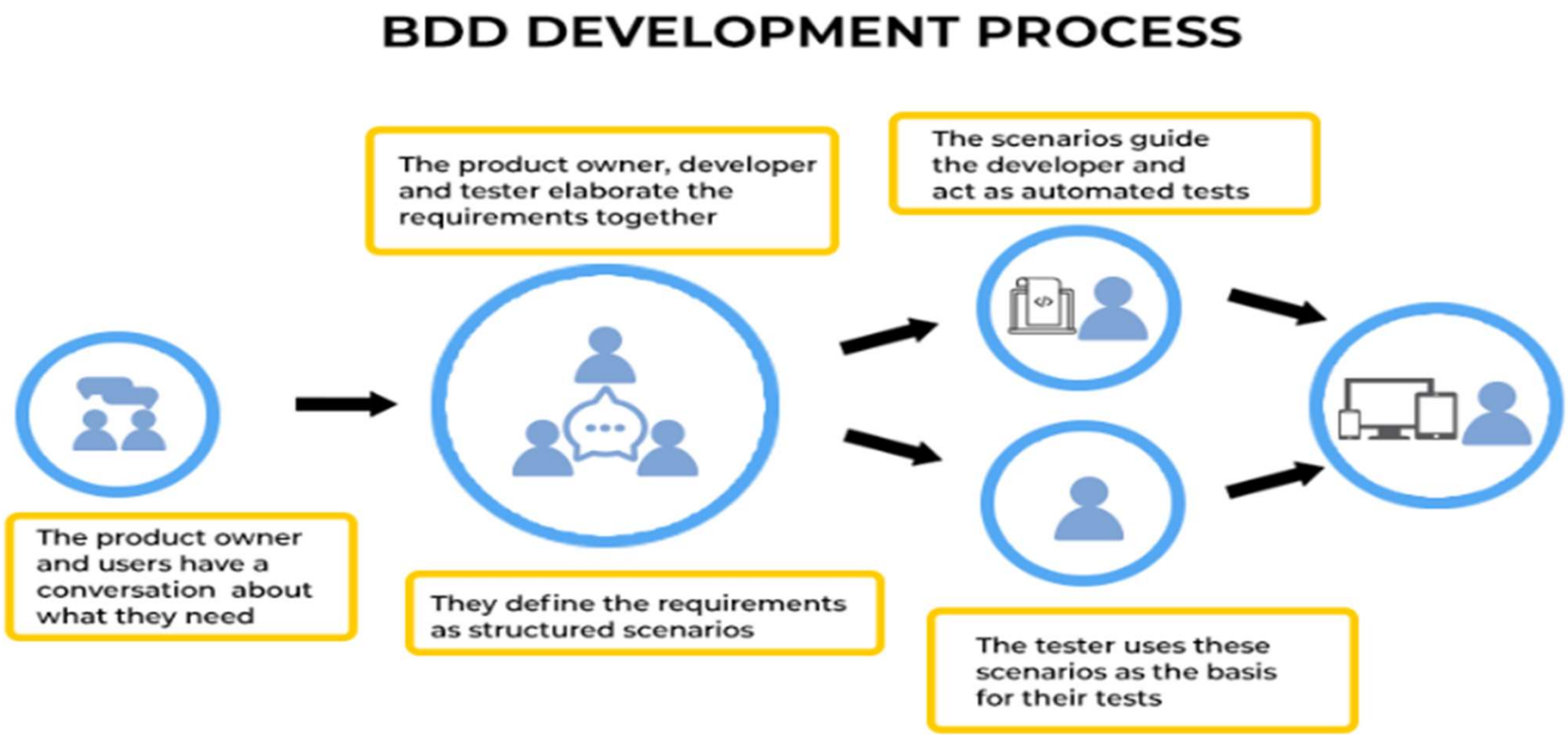
Test Driven Development



Feature Driven Development



Behaviour Driven Development



QUESTIONS AND DISCUSSION

THANK YOU