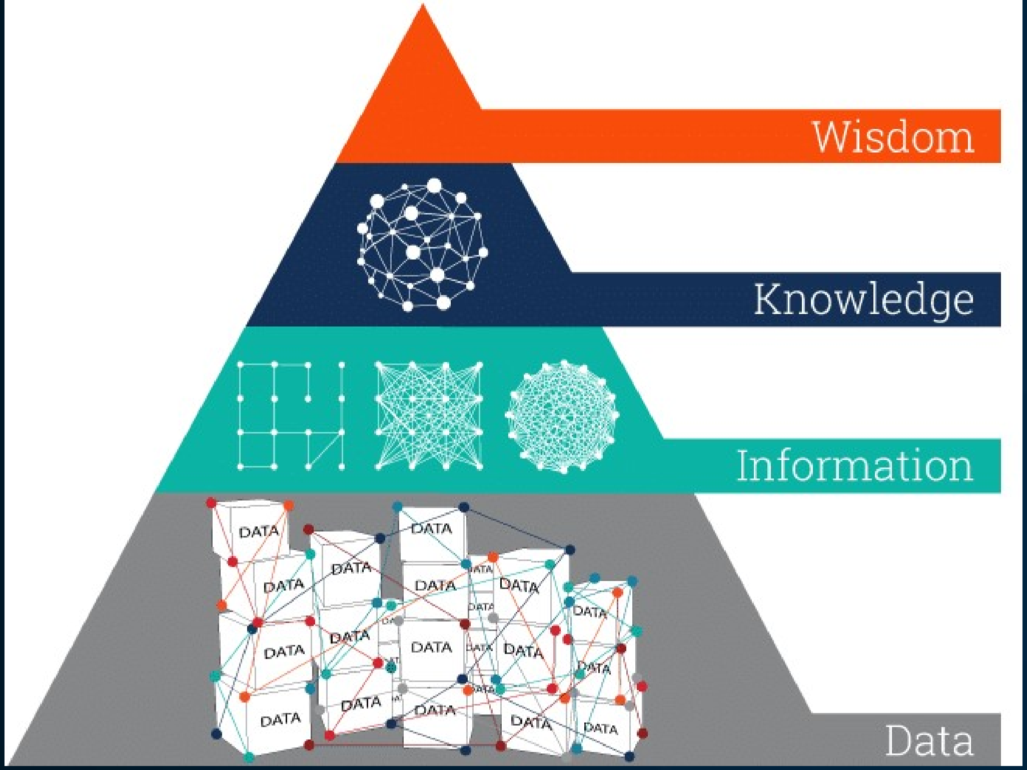
Software Industry Skills for Excellence

Unit 1
Mahesha BR Pandit
05-Oct-21, Version 1.0

Knowledge and Knowledge Industry



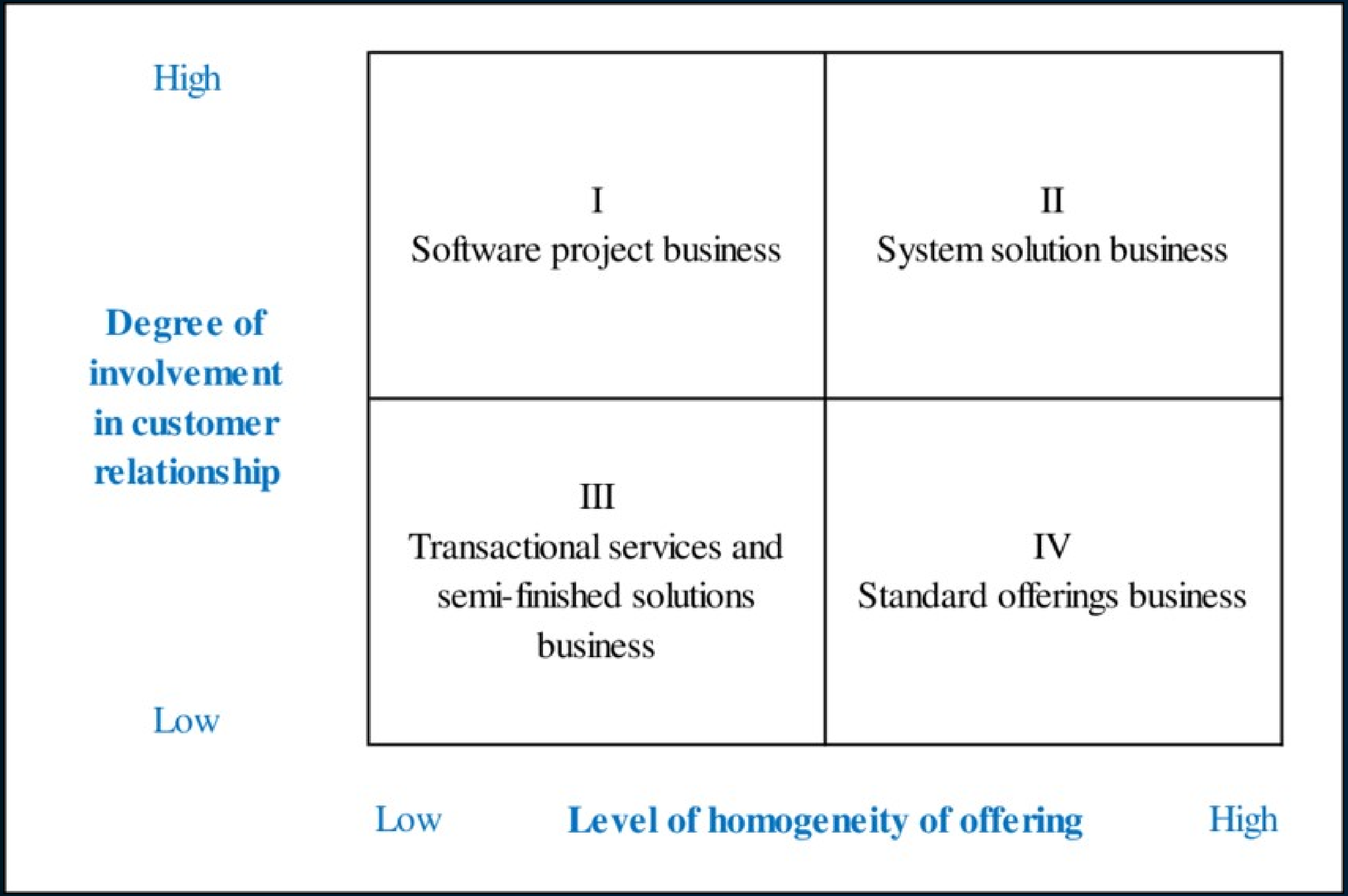
- Agrarian → Industrial → Knowledge Economy
- Intangible outputs
- Driven by thoughts
- Needs “high skills”
- Example: Software Industry, Educational Institution, Bank
- Few more examples?

Image: https://www.ontotext.com/knowledgehub/fundamentals/dikw-pyramid/

Organization of the Software Industry

- Products, Services, Consulting
- Geo based
- Functional, Projectized, Matrix
- MNCs, Captives, Startups
- Tier 1, Tier 2, Tier 3

Organization of the Software Industry



Degree of involvement in customer relationship

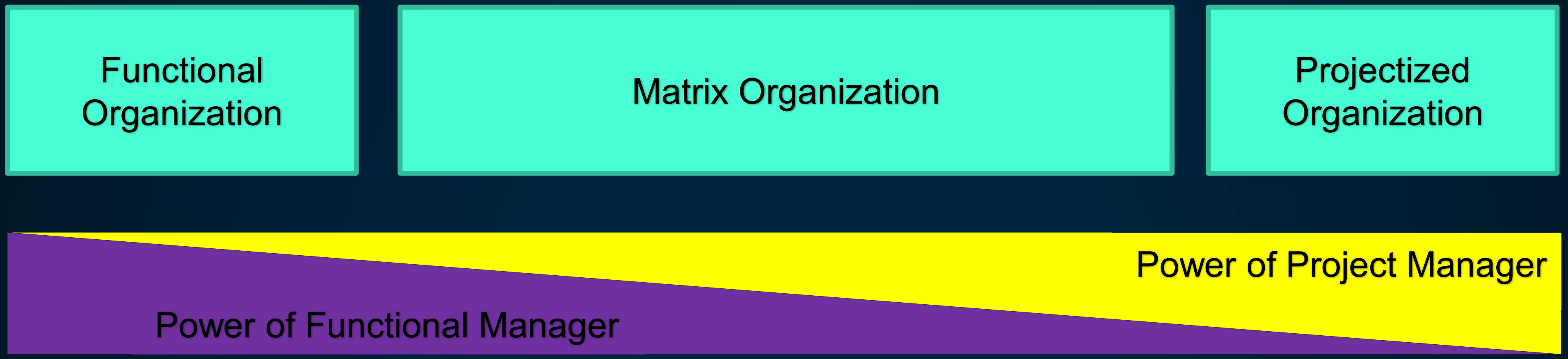
Level of homogeneity of offering

	Low	High
High	I Software project business	II System solution business
Low	III Transactional services and semi-finished solutions business	IV Standard offerings business

- How deep is your customer involvement?
- Can we interpret this with an example of a hotel or bakery business?

Image: https://www.researchgate.net/figure/Classification-of-different-types-of-business-models-in-the-software-industry_fig9_254429177

Software Industry – Common Organization Pattern



Functional Organization

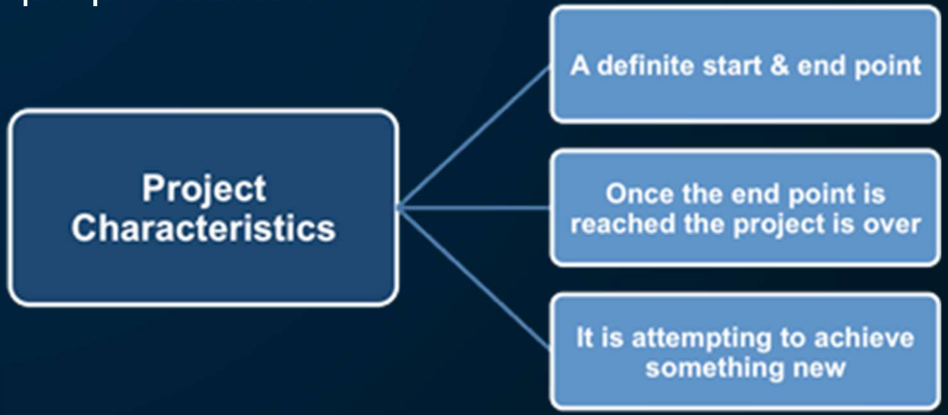
Matrix Organization

Projectized Organization

Power of Project Manager

Power of Functional Manager

Project: Temporary effort undertaken to produce a unique product or service



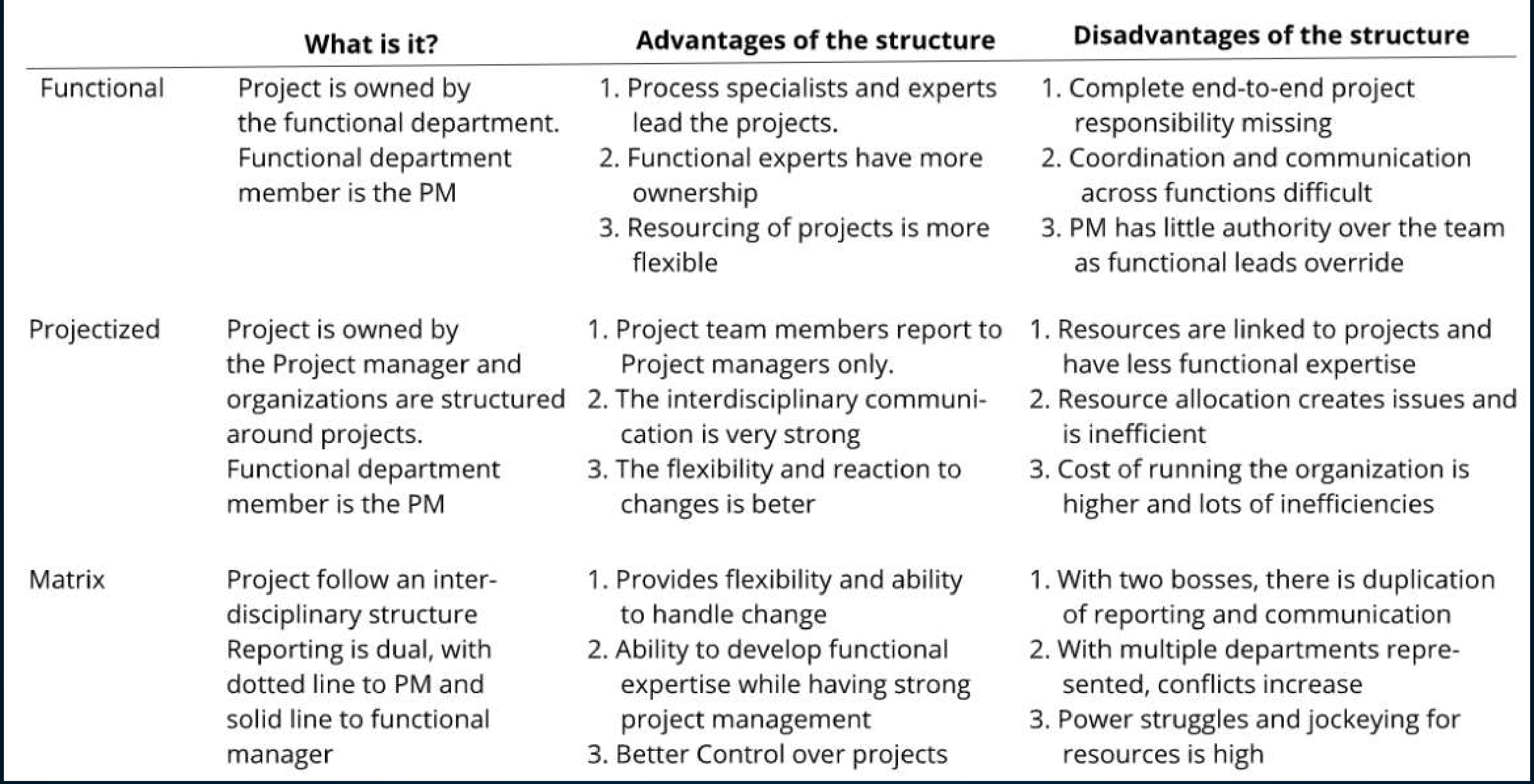
Project Characteristics

A definite start & end point

Once the end point is reached the project is over

It is attempting to achieve something new

Functional, Projectized, Matrix Organization



	What is it?	Advantages of the structure	Disadvantages of the structure
Functional	Project is owned by the functional department. Functional department member is the PM	1. Process specialists and experts lead the projects. 2. Functional experts have more ownership 3. Resourcing of projects is more flexible	1. Complete end-to-end project responsibility missing 2. Coordination and communication across functions difficult 3. PM has little authority over the team as functional leads override
Projectized	Project is owned by the Project manager and organizations are structured around projects. Functional department member is the PM	1. Project team members report to Project managers only. 2. The interdisciplinary communication is very strong 3. The flexibility and reaction to changes is beter	1. Resources are linked to projects and have less functional expertise 2. Resource allocation creates issues and is inefficient 3. Cost of running the organization is higher and lots of inefficiencies
Matrix	Project follow an interdisciplinary structure Reporting is dual, with dotted line to PM and solid line to functional manager	1. Provides flexibility and ability to handle change 2. Ability to develop functional expertise while having strong project management 3. Better Control over projects	1. With two bosses, there is duplication of reporting and communication 2. With multiple departments represented, conflicts increase 3. Power struggles and jockeying for resources is high

Your exam questions will be based on

1. Difference between – data and information, information and knowledge, knowledge and wisdom.
2. Typical patterns of software industry organization
3. Types of software industry based on extent of customer involvement.
4. Definition of a project.
5. Difference between a project and an operation.
6. Pattern on which software industries organize themselves.
7. Advantages and disadvantages of projectized organizations.

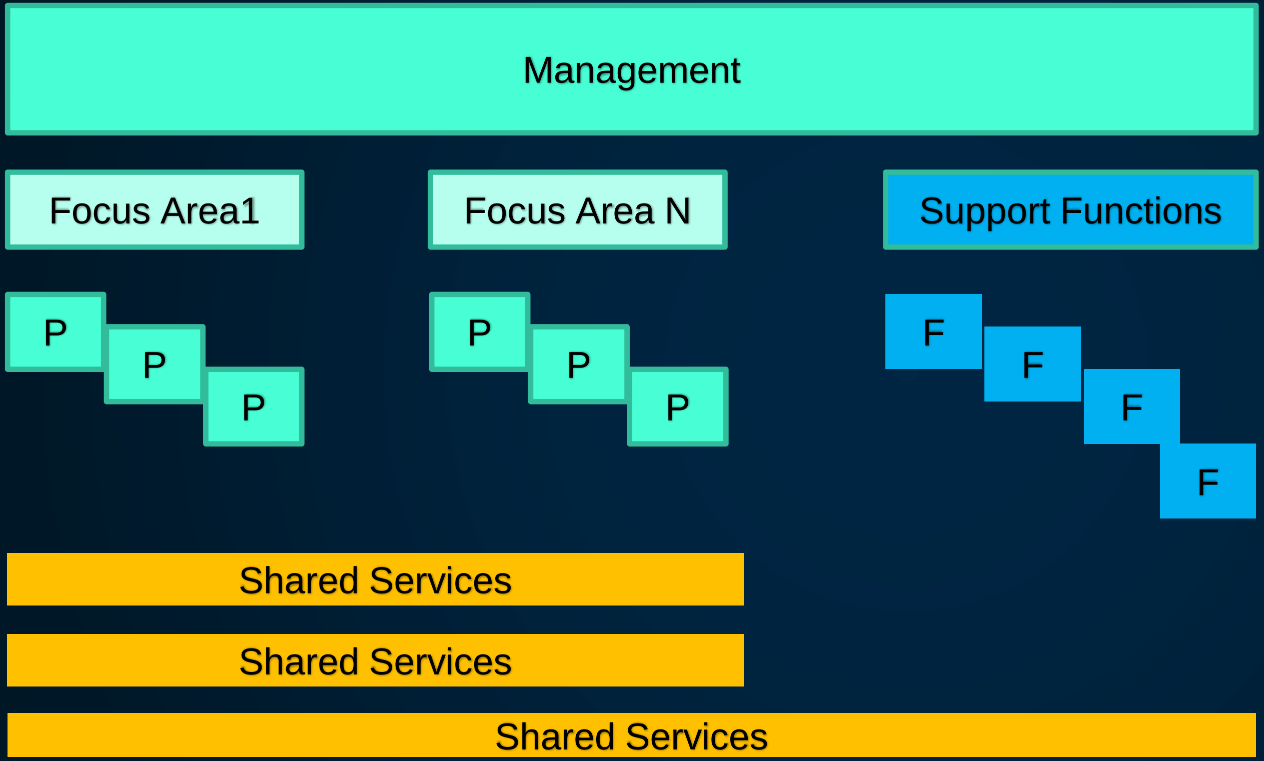
SIDDAGANGA INSTITUTE OF TECHNOLOGY
TUMKUR

Organization, Roles, Activities

Mahesha
BR Pandit

Unit 1.2
Mahesha BR Pandit
09-Oct-21, Version 1.0

General Organization of a Software Company



- Usually the upper layer is a management layer
- Focus Areas could be
  - Product, Consulting, Service
  - Geographies
  - Industries
  - Company’s own products
- Support Functions: HR, FA, LGL, PUR, IT, PMO, QAL, SEC, ADMIN, Traval

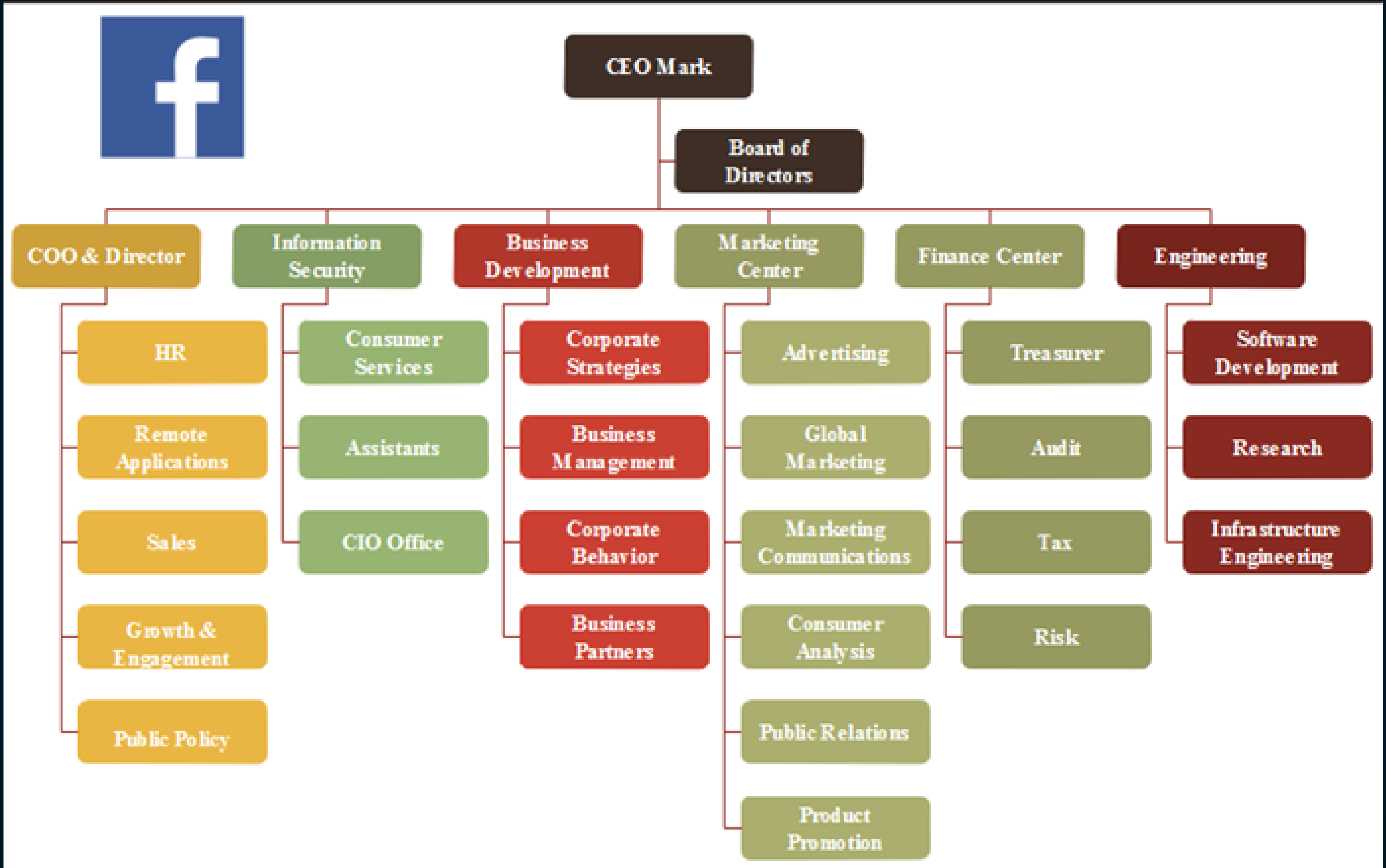
Support Functions

- Chief Executive
- Finance
- Operations
- Information
- Technology
- Accounting
- Privacy
- Security

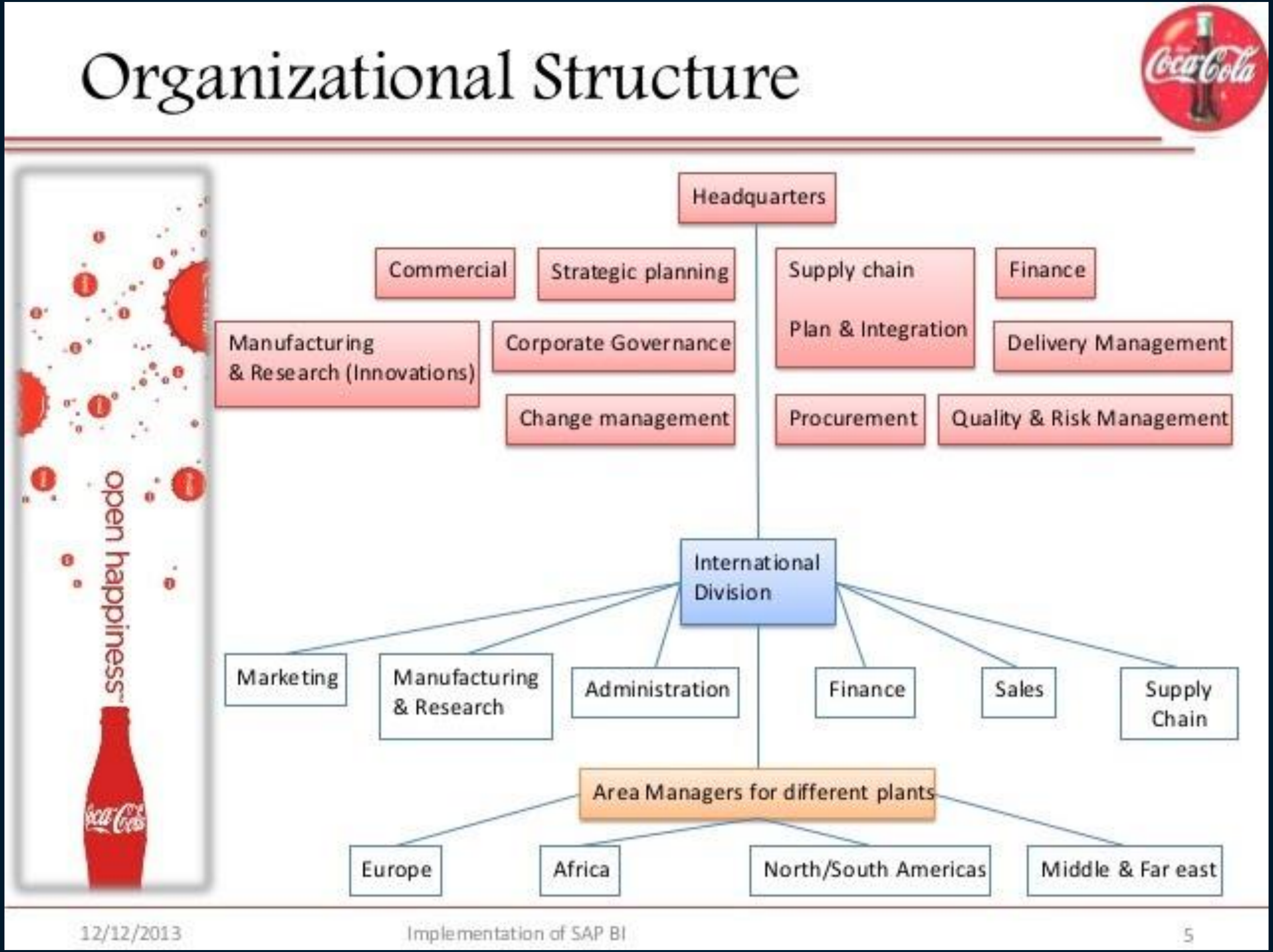
- Marketing
- Legal
- Business & Marketing
- Global Public Policy
- Investor Relations
- Product Management
- Human Resources

Popular Business Regions of the World

NA	North America
EMEA	Europe, the Middle East and Africa
APAC	Asia-Pacific
LATAM	Latin America
Africa	



- CEO Mark
  - Board of Directors
- COO & Director
  - HR
  - Remote Applications
  - Sales
  - Growth & Engagement
  - Public Policy
- Information Security
  - Consumer Services
  - Assistants
  - CIO Office
- Business Development
  - Corporate Strategies
  - Business Management
  - Corporate Behavior
  - Business Partners
- Marketing Center
  - Advertising
  - Global Marketing
  - Marketing Communications
  - Consumer Analysis
  - Public Relations
  - Product Promotion
- Finance Center
  - Treasurer
  - Audit
  - Tax
  - Risk
- Engineering
  - Software Development
  - Research
  - Infrastructure Engineering



Organizational Structure

Headquarters

Commercial

Strategic planning

Supply chain

Plan & Integration

Finance

Manufacturing & Research (Innovations)

Corporate Governance

Delivery Management

Change management

Procurement

Quality & Risk Management

International Division

Marketing

Manufacturing & Research

Administration

Finance

Sales

Supply Chain

Area Managers for different plants

Europe

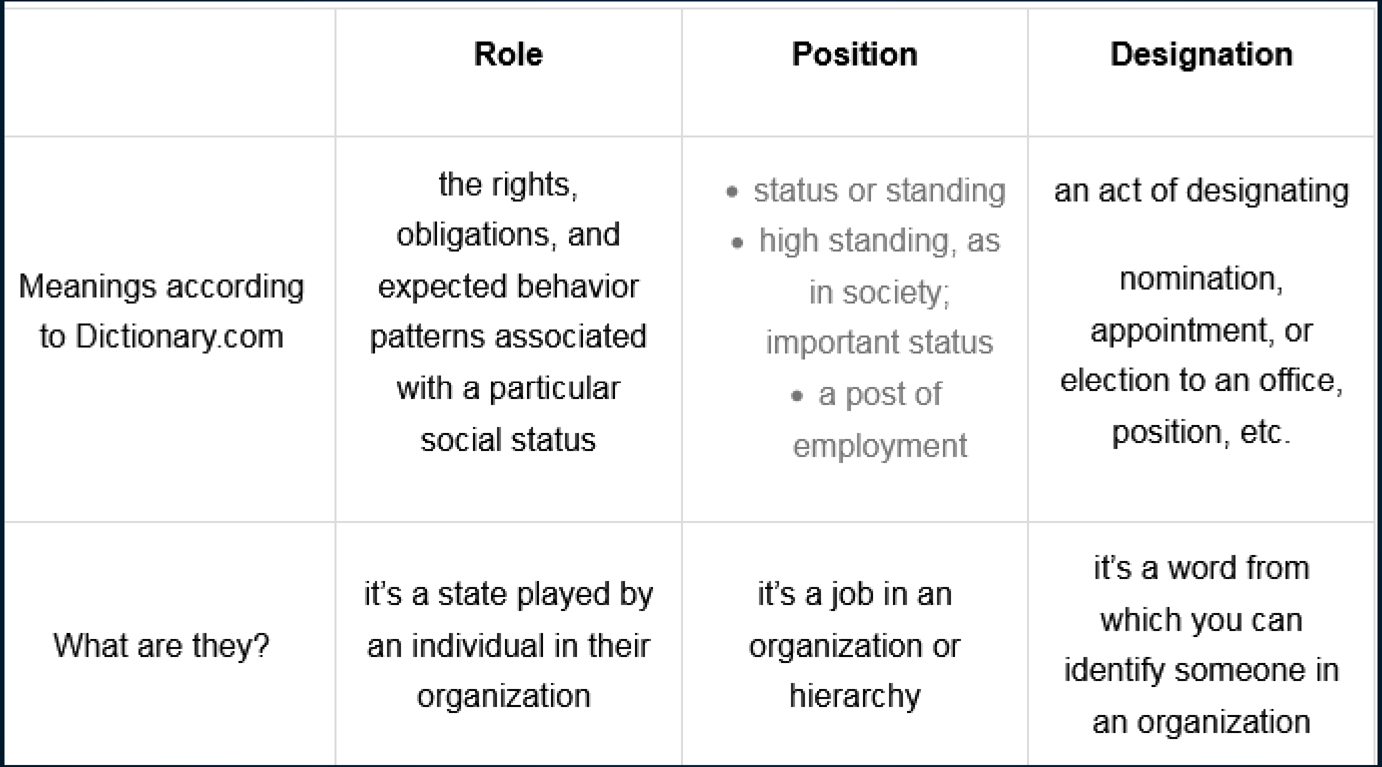
Africa

North/South Americas

Middle & Far east

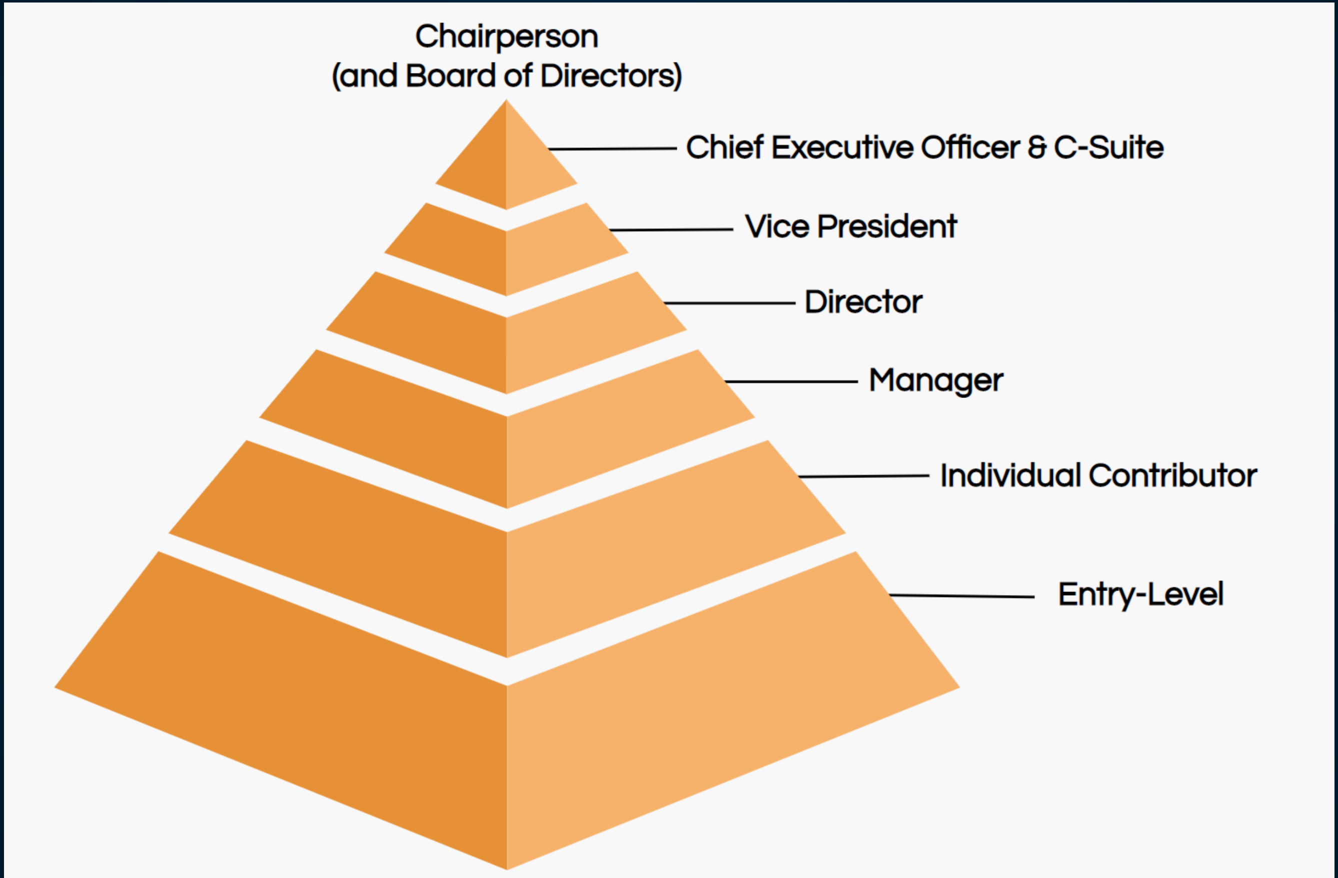
12/12/2013 Implementation of SAP BI 5

Role versus Designation



	Role	Position	Designation
Meanings according to Dictionary.com	the rights, obligations, and expected behavior patterns associated with a particular social status	• status or standing • high standing, as in society; important status • a post of employment	an act of designating nomination, appointment, or election to an office, position, etc.
What are they?	it's a state played by an individual in their organization	it's a job in an organization or hierarchy	it's a word from which you can identify someone in an organization

General Organization



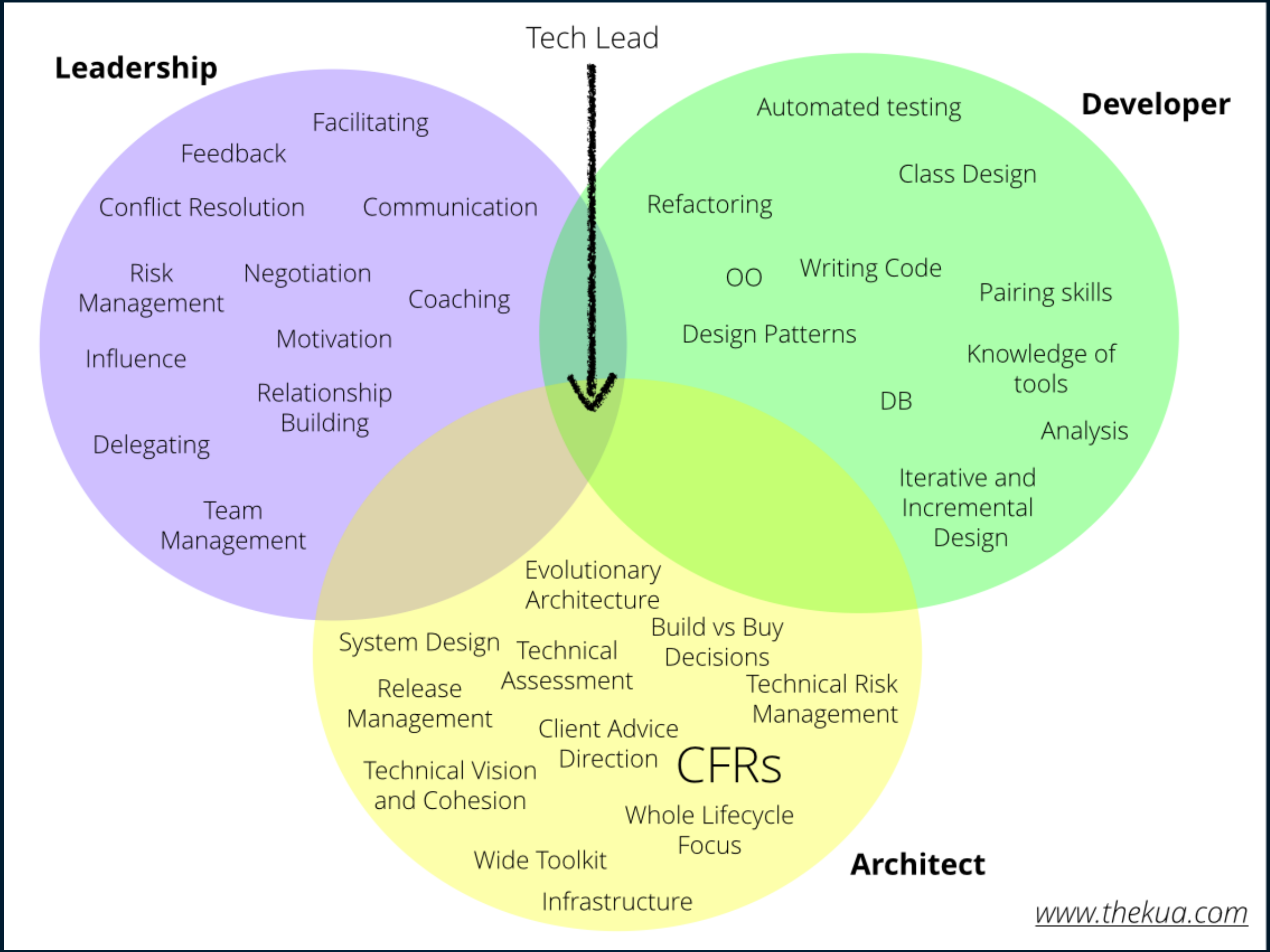
Technical Organization

- Project Manager
- Product Owner
- Team Leader
- Tech Leader
- Architect – business, solution, technical, enterprise
- Full Stack Developer
- Front End Developer
- Back End Developer
- QA Lead
- QA Engineer
- Tester
- UI Designer
- UX Designer

Few more terms

- MVP = Minimum Viable Product = Basic product which works
- Prototype
- Beta

Tech Lead



Tech Lead

Leadership

Facilitating
Feedback
Conflict Resolution
Communication
Risk Management
Negotiation
Coaching
Motivation
Influence
Relationship Building
Delegating
Team Management

Developer

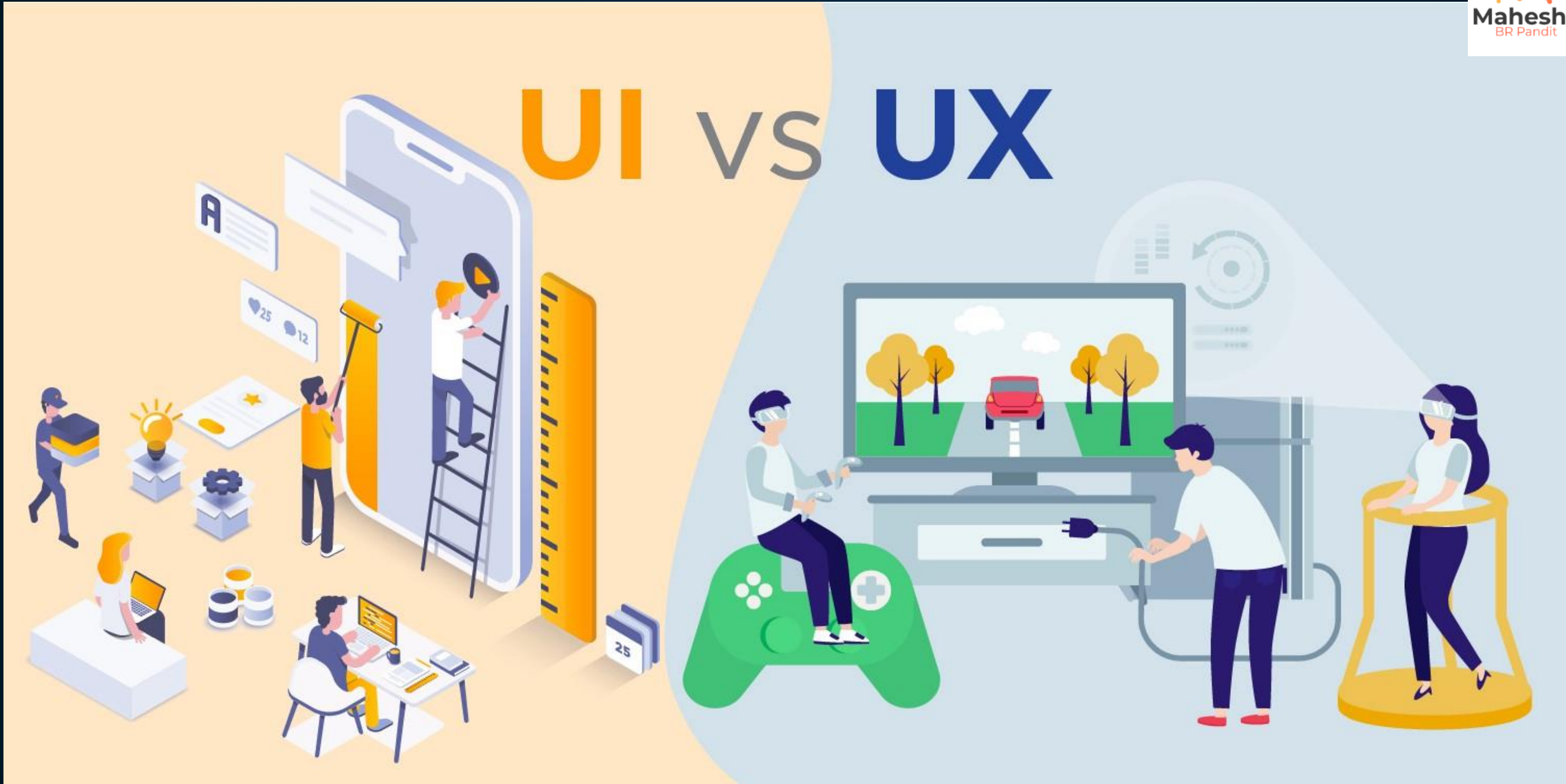
Automated testing
Class Design
Refactoring
OO
Writing Code
Pairing skills
Design Patterns
Knowledge of tools
DB
Analysis
Iterative and Incremental Design

Architect

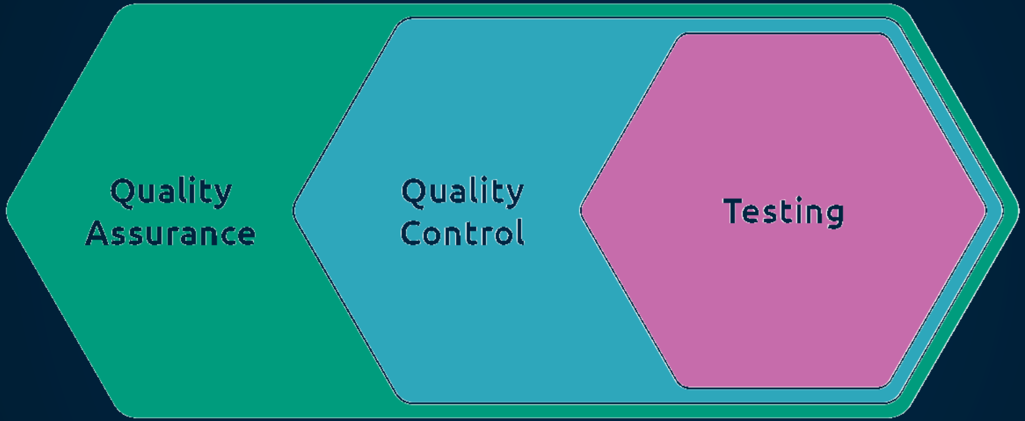
Evolutionary Architecture
System Design
Technical Assessment
Build vs Buy Decisions
Release Management
Technical Risk Management
Client Advice Direction
CFRs
Technical Vision and Cohesion
Whole Lifecycle Focus
Wide Toolkit
Infrastructure

www.thekua.com

UI vs UX



QA, QC, Testing



Question Bank – 1

1. Describe the General Organization of a Software Company.
2. Give examples of shared services.
3. Name the most common support functions.
4. Distinguish between Finance and Accounts.
5. Why does an IT company have an IT department within it?
6. Can Support Functions launch their own projects?
7. Name the popular business regions of the world
8. Distinguish between Role, Position and Designation with an example.
9. With the help of a Pyramidical structure, explain the general organizaition of a company.

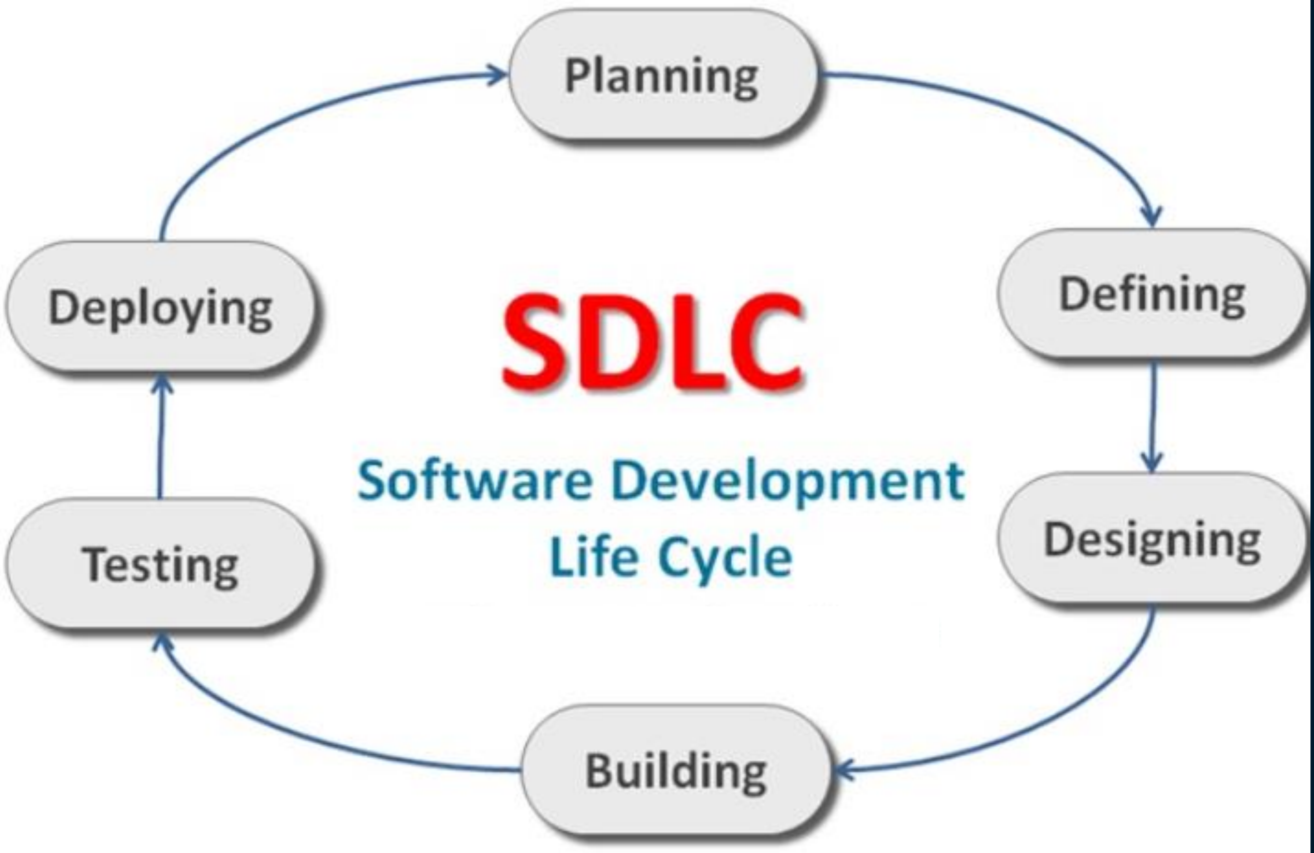
Question Bank – 2

10. Name the members of a technical organization.
11. Describe the role played by each member of the technical organization?
12. Can a project manager take a technical decision?
13. A computer science student is by definition, a full stack developer. Debate.
14. Distinguish between QA and QC and Testing
15. Distinguish between review and testing
16. Distinguish between QA Engineer and tester
17. Should the developer do testing?
18. Distinguish between UI and UX.
19. Describe the role played by a tech lead.

Processes within the Software Industry

Unit 1.3
Dr. Mahesha BR Pandit
12-Oct-21, Version 1.0

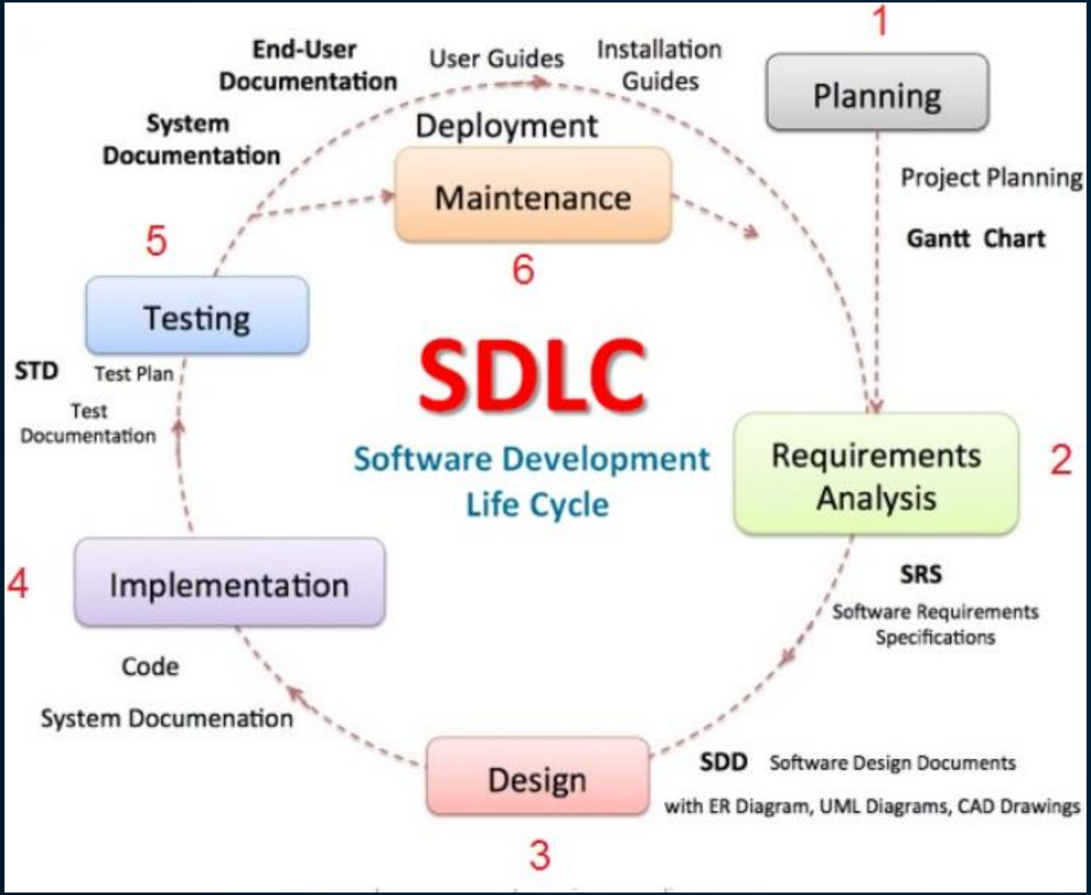
SDLC


Planning
Defining
Designing
Building
Testing
Deploying
SDLC
Software Development Life Cycle

Your Software Career

- Career Streams: Engineering, Technical, Process Oriented, Management
- Engineering: (Normal) – SE – SSE – TL – TM – PM – SPM – Director – VP – CIO, COO
- Process: SE (QA) – SSE (QA) – TL (QA) – PM – SPM – Director – VP -- COO
- Technical – Amazing – SE – SEE – TL – TA – EA-- – Tech. Director – VP(T) – CTO
- Business Process: BA – SBA – TL – PA – Director – VP – CIO, COO
- Business Management - PL – PM – SPM – Director – VP – COO
- COO, CIO, CMO, CDO ---- CEO
- Automation – SE (BA) – SEE (SBA) – TL – TA – EA-- – Tech. Director – VP(T) – CTO

Software Development Process: Generic



- Does this change with SDLC Model?
- Does it work for mobile app development?
- Does it hold good in agile setting?
- Is this old or no more valid?
- Will it change in the future?

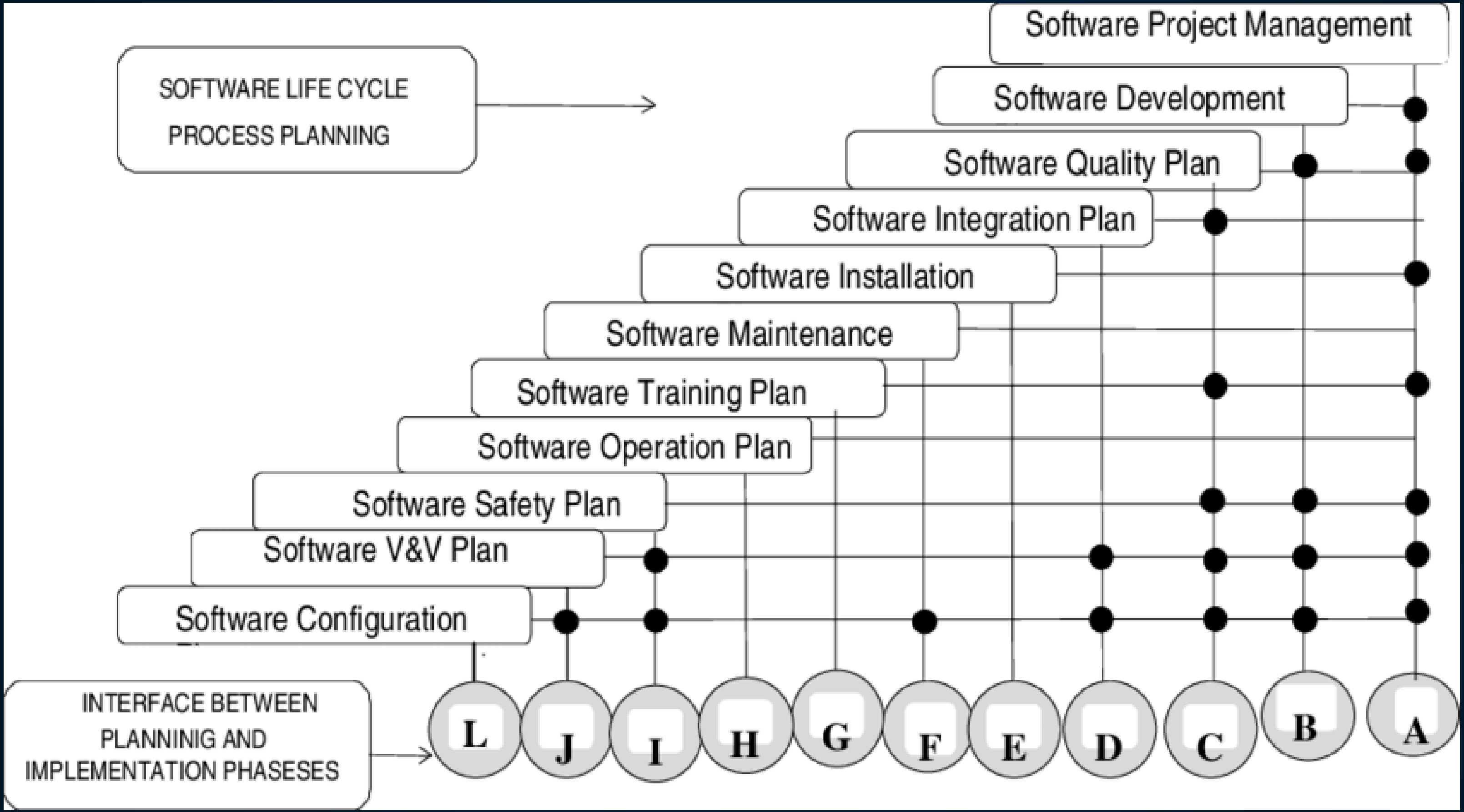
ASS U ME

Mother of all Processes: PDCA

PLAN
DO
CHECK
ACT
STANDARD
QUALITY

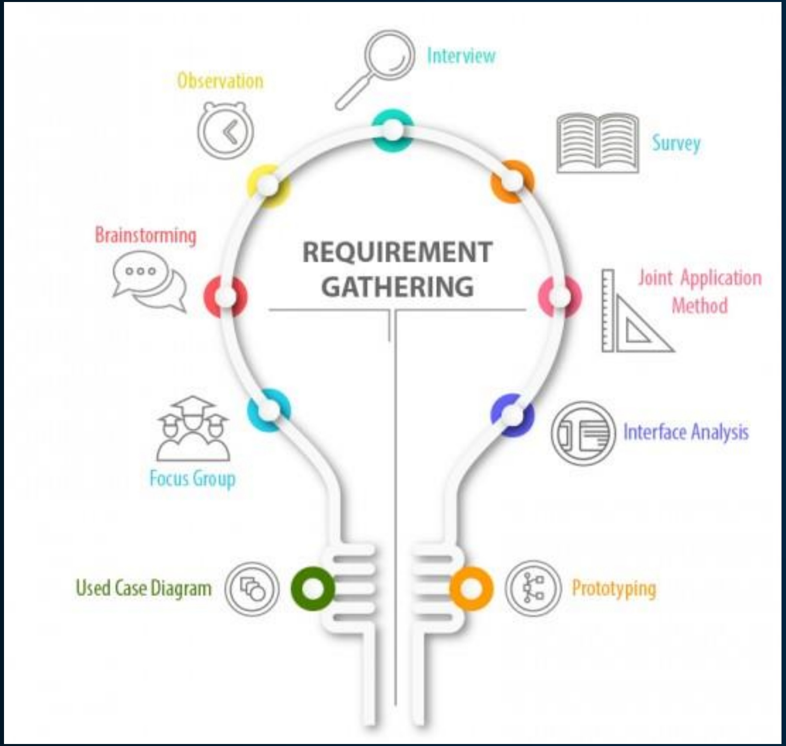
(c) Flipfine | Dreamstime.com

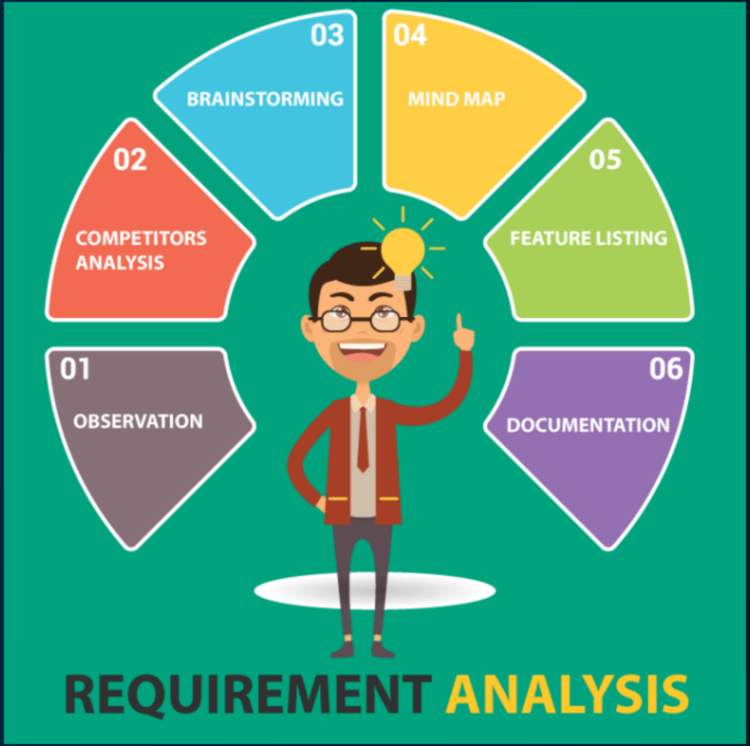
Planning



Modarres, Mohammad & Kececi, N.. (1998). Software Development Life Cycle Model to Ensure Software Quality.

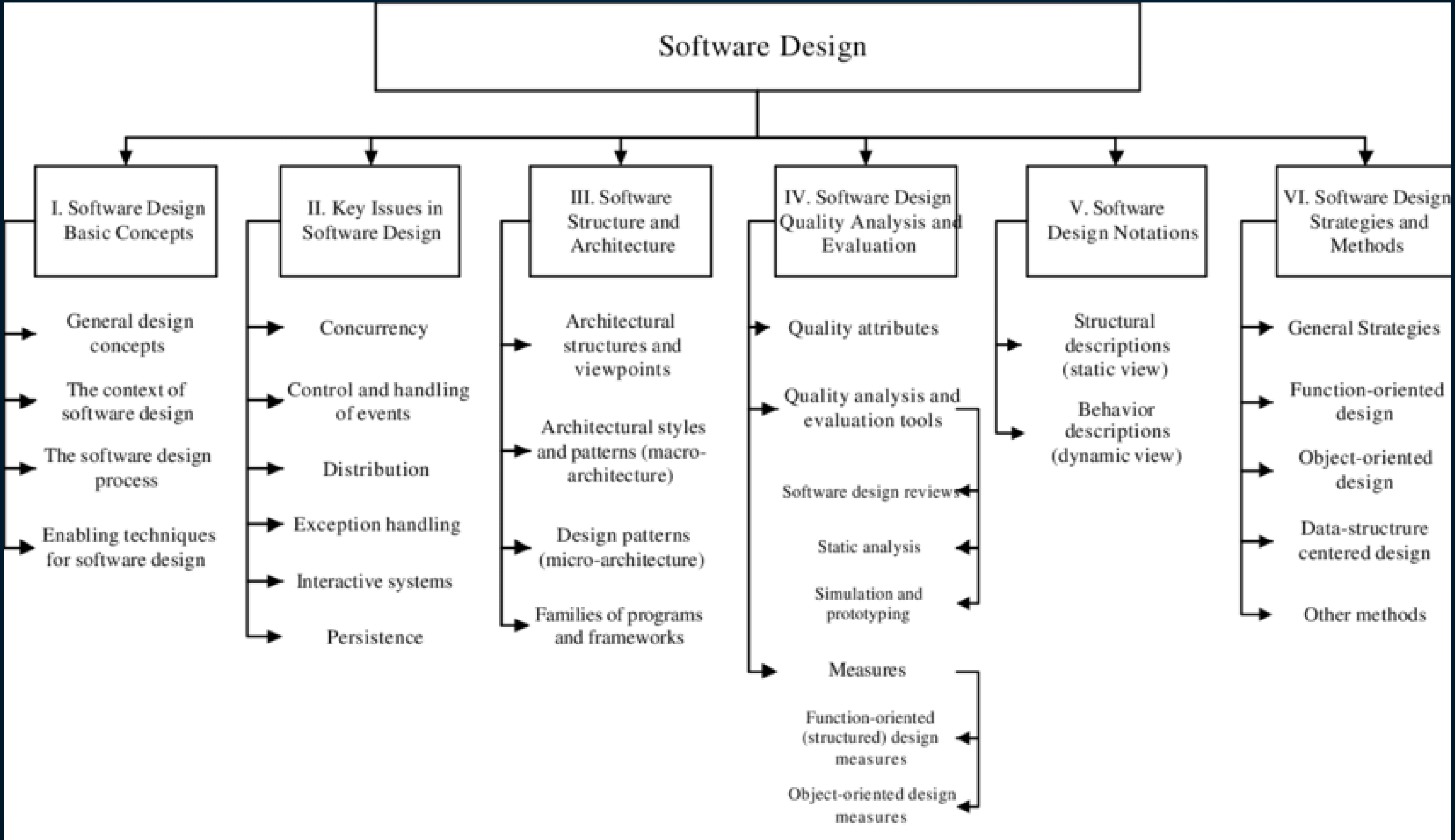
Requirements Analysis





https://nmgtechnologies.com/blog/requirement-gathering-solve-biggest-problems-consulting.html
https://www.fiverr.com/cpatel19184/provide-you-requirement-analysis-for-your-website-ideas

Software Design



Software Design

I. Software Design Basic Concepts
- General design concepts
- The context of software design
- The software design process
- Enabling techniques for software design

II. Key Issues in Software Design
- Concurrency
- Control and handling of events
- Distribution
- Exception handling
- Interactive systems
- Persistence

III. Software Structure and Architecture
- Architectural structures and viewpoints
- Architectural styles and patterns (macro-architecture)
- Design patterns (micro-architecture)
- Families of programs and frameworks

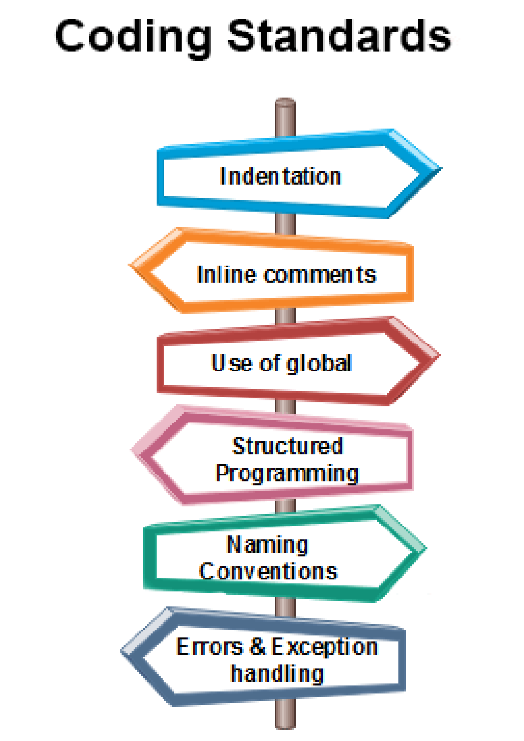
IV. Software Design Quality Analysis and Evaluation
- Quality attributes
- Quality analysis and evaluation tools
  - Software design reviews
  - Static analysis
  - Simulation and prototyping
- Measures
  - Function-oriented (structured) design measures
  - Object-oriented design measures

V. Software Design Notations
- Structural descriptions (static view)
- Behavior descriptions (dynamic view)

VI. Software Design Strategies and Methods
- General Strategies
- Function-oriented design
- Object-oriented design
- Data-structure centered design
- Other methods

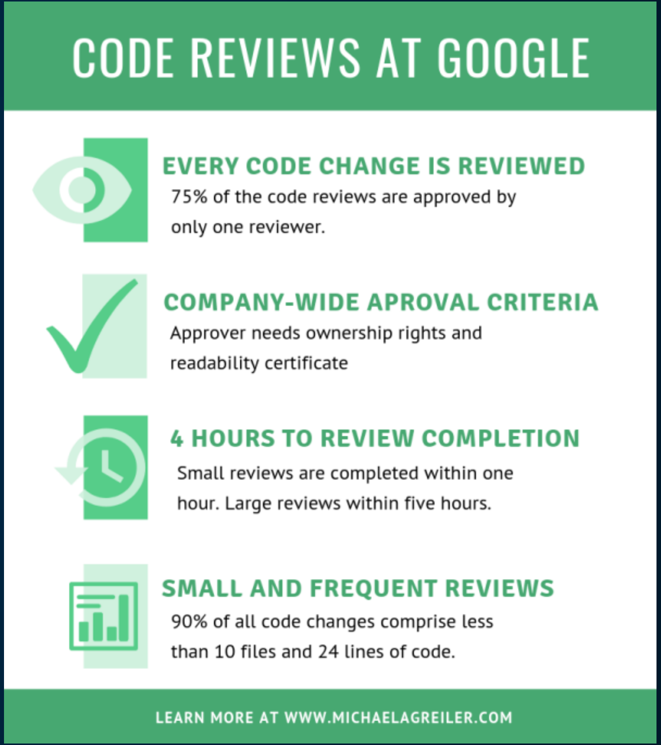
Abran, Alain & Moore, James & Bourque, Pierre & Dupuis, Robert. (2001). Guide to the Software Engineering Body of Knowledge - Trial Version (Version 0,9). 10.13140/RG.2.2.12202.44489.

Coding Standards



Coding Standards

- Indentation
- Inline comments
- Use of global
- Structured Programming
- Naming Conventions
- Errors & Exception handling



CODE REVIEWS AT GOOGLE

EVERY CODE CHANGE IS REVIEWED

75% of the code reviews are approved by only one reviewer.

COMPANY-WIDE APROVAL CRITERIA

Approver needs ownership rights and readability certificate

4 HOURS TO REVIEW COMPLETION

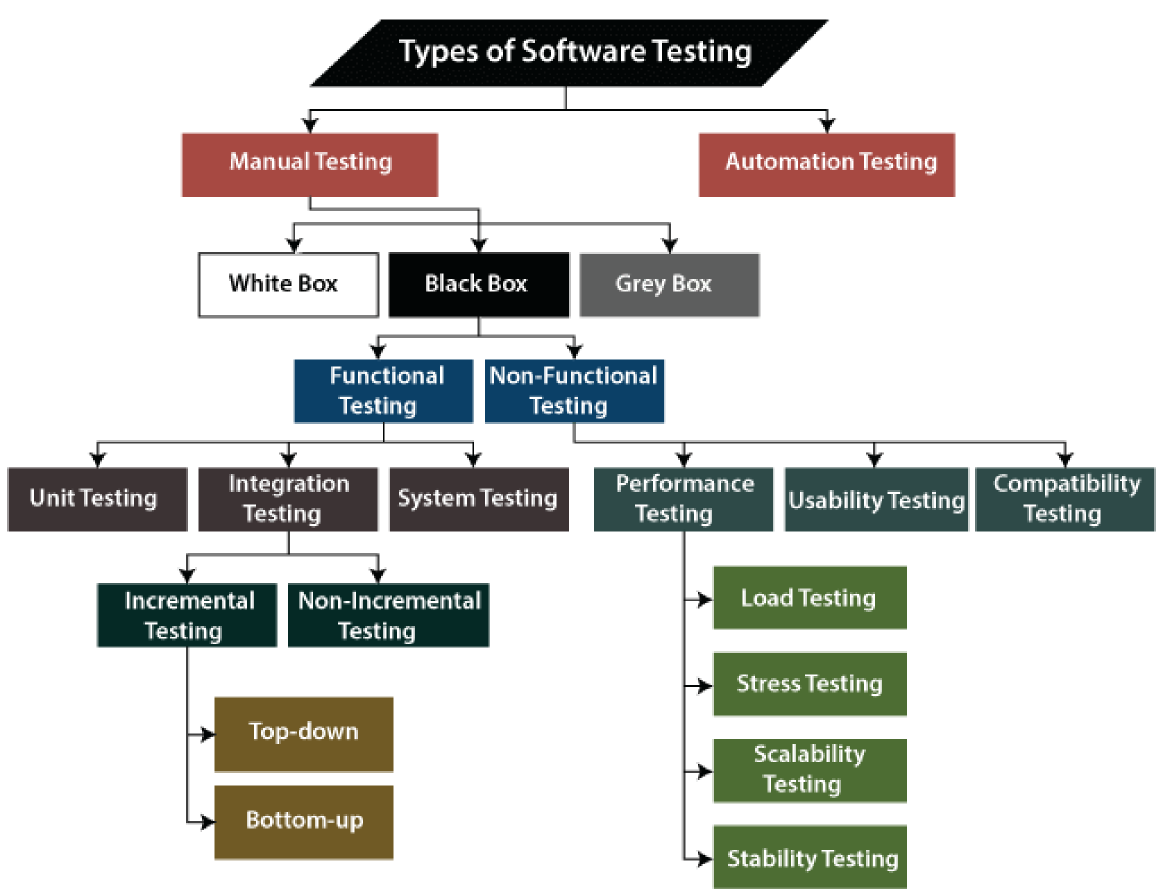
Small reviews are completed within one hour. Large reviews within five hours.

SMALL AND FREQUENT REVIEWS

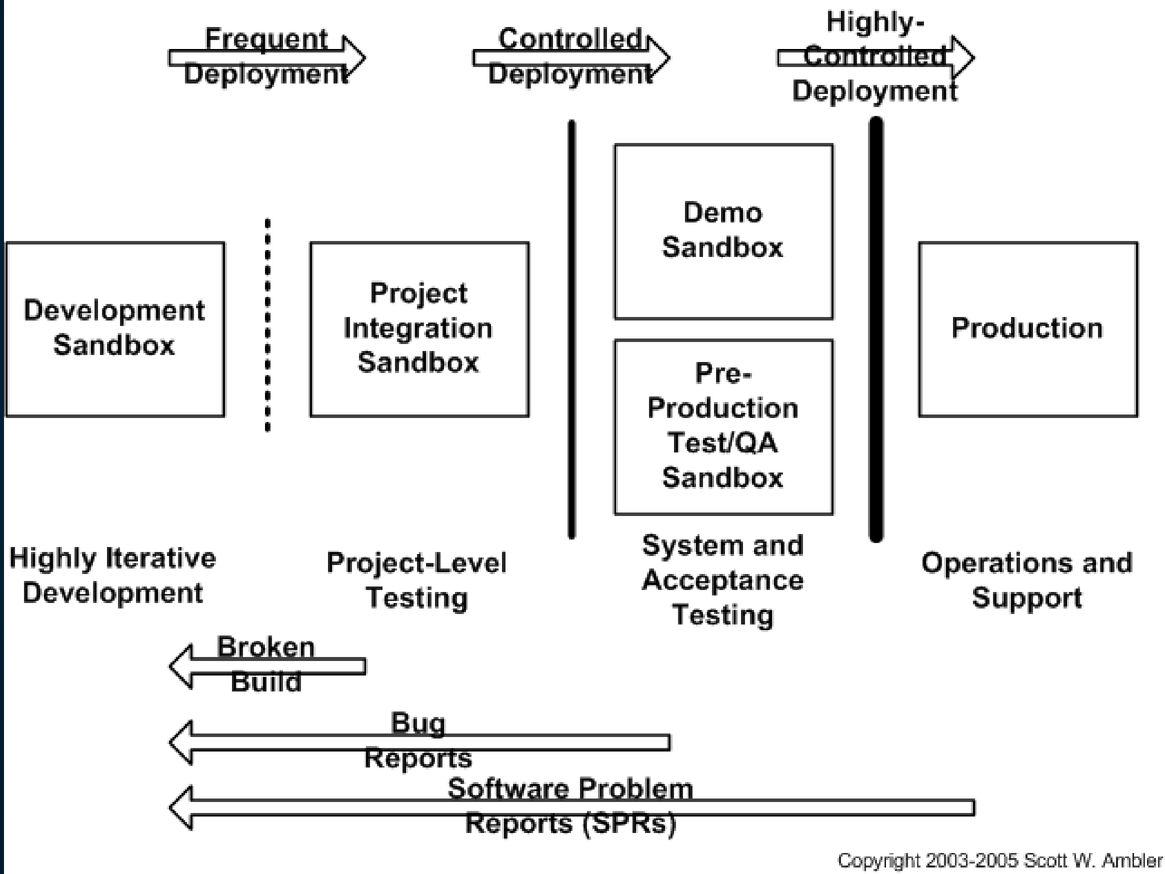
90% of all code changes comprise less than 10 files and 24 lines of code.

LEARN MORE AT WWW.MICHAELAGREILER.COM

Software Testing

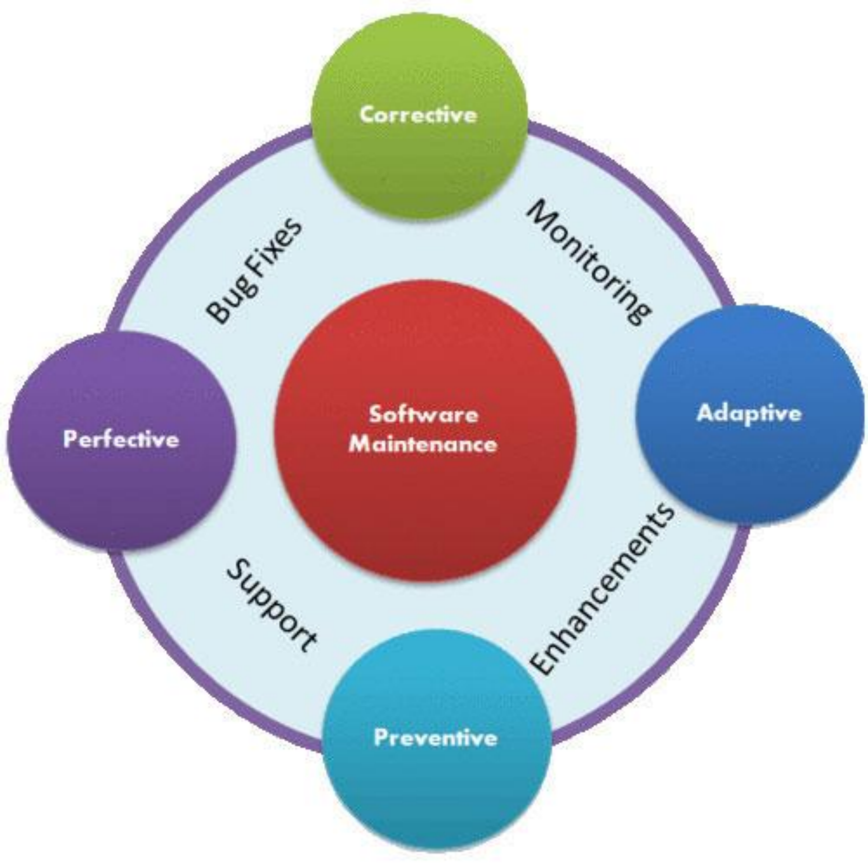


Software Deployment

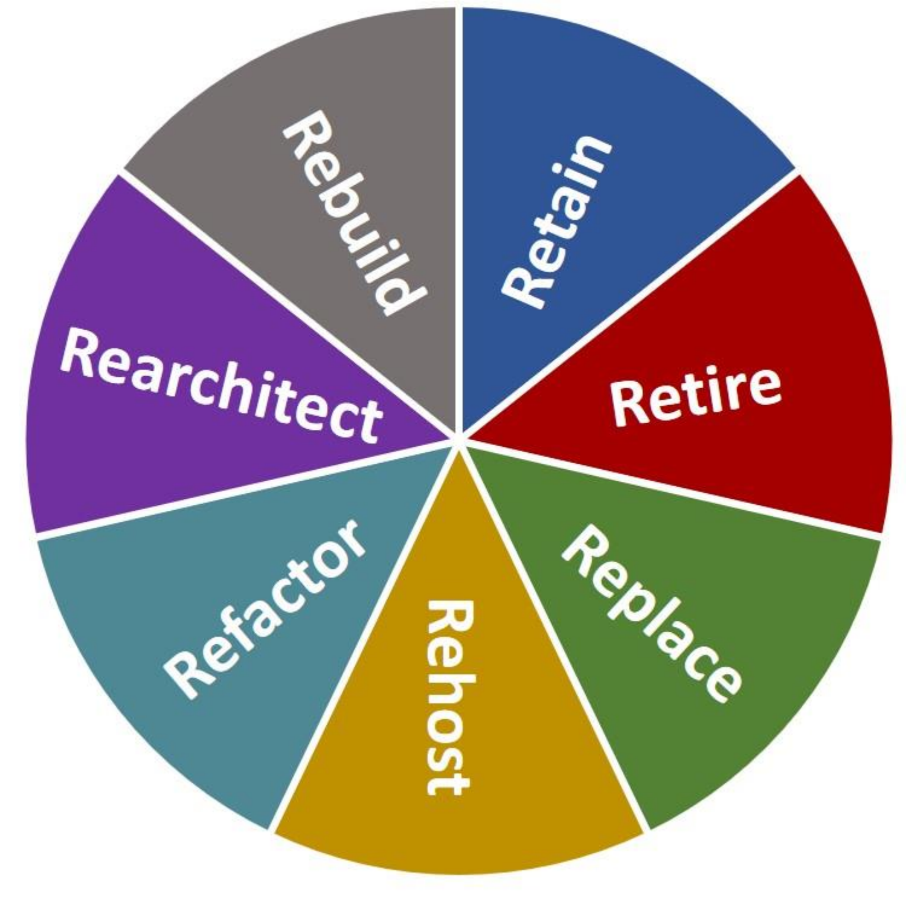

Frequent Deployment
Controlled Deployment
Highly-Controlled Deployment
Development Sandbox
Project Integration Sandbox
Demo Sandbox
Pre-Production Test/QA Sandbox
Production
Highly Iterative Development
Project-Level Testing
System and Acceptance Testing
Operations and Support
Broken Build
Bug Reports
Software Problem Reports (SPRs)

Copyright 2003-2005 Scott W. Ambler

Software Maintenance



Seven R Strategy



Question Bank – 1

1. What is a software development lifecycle model?
2. Name a few software development lifecycle models.
3. Describe six acivities of SDLC.
4. Explain in detail, the six steps of SDLC indicating inputs and outputs.
5. Do steps of SDLC change with SDLC model?
6. Does the 6 step SDLC model work for mobile app development?
7. Does the 6 step SDLC model hold good in agile setting?
8. What might be the future of the 6 step SDLC model?
9. Is there a standard process for achieving continuous improvement?
10. With the help of a picture, describe the standard process for achieving continuous improvement.

Question Bank – 2

11. Name components of a software plan.
12. Are the components of a software plan inter-related? Give examples.
13. To analyse requirements, you should first ________ it.
14. Explain the requirements gathering process.
15. Does requirement gathering process change between agile and traditional SDLCs?
16. Explain the requirements analysis process.
17. Distinguish between Requirements Gathering and Requirements Analysis processes.
18. Describe the code review process.
19. What are coding standards? Name a few of them.

Question Bank – 3

20. Can code reviews be done automatically? Explain with an example.
21. Can machine review software code written by humans?
22. What is the difference between review and testing?
23. Is software code review a type of software testing?
24. What is the main objective of software testing?
25. Draw the classification tree of software testing.
26. Distinguish between whitebox, gray box and black box testing.
27. What inputs do you need for whitebox, gray box and black box testing?
28. What is a sandbox?
29. Name and explain various types of sandboxes
30. In which sandbox unit testing is to be performed?

Question Bank – 4

31. In which sandbox integration testing is to be performed?
32. In which sandbox customer acceptance testing is to be performed?
33. Name the tests that can be performed on the production environment.
34. What is software maintenance?
35. What are the various types of software maintenance?
36. Describe the seven R strategy
37. What is code re-factoring?
38. An old web application software was converted to a mobile application platform. Name the maintenance strategy employed here.

Defects and SQA

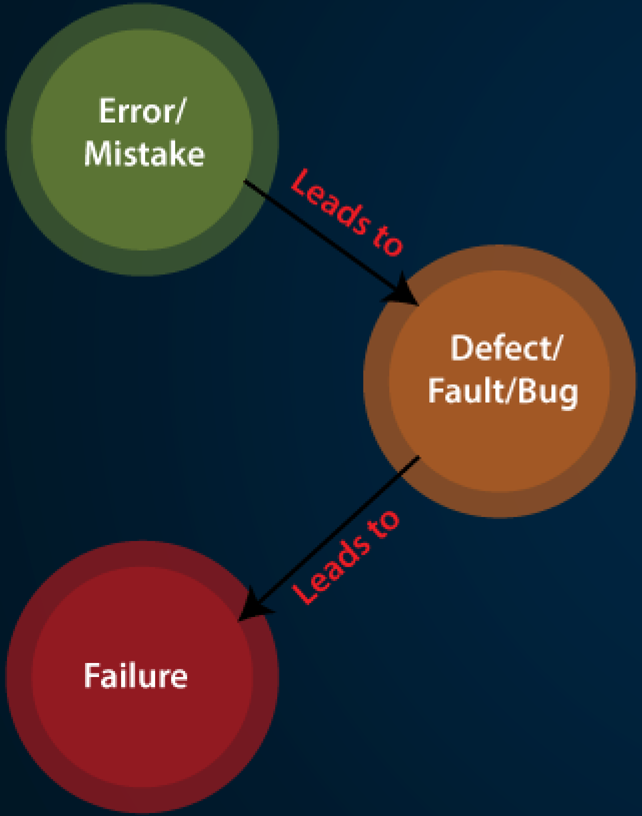
Unit 1.4
Dr. Mahesha BR Pandit
26-Oct-21, Version 1.0

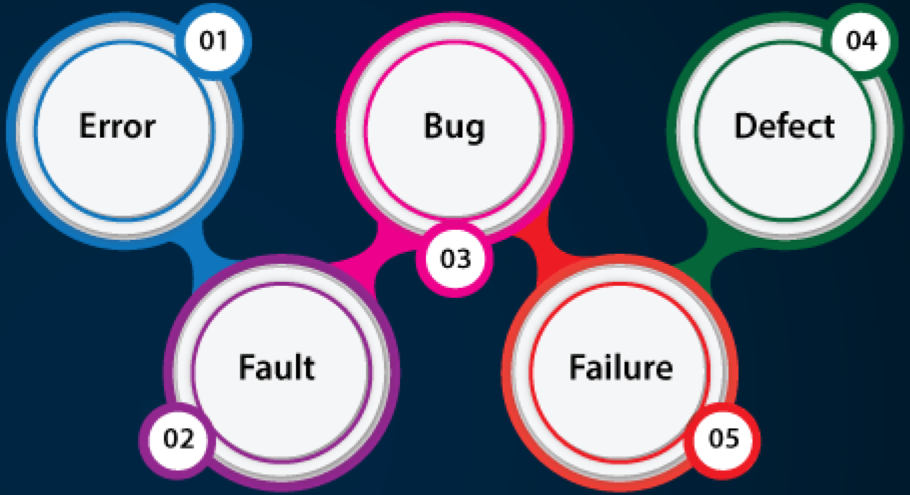
The Software Defect

- Deviation from specifications
- Major problem of the software industry
- Natural side effect of human effort
- No such thing as “zero defect software”
- Cost of Fixing = nearly 47% of the original effort
- It has its own life cycle
- There are techniques to prevent it
- It must be caught as early as possible
- World spends 4 T$ on software every year – 2.8 T$ is spent on fixing defects!
- Largest non-nuclear explosion = Software defect
- Read about Ariane Rocket Failure (France) Loss of 8 Billion USD because of numeric overflow error in one line of software!

WANTED
Blocker
Blocker
Critical
Critical
Major
Major
Minor
Minor
Trivial
Trivial
DEAD OR ALIVE

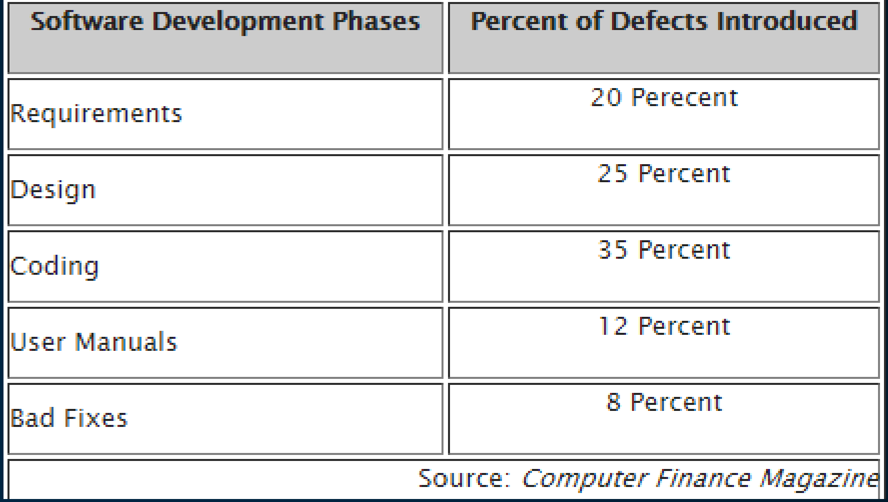
Terminology





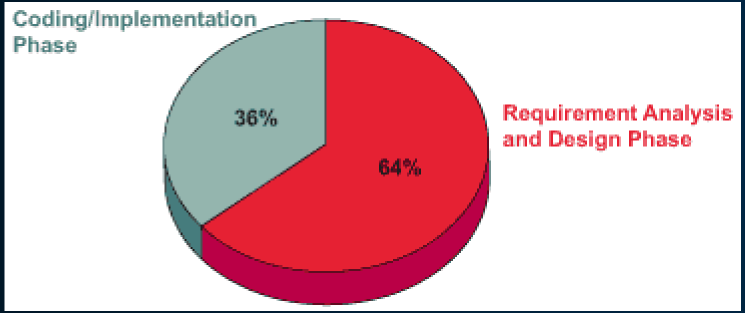
1. **Error**: Mistake made in the code
2. **Fault:** A state that causes the software to fail
3. **Bug**: An informal name for defect
4. **Defect**: difference between the actual outcomes and expected outputs
5. **Failure**: A state in which the software is unusable

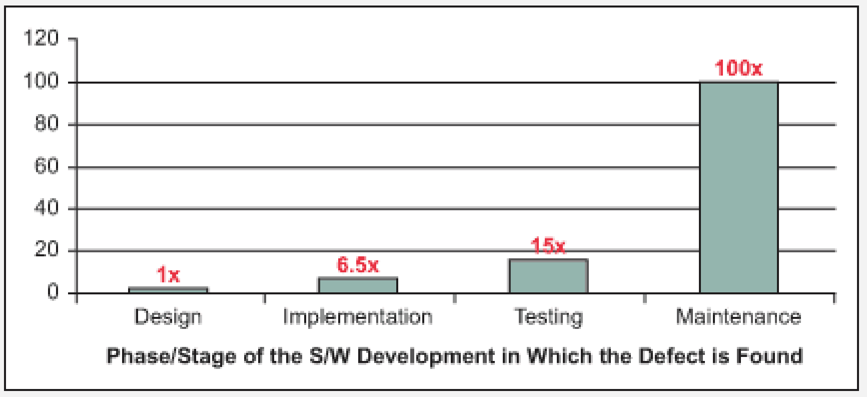
Defect Injection



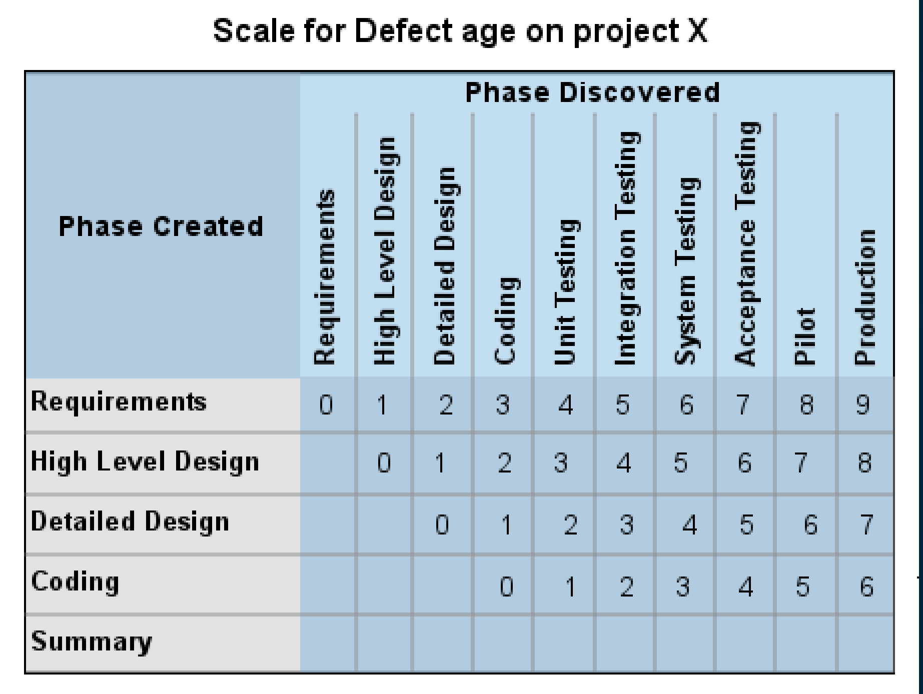
Software Development Phases	Percent of Defects Introduced
Requirements	20 Perecent
Design	25 Percent
Coding	35 Percent
User Manuals	12 Percent
Bad Fixes	8 Percent

Source: *Computer Finance Magazine*



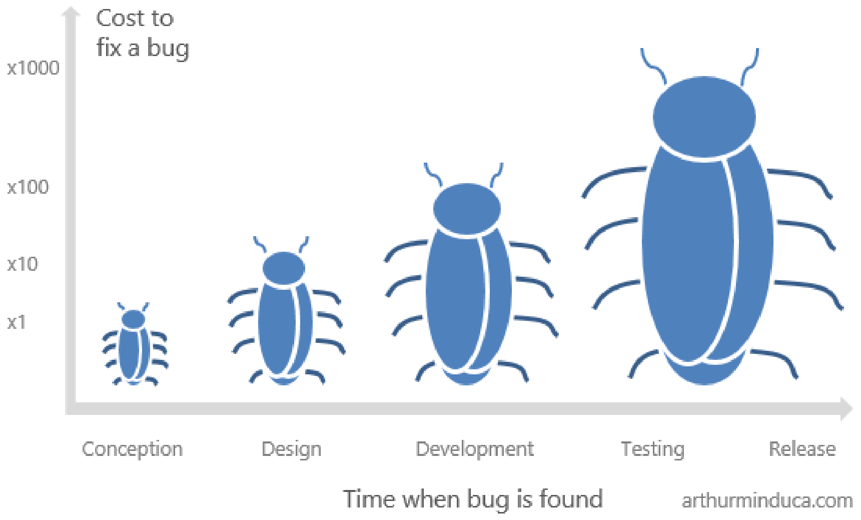


Catch it As Early As Possible



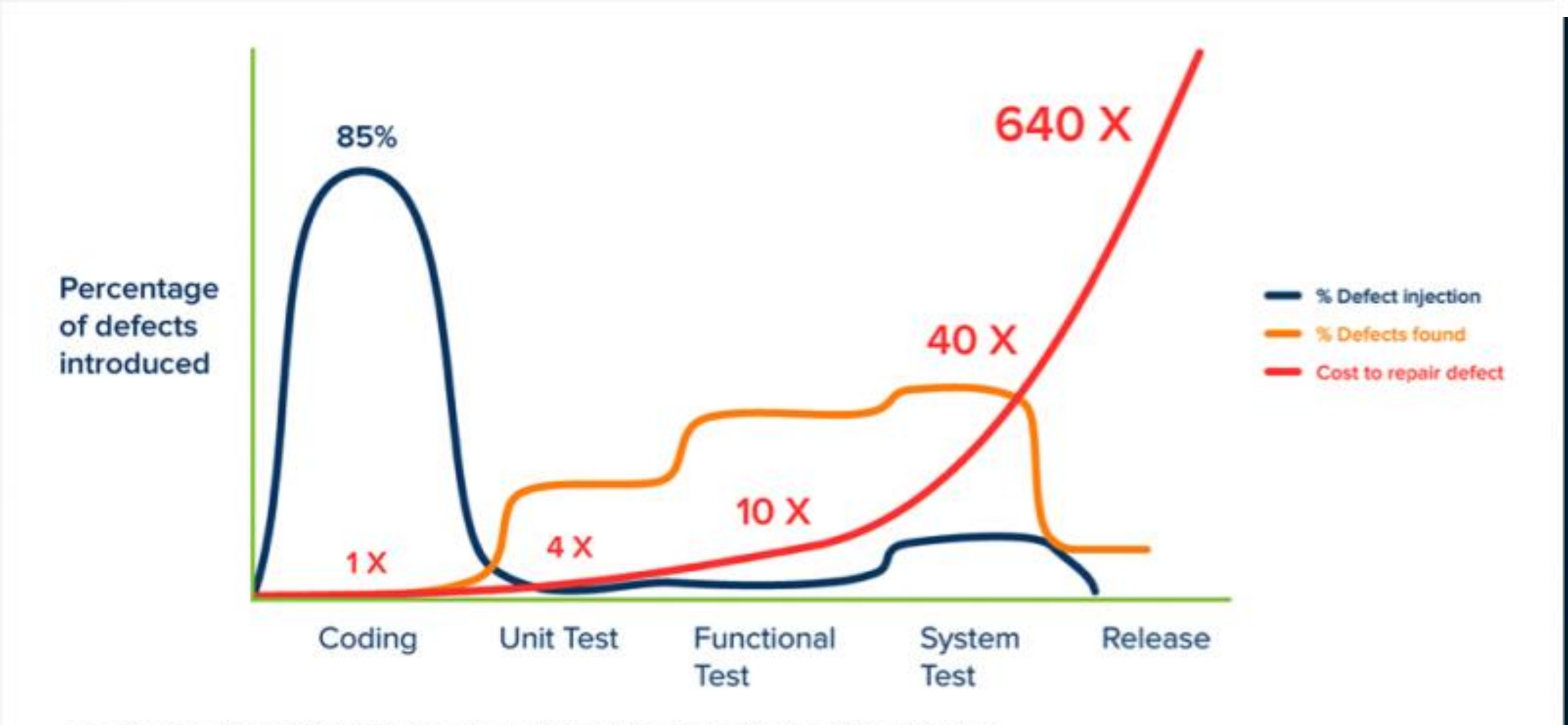
Scale for Defect age on project X

Phase Created	Phase Discovered									
	Requirements	High Level Design	Detailed Design	Coding	Unit Testing	Integration Testing	System Testing	Acceptance Testing	Pilot	Production
Requirements	0	1	2	3	4	5	6	7	8	9
High Level Design		0	1	2	3	4	5	6	7	8
Detailed Design			0	1	2	3	4	5	6	7
Coding				0	1	2	3	4	5	6
Summary										



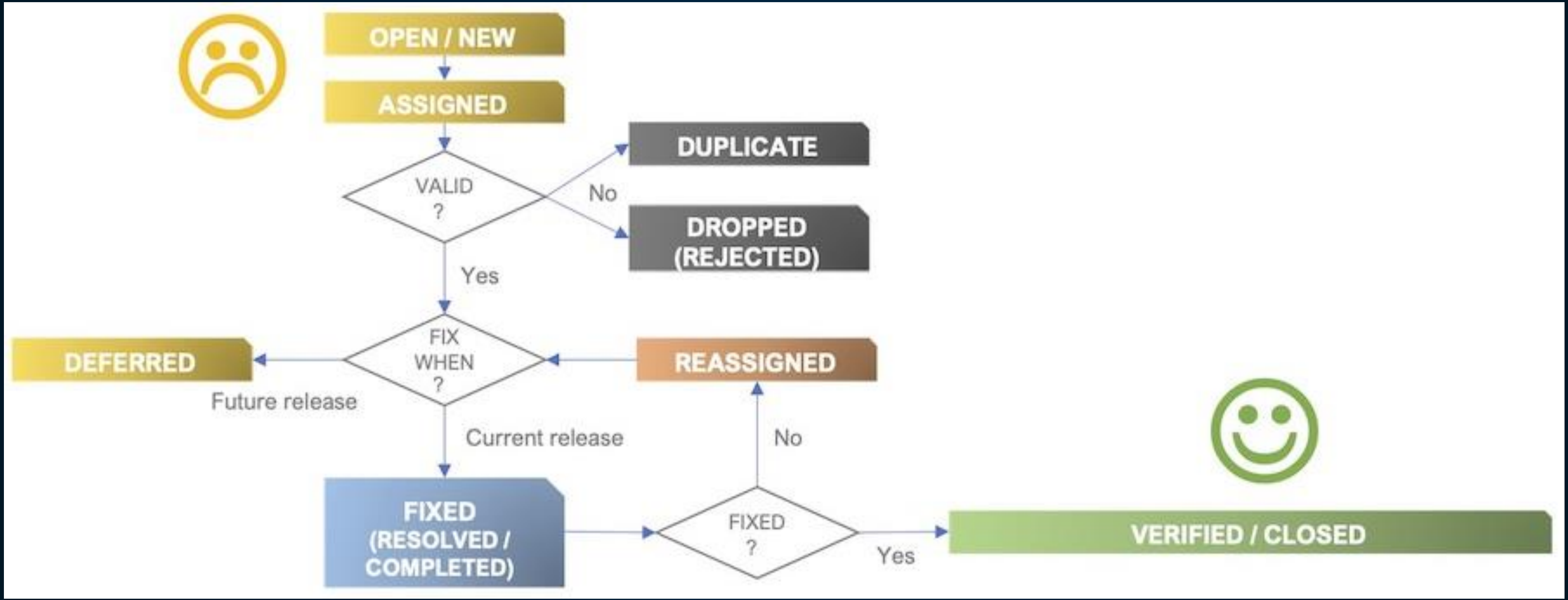
arthurminduca.com

Catch it As Early As Possible

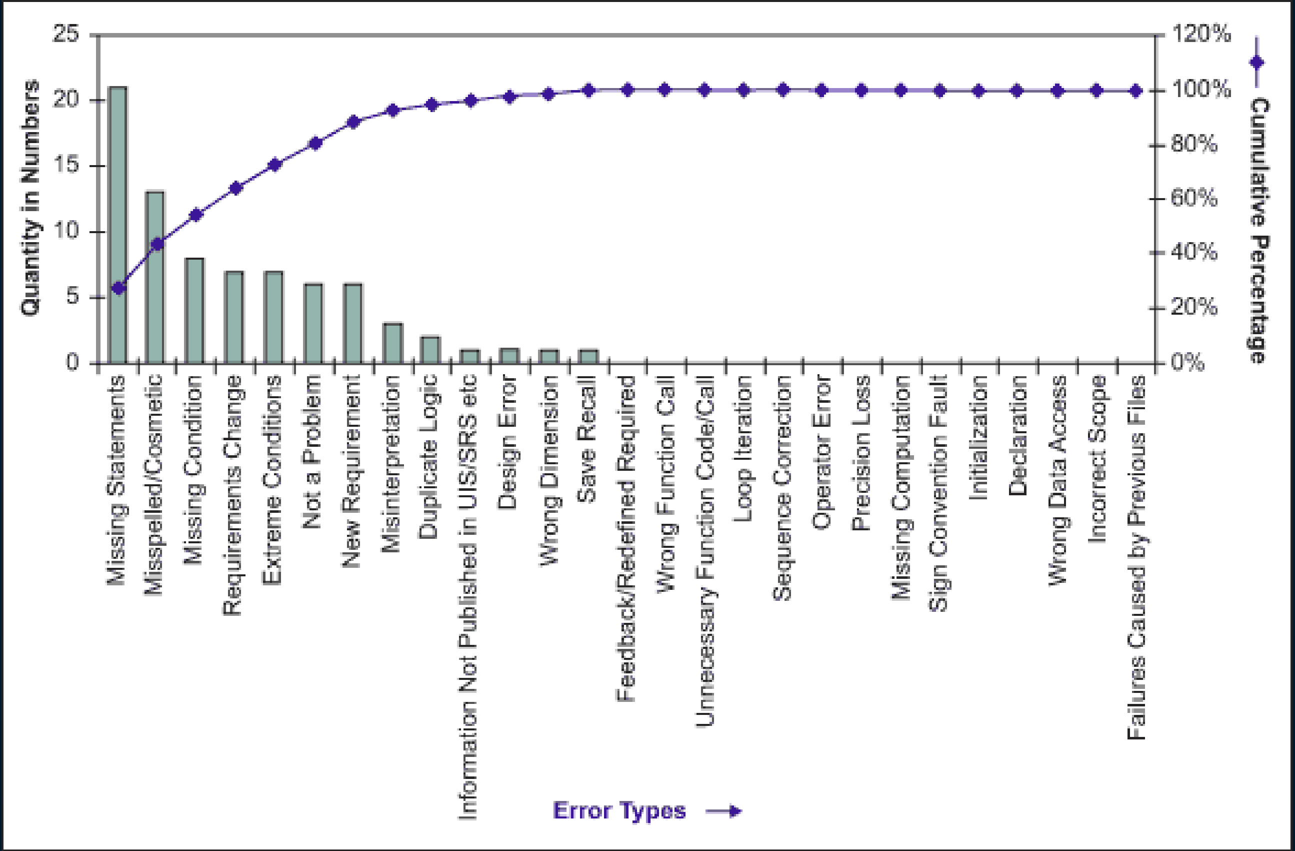


Jones, Capers. *Applied Software Measurement: Global Analysis of Productivity and Quality.*

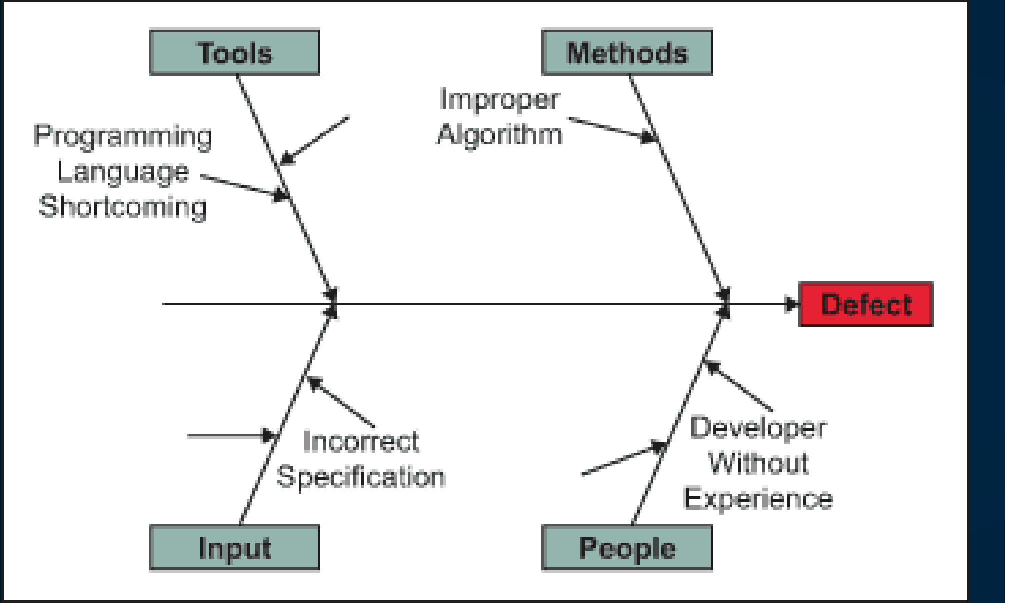
Defect Life Cycle



Types of Defects

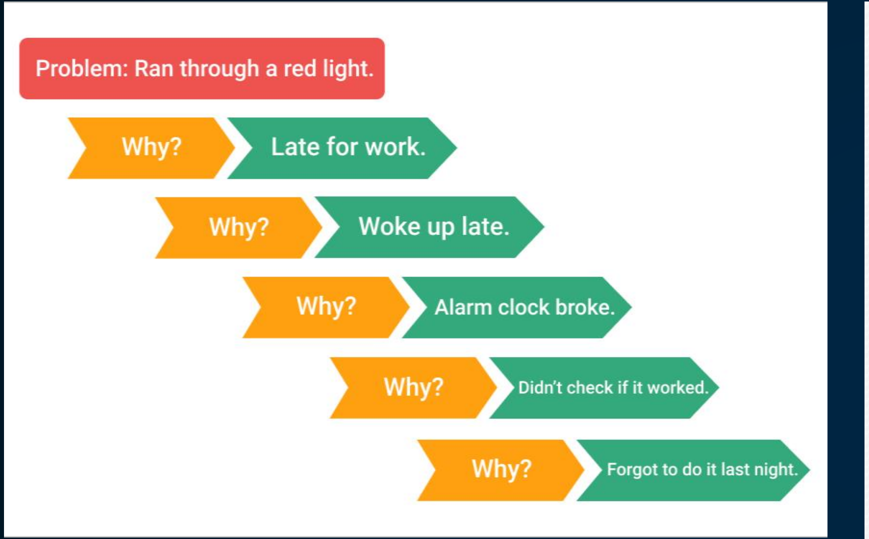


What Causes Defects?



Ishikawa Diagram / Fishbone

Root Cause Analysis



Problem: Ran through a red light.

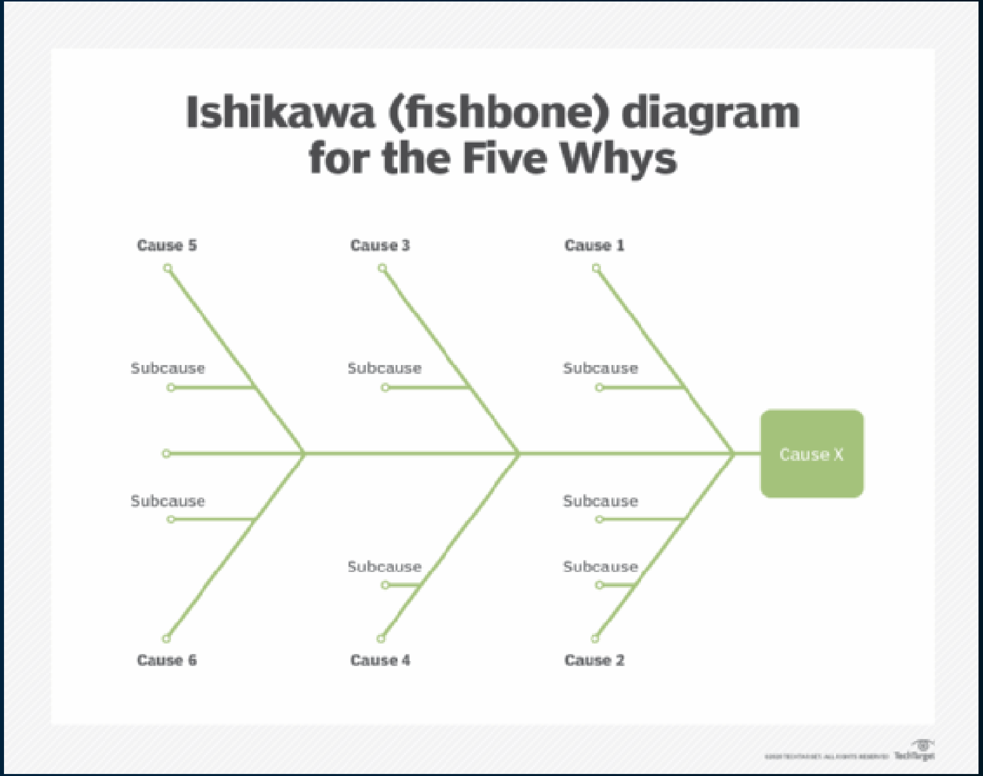
Why? Late for work.

Why? Woke up late.

Why? Alarm clock broke.

Why? Didn't check if it worked.

Why? Forgot to do it last night.



Ishikawa (fishbone) diagram for the Five Whys

Someone: “Hey, give me RC and CAPAs”

Someone = QA department

RC = Root Cause

CAPA = CA + PA

CA = Corrective Action

PA = Preventive Action

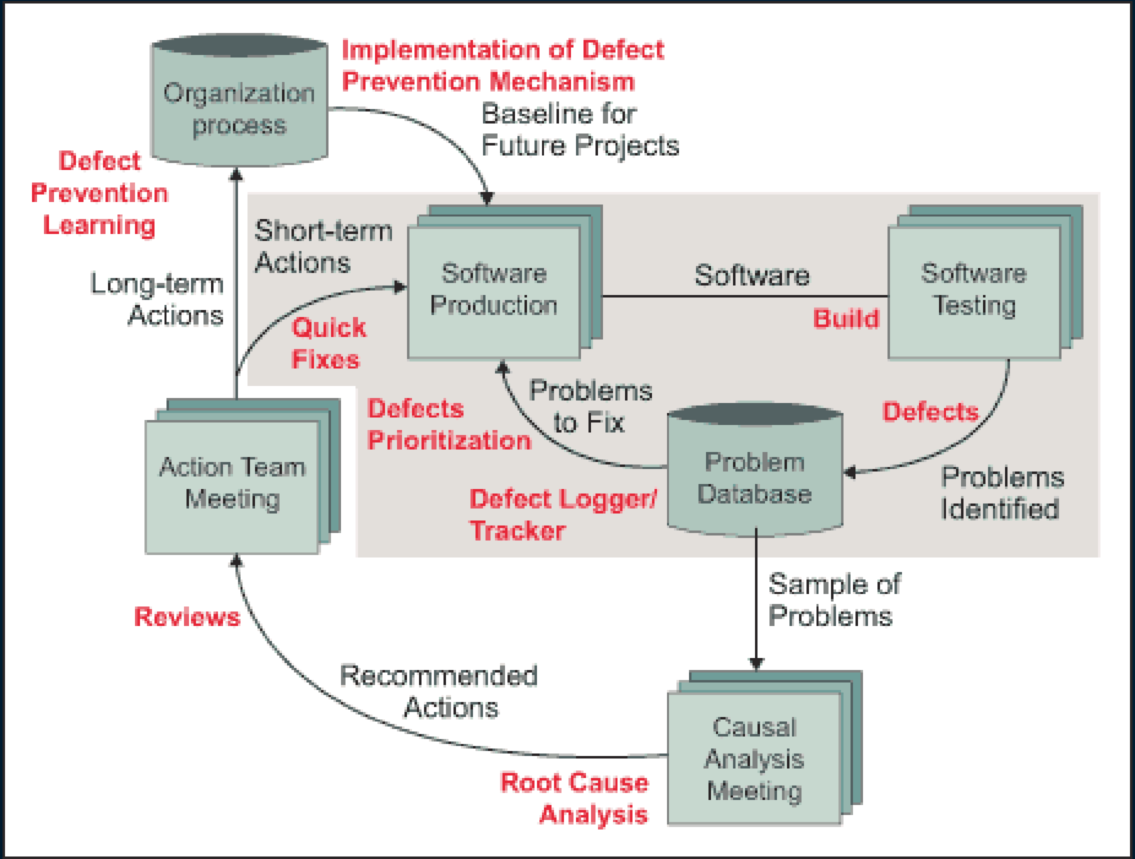
Root Causes for Software Defects

- “As many as 35% of production defects are caused by requirements problems” – Accenture, Jan 2021, Software Intelligence Forum.
- Miscommunication of requirements introduces error in code
- Unrealistic time schedule for development
- Lack of designing experience
- Lack of coding practices experience
- Human factors introduces errors in code
- Lack of version control
- Buggy third-party tools
- Last minute changes in the requirement introduce error
- Poor Software testing skill

Top 20 Reasons for Software Defects

1. Miscommunication or No Communication
2. Software Complexity
3. Programming Errors
4. Changing Requirements
5. Time Pressures
6. Egotistical or Overconfident People
7. Poorly Documented code
8. Software Development Tools
9. Obsolete Automation Scripts
10. Lack of Skilled Testers
11. Not having a proper test setup (test environment) for testing all requirements.
12. Writing code or test cases without understanding the requirements clearly.
13. The incorrect design leads to issues being carried out in all phases of the Software Development Cycle.
14. Releasing software patches frequently without completing the Software Testing Life Cycle.
15. Not providing training to resources for the skills required for developing or testing the application properly.
16. Giving very little or no time for Regression Testing.
17. Not Automating Repetitive Test Cases and depending on the testers for manual verification every time.
18. Not prioritizing test execution.
19. Not tracking the development and test execution progress continuously. Last-minute changes are likely to introduce errors.
20. Any wrong assumption made during coding and testing stages.

Defect Fixing Lifecycle

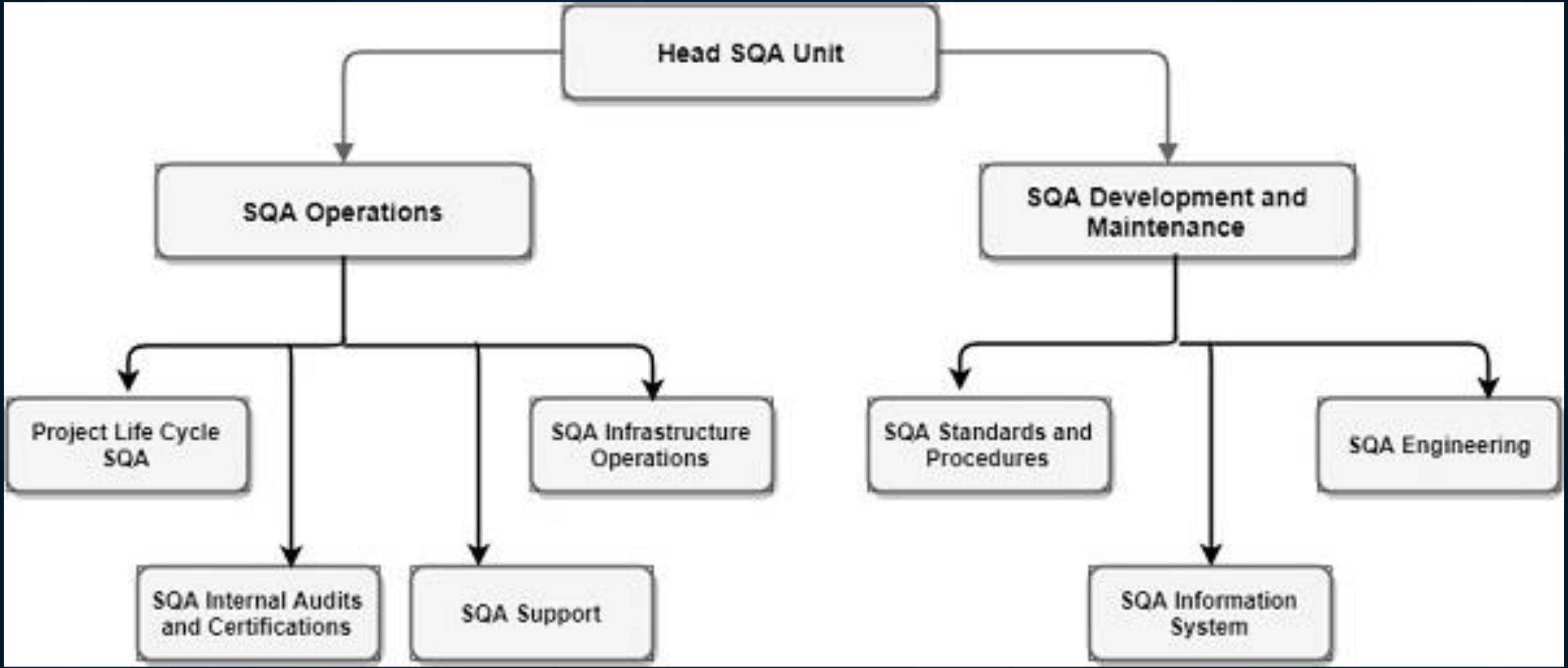


Source: 1998 IEEE Software Productivity Consortium

Reviews

- Everything can be reviewed and <u>needs to be</u> reviewed.
- Don’t accept any artefact (code/document/suggestion) without a review.
- Don’t release anything (code/doc/advice/suggestion/email) without a review.
- Reviews – Self or Peer, Supervisor, External Team
- Number One method to fix software defects! (Testing = #2)

Software Quality Assurance


Head SQA Unit
SQA Operations
SQA Development and Maintenance
Project Life Cycle SQA
SQA Infrastructure Operations
SQA Internal Audits and Certifications
SQA Support
SQA Standards and Procedures
SQA Engineering
SQA Information System

https://www.tutorialspoint.com/software_quality_management/software_quality_management_sqa_unit.htm

Software Audits

- Audit = Sampling Exercise
- Sampling = Take a few out of many
- E.g 100 units test cases exist. To review quality of unit test case, take 10 test cases (sample < population, 10 < 100).
- Audit = Checklist
- Result of Audit = “Finding”
- Finding = Defect or Observation
- Defects are more serious than observations
- Findings are classified = Major (Ma), Minor (mi)
- You need to report RC, CAPA for each defect.

Question Bank – 1

1. What is a software defect?
2. How is a defect usually classified?
3. Distinguish between an error and a fault.
4. Distinguish between fault and a failure
5. Give three examples of software defects.
6. Give the typical defect injection rates across SDLC phases.
7. A defect was injected in the coding phase. It was not detected in unit testing, integration testing, system testing but, it was caught in acceptance testing. What is the age of the defect?
8. Draw and explain the defect life cycle flowchart.
9. Name a few types of software defects.
10. Describe the root cause analysis process with an example and Ishikawa

Question Bank – 1

10. Describe the root cause analysis process with an example and Ishikawa diagram.
11. Distinguish between corrective and preventive actions.
12. Name the popular root causes for software defects.
13. Name two techniques which can be helpful in reducing defects.
14. Name a few activities that SQA department of a software unit perform.
15. What is a software audit?
16. How are audit findings classified?
17. What actions are taken on an audit finding?

Software Process Maturity Models

Unit 1.5
Dr. Mahesha BR Pandit
26-Oct-21, Version 1.0

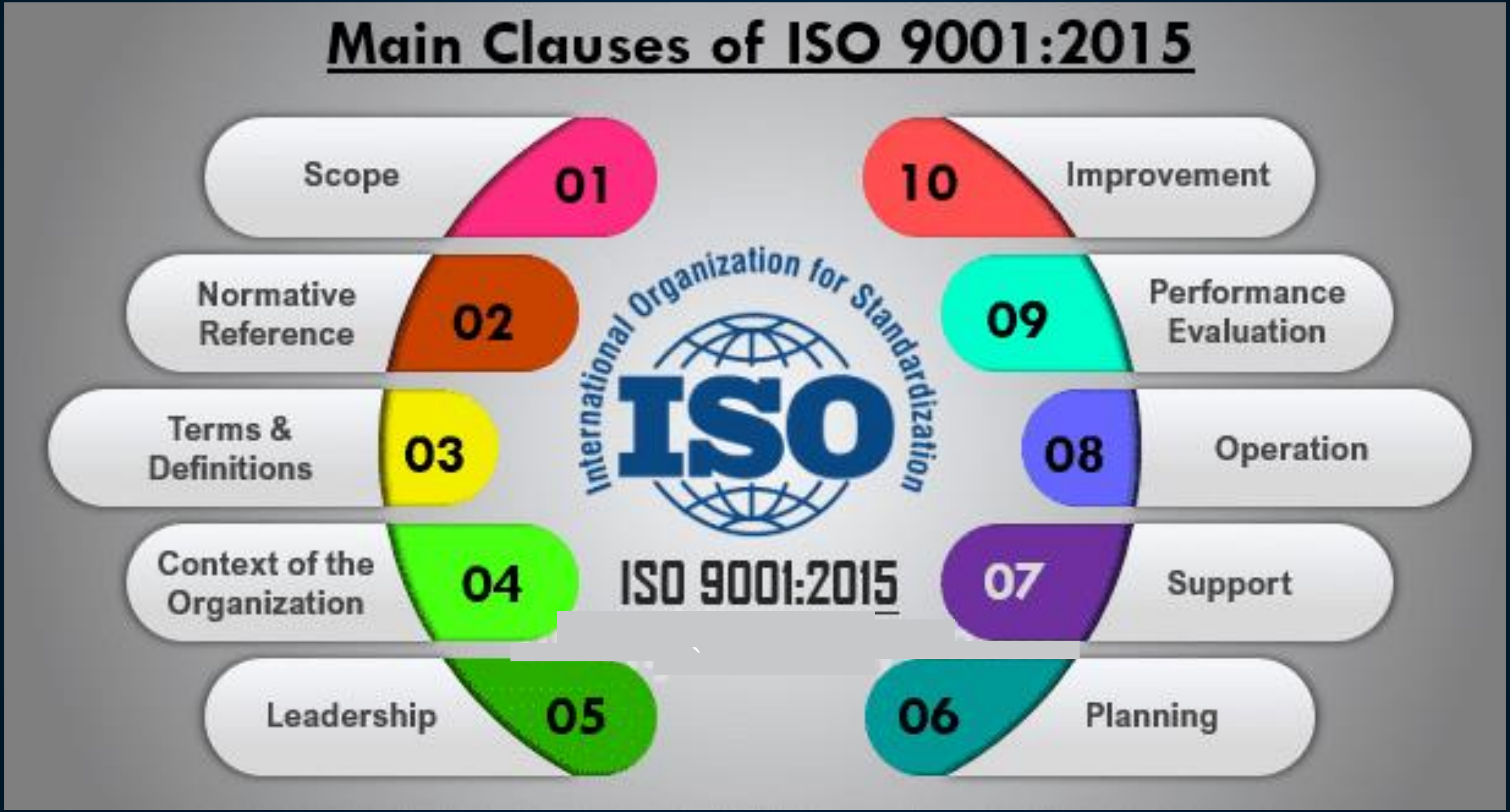
Overview

- Software Maturity – Definition
- Software Maturity Models
    - ISO 9001
    - CMMI

Software Maturity

- Software Maturity = Software Process Maturity
- Software process maturity is the extent to which a specific process is explicitly defined, managed, measured, controlled, and effective.

ISO 9001



Main Clauses of ISO 9001:2015

01 Scope

02 Normative Reference

03 Terms & Definitions

04 Context of the Organization

05 Leadership

06 Planning

07 Support

08 Operation

09 Performance Evaluation

10 Improvement

ISO 9001: Seven Key Principles

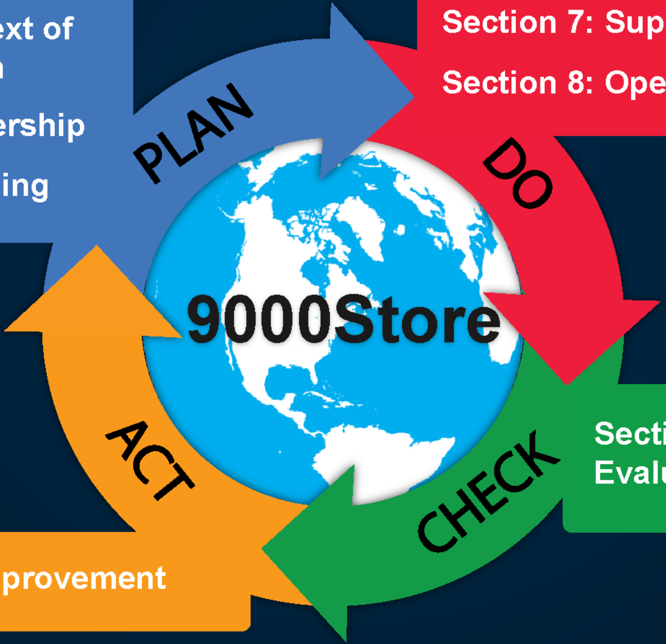
- Customer: To meet and exceed customer expectations.
- Leadership: Create the quality goals and build the processes and consensus that will achieve them.
- Engagement: All workers at all levels need to buy-in to the company's quality goals for any chance of success.
- Process: Define standard and measurable processes that support leadership vision and corporate quality goals.
- Improvement: Quality improvement drives innovation and responsiveness, which are key components of a QMS.
- Decision-making: Measurement and evaluation are necessary for quality improvement and the Plan-Do-Check-Act cycle.
- Relationships: Successful companies need to manage their relationships with suppliers and other stakeholders as much as their customers.

ISO 9001 and PDCA

Section 1: Scope

Section 2: Normative References

Section 3: Terms and Definitions



Section 4: Context of the Organization

Section 5: Leadership

Section 6: Planning

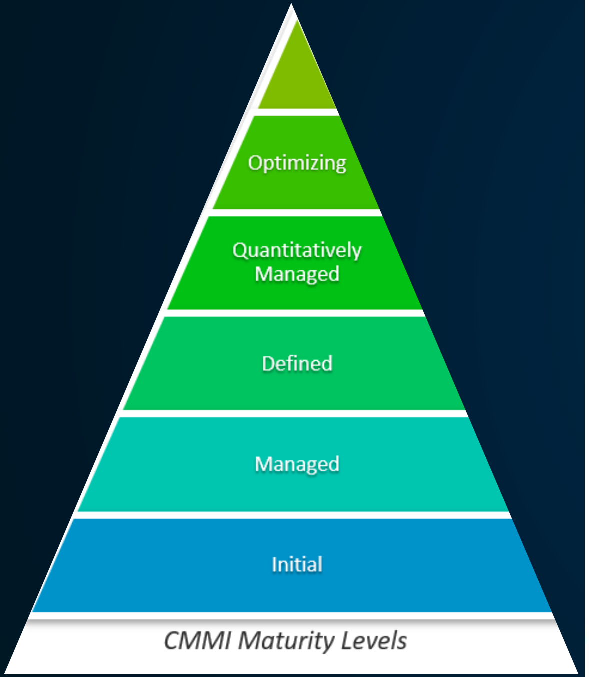
Section 7: Support

Section 8: Operation

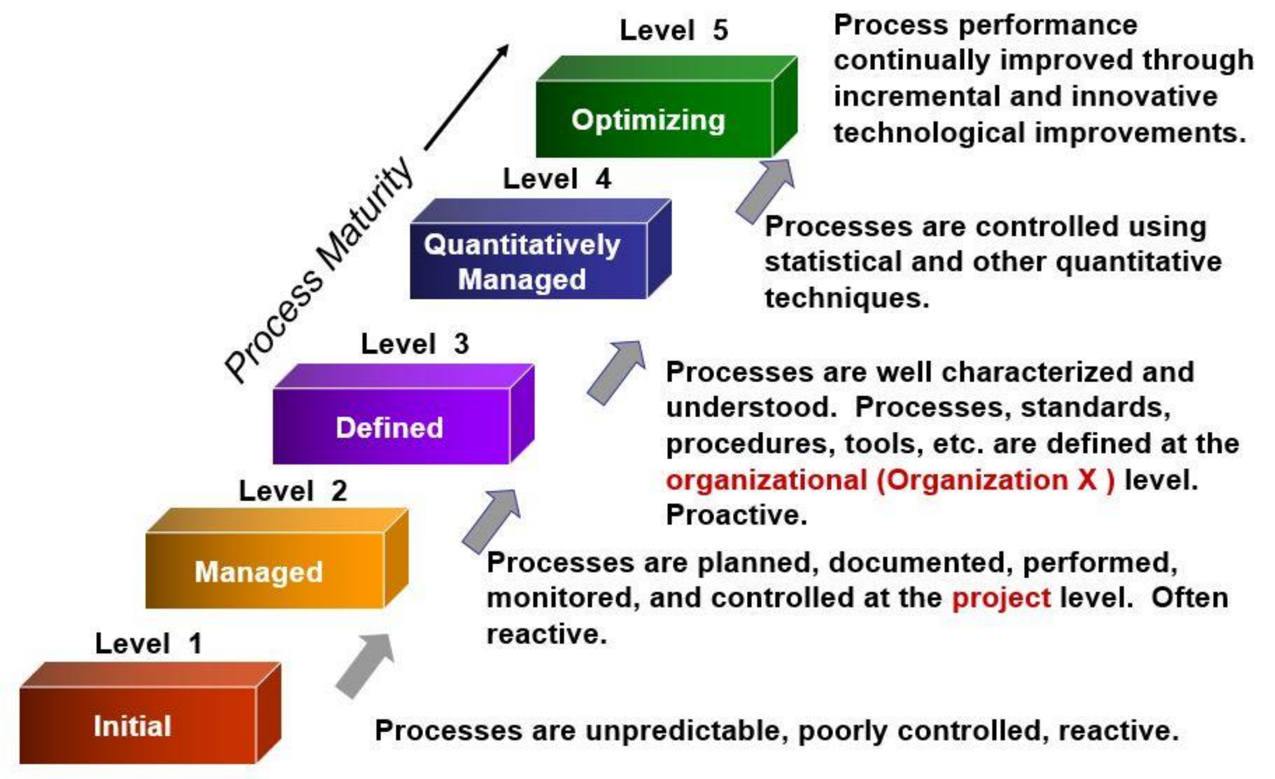
Section 9: Performance Evaluation

Section 10: Improvement

Capability Maturity Model



CMMI Maturity Levels



Process performance continually improved through incremental and innovative technological improvements.

Processes are controlled using statistical and other quantitative techniques.

Processes are well characterized and understood. Processes, standards, procedures, tools, etc. are defined at the organizational (Organization X) level. Proactive.

Processes are planned, documented, performed, monitored, and controlled at the project level. Often reactive.

Processes are unpredictable, poorly controlled, reactive.

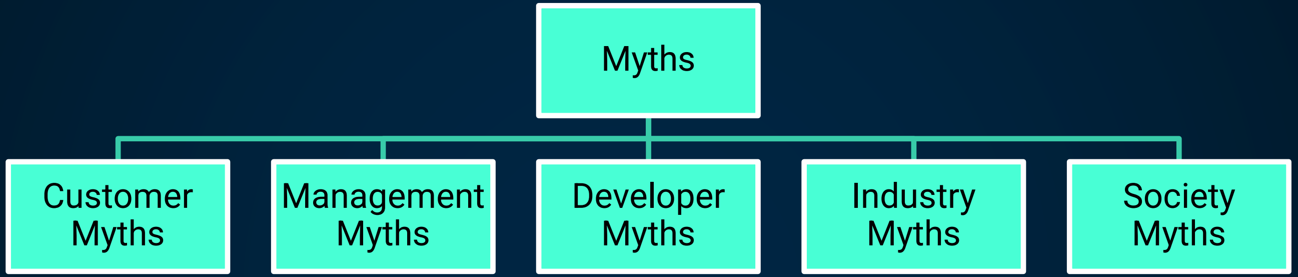
Question Bank – 1

1. Define software maturity. Does it apply to software product or process?
2. Name the seven principles of ISO 9001.
3. How may levels of maturity does the CMMI model recognize? Name them.
4. Does SIT have ISO 9001 certificate?

Software Industry Myths

Unit 1.6
Dr. Mahesha BR Pandit
29-Oct-21, Version 1.0

Software Industry Myths



- Software is flexible
- Software is cheap

- I can put more people to get things done faster

- I must develop code faster
- Coding is the only job I have

- We must use the latest technology
- Quality tools build quality products

- Software people are unnecessarily overpaid

Chinese Proverb

- A fool with a tool is still a fool

Software Industry Myths

1. The Most Popular Language is the Best One
2. Coding Knowledge is enough to Build a Product
3. Software Development is Cheap
4. Latest Tools are the Best
5. More People in the Team means Faster Time to Market
6. The Project is Done once it goes Live
7. Remote Software Development is Cheap
8. Agile development method is easy and the best
9. Quality Tools build Quality Products
10. Outsourcing is a one-stop solution for all IT problems
11. Software is flexible
12. I can add more resources to get things done fast
13. Requirements and design are just documentation
14. A software change is just changing a few lines of code
15. Customer knows what needs to be done

Question Bank – 1

1. Name the types of software myths with a few examples.