Software Testing (CS491)

Introduction

Introduction to Software Quality & Testing

Prof Raghu B. A. Rao

Department of Computer Science and Engineering

Lecture Agenda

What is Software Quality?

Software Quality Concepts

Verification and Validation

Software Quality Assurance

Cost of Quality

Software

Software is a collection of Computer Programs that performs a task.

Eg) Mobile without Apps

Programs- Set of Instructions

Softwares-

1. System Softwares (Device Drivers, Utility Files, (Docs) ,Operating System,Servers)

- Will helps to run the Systems

Accessories connected to Computer(Mouse/Keyboard/Printer)

2. Programming Software-Compilers, Debuggers, Interpreters

-> Input-> Process->Output { Internal Process}

->Internal Softwares

3. Application Software- The users will use the Software

Web Applications (Amazon/Flipkart)

Mobile Applications (Playstore Downloads)

Desktop Applications (MS word,Notepad)

Software Types

1) System software Ex: Device drivers, Operating Systems, Servers, Utilities, etc.

2) Programming software Ex: compilers, debuggers, interpreters, etc.

3) Application software Ex: Web Applications, Mobile Apps, Desktop Applications etc.

X Bank(company)---------> IT Company---> Develop--->Test--->Deliver --->XBank

What is software Testing?

Software Testing is a part of software development process. Software Testing is an activity to detect and identify the defects in the software. The objective of testing is to release quality product to the client.

Software Quality

Bug Free

Delivered On Time

Within Budget

Meet the Requirements

Maintainable

Project vs Product

PROJECT: If Software application developed for specific customer/specific customer requirements ,then it is called PROJECT. That project is only used by that specific organization.

Ex) Finacle

Service Based Companies (TCS, Accenture, Infosys)

PRODUCT: If Software application developed for the market requirements, and that is used by multiple customer then it is called PRODUCT.

Ex) Mobile Apps, MS Word, PPT

Product Based Companies (Microsoft, Google, Oracle)

What is Software Quality?

Software quality is defined as **a field of study and practice that describes the desirable attributes of software products**.

There are two main approaches to software quality: defect management and quality attributes.

“fitness of purpose” isn’t a completely satisfactory definition of quality.

ISO/IEC 25010:2011 Software Quality Model : This standard describes a hierarchy of eight quality characteristics, each composed of sub-characteristics:

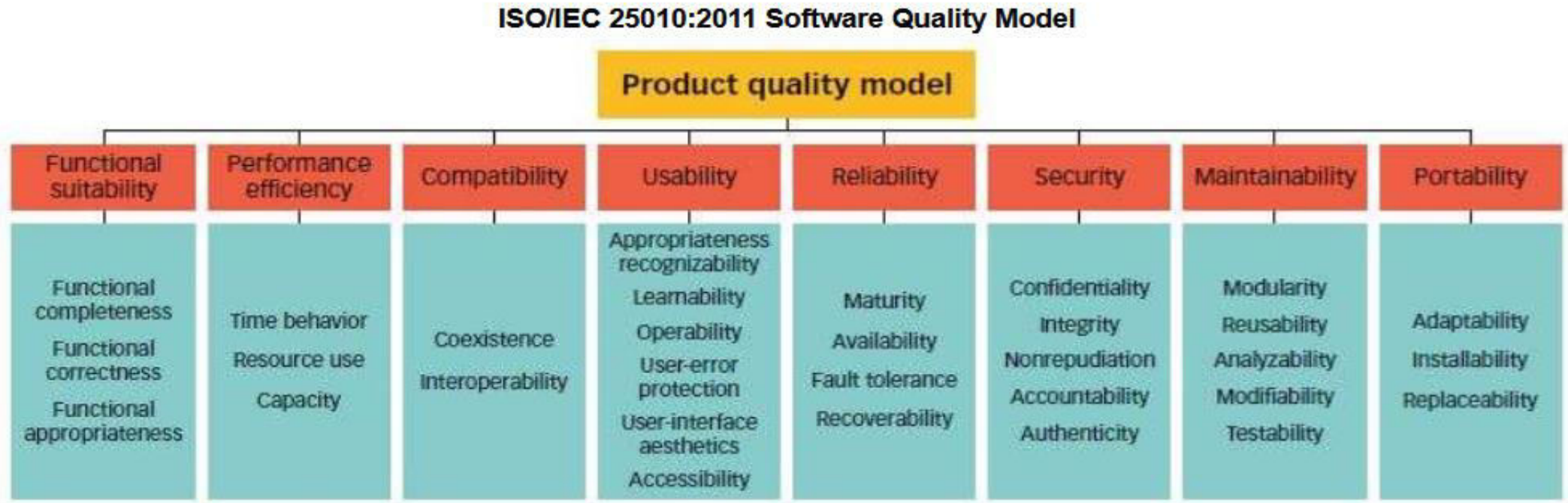
-> Functional suitability
-> Reliability
-> Operability
-> Performance efficiency
-> Security
-> Compatibility
-> Maintainability
-> Transferability

Quality Concepts

Additionally, the standard defines a quality-in-use model composed of five characteristics:

- Effectiveness
- Efficiency
- Satisfaction
- Safety
- Usability

Software Quality Model



Quality Concepts

- According to PMBOK, quality is a conformance to standards or requirements.

- ISO 9000 defines it as the
  - “totality of features and characteristics of a product or service that bear on its ability to satisfy stated and implied needs.”

- In some industries, government agencies, and educational institutions, quality is described as
  - Fitness for use.
  - Fitness for purpose
  - Customer satisfaction,
  - Conformance to requirements/specifications

Quality Concepts

Dr. Herald Kerzner States that most organizations today view quality as a process rather than product. It is a continuous process of improvement and the use of lessons learned to enhance the manufacturing of products or services in order to

- Retain existing customers
- Win back lost customers
- Win new customers

Quality Concepts

Deming's PDCA Cycle (Plan, Do, Check, and Act) is another well known continuous improvement process. Deming states that:

- **Plan** Improvements to present practices (Establish objectives and processes required to deliver the desired results.)
- **Do** Implement the plan (Carry out the objectives from the previous step.)
- **Check** Test to see if the desired results are achieved (Data is compared to the expected outcomes to see any similarities and differences.)
- **Act** Implement the corrective action (Also called "Adjust", this act phase is where a process is improved.)

What is Software Quality?

- Conformance to requirements
- Lack of defects
- Low defect rate (# of defects/size unit)
- High reliability (number of failures per N hours of operation)
- Probability of failure-free operation in a specified time
- Measured as Mean Time To Failure (MTTF)
    - this metric indicates how long the system operates until failure.

- **MTBF** (Mean Time Between Failures) describes the time between two failures
- **MTTF** (Mean Time To Failure) describes the time up to the first failure.

Generic Reasons for Software Failure

- Uniqueness of the software product
- High Complexity
- Limited Opportunity to detect bug
- Environment in which software is developed
- Requirement of teamwork

Quality Factors

- Correctness
- Reliability
- Efficiency
- Integrity
- Usability
- Maintainability
- Flexibility
- Testability
- Portability
- Interoperability
- Reusability

Quality Management

According to PMBOK,

- “Project Quality Management includes the processes required to ensure that the project will satisfy the needs for which it was undertaken.
- It includes “all activities of which the overall management function that determine the quality policy, objectives, and responsibilities and implements them by means such as quality planning, quality control, quality assurance, and quality improvement, within the quality system.”

Project Quality Management must address both the management of the project and product of the project.

Quality Management

Project Quality Management consists of the following major processes

- Quality Planning (Planning Process)
- Quality Assurance (Execution Process)
- Quality Control (Control Process)

***QA is process-oriented**, and it focuses on preventing quality issues. QC is product-oriented and focused on identifying quality issues in manufactured products. ... QA involves the actions which create the product, while QC is focused on the resulting product.*

Cost of Quality

- The cost of quality is the total price of all efforts to achieve product or service quality. This includes the work to build a product or service that conforms to the requirements and all work resulting non-conformance to the requirements.
- A typical project should have a goal of 3 to 5 percent of the total value devoted to the cost of a quality program.

Cost of Quality

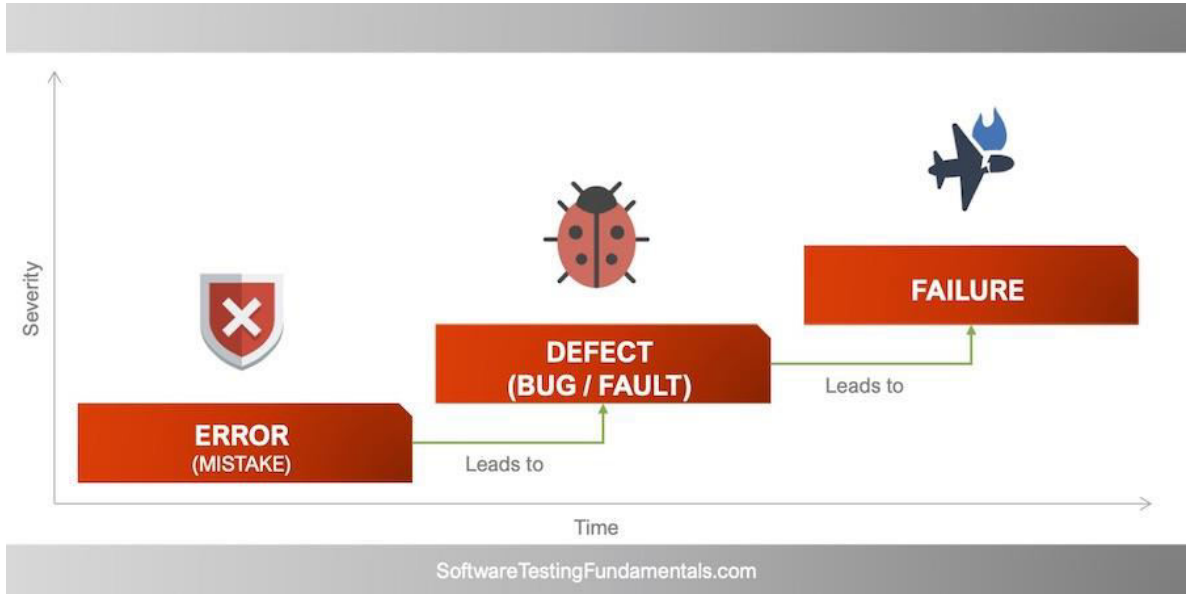
- Prevention cost
- Appraisal Cost
- Internal Failure cost
- External failure cost

Reasons for Testing

- Software contains defects
- Defects may cause software failure
- Failure can cause disaster

- Testing ensures quality of software
- Testing accelerates software development.

➢ ……..And much more!!!



Error/Bug/Failure

ERROR: Human Mistake, incorrect human action.

Developer has made some mistake ,typing something wrong

BUG: Deviation from the expected behavior and actual behavior (Defect)

Bug and Failure related to application

Gmail: Valid user name and valid password (Login)

If we pass invalid data, it is a mismatch

FAILURE: Deviation identified by the end user is Failure (Production Environment)

Reasons for Failure

Miscommunication/No Communication

Complexity of the Software

Programming Error (Responsibility of the Developer)- Incorrect Output

Frequently changing Requirements

Lack of skillset : Tester skillset

Infamous Failures

February 2020: Heathrow disruption

More than 100 flights to and from London's Heathrow airport were disrupted on Sunday 16 February, 2020, after it was hit by technical issues affecting departure boards and check-in systems, leaving passengers with little information about their flights and limiting the use of electronic tickets.

A spokesperson for Heathrow said that it could not share any more details as to which systems had been affected "due to the sensitive nature of those processes".

"Following yesterday's technical issue Heathrow's systems are stable and the airport is operating as normal. We apologise for the inconvenience this caused our passengers. Our teams continue to closely monitor our systems and will be on hand across our terminals to provide assistance to passengers," the spokesperson added by way of a statement.

Process Failure

- Human errors
  - Failures in development (e.g., poor development methodologies)
  - error in operation
- Therac-25
  - Radiation treatment machine malfunction
  - Delivers small doses of radiation through filters to treat cancers, tumors
  - Six deaths due to lethal dose of radiation before fixed

Korean Air Flight 801

- On August 6, 1997, Korean Air Flight 801 crashed into a rocky hillside in the middle of Guam. 228 of the 254 passengers on board were killed.
- Primary cause of the crash due to negligence of the flight crew, but the Minimum Safe Altitude Warning system also failed
  - FAA system responsible for warning officials when aircrafts approach too near to the ground
  - "The altitude warning system is designed to cover a circular area with a radius of 55 nautical miles (102 kilometers). However, since the software was modified, the system only covered a mile-wide circular strip that ran the circumference of that area. Flight 801 was not covered when it crashed. Black [US National Transportation Safety Board investigator] said the software was modified to stop the system from giving too many false alarms. 'The modification modified too much,' he said." (Delaney, 1997)

Famous Software Testing Failures Examples

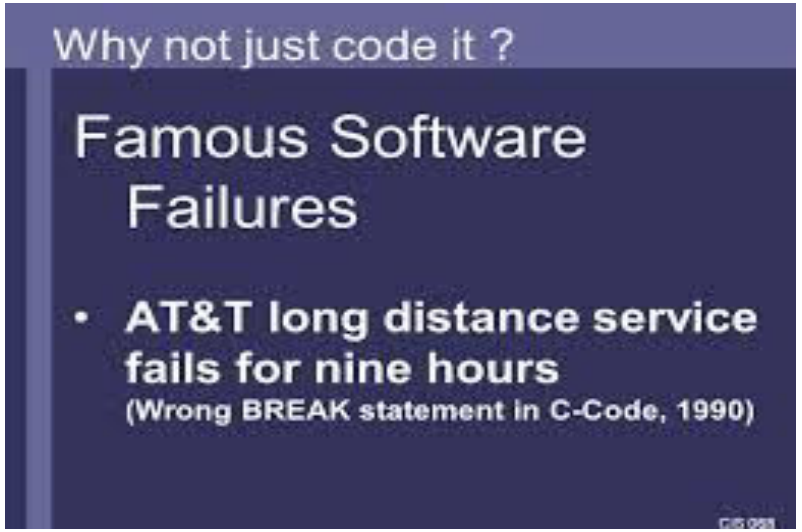
Ariane 5 Flight 501

Soviet Gas Pipeline Explosion

https://raygun.com/blog/costly-software-errors-history/

Other Famous Software Failures

- 1990 AT&T long distance calls fail for 9 hours
  - Wrong location for C break statement
- 1996 Ariane rocket explodes on launch
  - Overflow converting 64-bit float to 16-bit integer
- 1999 Mars Climate Orbiter crashes on Mars
  - Missing conversion of English units to metric units
- Other Failures available at:
  - http://www.sundoginteractive.com/sunblog/posts/top-ten-most-infamous-software-bugs-of-all-time/
  - http://www.net-security.org/secworld.php?id=10354



Do the Software Testing Right

- Testing has to be planned
- Testing has to be performed in a reasonable way
- Testing shall be independent of the Objective
- Testing has to be managed.

SOFTWARE TESTING IS A FUNDAMENTAL PART OF THE PROFESSIONAL SOFTWARE DEVELOPMENT

Software Testing (CS491)

Introduction II

Software Development Lifecycle

Prof Raghu B. A. Rao

Department of Computer Science and Engineering

Software Development Lifecycle

Process used by the Software Industry to deliver the project to the customer

3 Pillars

P- People

P- Process (Stages)

P- Product (Deliverables)

SDLC –Phases

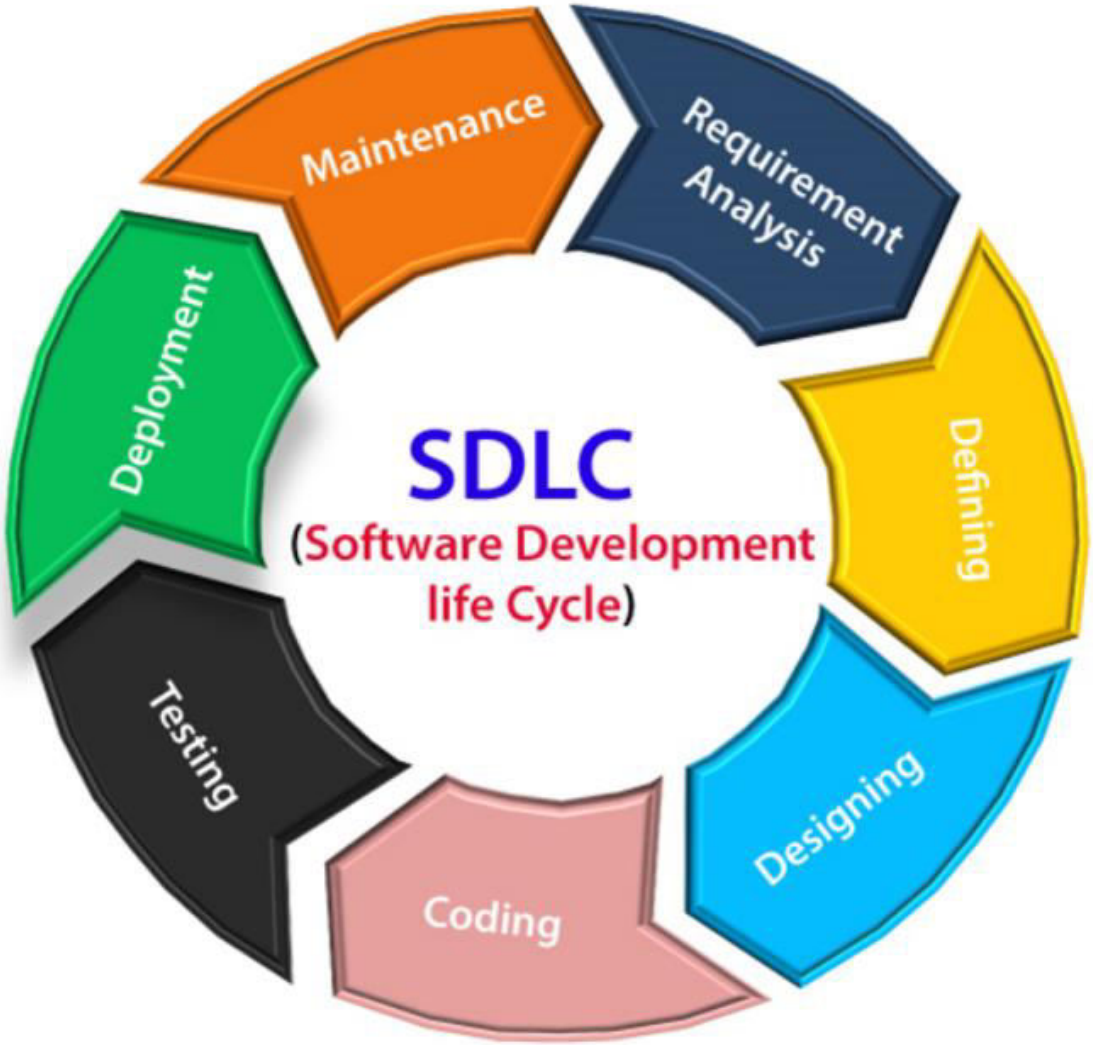
Requirements Analysis -> Design->Development (Coding)-> Testing-> Implementation-> Maintenance

Software Development Life Cycle (SDLC) is a framework that defines tasks performed at each step in the development process.

Originated in the 1960s, SDLC is a structured, standardized set of processes for developing and maintaining business solutions.

From inception to completion, it details out; how to plan, build, and maintain specific software.

SDLC Phases



REFERENCE

Why SDLC?

- Project Tracking and control
- Increased visibility
- Enhanced Development speed
- Improved client relation
- Decreased the project risk

To manage the project complexities, number of SDLC models was created like waterfall, incremental, Agile, Prototype, etc.

The method of implementing SDLC varies vastly between methodologies.

SDLC is a crucial consideration before the actual software development process.

Aims of SDLC

- Deliver high-quality system
- Work efficiently with the existing and planned infrastructure
- Maximize productivity
- Provide strong management controls
- Lower the cost the development

SDLC Phases

Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

SDLC Phases

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules.The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

SDLC Phases

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

SDLC Phases

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market.

Sometimes product deployment happens in stages as per the business strategy of that organization.

The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Process Models

1. WATERFALL MODEL

Traditional model

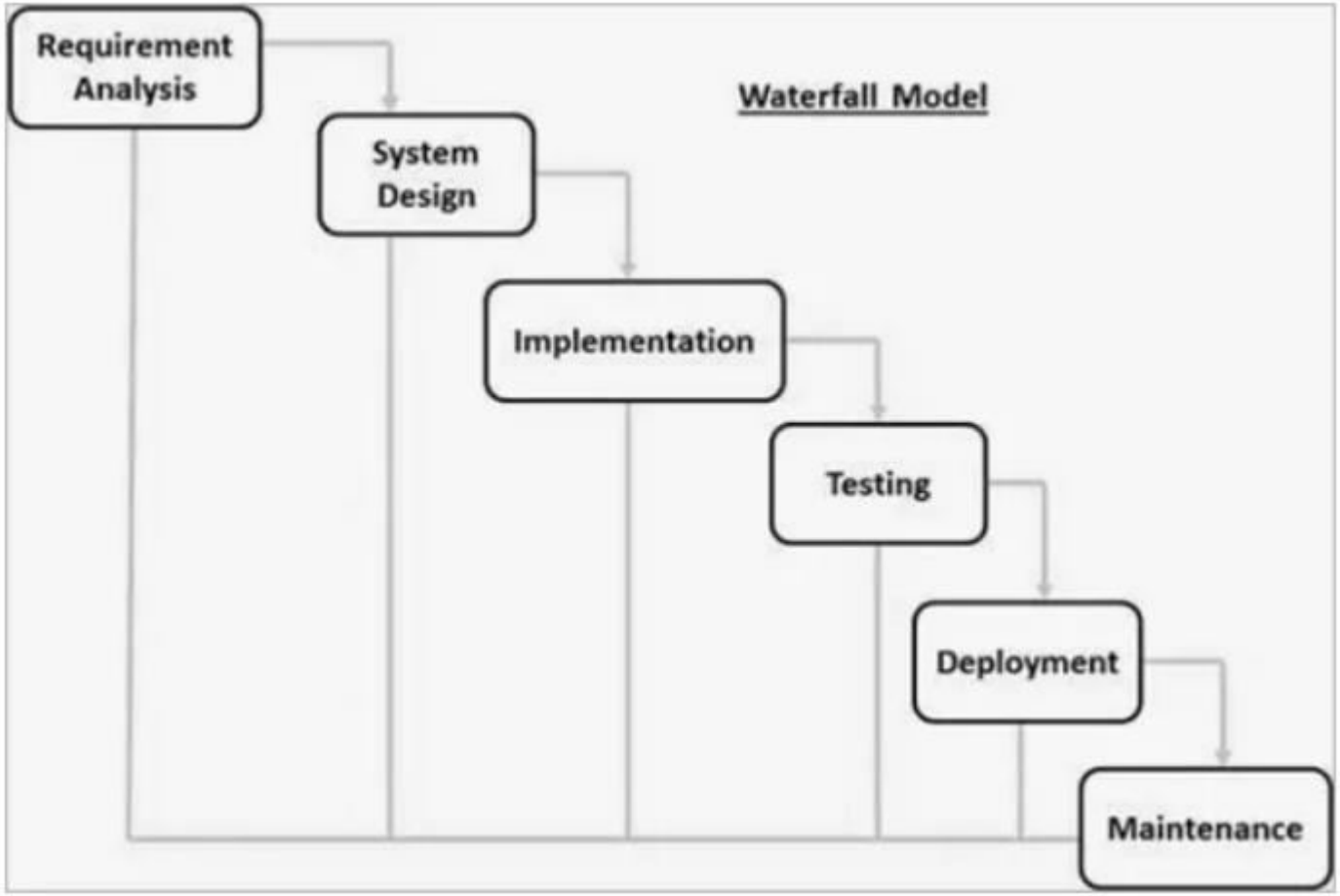
Linear Model

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a **linear-sequential life cycle model**.

In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

Waterfall Model



Advantages & Disadvantages of Waterfall Model

Advantages

Quality of the Product will be Good

Since requirements changes are not allowed chances of finding bugs will be less

Initial investments is less since the testers are hired at the later stages

Preferred for smaller projects where requirements are freezed

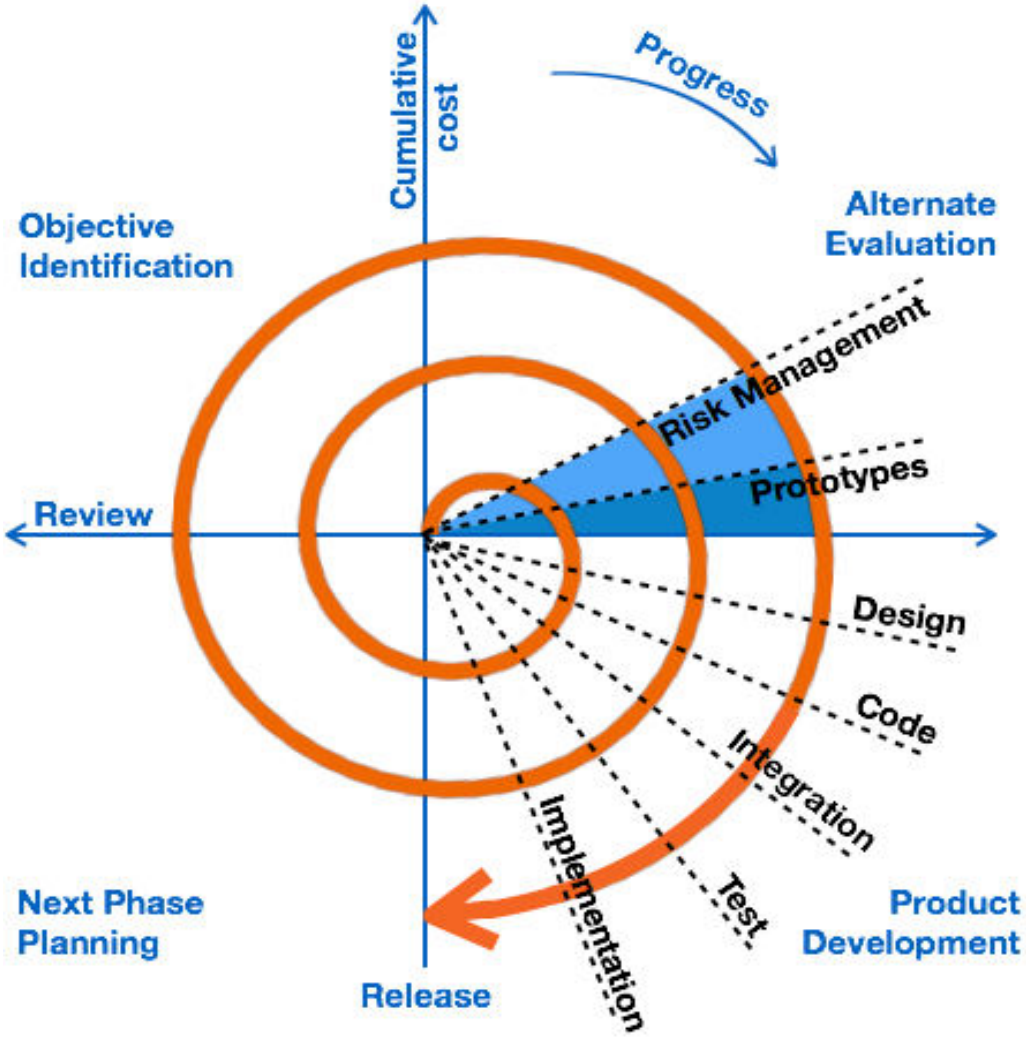
Disadvantages

Requirement changes are not allowed

Backtracking is not possible

If there is defect in the requirement phases, it will continue in the later phases

Spiral Model



Spiral Model

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

Identification

This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

Design

The Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and the final design in the subsequent spirals.

Construct or Build

The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

Spiral Model

Evaluation and Risk Analysis

Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun.

After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

Advantages & Disadvantages of Spiral Model

Advantages of Spiral Model

Software will be released in multiple versions, Version Control Model

Testing is done in every cycle before go to the next cycle'

Customer will get to use the software for every module

Requirement Changes are allowed after every cycle before going to the next cycle.

Eg) Gmail(Compose mail, Sent Box-Dependency)

Disadvantages of Spiral Model

Requirement Changes are allowed in between cycle.

Internally we are following a waterfall model.

There is no testing in every phase.(every cycle , we have testing)

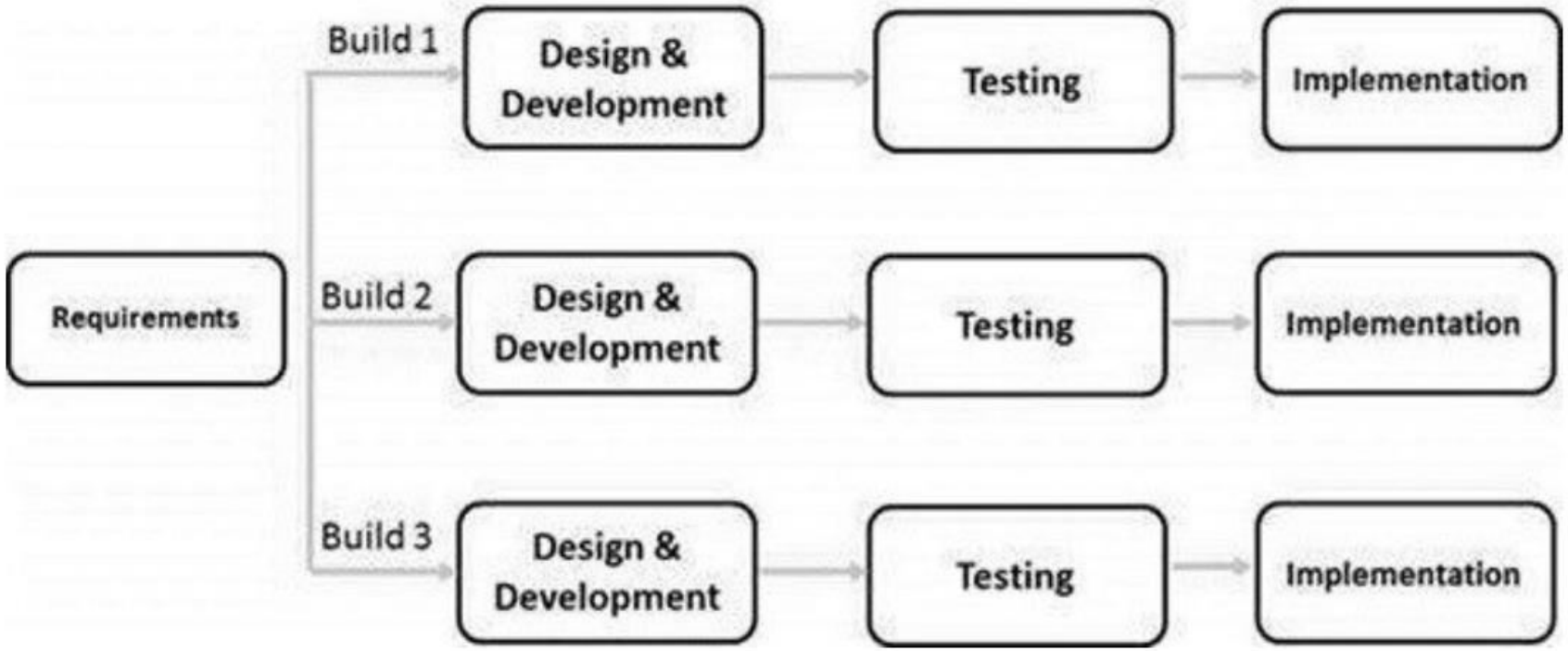
Iterative Model

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented.

At each iteration, design modifications are made and new functional capabilities are added.

The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

Iterative Model - Design



Prototyping & RAD Model

The Rapid Application Development Model was first proposed by IBM in the 1980s.

A software project can be implemented using this model if the project can be broken down into small modules wherein each module can be assigned independently to separate teams.

These modules can finally be combined to form the final product.

Development of each module involves the various basic steps as in the waterfall model i.e analyzing, designing, coding, and then testing, etc.

Prototype is a sample , blueprint of the software.

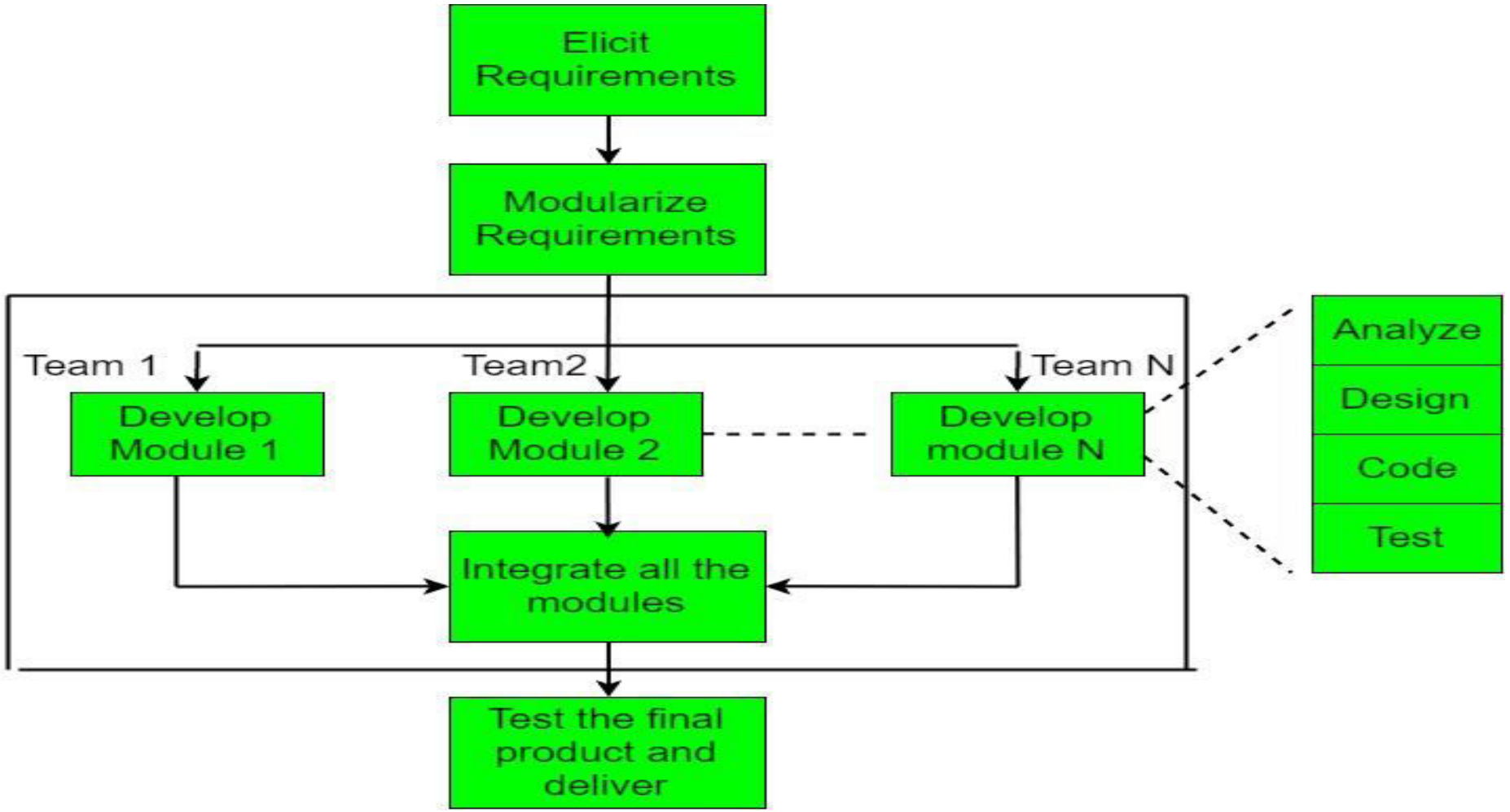
Get the initial requirements from customer->develop the prototype->show this to customer->we will start design, coding, testing

Eg) Model House

4 Phases

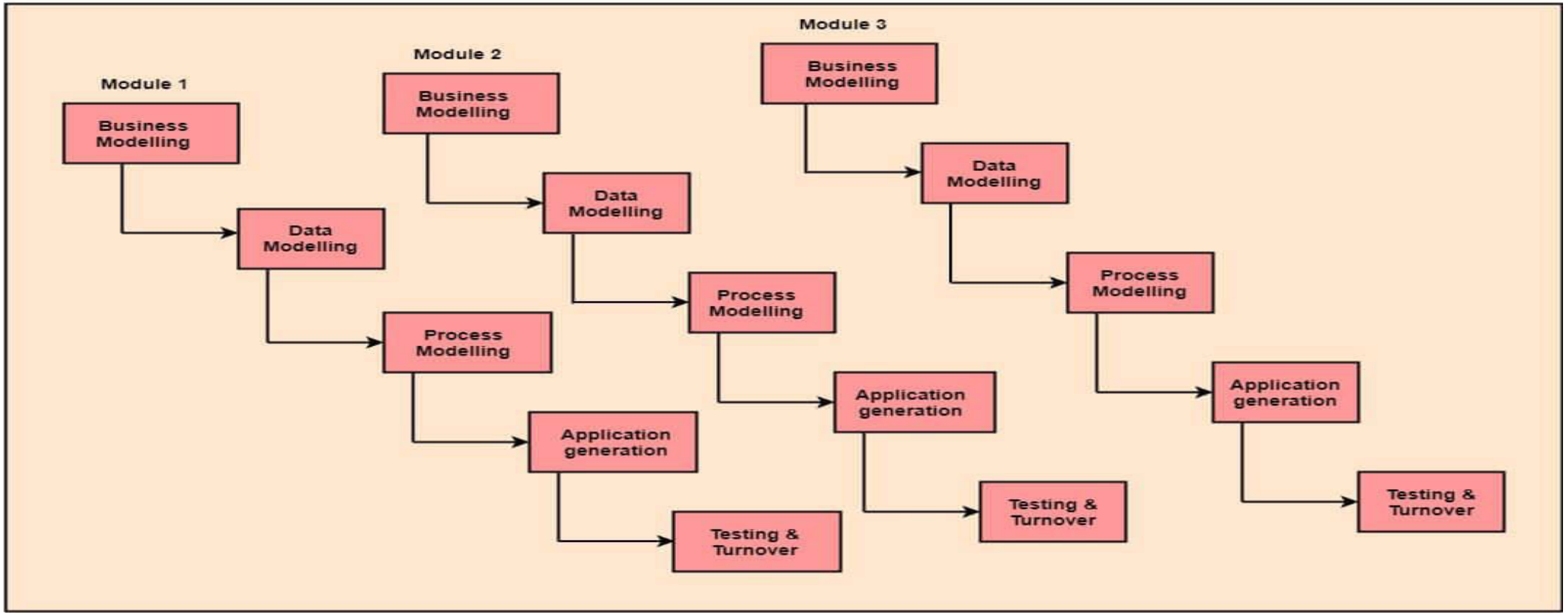
1. **Requirements Planning –** It involves the use of various techniques used in requirements elicitation like brainstorming, task analysis, form analysis, user scenarios, FAST (Facilitated Application Development Technique), etc.
2. **User Description –** This phase consists of taking user feedback and building the prototype using developer tools. In other words, it includes re-examination and validation of the data collected in the first phase.
3. **Construction –** In this phase, refinement of the prototype and delivery takes place. It includes the actual use of powerful automated tools to transform process and data models into the final working product.
4. **Cutover –** All the interfaces between the independent modules developed by separate teams have to be tested properly.

Schematic Representation (RAD Model)



RAD Model

Fig: RAD Model



When to use RAD Model?

- When the system should need to create the project that modularizes in a short span time (2-3 months).
- When the requirements are well-known.
- When the technical risk is limited.
- When there's a necessity to make a system, which modularized in 2-3 months of period.
- It should be used only if the budget allows the use of automatic code generating tools.

V-V Model

The model is similar to V shape

In every phase we conduct testing (All SDLC phases conducts testing)

BRS/CRS/URS : Documents contains the requirements from the customer

Business Unit will prepare the document

We conduct testing called UAT with this document

SRS and BRS? What is the difference?

Based upon SRS->HLD->LLD is prepared by designers

HLD->Main Modules of the application

LLD-> Low level modules

V-V Model

BRD->Business Unit People

SRS-> Product Manager/Project Managers

HLD & LLD->Designers

In V model, we have testing.

BRD->Reviews (To ensure the correctness and Completeness)

SRS, LLD,HLD ->Testing Techniques like Reviews, Walkthrough, Inspection (Testing the documents)

Static Testing : Testing the project related documents

Why static?

Review, Walkthrough and Inspection -> Following slides

Reviews

Conducts on documents to ensure correctness and completeness
-> Requirements Review
->Design review
-> Code Review
->Test Plan review
-> Test Code Review

Walkthroughs

Informal review, we don’t have any specific plan
Author- The person who has created the documents
Review the document with peers
Review : Only Author will be there
Walkthrough: Informal review With Peers

Walkthrough - It is a informal review.
Author reads the documents or code and discuss with peers. It's not pre-planned and can be done whenever required. Also walkthrough does not have minutes of the meet

Inspection

It is most formal review Type

Minimum 3- 8 people sit together and review –Reader, Writer, Moderator

Inspection will have a proper schedule and proper intimation via email to the concerned tester /Developer

Verification & Validation

Dynamic Testing?

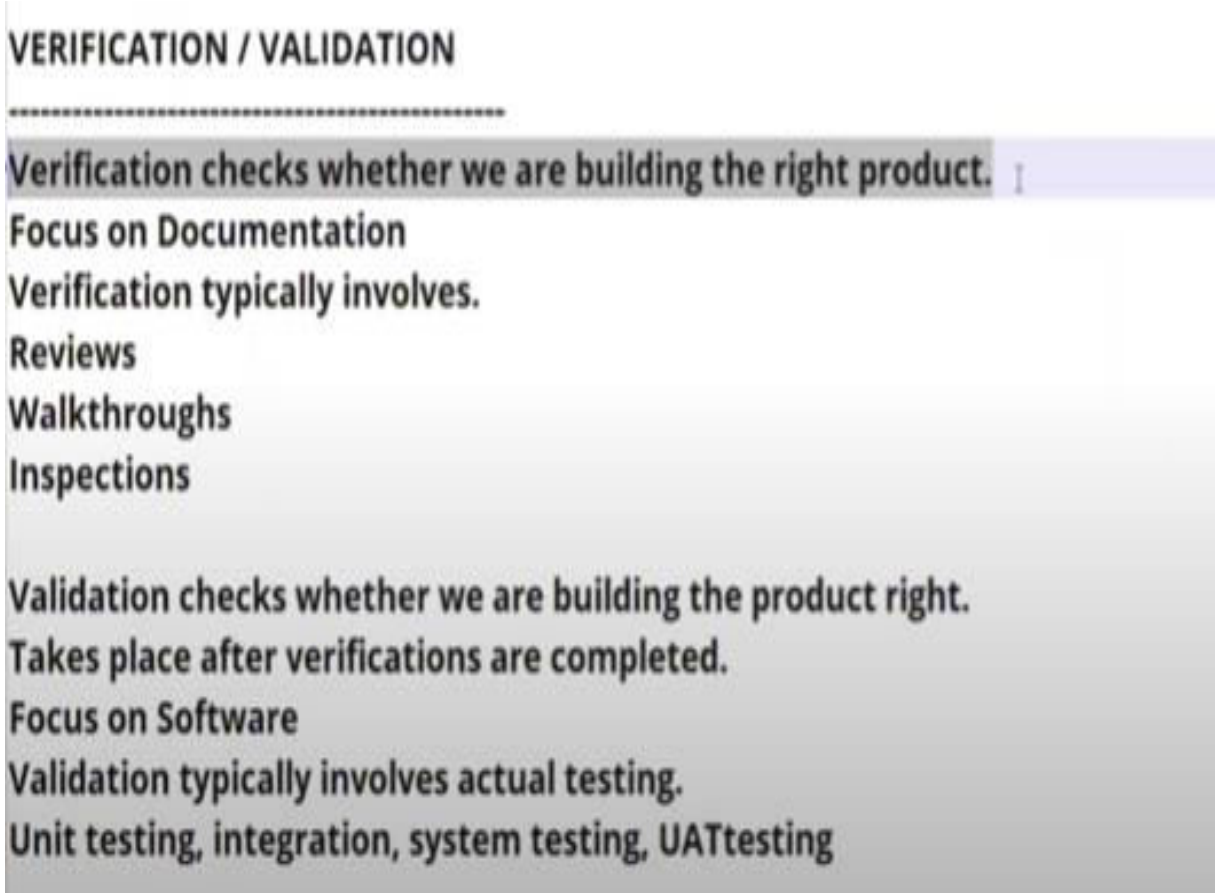
Testing the actual Software

Techniques: Unit Testing, Integration Testing, UAT

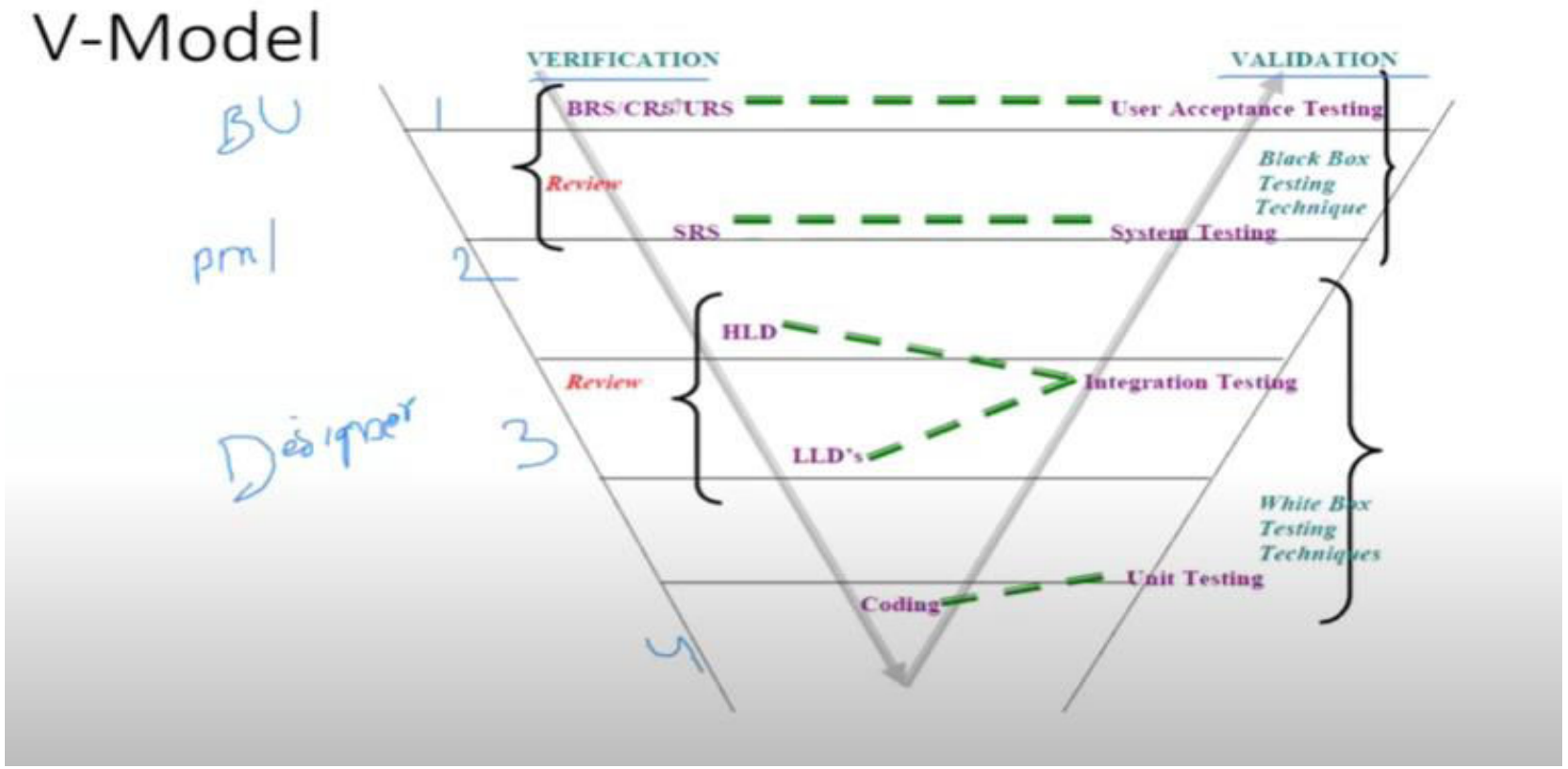
Verification and Validtion?

Verification?-Will be done before software ready

Validation? – Will be done after Software is ready



V-V Model

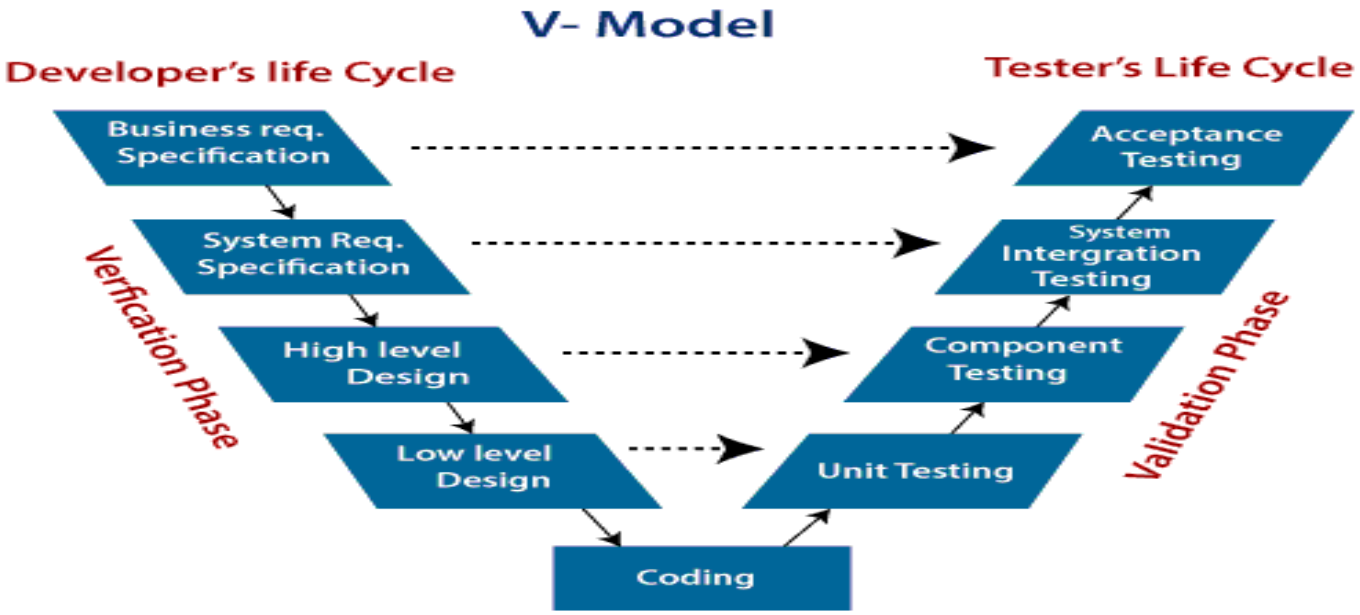

V-Model
VERIFICATION
VALIDATION
BRS/CRS/URS
User Acceptance Testing
Review
Black Box Testing Technique
SRS
System Testing
HLD
Review
Integration Testing
LLD's
White Box Testing Techniques
Unit Testing
Coding

V Model

V-Model also referred to as the Verification and Validation Model. In this, each phase of SDLC must complete before the next phase starts.

It follows a sequential design process same as the waterfall model.

Testing of the device is planned in parallel with a corresponding stage of development.



Verification & Validation

Verification: It involves a static analysis method (review) done without executing code. It is the process of evaluation of the product development process to find whether specified requirements meet.

Validation: It involves dynamic analysis method (functional, non-functional), testing is done by executing code. Validation is the process to classify the software after the completion of the development process to determine whether the software meets the customer expectations and requirements.

Advantages & Disadvantages of V Model

Advantages of V Model

Testing is involves in every phase, so less number of chances to get bugs in later phases

Disadvantages

Initial investment is high ,in every phase we need people developers,testers from beginning

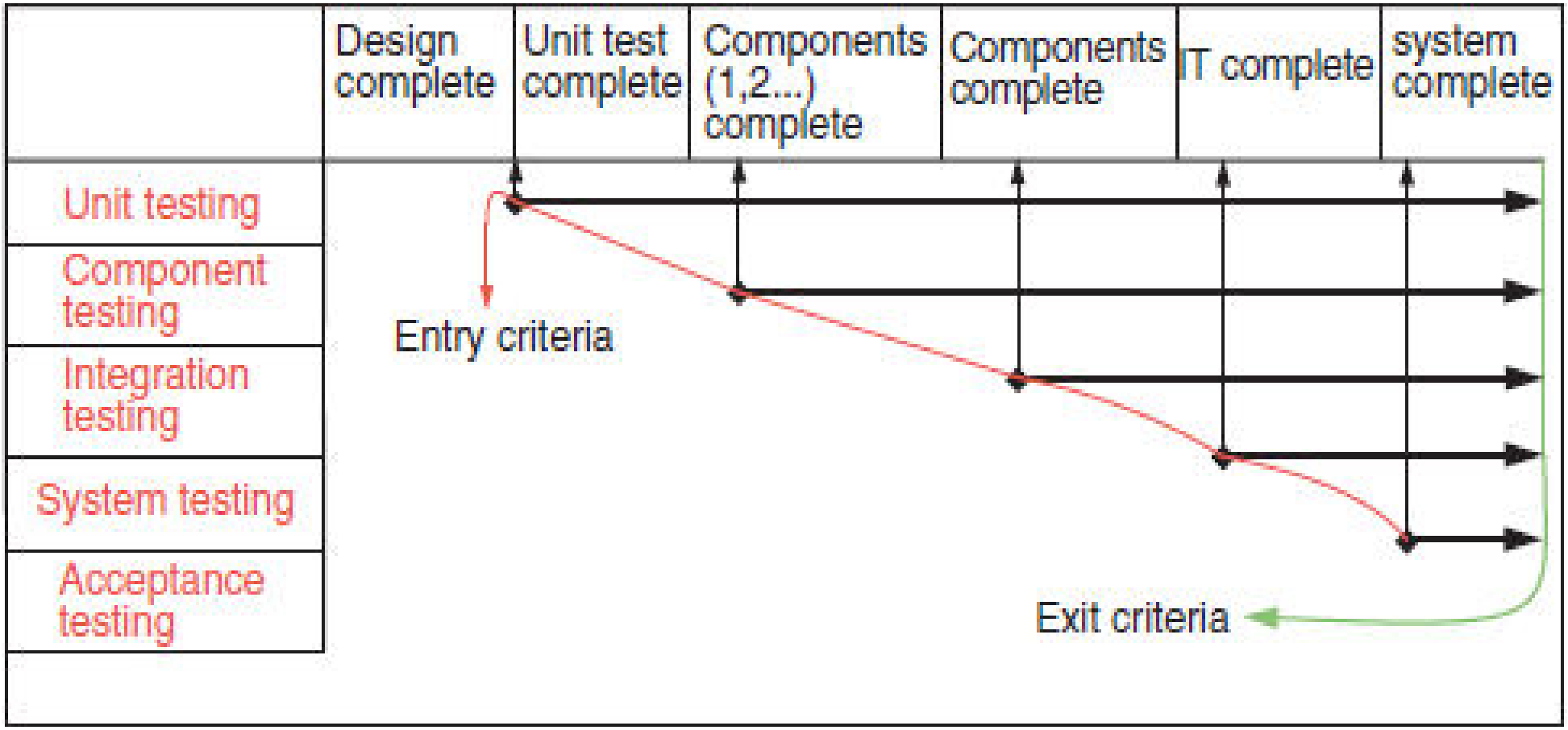
Documentation is more

All activities should go in parallel

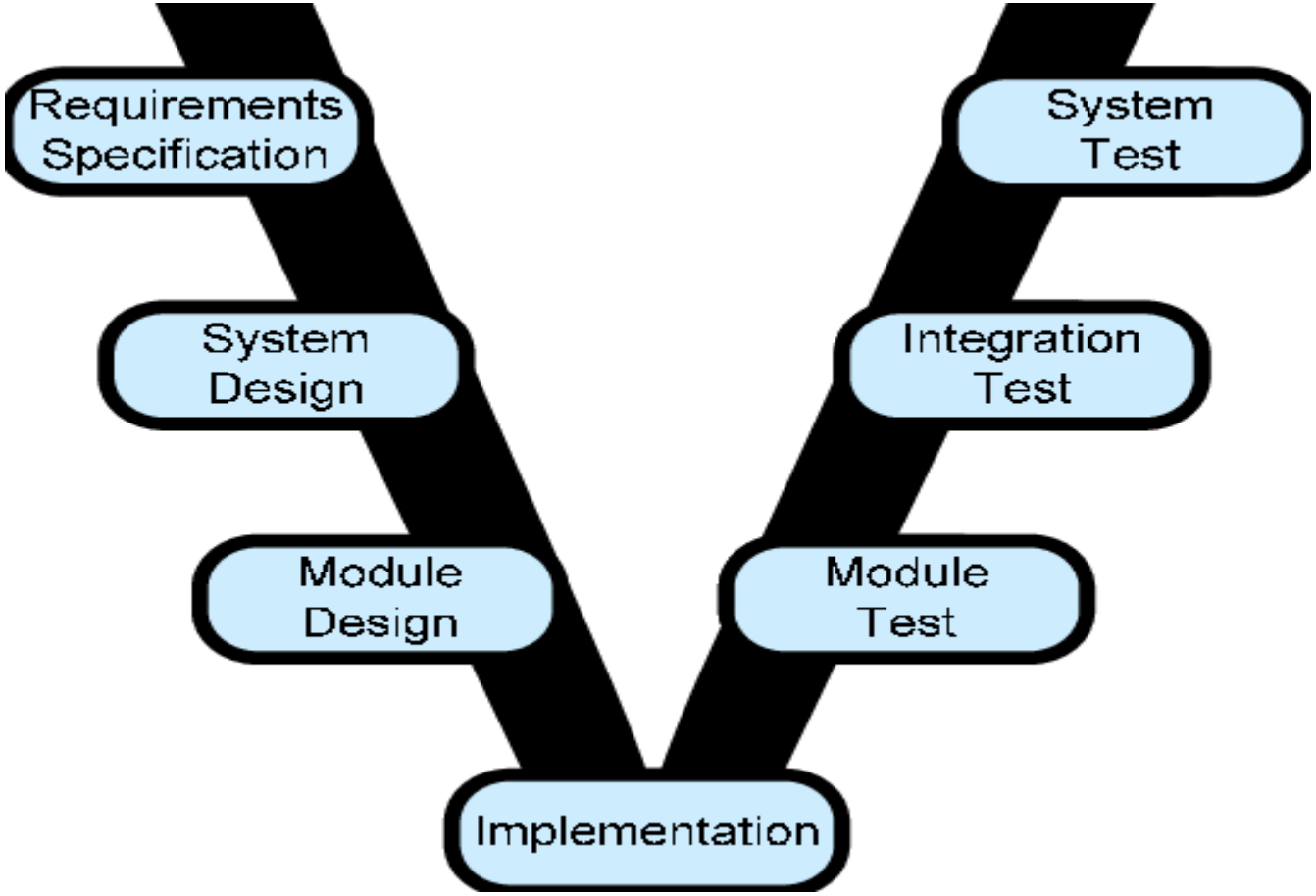
Modified V Model

- Modified V model recognizes that different parts of a product are in different stages of evolution
- Each part enters the appropriate testing phase as unit testing, component testing and so on when the appropriate entry criteria are met

Modified V Model



Modified V Model



THANK YOU

Introduction to Software Quality & Testing

Prof Raghu B. A. Rao

Department of Computer Science and Engineering

Software Testing (CS491)

Introduction

Quality Management

Prof Raghu B. A. Rao

Department of Computer Science and Engineering

Lecture Agenda

1.Levels of Testing
2.QA & QC
3.Testing Models

Levels of Testing

- Unit Testing
- Integration Testing
- System Testing
- User Acceptance Testing (UAT)

Unit Testing

A unit is a single component or module of a software.
Unit Testing is conducted on a single program or single module.
Unit Testing is a white box testing technique.
It is conducted by the developers

Unit Testing techniques :

1. Basis Path Testing
2. Control Structure testing
    - Conditional Coverage
    - Loops Coverage
1. Mutation Testing

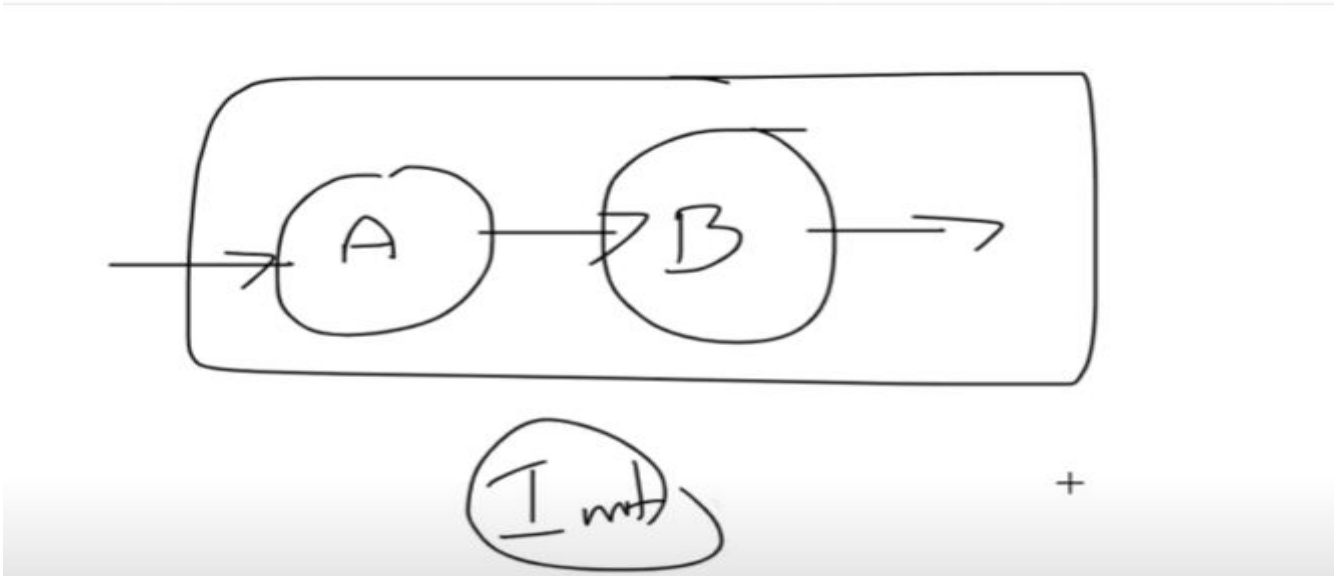
Integration Testing

Integration Testing

Integration testing performed between 2 or more modules.

Integration testing focuses on checking data communication between multiple modules.

Integrated Testing is white box testing technique.



System Testing

System Testing

Testing over all functionality of the application with respective client requirements.

It is a black box testing technique.

This testing is conducted by testing team.

After completion of component and integration level testing's we start System testing.

Before conducting system testing we should know the customer requirements.

System Testing focusses on below aspects.

User Interface Testing (GUI)

Functional Testing

Non-Functional Testing

Usability Testing

UAT Testing

After the completion of the system testing , UAT team conducts acceptance testing in 2 levels

- **Alpha Testing** : The customers will come back to the company and do some testing
- **Beta Testing** : Install the software in user environment and do testing

Quality Management

Project Quality Management consists of the following major processes :

- Quality Planning (Planning Process)
- Quality Assurance (Execution Process)
- Quality Control (Control Process)
- Quality Improvement.

QA & QC

Quality Assurance

- Attempts defect prevention by concentrating on the process of producing the product rather than working on defect detection / correction after the product is built.
- QA is process oriented ,define the process (high level management people)
- QA will be involved throughout the development process
- Focuses on building the quality
- QA is for preventing the defects

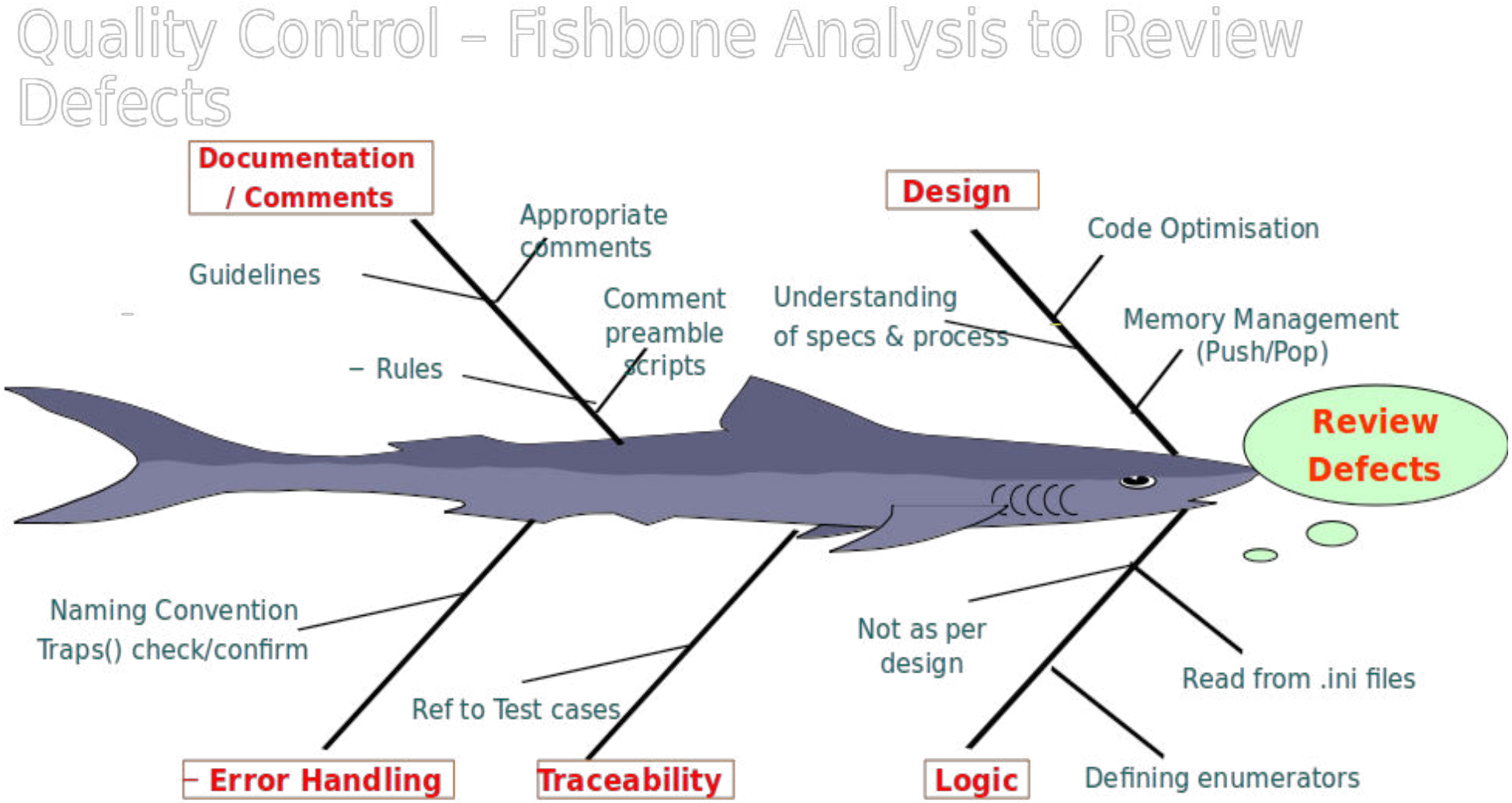
Quality Control

- Attempts to build a product, test it for expected behavior after it is built and if expected behavior is not same as actual behavior of the product, fixes the product as is necessary and rebuilds the product.
- Associated with people, testers
- QC will focuses on testing the quality
- QC is for detecting the defects/bugs

Difference between Quality Assurance and Quality Control

#	QA	QC
1.	Concentrates on the process of producing products	Concentrates on specific products
2.	Defect prevention oriented	Defect-detection and correction
3.	Usually done throughout the life cycle	Usually done after the product is built
4.	Usually a staff function	Usually a line function
5.	Examples: reviews and audits	Software testing at various levels

Quality Control- Fish Bone Analysis to review defects



Fish Bone Analysis to review defects (contd.)

The team using the fishbone diagram tool should carry out these steps.

- Agree on the problem statement (also referred to as the **effect**). This is written at the mouth of the "fish." Be as clear and specific as you can about the problem. Beware of defining the problem in terms of a solution (e.g., we need more of something).
- Agree on the major categories of **causes** of the problem (written as branches from the main arrow). Major categories often include: equipment or supply factors, environmental factors, rules/policy/procedure factors, and people/staff factors.
- Brainstorm all the possible causes of the problem. Ask "Why does this happen?" As each idea is given, the facilitator writes the causal factor as a branch from the appropriate category (places it on the fishbone diagram). Causes can be written in several places if they relate to several categories.
- Again asks "Why does this happen?" about each cause. Write sub-causes branching off the cause branches.
- Continues to ask "Why?" and generate deeper levels of causes and continue organizing them under related causes or categories. This will help you to identify and then address root causes to prevent future problems.

Ref: https://www.cms.gov/medicare/provider-enrollment-and-certification/qapi/downloads/fishbonerevised.pdf

Cost of Quality

- The cost of quality is the total price of all efforts to achieve product or service quality. This includes the work to build a product or service that conforms to the requirements and all work resulting non-conformance to the requirements.
- A typical project should have a goal of 3 to 5 percent of the total value devoted to the cost of a quality program.

Verification and Validation

Verification	Validation
Build the product right	Build the right product
Reactive	Proactive
Design v/s SRS Code v/s Design	Product V/s Need Product V/s Attributes
Mostly done by Team / Technical	Team and Customer & Domain Experts
Required	Required

Process Model to Represent Different Phases

It is a way to represent any given phase of software development that effectively builds in the concepts of validation and verification to prevent and minimize the delay between defect injection and defect detection. In this model. each phase of a software project is characterized by following:

1. **E**ntry
2. **T**ask
3. **V**erification
4. e**X**it

This model is also known as **ETVX.**

ETVX model applied to design

- ETVX Model for Design Phase
- (Entry, Task, Verification, Exit)

- Entry Criteria
- Customer approved SRS

- Input: SRS

- Tasks
- Evolve Architecture
- Create High Level Design
- Create Detailed Design
- Create Program Specs

- Exit Criteria
- Traceability matrix, Output ready for next phase

- Output:
- Design docs, Program Specs

THANK YOU

Software Testing (CS491)

Introduction

STLC and Testing Types

Prof Raghu B. A. Rao

Department of Computer Science and Engineering

Lecture Agenda

1. Why is Testing Important?
2. STLC
3. STLC Phases
4. Test Cases
5. Types of Testing

Why Is Testing Important?

- Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do.
- The benefits of testing include preventing bugs, reducing development costs and improving performance.
- The testing is important since it discovers defects/bugs before the delivery to the client, which guarantees the quality of the software.
- It makes the software more reliable and easy to use.
- Thoroughly tested software ensures reliable and high-performance software operation.

Testing

Testing in Software Engineering

As per ANSI/IEEE 1059, **Testing in Software Engineering** is a process of evaluating a software product to find whether the current software product meets the required conditions or not.

The testing process involves evaluating the features of the software product for requirements in terms of any missing requirements, bugs or errors, security, reliability and performance.

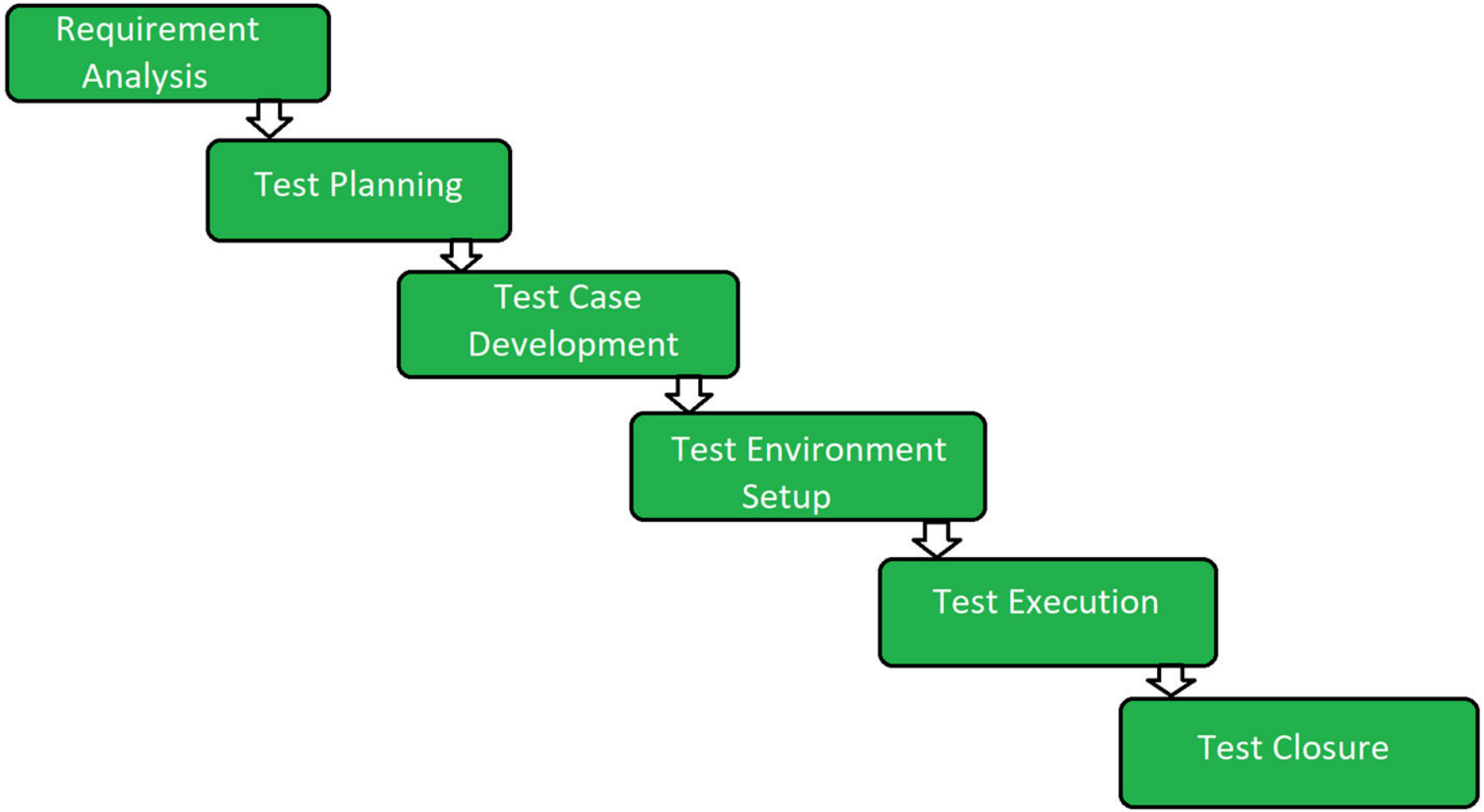
Types of Software Testing

- Functional Testing
- Non-Functional Testing or Performance Testing
- Maintenance (Regression and Maintenance)

STLC

- Software Testing Life Cycle (STLC) is a process used to test software and ensure that quality standards are met.
- Tests are carried out systematically over several phases.
- Software Testing Life Cycle (STLC) is a sequence of specific activities conducted during the testing process to ensure software quality goals are met.
- STLC involves both verification and validation activities.

STLC


Requirement Analysis
Test Planning
Test Case Development
Test Environment Setup
Test Execution
Test Closure

STLC Phases

There are following six major phases in every Software Testing Life Cycle Model (STLC Model):

1. Requirement Analysis
2. Test Planning
3. Test case development
4. Test Environment setup
5. Test Execution
6. Test Cycle closure

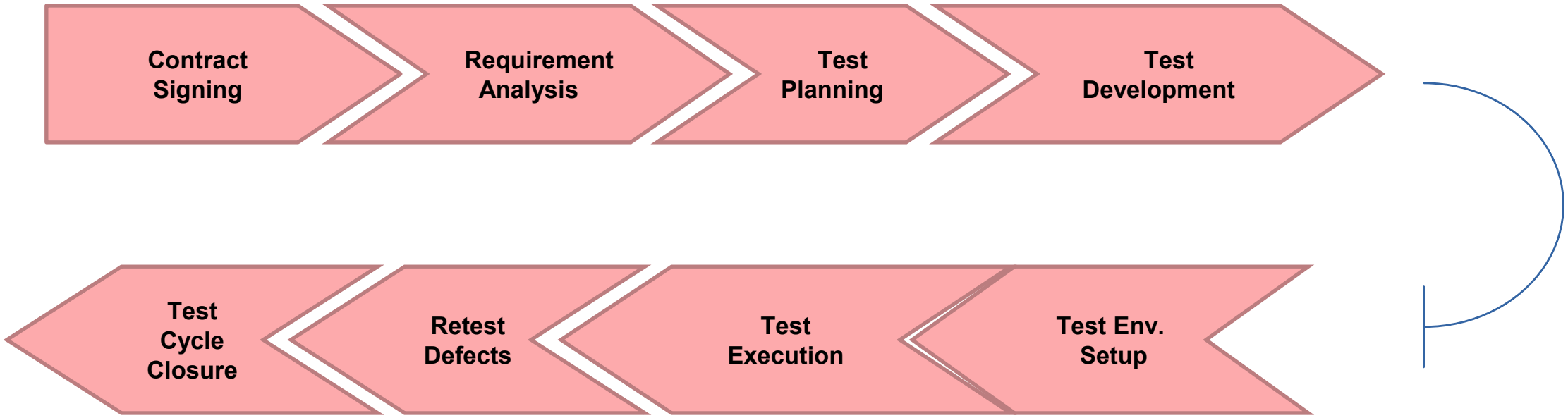
What is Entry and Exit Criteria in STLC?

Entry Criteria: Entry Criteria gives the prerequisite items that must be completed before testing can begin.

Exit Criteria: Exit Criteria defines the items that must be completed before testing can be concluded

Software Testing Life Cycle - STLC

The course of software being tested in a well-planned manner is known as Software Test Life Cycle (STLC). Each stage has a well defined Entry and Exit Criteria, Activities & Deliverables associated with it.



STLC Phases

1. **Requirement Phase Testing**

Requirement Phase Testing also known as Requirement Analysis in which test team studies the requirements from a testing point of view to identify testable requirements and the QA team may interact with various stakeholders to understand requirements in detail.

Activities in Requirement Phase Testing

- Identify types of tests to be performed.
- Gather details about testing priorities and focus.
- Prepare Requirement Traceability Matrix (RTM).
- Identify test environment details where testing is supposed to be carried out.
- Automation feasibility analysis (if required).

Deliverables of Requirement Phase Testing

- RTM
- Automation feasibility report. (if applicable)

STLC Phases

2. Test Plan

Test Planning in STLC is a phase in which a Senior QA manager determines the test plan strategy along with efforts and cost estimates for the project.

Moreover, the resources, test environment, test limitations and the testing schedule are also determined.

Test Planning Activities

- Preparation of test plan/strategy document for various types of testing
- Test tool selection
- Test effort estimation
- Resource planning and determining roles and responsibilities.
- Training requirement

Deliverables of Test Planning

Test Plan /strategy document : TestPlan.doc

STLC Phases

3. Test Case Development Phase

The Test Case Development Phase involves the creation, verification and rework of test cases & test scripts after the test plan is ready.

Test Case Development Activities

- Create test cases, automation scripts (if applicable)
- Review and baseline test cases and scripts
- Create test data (If Test Environment is available)

Deliverables of Test Case Development

- Test cases/scripts
- Test data

STLC Phases

4. Test Environment Setup

Test Environment Setup decides the software and hardware conditions under which a work product is tested.

It is one of the critical aspects of the testing process and can be done in parallel with the Test Case Development Phase.

Test Environment Setup Activities

- Understand the required architecture, environment set-up and prepare hardware and software requirement list for the Test Environment.
- Setup test Environment and test data
- Perform smoke test on the build

Deliverables of Test Environment Setup

- Environment ready with test data set up
- Smoke Test Results.

STLC Phases

5. Test Case Execution Phase

Test Execution Phase is carried out by the testers in which testing of the software build is done based on test plans and test cases prepared.

Test Execution Activities

- Execute tests as per plan
- Document test results, and log defects for failed cases
- Map defects to test cases in RTM
- Retest the Defect fixes
- Track the defects to closure

Deliverables of Test Execution

- Completed RTM with the execution status
- Test cases updated with results
- Defect reports

STLC Phases

6. Test Cycle Closure

Test Cycle Closure phase is completion of test execution which involves several activities like test completion reporting, collection of test completion matrices and test results.

Testing team members meet, discuss and analyze testing artifacts to identify strategies that have to be implemented in future, taking lessons from current test cycle.

Test Cycle Closure Activities

- Evaluate cycle completion criteria based on Time, Test coverage, Cost,Software, Critical Business Objectives, Quality
- Prepare test metrics based on the above parameters.
- Document the learning out of the project
- Prepare Test closure report
- Test result analysis to find out the defect distribution by type and severity.

Deliverables of Test Cycle Closure

- Test Closure report
- Test metrics

STLC Stages

STLC Stage	Entry Criteria	Activity	Exit Criteria	Deliverables
Requirement Analysis	- Requirements Document available (both functional and non functional) - Acceptance criteria defined. - Application architectural Document available.	- Identify types of tests to be performed. - Gather details about testing priorities and focus. - Prepare Requirement Traceability Matrix (RTM). - Identify test environment details where testing is supposed to be carried out. - Automation feasibility analysis (if required).	- Signed off RTM - Test automation feasibility report signed off by the client	- RTM - Automation feasibility report (if applicable)
Test Planning	- Requirements Documents - Requirement Traceability matrix. - Test automation feasibility document.	- Preparation of test plan/strategy document for various types of testing - Test tool selection - Test effort estimation - Resource planning and determining roles and responsibilities. - Training requirement	- Approved test plan/strategy document. - Effort estimation document signed off.	- Test plan /strategy document. - Effort estimation document.

STLC Stages

STLC Stage	Entry Criteria	Activity	Exit Criteria	Deliverables
Test case development	- Requirements Documents - RTM and test plan - Automation analysis report	- Create test cases, automation scripts (where applicable) - Review and baseline test cases and scripts - Create test data	- Reviewed and signed test Cases/scripts - Reviewed and signed test data	- Test cases/scripts - Test data
Test Environment setup	- System Design and architecture documents are available - Environment set-up plan is available	- Understand the required architecture, environment set-up - Prepare hardware and software requirement list - Prepare environment setup checklist - Setup test Environment and test data - Perform smoke test on the build - Accept/reject the build depending on smoke test result	- Environment setup is working as per the plan and checklist - Test data setup is complete - Smoke test is successful	- Environment ready with test data set up - Smoke Test Results.

STLC Stages

STLC Stage	Entry Criteria	Activity	Exit Criteria	Deliverables
Test Execution	- Baselined RTM, Test Plan , Test case/scripts are available - Test environment is ready - Test data set up is done - Unit/Integration test report for the build to be tested is available	- Execute tests as per plan - Document test results, and log defects for failed cases - Update test plans/test cases, if necessary - Map defects to test cases in RTM - Retest the defect fixes - Regression testing of application - Track the defects to closure	- All tests planned are executed - Defects logged and tracked to closure	- Completed RTM with execution status - Test cases updated with results - Defect reports
Test Cycle closure	- Testing has been completed - Test results are available - Defect logs are available	- Evaluate cycle completion criteria based on - Time, Test coverage , Cost , Software - Quality , Critical Business Objectives - Prepare test metrics based on the above parameters. - Document the learning out of the project - Prepare Test closure report - Qualitative and quantitative reporting of quality of the work product to the customer. - Test result analysis to find out the defect distribution by type and severity	Test Closure report signed off by client	- Test Closure report - Test metrics

Sample Test Case Document

Project Name:	**Google Email**
Module Name:	**Login**
Reference Document:	**If any**
Created by:	**Rajkumar**
Date of creation:	**DD-MMM-YY**
Date of review:	**DD-MMM-YY**

www.SoftwareTestingMaterial.com

TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/ FAIL)
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Valid User Name>	Successful login	Gmail inbox is shown		
				2. Enter Password	<Valid Password>				
				3. Click "Login" button					
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Valid User Name>	A message "The email and password you entered don't match" is shown			
				2. Enter Password	<Invalid Password>				
				3. Click "Login" button					
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Invalid User Name>	A message "The email and password you entered don't match" is shown			
				2. Enter Password	<Valid Password>				
				3. Click "Login" button					
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Invalid User Name>	A message "The email and password you entered don't match" is shown			
				2. Enter Password	<Invalid Password>				
				3. Click "Login" button					

Different Types of Test Cases

Test Case Type	Description	Test Step	Expected Result	Status
Functionality	Area should accommodate up to 20 characters	Input up to 20 characters	All 20 characters in the request should be appropriate	Pass or Fail
Security	Verify password rules are working	Create a new password in accordance with rules	The user's password will be accepted if it adheres to the rules	Pass or Fail
Usability	Ensure all links are working properly	Have users click on various links on the page	Links will take users to another web page according to the on-page URL	Pass or Fail

Test Cases

Test cases have a few integral parts that should always be present in fields, 8 basic steps.

Step 1: Test Case ID

Test cases should all bear unique IDs to represent them. In most cases, following a convention for this naming ID helps with organization, clarity, and understanding.

Step 2: Test Description

This description should detail what unit, feature, or function is being tested or what is being verified.

Step 3: Assumptions and Pre-Conditions

This entails any conditions to be met before test case execution. One example would be requiring a valid Outlook account for a login.

Test Cases

Step 4: Test Data

This relates to the variables and their values in the test case. In the example of an email login, it would be the username and password for the account.

Step 5: Steps to be Executed

These should be easily repeatable steps as executed from the end user's perspective.

For instance, a test case for logging into an email server might include these steps:

1. Open email server web page.
2. Enter username.
3. Enter password.
4. Click "Enter" or "Login" button.

Test Cases

Step 6: Expected Result

This indicates the result expected after the test case step execution. Upon entering the right login information, the expected result would be a successful login.

Step 7: Actual Result and Post-Conditions

As compared to the expected result, we can determine the status of the test case. In the case of the email login, the user would either be successfully logged in or not. The post-condition is what happens as a result of the step execution such as being redirected to the email inbox.

Step 8: Pass/Fail

Determining the pass/fail status depends on how the expected result and the actual result compare to each other.

Test Case Import

https://www.youtube.com/watch?v=Lz9JEX_tr-k

Types of Software Testing

1. Testing type based on the **method** used
    a. White-box testing
    b. Black-box testing

2. Testing levels based on **requirement type**
    a. Functional
    b. Non-functional

3. Testing targets based on **life cycle phase**
    a. Unit testing
    b. Integration testing
    c. System testing

4. Testing types based on **needs**
    a. Regression testing
    b. Acceptance testing
    c. Alpha & Beta testing

Testing Types – Based on Method Used

- Testing type based on the method used :
  - White-box – Focused on Code
  - Black-box – Focused on Requirements
- Objective being to achieve higher test coverage (both requirements and code coverage)

White Box Testing

- White box testing which also known as glass box is **testing, structural testing, clear box testing, open box testing and transparent box testing**.
- It tests internal coding and infrastructure of a software focus on checking of predefined inputs against expected and desired outputs.
- It is based on inner workings of an application and revolves around internal structure testing.
- In this type of testing programming skills are required to design test cases.
- The term 'white box' is used because of the internal perspective of the system. The clear box or white box or transparent box name denote the ability to see through the software's outer shell into its inner workings.
- Examples : Path testing, Loop testing, Condition testing

White Box Testing

White box testing involves the testing of the software code for the following

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output
- The functionality of conditional loops
- Testing of each statement, object, and function on an individual basis

Popular White Box Testing Tools :

- Parasoft Jtest
- EclEmma
- NUnit
- PyUnit
- HTMLUnit
- CppUnit

White Box Testing Example

```
Printme (int a, int b) {                    # Printme is a function
   int result = a+ b;
   If (result> 0)
         Print ("Positive", result)
   Else
         Print ("Negative", result)
   }
----------   End of the source code   ------------------
```

Black Box Testing

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths.

Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.

How to do Blackbox Testing

Here are the generic steps followed to carry out any type of Black Box Testing.

1. Initially, the requirements and specifications of the system are examined.
2. Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
3. Tester determines expected outputs for all those inputs.
4. Software tester constructs test cases with the selected inputs.
5. The test cases are executed.
6. Software tester compares the actual outputs with the expected outputs.
7. Defects if any are fixed and re-tested.

Types of Black Box Testing

- **Functional testing** – This black box testing type is related to the functional requirements of a system; it is done by software testers.
- **Non-functional testing** – This type of black box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.
- **Regression testing** – Regression Testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

Tools used for Black box testing largely depends on the type of black box testing you are doing.

- For Functional/ Regression Tests you can use – QTP, Selenium
- For Non-Functional Tests, you can use – LoadRunner, Jmeter

Testing Types – Based on Requirement Type

- Testing levels based on requirement type
  - Functional
  - Non-functional
- Functional testing could either be Black-box or White-box testing
- Non-functional testing could be any of the following
  - Load/ Stress Testing
  - Usability/ Compatibility Testing
  - Localization Testing etc.

Functional and Non-functional Testing

Functional Testing

- Functional testing is a type of software testing in which the system is tested against the functional requirements and specifications
- Functional testing ensures that the requirements or specifications are properly satisfied by the application.

Non-Functional Testing

- Non-functional testing is a type of software testing to test non-functional parameters such as reliability, load test, performance and accountability of the software.
- The primary purpose of non-functional testing is to test the reading speed of the software system as per non-functional parameters.
- Functional testing checks the correctness of internal functions while Non-Functional testing checks the ability to work in an external environment.

Differences between Functional and Non-functional Testing

Functional Testing	Non-functional Testing
It verifies the operations and actions of an application.	It verifies the behavior of an application.
It is based on requirements of customer.	It is based on expectations of customer.
It helps to enhance the behavior of the application.	It helps to improve the performance of the application.
Functional testing is easy to execute manually.	It is hard to execute non-functional testing manually.
It tests what the product does.	It describes how the product does.
Functional testing is based on the business requirement.	Non-functional testing is based on the performance requirement.
Examples : Unit Testing, Smoke Testing, Integration Testing, Regression Testing	Examples : Performance Testing, Load Testing, Stress Testing, Scalability Testing

Testing Types - Based on life cycle phase

- Testing targets based on life cycle phase
  - Unit testing – using White-box testing
  - Integration testing – using White/Black-box testing
  - System testing – using Black-box testing

- Focus changes from module-level to sub-system level to entire system

Testing Types - Based on needs

- Testing types based on needs
  - Regression testing
  - Acceptance testing
  - Alpha & Beta testing

Performed with a specific need or focus

●Regression testing – to ensure changes have not impacted existing functionality or behavior

●Acceptance testing – performed by customer before accepting the software into production

●Alpha & Beta testing
  - Alpha – with limited user base at development site
  - Beta – with larger end user base at customer site

Regression Testing

Regression testing is re-running functional and non-functional tests to ensure that previously developed and tested software still performs after a change.
This testing is done to make sure that new code changes should not have side effects on the existing functionalities.
It ensures that the old code still works once the latest code changes are done.

When can we perform Regression Testing?

- When new functionality added to the application.
- When there is a Change Requirement.
- When the defect fixed
- When there is a performance issue fix
- When there is an environment change

https://www.javatpoint.com/regression-testing

Regression testing across the release

The regression testing process starts whenever there is a new Release for same project because the new feature may affect the old elements in the previous releases.

To understand the regression testing process, we will follow the below steps:

Step 1 : There is no regression testing in **Release#1** because there is no modification happen in the Release#1 as the release is new itself.

Step 2 : The concept of Regression testing starts from **Release#2** when the customer gives some **new requirements**.

Step 3 : After getting the new requirements (modifying features) first, they (the developers and test engineers) will understand the needs before going to the **impact analysis**.

Step 4 : After understanding the new requirements, we will perform one round of **impact analysis** to avoid the major risk, but here the question arises who will do the Impact analysis?

Regression testing across the release

Step 5 : The impact analysis is done by the **customer** based on their **business knowledge**, the **developer** based on their **coding knowledge**, and most importantly, it is done by the **test engineer** because they have the **product knowledge**.

Step 6 : Once we are done with the **impact area**, then the developer will prepare the **impact area (document)**, and the **customer** will also prepare the **impact area document** so that we can achieve the **maximum coverage of impact analysis**.

Step 7 : After completing the impact analysis, the developer, the customer, and the test engineer will send the **Reports#** of the impact area documents to the **Test Lead**.

Regression testing across the release

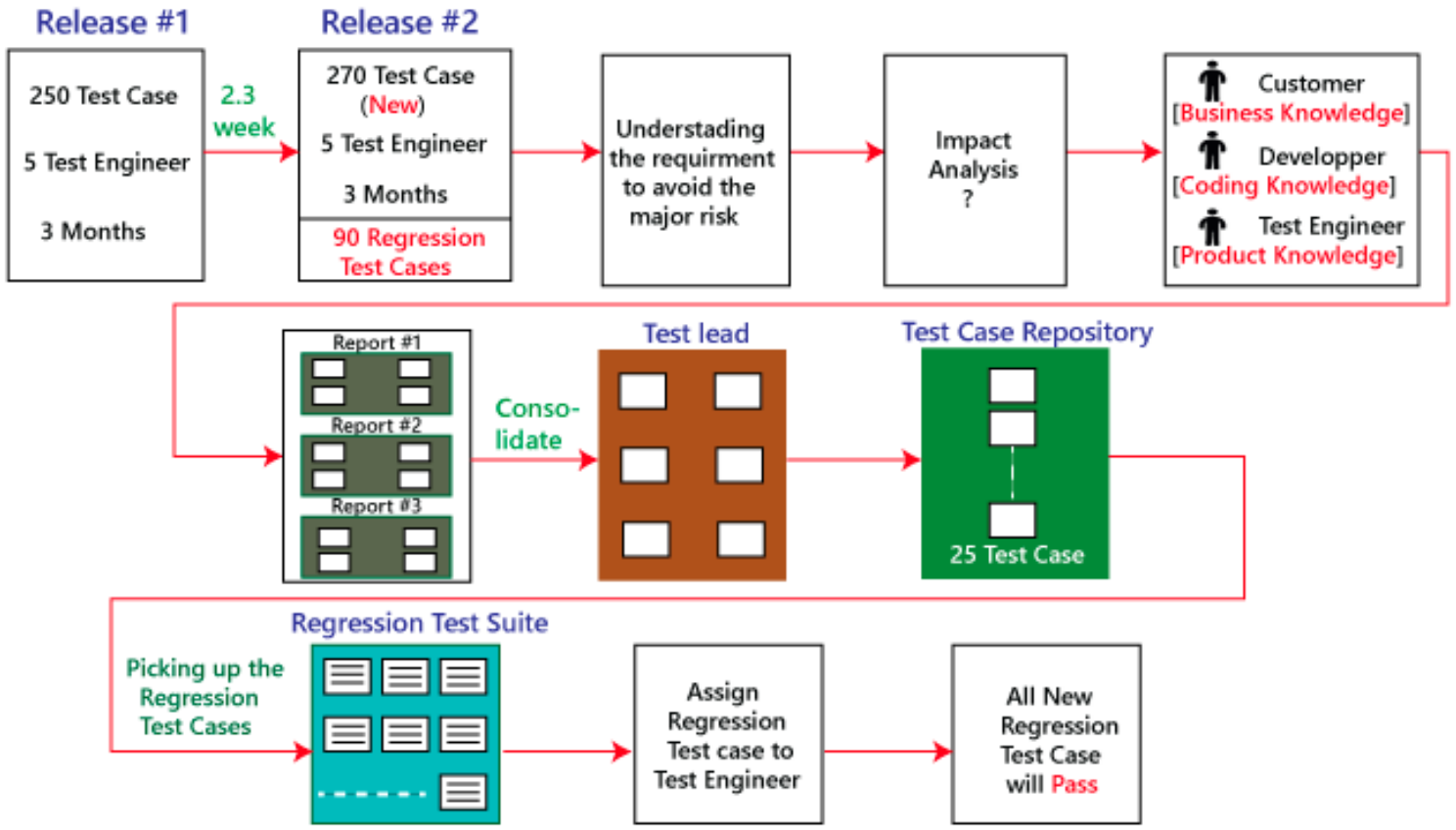
Step 8 : Once the Test lead gets the Reports#, he/she will consolidate the reports and stored in the test case requirement repository for the release#1.

Step 9 : After that, the Test Lead will take the help of RTM and pick the necessary regression test case from the test case repository, and those files will be placed in the Regression Test Suite.

Step 10 : After that, when the test engineer has done working on the new test cases, the test lead will assign the regression test case to the test engineer.

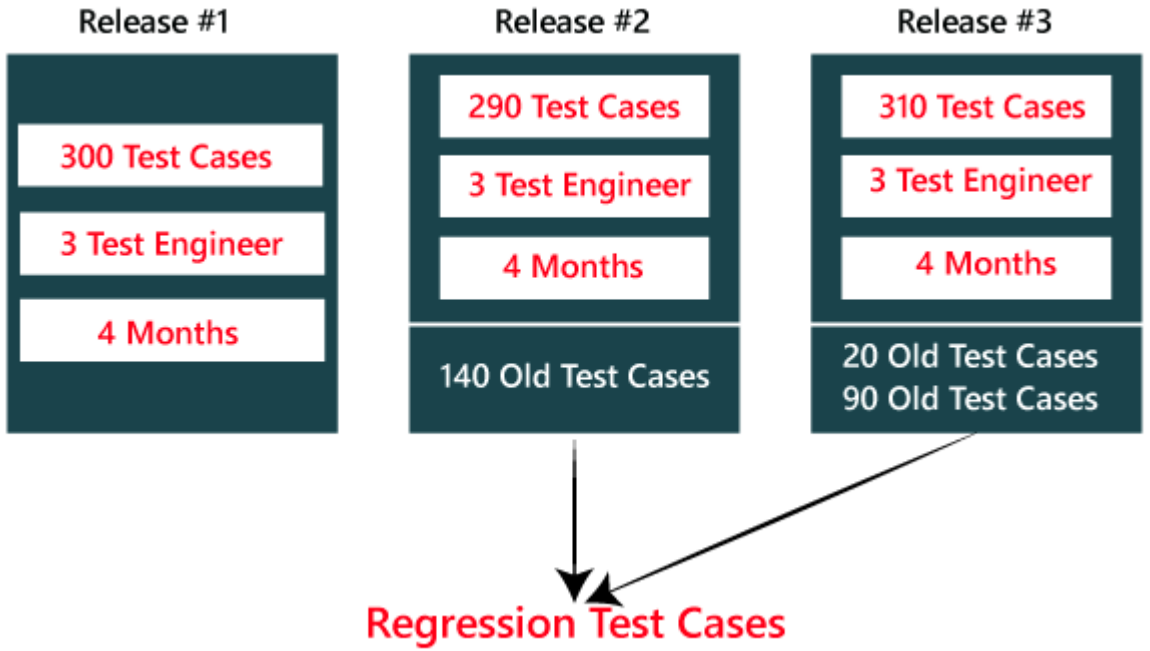
Step 11 : When all the regression test cases and the new features are stable and pass, then check the impact area using the test case until it is durable for old features plus the new features, and then it will be handed over to the customer.

Regression testing across the release



Regression testing across the release

Regression Test Cases: These are the test cases of the old releases text document which need to be re-executed as we can see in the below image



Unit Regression Testing [URT]

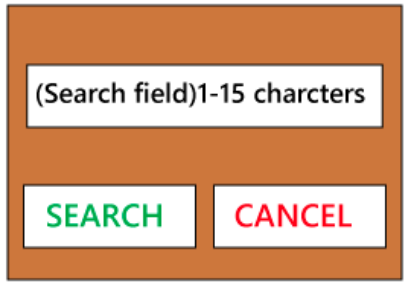
In this, we are going to test only the changed unit, not the impact area, because it may affect the components of the same module.

Example:

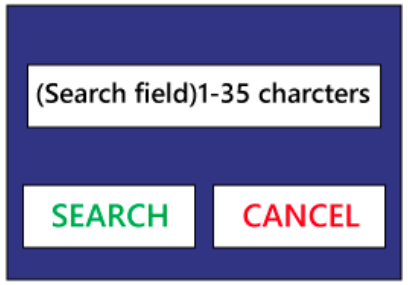
In the below application, and in the first build, the developer develops the **Search** button that accepts **1-15 characters**. Then the test engineer tests the Search button with the help of the **test case design technique**.

Now, the client does some modification in the requirement and also requests that the **Search button** can accept the **1-35 characters**.

The test engineer will test only the Search button to verify that it takes 1-35 characters and does not check any further feature of the first build.



Build 1-B001



Build 1-B002

Example1 https://www.javatpoint.com/regression-testing

Acceptance Testing

User acceptance testing (UAT) is a type of testing, which is done by the customer before accepting the final product. Generally, UAT is done by the customer (domain expert) for their and check whether the application is working according to given business scenarios, real-time scenarios.

Watir: Acceptance testing uses this tool for the execution of automated browser-based test cases. It uses Ruby language for the inter-process communication.

Fitness tool: This tool is used to enter input values and generate test cases automatically. The user needs to input values, these values used by the tool to execute test cases and to produce output.

Alpha Testing

- Alpha testing is conducted in the organization and tested by a representative group of end-users at the developer's side and sometimes by an independent team of testers.
- Alpha testing is a type of acceptance testing.
- Alpha testing is happening at the stage of the completion of the software product.
- Alpha testing is in-house testing, which is performed by the internal developers and testers within the organization.

THANK YOU