

**FACE-to-PHASE:**

**Getting to Know the Phases of Systems Development**

**UNIT 1**

**LESSON 1**

**Requirements**

**Learning Outcomes**

At the end of the lesson, you should be able to:

* Identify correctly and discuss the phases of the Requirements Life Cycle.
* Illustrate the steps of an existing system using a USE CASE Model.
* Create Activity Diagram of a system.

What are requirements?

* Requirements are statements that identify the essential needs of a system in order for it to have value and utility.

Characteristics of Good ReqUIREMENts

1. Describes What, Not How.

2. Atomic. i.e., it should have a single purpose

3. Unique.

4. Documented and Accessible.

5. Identifies Its Owner.

6. Approved. After a requirement has been revised, reviewed, and rewritten, it must be approved by its owner.

7. Traceable. A good requirement is traceable; it should be possible to trace each requirement back to its source.

8. Necessary.

9. Complete.

10. Unambiguous

11. Quantitative and testable

12. Identifies applicable states

14. States Assumptions. All assumptions should be stated.

15. Use of Shall, Should, and Will. A mandatory requirement should be expressed using the word shall (e.g., "The system shall conform to all state laws

16. **Avoids Certain Words**. The words optimize, maximize, and minimize should not be used in stating requirements, because we could never prove that we had achieved them.

**Requirements Life cycle**

**The User**

**Raw Req’ts**

**Organised Req’ts**

**Analysed Req’ts**

**Complete user Req’ts**

**SPECS**

C:\Program Files\Microsoft Office\MEDIA\CAGCAT10\j0195384.wmf

* **Elicitation Phase**

The starting point of the requirements engineering process is an elicitation process that involves a number of people to ensure consideration of a broad scope of potential ideas and candidate problems

* **Organization Phase**

In this step there is no transformation of the requirements, but simple classification and categorization. For example, requirements may be grouped into functional vs. nonfunctional requirements.

* **Analysis Phase**

This represents a transformation.

* **Prototype Phase**

In this way poorly understood requirements may be tested and perhaps strengthened, corrected, or refined. This activity is often done as a proof of concept and serves to induce feedback from both the stakeholders and engineers.

* **Requirements documentation and specification**

This represents the requirements as the finished product of the stakeholder requirements team. The requirements are compiled into a requirements list or into some equivalent document format. These collected requirements are then transformed into a specification.

* + 1. **Requirements Elicitation, Documentation, and Maintenance**

**Requirements elicitation**

* Requirements determination addresses the gathering and documentingof the true and real requirements for the Information System being developed.
* Requirements is the wants and /or needs of the user within a problem domain.

**Requirements determination questions**

* Who does it?
* What is done?
* Where is it done?
* When is it done
* How is it done
* Why is it done?

**Systems Requirements**

* Characteristics or features that must be included to satisfy business requirements
  + Outputs
  + Inputs
  + Processes
  + Timing
  + Controls
  + Volumes. sizes, and frequencies
* Data/Information collected can be about; people, organization, work and work environment.

**Fact – Finding Methods**

* Sampling (of existing documentation, forms, and databases).
* Research and site visits. (Participation)
* Observation of the work environment.
* Questionnaires.
* Interviews.
* Prototyping.
* JAD/Joint requirements planning (JRP).

**Types of Requirements**

* **User Requirements:** these are statements in Natural language plus diagrams of services the system provides, together with its operational constraints. These can be categorised into 2; functional requirements and non-functional requirements
  + **Functional requirements** 
    - Describe ***what*** the system should do
  + **Non-functional requirements** 
    - Consists of **C*onstraints*** that must be adhered to during development (design and implementation)
    - Remember ‘**Constraints**.’
* **System requirements**
  + What we agree to provide
  + Describes system services
  + Contract between Client and contractor

**Functional requirements**

* + What *inputs* the system should accept
  + What *outputs* the system should produce
  + What data the system should *store* that other systems might use
  + What *computations* the system should perform
  + The *timing and synchronization* of the above

**Non-functional requirements**

* Non-functional requirements are global constraints on a computer system
  + e.g. development costs, operational costs, performance, reliability,
* The challenge of Non-functional requirements:
  + Hard to model
  + Usually stated informally, and so are:
    - often contradictory,
    - difficult to enforce during development
    - difficult to evaluate for the customer prior to delivery
* Define system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations.
* Process requirements may also be specified mandating a particular programming language or development method
* Non-functional requirements may be more critical than functional requirements. If these are not met, the system is useless.

**Examples of NFR**

* Interface requirements
  + How will the new system interface with its environment?
  + User interfaces and “user-friendliness”
  + Interfaces with other systems
* Performance requirements
  + Time - response time
  + Output - transactions per second
* Security
  + permissible information flows
  + Or who can do what
  + Survivability – e.g. system will need to survive fire natural catastrophes, etc
* Operating requirements
  + Physical constraints (size, weight),
  + Personnel availability & skill level
  + Accessibility for maintenance
  + Environmental conditions
* Lifecycle requirements
  + Maintainability, Enhanciability, Portability, expected market or product lifespan
* limits on development
  + E.g. development time limitations, resource availability and methodological standards.
* Economic requirements
  + E.g. restrictions on immediate and/or long-term costs**.**

**Requirements Documentation**

* There are basically two types of documents realized from the requirements elicitation phase. These include;
  + User Requirements Specification Document
  + System requirements specification Document

**User Requirements Specification –URS/URD**

* The URS document outlines precisely what the User (or customer) is expecting from this system.
* User Requirement Specification may incorporate the functional requirements of the system or may be in a separate document labelled the Functional Requirements Specification - the FRS.

The URD has the following information:

* 1. Functional Requirements
  2. Non-Functional Requirements

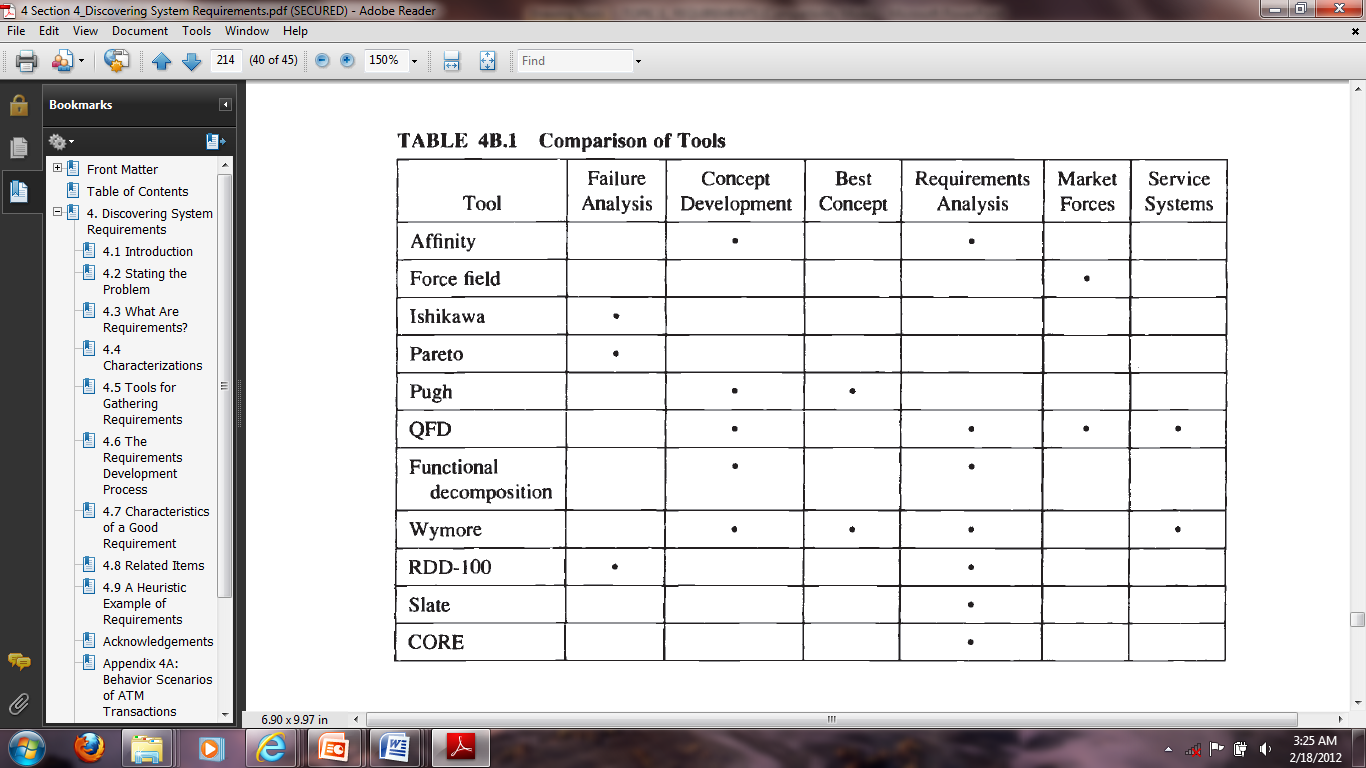
**System Requirements Specification Document**

A detailed description of the system services.

* What do we agree to provide?
* A structured document setting out detailed descriptions of the system services.
* Written as a contract between client and contractor.

**TOOLS THAT AID IN DEVELOPING & UNDERSTANDING SYSTEM REQuiremenTS**

* Affinity diagrams
* Force-field analysis
* Ishikawa fishbone (cause-and-effect) diagrams
* Pareto diagrams
* Pugh charts
* Quality function deployment (QFD)



* + 1. **Modelling Requirements**

**We build models in requirements analysis to understand**

* current systems or business processes which we are trying to automate
* how users will use a new system

The **software requirements** document is the official statement of what is required of the system developers.

* Should include both a definition of user requirements and a specification of the system requirements.
* It is NOT a design document. As far as possible, it should set WHAT the system should do rather than HOW it should do it.

**Requirements Document Variability**

Information in requirements document depends on the type of system and the approach to development used.

Systems developed incrementally will, typically, have less detail in the requirements document.

Requirements documents standards have been designed e.g. IEEE standard. These are mostly applicable to the requirements for large systems projects.

**The Structure of a Requirements Document**

|  |  |
| --- | --- |
| **Chapter** | **Description** |
| Preface | This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version. |
| Introduction | This should describe the need for the system. It should briefly describe the system’s functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software. |
| Glossary | This should define the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader. |
| User requirements definition | Here, you describe the services provided for the user. The nonfunctional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified. |
| System architecture | This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted. |

|  |  |
| --- | --- |
| System requirements specification | This should describe the functional and nonfunctional requirements in more detail. If necessary, further detail may also be added to the nonfunctional requirements. Interfaces to other systems may be defined. |
| System models | This might include graphical system models showing the relationships between the system components and the system and its environment. Examples of possible models are object models, data-flow models, or semantic data models. |
| System evolution | This should describe the fundamental assumptions on which the system is based, and any anticipated changes due to hardware evolution, changing user needs, and so on. This section is useful for system designers as it may help them avoid design decisions that would constrain likely future changes to the system. |
| Appendices | These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions. Hardware requirements define the minimal and optimal configurations for the system. Database requirements define the logical organization of the data used by the system and the relationships between data. |
| Index | Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on. |

* + 1. **Modelling Tools and Methodologies**

**Unified Modeling Language (UML)**

* Use case diagrams
* Class diagrams
* Sequential diagrams
* State Diagrams

**USE CASE MODEL**

* Describes the proposed functionality of the new system
* Represent a discrete unit of interaction between a user and the system
* A single unit of meaningful work

Customer

**Use Case Model**

<<extends>>

A **Use Case** may ‘include’ another Use Case’s functionality or ‘extend’ another Use Case with its own behavior.

Example:

**Login to system**

**Register with system**

**Create order**

A Use Case description will generally include:

* General comments and notes describing the Use Case
* Requirements- things that the Use Case must allow the user to do such as

<ability to update order>

<ability to modify order> and others.

A Use Case may include:

* Constraints- rules about what can and cannot be done:
  + Pre-conditions that must be true before the Use Case is run

–e.g. <create order> must precede <modify order>;

* Post-conditions that must be true before once the Use Case is run

–e.g. <order is modified and consistent>;

* Invariants: these are always true

–e.g. an order must always have a customer number;

* Scenarios: Sequential descriptions of the steps taken to carry out the Use Case. May include multiple scenarios, to cater for exceptional circumstances and alternate processing paths;
* Scenario Diagrams: Sequence diagrams to depict the workflow – as above but graphically portrayed;
* Additional attributed such as implementation phase, version number, complexity rating, stereotype and status.

**An ACTOR:**

* Is a user of the system
* This includes both human user and other computer systems
* Uses a Use Case to perform some piece of work which is of value to the business
* The set of Use Cases an actor has access to, defines their overall role in the system and the scope of their action

Person

Student

* Actors can participate in a generalization with other actors

Is a generalization of

Student

* Actors may be connected to Use Cases only by associations

Student

Billing System

Registrar

* Here we have a Student interacting with the Registrar and the Billing System via a “Register for Courses” Use Case

**Requirements, Constraints & Scenarios**

* **Requirements**:

These are the formal functional requirements that a Use Case must provide to the end user. They correspond to the functional specifications found in structured methodologies. A requirement is a contract that the Use Case will perform some actions or provide some value to the system.

* **Constraints**:

These are the formal rules and limitations that a Use Case operator under, and includes pre- post- and invariant conditions.

A **pre-condition** specifies what must have already occurred or be in place before the Use Case may start.

A **post-condition** documents what will be true once the Use Case is complete. An **invariant** specifies what will be true throughout the time the Use Case operates.

* **Scenarios**:

These formal descriptions of the flow of events that occurs during a Use Case instance.

Usually described in text and correspond to a textual representation of the Sequence Diagram.

* **Includes and Extends Relationships Between Use Cases**
  + May include the functionality of another as part of its normal processing
  + It is assumed that the included Use Case will be called every time the basic path is run.

*Example* <list order> Use Case may be included every time the <modify order> Use Case is run.

* + A Use Case may be included by one or more Use Cases. This helps to reduce duplication of functionality by factoring out common behavior into Use Cases that are re-used many times.

**SEQUENCE DIAGRAMS**

* + A sequence diagram is an interaction diagram that emphasizes the time ordering of messages
  + It shows a set of objects and the messages sent and received by those objects
  + Sequence diagrams can be used to document Use Case Scenarios
  + Captures required objects early in analysis and verify object usage later in design
  + Shows the flow of messages from one object to another, and as such correspond to the methods and events supported by a class/object
  + Graphically, a sequence diagram is a table that shows objects arranged along the X axis and messages, ordered in increasing time, along the Y axis.

**Sequence Diagrams – Objects Symbols**

* An object in a sequence diagram is rendered as a box with a dashed line descending from it.

An Order Line

* The line is called the object lifeline, and it represents the existence of an object over a period of time

**Sequence Diagrams – Message Indicators**

An Order Line

A Stock Item

Check ()

[Check=”True”]

* Messages are rendered as horizontal {check()} arrows being passed from object to object as time advances down the object lifelines.
* Conditions (such as [check = “True”] indicate when a message gets passed.
* Notice that the bottom arrow is not solid, and there is no accompanying message, this indicates return from a previous message not new message

**Sequence Diagrams – Iteration Marker**

An Order Line

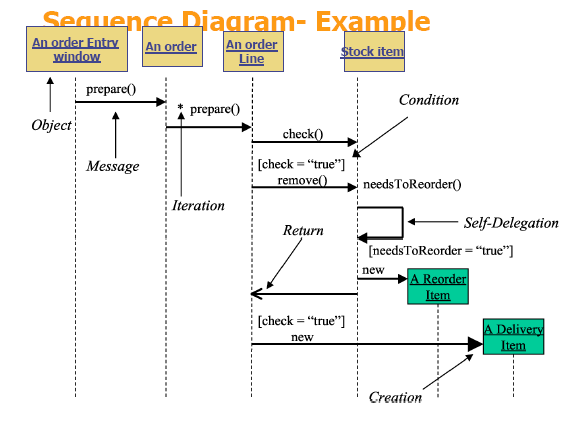
A Stock Item

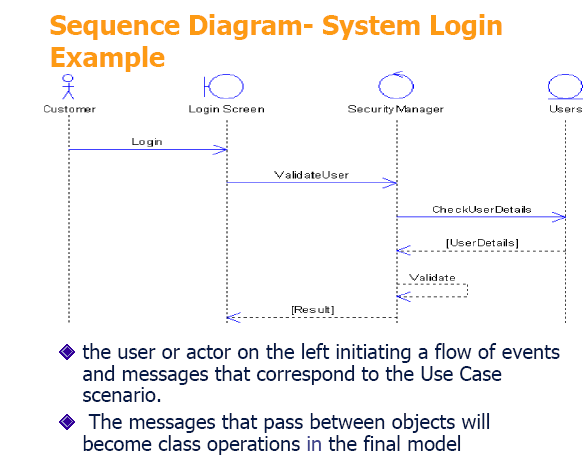
\*prepare ()

* An iteration marker, such as \* (as shown) or \*[I=1…n], indicates that a message will be repeated as indicated

Iteration Marker

**Sample Sequence Diagrams:**

**(a)**

**(b)**

**ACTIVITY DIAGRAM**

* Models the procedural flow of actions that are part of a larger activity
* Used to model system-level functions
* Focuses on the action sequence of execution and the conditions that trigger or guard those actions.
* Has a very similar notations to that of a state machine diagram

Customer Calls Ticket Office

* An action is indicated by a “capsule” shape – a rectangular object with semi-circular left and ends.
* The text inside it indicates the action (e.g., Customer Calls Ticket Office or Registration Office Opens).

First Action To Do

* The initial state is drawn as a solid circle with a transition line (arrow) that connects it to the first action in the activity’s sequence of actions.
* It is important to note that there can be only one initial state on an activity diagram and only one transition line connecting the initial state to an action.

Action 1

Action 2

The Figure above indicates *INCORRECT* rendering of an initial state within an activity diagram.

* Arrows indicate directions, the transition lines on an activity diagram show the sequential flow of actions in the modelled activity. It always point to the next action in the activity’s sequence.

**Activity Diagram – How a customer books a concert ticket.**

Customer Calls Ticket Office

Tickets Rep Asks What Event Person Wants Ticket

Customer Tells Rep Which Event

Ticket Rep Tells Customer Available Seats and Prices

Customer Tells Rep Seats

Ticket Rep Reserves Seats

Ticket Rep Asks For Credit Card and Billing Address

Customer Gives Requested Information

Ticket Rep Charges Credit Card

Ticket Rep Mails Tickets

The sample activity diagram documents the activity “Booking a Concert Ticket,” with actions in the following order:

1. Customer calls ticket office.
2. Ticket rep asks what event person wants ticket for.
3. Customer tells rep event choice.
4. Ticket rep tells customer available seats and prices.
5. Customer tells rep seating choice.
6. Ticket rep reserves seats.
7. Ticket rep asks for credit card and billing address.
8. Customer gives requested information.
9. Ticket rep charges credit card.
10. Ticket rep mails tickets.

The action order is clear from the diagram because it shows an initial state (starting point), and from that point one can follow the transition lines as they connect the activity’s action. It is possible for an activity diagram to show multiple final sates. Unlike initial state symbols, of which there can be only one on an activity.

**Decision points**

Typically, decisions need to be made throughout an activity, depending on the outcome of a specific prior action.

Customer Orders Drink

[Drink is Alcoholic]

[else]

Make Sure Customer is at Least 21 years Old

Get Drink for Customer

* Each transition line involved in a decision point must be labelled with text above it to indicate “guard conditions,” commonly abbreviated as guards.
* Guard condition text is always placed in brackets—for example, [guard condition text]
* A guard condition explicitly tells when to follow a transition line to the next action

**Merge points**

Sometimes the procedural flow from one decision path may connect back to another decision path. In these cases we connect two or more action paths together using the same diamond icon with multiple paths pointing to it, but with only one transition line coming out of it. This indicate a *merge* decision point.

Tell Customer to Order Non-Alcoholic Drink

[Customer’s Age <21]

Make Sure Customer is at Least 21 years Old

Customer Orders Drink

[else]

[Customer’s Age >=21]

[Drink is Alcoholic]

Get Drink for Customer

A partial activity diagram, showing two decision points:

(“Drink is alcoholic” and “Customer’s Age <21”) and one merge

(“else” and “Customer’s Age >=21”)

* + 1. **Testing**

**What is Testing?**

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

**Who does Testing?**

It depends on the process and the associated stakeholders of the project(s). In the IT industry, large companies have a team with responsibilities to evaluate the developed software in context of the given requirements. Moreover, developers also conduct testing which is called Unit Testing. In most cases, the following professionals are involved in testing a system within their respective capacities:

 Software Tester

 Software Developer

 Project Lead/Manager

 End User

Different companies have different designations for people who test the software on the basis of their experience and knowledge such as Software Tester, Software Quality Assurance Engineer, QA Analyst, etc.

It is not possible to test the software at any time during its cycle. The next two sections state when testing should be started and when to end it during the SDLC.

**When to Start Testing?**

An early start to testing reduces the cost and time to rework and produce error-free software that is delivered to the client. However in Software Development Life Cycle (SDLC), testing can be started from the Requirements Gathering phase and continued till the deployment of the software.

It also depends on the development model that is being used. For example, in the Waterfall model, formal testing is conducted in the testing phase; but in the incremental model, testing is performed at the end of every increment/iteration and the whole application is tested at the end.

Testing is done in different forms at every phase of SDLC:

* During the requirement gathering phase, the analysis and verification of requirements are also considered as testing.
* Reviewing the design in the design phase with the intent to improve the design is also considered as testing.
* Testing performed by a developer on completion of the code is also categorized as testing.

**When to Stop Testing?**

It is difficult to determine when to stop testing, as testing is a never-ending process and no one can claim that a software is 100% tested. The following aspects are to be considered for stopping the testing process:

 Testing Deadlines

 Completion of test case execution

 Completion of functional and code coverage to a certain point

 Bug rate falls below a certain level and no high-priority bugs are identified

 Management decision

**Verification & Validation**

These two terms are very confusing for most people, who use them interchangeably. The following table highlights the differences between verification and validation.

|  |  |  |
| --- | --- | --- |
| **S.N.** | **Verification** | **Validation** |
| 1 | Verification addresses the concern: "Are you building it right?" | Validation addresses the concern: "Are you building the right thing?" |
| 2 | Ensures that the software system meets all the functionality. | Ensures that the functionalities meet the intended behavior. |
| 3 | Verification takes place first and includes the checking for documentation, code, etc. | Validation occurs after verification and mainly involves the checking of the overall product. |
| 4 | Done by developers. | Done by testers. |
| 5 | It has static activities, as it includes collecting reviews, walkthroughs, and inspections to verify a software. | It has dynamic activities, as it includes executing the software against the requirements. |
| 6 | It is an objective process and no subjective decision should be needed to verify a software. | It is a subjective process and involves subjective decisions on how well a software works. |

**Software Testing - Myths**

Given below are some of the most common myths about software testing.

**Myth 1: Testing is Too Expensive**

**Reality**: There is a saying, pay less for testing during software development or pay more for maintenance or correction later. Early testing saves both time and cost in many aspects, however reducing the cost without testing may result in improper design of a software application rendering the product useless.

**Myth 2: Testing is Time-Consuming**

**Reality**: During the SDLC phases, testing is never a time-consuming process. However diagnosing and fixing the errors identified during proper testing is a time-consuming but productive activity.

**Myth 3: Only Fully Developed Products are Tested**

**Reality**: No doubt, testing depends on the source code but reviewing requirements and developing test cases is independent from the developed code. However iterative or incremental approach as a development life cycle model may reduce the dependency of testing on the fully developed software.

**Myth 4: Complete Testing is Possible**

**Reality**: It becomes an issue when a client or tester thinks that complete testing is possible. It is possible that all paths have been tested by the team but occurrence of complete testing is never possible. There might be some scenarios that are never executed by the test team or the client during the software development life cycle and may be executed once the project has been deployed.

**Myth 5: A Tested Software is Bug-Free**

**Reality**: This is a very common myth that the clients, project managers, and the management team believes in. No one can claim with absolute certainty that a software application is 100% bug-free even if a tester with superb testing skills has tested the application.

**Myth 6: Missed Defects are due to Testers**

**Reality**: It is not a correct approach to blame testers for bugs that remain in the application even after testing has been performed. This myth relates to Time, Cost, and Requirements changing Constraints. However the test strategy may also result in bugs being missed by the testing team.

**Myth 7: Testers are Responsible for Quality of Product**

**Reality**: It is a very common misinterpretation that only testers or the testing team should be responsible for product quality. Testers’ responsibilities include the identification of bugs to the stakeholders and then it is their decision whether they will fix the bug or release the software. Releasing the software at the time puts more pressure on the testers, as they will be blamed for any error.

**Myth 8: Test Automation should be used Wherever Possible to Reduce Time**

**Reality**: Yes, it is true that Test Automation reduces the testing time, but it is not possible to start test automation at any time during software development. Test automaton should be started when the software has been manually tested and is stable to some extent. Moreover, test automation can never be used if requirements keep changing.

**Myth 9: Anyone can Test a Software Application**

**Reality**: People outside the IT industry think and even believe that anyone can test a software and testing is not a creative job. However testers know very well that this is a myth. Thinking alternative scenarios, try to crash a software with the intent to explore potential bugs is not possible for the person who developed it.

**Myth 10: A Tester’s Only Task is to Find Bugs**

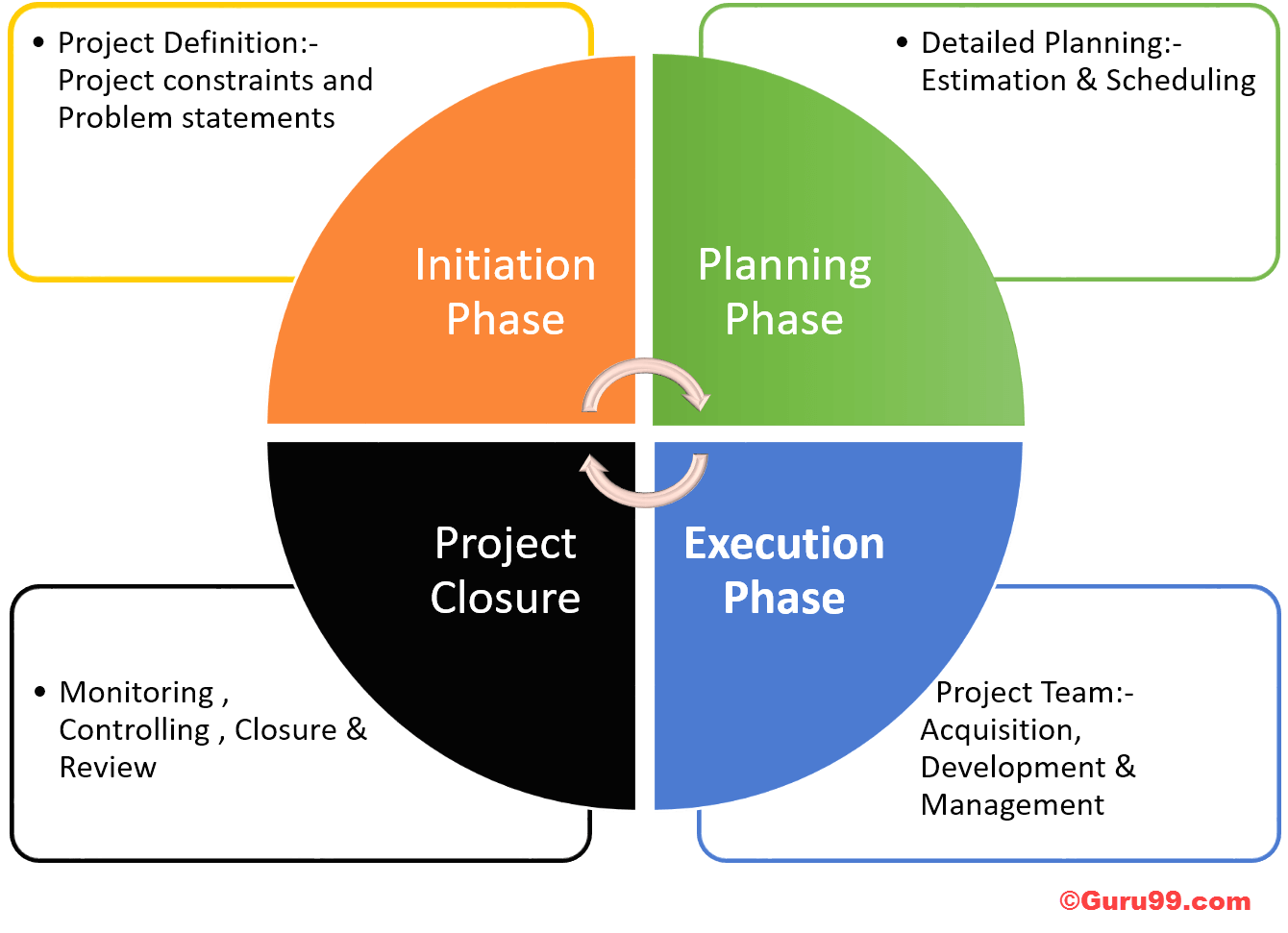
**Reality**: Finding bugs in a software is the task of the testers, but at the same time, they are domain experts of the particular software. Developers are only responsible for the specific component or area that is assigned to them but testers understand the overall workings of the software, what the dependencies are, and the impacts of one module on another module.

* + 1. **Project Lifecycle Phases**

**What is Project Life Cycle?**

The Project Life Cycle is a series of activities which are essential for accomplishing project objectives or targets. Projects may have different dimensions and difficulty level, but, whatever the size: large or small, may be all projects could be mapped to the given lifecycle structure. This life cycle for the project includes four phases-

* [Initiation Phase](https://www.guru99.com/initiation-phase-project-management-life-cycle.html#1)
* [Planning Phase](https://www.guru99.com/initiation-phase-project-management-life-cycle.html#2)
* [Execution Phase](https://www.guru99.com/initiation-phase-project-management-life-cycle.html#3)
* [Monitoring, Controlling & Closing Phase](https://www.guru99.com/initiation-phase-project-management-life-cycle.html#4)

[](https://www.guru99.com/images/projmgmt.png)

**Project Life Cycle Diagram**

1. **Project Initiation Stage**

Initiation phase defines those processes that are required to start a new project. The purpose of the project initiation phase is to determine what the project should accomplish.

This phase mainly composed of two main activities

* Develop a Project Charter and
* Identify Stakeholders

All the information related to the project are entered in the Project Charter and Stakeholder Register. When the project charter is approved, the project becomes officially authorized.

Project Charter

The Project Charter defines the project's main elements

* Project goals
* Project constraints and Problem statements
* Assign project manager
* Stakeholder list
* High-level schedule and budget
* Milestones
* Approvals

This document allows a project manager to utilize organizational resources for the sake of the project. To create a project charter, the inputs required will be enterprise environment factor, business case, agreements, a project statement of work and organizational process assets.

Identifying Stakeholders

A [stakeholder](https://www.guru99.com/stakeholder-needs-analysis.html) can influence the success and failure of the project. To note down the information about the stakeholder, a Stakeholder Register is used.

The stakeholder register will have information like

* Type of stakeholder
* Expectation of stakeholder
* Role in Project ( Business Analyst, Tech architect, Client PM)
* Designation (Director, Business Lead, etc.)
* Type Communication ( Weekly/Monthly)
* Influence on the project ( Partial/Supportive/Influensive)

The other activities involved in initiating process group are:

* Assigning the project manager
* Determining the stakeholder needs, expectations and high-level requirements
* Define the project success criteria
* Identify particular budget for particular stage
* Make sure that the project is aligned with the organizations strategic goal

The stakeholder register and project charter are used as inputs to the other development groups such as planning process group.

1. **Project Planning Stage**

Project Planning phase covers about 50% of the whole process. Planning phase determines the scope of the project as well as the objective of the project. It begins with the outputs of initiation phase (charter, preliminary scope statement, and project manager). The output of the planning phase serves as the input for the execution phase.

The important aspects of planning process are

* Planning phase should not be executed before your initial planning is finished
* Until the execution process does not start, you should not stop revising plans

Create Work Breakdown Structure (WBS)

For any successful project WBS (Work Breakdown Structure) is important. Following are steps to create WBS.

* Conduct a brainstorm to list all the tasks
* Involve your whole team for brainstorming
* Write down the structure tree of the task also known as WBS (work breakdown structure)
* Further breakdown your top WBS into a hierarchical set of activities, for instance, categories, sub-categories, etc. For example hardware, software, trainee, management teams, etc.
* Define how to record the items into your WBS
* Ask other people - it can be an expert, experienced personnel, etc.
* Granularity- how detailed your task should you have? Estimating cost and time for higher granularity is hard while for lower granularity it will be bogged down with too detailed information
* Granularity should be of right level not too high or not too low

Planning Schedule Management

Plan Scheduling is the process of establishing the procedure, policies and documentation for planning, managing, executing and controlling the project schedule. The inputs in these activities include

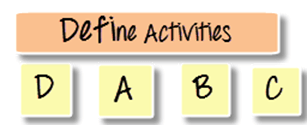
* Project management plan
* Project Charter
* Enterprise environmental factors
* Organizational process assets

The output of the Planning Schedule Management includes

* Schedule management plan

Defining Activities

Defining Activities is the procedure for documenting and identifying specific actions to be performed to produce the project deliverables.

[](https://www.guru99.com/images/9-2015/082815_0706_PlanningPha1.png)

In define activities, each work packages is broken down into individual work schedule activities. The inputs of the defining activities include

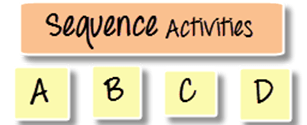
* Schedule management plan
* Scope baseline
* Enterprise environmental factors
* Organizational process assets

While the outputs of these activities are

* Activity list
* Activity attributes
* Milestone list

Sequence Activities

Sequence activities is nothing but logically organizing the output of "define activities". It determines the order in which the activities needs to be performed.

[](https://www.guru99.com/images/9-2015/082815_0706_PlanningPha2.png)

The main output from the sequence activity process is "Network Diagram".

Network diagram is nothing but posting the task on a board in a logical order.

For example, you want to start a business in foreign country what will be your list of activities and what will be the order it should be done?

You will perform activities in these order

1. Choose a country
2. Get business permit
3. Hiring a manager
4. Buying a property
5. Buying the furniture etc.
6. Opening the business

Estimating Activity Resources

This stage explains the process of estimating the work effort and resources required to complete the task. The other factor that has to be considered at this stage is the availability of the resources.

While estimating resources, the focus should be on the longest path of the plan (Critical Path), which going to consume more time and money.

You have to estimate resources for two tasks

* Critical tasks
* Floating tasks

Make sure that your critical tasks are accurately estimated (completion time).

There are five inputs used to estimate activity resources

* Schedule Management Plan
* Activity list
* Resource Calendar
* Enterprise environmental factors
* Organizational process assets

The output of this stage is

* Activity resource requirements
* Resource breakdown structure
* Project documents updates

NOTE: All the activity that is done so far (define activities + sequence activities + Estimate activity resources) is going to help in "Develop Schedule."

Estimating Activity Durations

Estimating Activity Duration is the process of estimating the number of work periods (weeks/months) required to complete the individual task with estimated resources. This step defines how much time an individual task will take to complete.

You cannot calculate activity duration without calculating the work effort and resources required to complete the task. Estimating process should be done in this order

* Estimate work effort first
* Followed by estimating the resources
* Followed by Estimating the duration of task

To estimate activity durations, you need inputs

* Activity list
* Activity attributes
* Resource calendars
* Project scope statement
* Organizational process assets
* Enterprise environmental factors

While there are two main outputs

* Estimate activity durations
* Estimate activity durations-project document updates

This technique is also referred as PERT (Project Evaluation and Review Techniques) estimates.

Develop Schedule

Develop Schedule is the process of analyzing activity sequences, resource requirements, durations and schedule constraints to create the project schedule model. For scheduling each task, three main factors are taken into consideration

* Duration
* Task dependencies
* Constraints

Using these factors project calculates the start date and finish date for each task.

A scheduling software can be used to create a schedule. It generates a schedule model with planned dates for completing project activities.

The input of this tool includes

* Schedule management plan
* Activity list
* Activity attributes
* Project schedule –network diagrams
* Activity resource requirements
* Resource calendars
* Activity duration estimates
* Project scope statement
* Risk register
* Project staff assignments
* Resource breakdown structure
* Enterprise environmental factors
* Organizational process assets

The output from this would be

* Project Schedule
* Project network diagram
* Gantt charts or Bar charts
* Milestone chart
* Schedule baseline
* Scheduled data
* Project document updates

Control Schedule

The last stage of the planning phase is Control Schedule. It is the process of monitoring the status of project activities to update project process and manage changes to the schedule baseline.

If changes are required to the schedule, they must go through the change control process. The schedule should be managed or controlled by manager proactively.

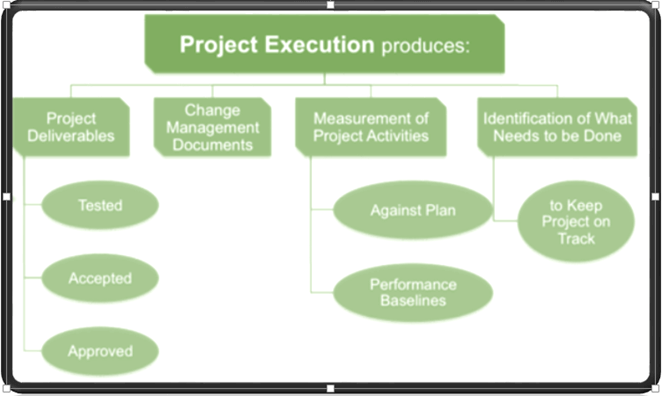
There are four main outputs of control schedule process

* Project management plan
* Schedule baseline
* Schedule management plan
* Project schedule
* Work performance information
* Organizational process assets

There are five outputs of control schedule

* Work performance management
* Organizational process assets updates
* Change request
* Project management plan updates
* Project document updates

1. **Project Execution Stage**

[](https://www.guru99.com/images/9-2015/082815_0747_ExecutionPh1.png)The executing phase consists of those activities that are defined in project management plan. This process involves managing stakeholder expectations, coordinating with people and resources, as well as performing other activities related to project deliverables.

During the execution phase, the result may require re-baselining and updates to existing project requirements. Action taken in execution phase may affect the project management plan or documents.

Direct and Manage Project Execution

This stage consumes most of the project cost, time, and resources as this is the process that produce project deliverables.

There are four inputs to Direct and Manage Project Execution

* Project Management Plan
* Approved change request
* EEFs (Enterprise Environmental Factors)
* OPAs (Organizational Process Assets)

While there are five outputs

* Deliverables
* Work performance data
* Change request
* Project Management plan updates
* Project documents updates

During this stage, expert's judgments, meetings, and reporting KPI (Key Performance Indicators) are of prime importance.

Performing Quality Assurance

Performing[Quality Assurance](https://www.guru99.com/all-about-quality-assurance.html)is the process of auditing the quality requirements and the results from quality control measurements. It is the process of recording and monitoring results of the quality activities to assess performance. Various tools like control charts, cost-benefit analysis, flowcharting, run charts, scatter diagrams, inspection & reviews, etc., can be used for this process.

The main input to this is

* Project management plan
* Quality metrics
* Quality control measurements
* Work performance information

While, the output of this is

* Change request
* Project management plan updates
* Project document updates
* Organizational process assets updates

Acquiring Project Team

During the execution phase, project team acquiring takes place, this is because it is more likely that individuals with different skill set will be required during the process.

There are three main inputs to acquire project team

* Roles and responsibilities
* Project organization chart
* Staffing management plan

While there are three outputs

* Project staff assignments
* Resource calendars
* Project management plan updates

Develop Project Team

The majority of human resource processes involves in executing process, developing project team is also a part of it. The main purpose of developing project team is to improve the overall performance of team members. This stage must start early on in the project.

The inputs in project development team include

* Human resource management plan
* Project staff assignments
* Resource calendars

Output of this process include

* Team performance assessments
* EEFs Updates

Manage project team

Managing project team is one of the important parts of project management. It is the most complex area of project management because many times managers would not be in direct contact with team members, in such situation to analyze their performance and deciding their remuneration becomes difficult.

There are five inputs to manage project team process

* Project staff assignments
* Team performance assessments
* Performance reports
* Project management plan
* Organizational process assets

There are four main outputs

* Organizational process assets updates
* Enterprise environmental factors updates
* Change request
* Project management plan updates

Manage Communications

Out of three communication attributes, one falls in the execution process. In communication management program, there are three main communication aspects that need to monitor.

1. Project team members to the project manager
2. Project managers to the program manager
3. Program manager to stakeholders or other sponsors

The input of managing communications include

* Communications management plan
* Work performance reports
* EEFs
* OPAs

The output of this stage would be

* Project communications
* Project management plan updates
* Project documents updates
* OPAs updates

Conduct Procurements

In this stage, there are two main roles involved the buyer and the seller. During the procurement process the activities involved are

1. Issue the bid package to potential sellers
2. Hold bidder conferences
3. Evaluate potential seller proposals
4. Select the winning seller proposals

The output of the procurement process include

* Project management plan
* Conduct procurement documents
* Source selection criteria
* Qualified seller list
* Seller proposals
* Project documents
* Make or buy decisions
* Partnership agreement (teaming agreement)
* Organizational process assets

While, you will have six outputs

* Selected sellers
* Procurement contract award
* Resource calendars
* Change requests
* Project management plan updates

Manage Stakeholder Engagement

This stage includes actively managing stakeholders throughout the project. To avoid unexpected project delay or abandoning the project in between, stakeholder expectation is identified and quickly resolved.

There are five inputs to manage stakeholder process

* Stakeholder register
* Stakeholder management strategy
* Project management plan
* Issue log
* Change log
* Organizational process assets

The output of this process include

* Organizational process assets updates
* Change request
* Project management plan updates
* Project documentation updates

Project Phase Review

At the end of execution phase, project phase review is done. It helps you to document in following activities

* Document the result of your project management review
* Inform the sponsor about the progress of the project
* Identifying any risk or issues that impacted the project
* Shows deliverable to stakeholder produced during the project
* Seek approval to proceed to the next phase

1. **Project Monitoring and Controlling & Closing Stage**

After execution phase, to check the project is on right track, monitoring and controlling phase becomes active. During this phase various changes and reviews to enhance the project performance is done.

Monitor and Control Project Work

This stage involves tracking, reviewing and regulating the progress in order to meet the objective of the project. It also ensures that the deliverables are according to the project management plan. The main focus of this step is to identify any changes made from the point of project management plan to determine appropriate preventive action.

The inputs for this stage include

* Project management plan
* Performance reports
* Cost forecasts
* Schedule forecasts
* Validate changes
* Enterprise environmental factors
* Organizational process assets

While the output includes

* Change requests
* Project management plan updates
* Project document updates

Perform Integrated Change Control

It is one of the most important process of project management. It is in this stage where the impact of any change is assessed against the project. If a change in this stage occurs at any one part of a project, the whole project will be assessed. It is better to implement changes at an early stage of the project, because as the project progresses, the cost of implementing changes also increases.

The input of this stage includes

* Project management plan
* Work performance reports
* Change requests
* EEFs
* OPAs

While the outputs are

* Approved change requests
* Change log
* Project management plan updates
* Project document updates

Validate Scope

Validating scope involves verifying whether the deliverables meet the customer acceptance criteria. The external checking with the customer or stakeholders are part of Validating Scope Management.

The inputs for validating scope includes

* Project management plan
* Requirements
* Documentation
* Requirements traceability matrix
* Verified deliverables
* Work performance data

While the output of the scope validation includes

* Accepted deliverables
* Change requests
* Work performance information
* Project document updates

Control Scope

Control scope ensures that it is the only work identified as being in scope that is delivered. The actual result is compared against the scope baseline and ensures that all of the approved scope is in fact being delivered.

The inputs to control scope process includes

* Project management plan
* Work performance information
* Requirement documentation
* Requirements traceability matrix
* Organizational process assets

While the output includes

* Work performance measurements
* Organizational process assets updates
* Change requests
* Project management plan updates
* Project document updates

Control Schedule

Control Schedule process helps you in many ways. It helps you to capture current schedule status, determine the variance from the schedule baseline, understand the nature of the variance and respond by taking appropriate action.

If changes are needed to the schedule then they must go through the change control process, the change should be re-evaluated and only then it should be used to update the schedule baseline.

There are four main inputs to the control schedule

* Project management plan
* Schedule baseline
* Schedule management plan
* Project schedule
* Work performance information
* Organizational process assets

The output includes

* Work performance measurements
* Organizational process assets updates
* Change requests
* Project management plan updates
* Project document updates

Control Cost

Control cost is comparing baseline cost for each deliverable against the actual cost. The cost baseline should change only in response to a change request that has gone through the Perform Integrated Change Control process. Control cost ensures that your project stay within funding limitations.

The inputs for the Control Cost include

* Project management plan
* Project funding requirements
* Work performance information
* Control Cost Organizational process assets

The output for this include

* Earned value work performance measurements
* Earned value budget forecasts in control costs
* Change requests
* Project management plan updates
* Project document updates
* Organizational process assets updates

Control Quality

The control quality ensures that the project and product are delivered with the quality management plan. It ensures that whether the work is performed correctly. The major output of the control quality is Quality management plan. While the other information that will be helpful are

* Existing flowchart
* Upper and lower control and specification limits contained within the control charts
* Information is referenced such as sample criteria, sampling numbers, measurements and variable sampling
* Quality metrics- it is a standard measurement to meet the quality requirements
* It ensures that the proper steps are being followed in order to comply with aspects such as process, policies or regulations

There are four main outputs from the perform quality control process:

* Integrated change control
* Approved change requests
* Approved change requests review
* Validated changes

Control Communications

Control communication ensures that the right information reaches to the stakeholder. Control communication information includes inputs, tools and techniques and output that belong to this process.

Control communication can be in any format, it can be

* Trending data
* Tabulated information
* S-curve
* Dashboard formats
* Use histogram

In control communication process, work information is taken from various other processes, and the performance report is used as an input for various monitoring and managing processes. The main deliverables from the control communication process is the performance record.

Control Risks

Throughout the project cycle, risk analysis is a continuous process. It is important that you continuously analyze, identify and respond to risks. The activities include in control risk are

* Tracking existing risks
* Monitoring residual risks
* Identifying new risks
* Implementing risk response plans
* Continuously evaluating risk process

The input for control risk are

* Risk register
* Work performance information
* Performance reports
* Reserve analysis
* Risk Audits

The output for the control risk are

* Updating risk register
* Risk management plan

Control Procurements

Out of four procurement plan, the third process of procurement falls in Monitoring & Executing process group. This stage involves monitoring the vendor's performance and ensuring that all contract requirements are being met.

The control procurement process involves verifying

* Whether goods or service being delivered
* Whether it is delivered on time
* Whether invoice charged is for correct quantity
* Whether all conditions of the contract being met
* Whether the relationship between buyer or seller are managed properly

The major input for procurement process are

* Project management plan
* Procurement documents
* Agreements
* Approved change requests
* Work performance reports
* Work performance data

The output for procurements are

* Work performance information
* Change requests
* Project management plan updates
* Project document updates
* OPAs updates

Control Stakeholder Management

Many project stumble due to inadequate management of stakeholders. If the stakeholders are managed properly, there are more chances for project success. In this process, we monitor the current engagement level of stakeholders and take actions accordingly.

The input and output for all these activities include

|  |  |
| --- | --- |
| Input | Output |
| * Plan Stakeholder Management | * Work performance information |
| * Issue log | * Change requests |
| * Work performance data | * Project management plan updates |
| * Project documents | * Project documents updates |
|  | * OPAs updates |

Closing- Phase

Closing phase is the process that performs a controlled shut down of the project at the end. In a project, there are three closure activities that are going on

* Closure of the product- Getting the customer to accept the final deliverables, if the project is external
* Closure of the project- This include formally closing of administrative procedures, updating project documents and archiving those databases & documents
* Closure of the resource behind the project- The financial closure of the project, resources assigned to the project should be returned

The inputs for this process include

* Project Management Plan
* Accepted Deliverables
* OPAs

The output of this process include

* Final output, service or result transition
* OPAs updates

Close Procurements

For each project development life cycle phase- planning, executing, monitoring and controlling & closing there is one procurement process. The final closing procurement is done as per the contract between the seller and buyer.

The closing activities and deliverables include:

* Project performance reviews including management of risks and issues
* Updated project management plan to reflect actual results
* Final reports distributed to appropriate stakeholders

The input for closing procurement include

* Project management plan
* Procurement documents

While the output include

* Closed procurement
* OPAs updates

Project Management Ethic of code and conduct

In the end, you will come across project management ethic of code and conduct which deals various human behavioral aspects such as

* Responsibility
* Respect
* Fairness
* Honesty
* Cultural Competence

This code is practiced to induce the confidence and bring a common frame of behavior in the project manager.

Summary:

Initiation phase defines those processes that are required to start a new project. It defines what project should accomplish in due course of time.

The initiation phase mainly composed of two main activities

* Develop a Project Charter
* Identify Stakeholders

The stakeholder register and project charter are also useful in other process groups of project management like planning process.

Planning phase determines the scope as well as the objective of the project. It involves creating a set of plans that guides you through the execution and closure phases of the project.

The executing phase consists of those activities that are defined in project management plan. It is the longest phase of the project life cycle and consumes maximum energy and resources. Action taken in execution phase may affect the project management plan or documents.

Key task in execution phase are

* Execute Project Management Plans
* Direct and Manage Project Execution
* Execute Task Assignments
* Conduct Progress Status Meetings, etc.

During the execution phase, the result may require re-baselining and updates to existing project requirements.

Monitoring and controlling stage ensures that the deliverables are according to the project management plan before closing phase.

The main focus of this phase is to identify any changes made from the point of project management plan to determine preventive action against any unexpected result.

Closing phase is the process that performs a controlled shut down of the project at the end.

REFERENCES

https://www.enlightened.com/wp-content/uploads/2017/10/ENL-Build-vs-Buy-White-Paper.pdf

http://ksuweb.kennesaw.edu/~svandev/AcqLfeCycle.pdf

https://www.tutorialspoint.com/software\_testing/software\_testing\_tutorial.pdf

https://muele.mak.ac.ug/course/view.php?id=1563

https://opentextbc.ca/projectmanagement/chapter/chapter-3-the-project-life-cycle-phases-project-management/

https://www.nibusinessinfo.co.uk/content/advantages-and-disadvantages-outsourcing

https://fourcornerstone.com/middleware-advantages-disadvantages/

http://www.cems.uwe.ac.uk/~kg-doyle/tdrewry/modeling.htm

https://www.slideshare.net/koko3159/modeling-system-requirement

https://www.slideshare.net/meenakshi24/11-system-development-models

\*https://www.tutorialspoint.com/software\_testing\_dictionary/system\_testing.htm

https://blog.careerminds.com/insource-vs-outsource

https://qualityandinnovation.com/2008/10/18/quality-system/

https://www.tmap.net/wiki/quality-characteristics