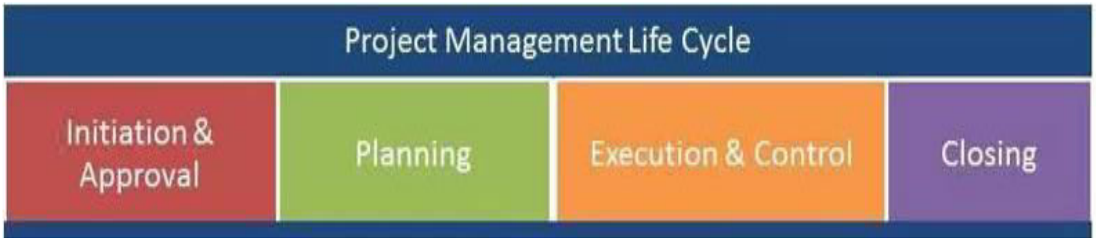
UE17CS402:
SOFTWARE ENGINEERING (4-0-0-0-4)

UNIT 1

Question and Answers

1. **Discuss the different phases involved with Project Management**

   Soln.



Discuss each phase e.g.

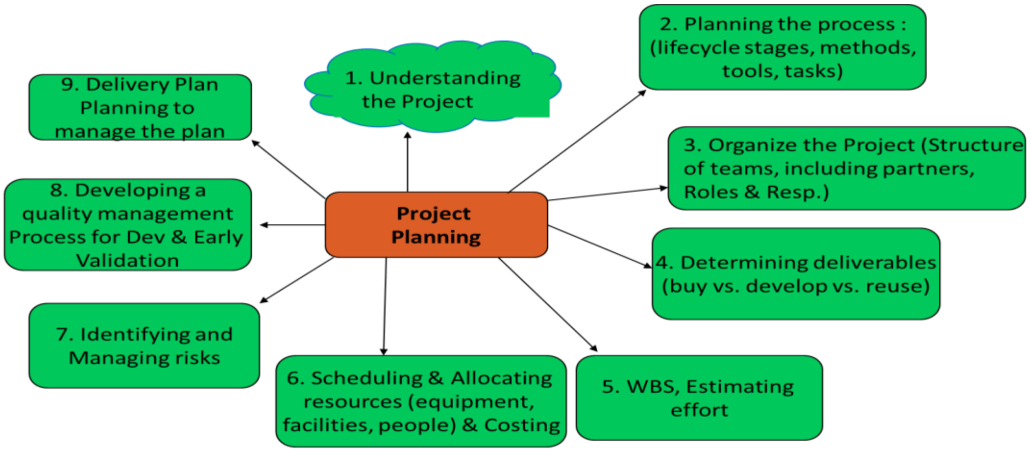
Initiation and Approval:

This happens during the Feasibility study activity in the early part of requirements engineering phases. Entry criteria: Feasibility Report available and approved for development

Planning:

There are 9 steps which are involved in planning as listed below. Discuss each of these in terms of what needs to be achieved and why.

Ex. Understanding the project needs to understand the expectations of stakeholders, customers to ensure that planning factors in requirements for all stakeholders like finance, Development, Management etc.


Project Planning
1. Understanding the Project
2. Planning the process : (lifecycle stages, methods, tools, tasks)
3. Organize the Project (Structure of teams, including partners, Roles & Resp.)
4. Determining deliverables (buy vs. develop vs. reuse)
5. WBS, Estimating effort
6. Scheduling & Allocating resources (equipment, facilities, people) & Costing
7. Identifying and Managing risks
8. Developing a quality management Process for Dev & Early Validation
9. Delivery Plan Planning to manage the plan

.....

2. **Illustrate upstream and downstream dependencies in a project with at least two examples each**

Soln:

- Upstream dependencies in a project are things that your project is reliant on happening before something else can happen. Eg. Implementation cannot be complete before Design is complete. Testing cannot happen before reviews are complete. So in these examples Design and Reviews become upstream dependencies.
- Downstream dependencies are those subsequent tasks which are waiting for this task to be completed. Eg. Testing would be a downstream partner for reviews in the above example. Acceptance testing would be a downstream partner for Implementation.

3. **Explain the characteristics of Software architecture?**

Soln.

- Multitude of stakeholders: Software systems architecture involves dealing with a broad variety of concerns and variety of stakeholders such as business managers, owners, users and operators who could be multidisciplinary nature.
- Separation of concerns to drive the design: Architecture documentation shows different stakeholder concerns addressed by modelling and describing the architecture from separate points of view associated.(called architectural views).
- Quality-driven: Architecture of a software system is more closely related to its quality attributes such as fault-tolerance, backward compatibility, extensibility, reliability, maintainability, availability, security, usability, and other such –ilities. Stakeholder concerns often translate into requirements on these quality attributes, which are variously called non-functional requirements, extra-functional requirements, system quality requirements or constraints.
- Recurring styles: Software architecture discipline has developed standard ways to address recurring concerns which are called as recurring solutions are architectural style, strategy or tactic, reference architecture and architectural pattern.
- Conceptual integrity: Architecture of a software system represents an overall vision of what it should do and how it should do it while separating the implementation. The architect would help keeping the vision and hence preserving conceptual integrity.

4. **Illustrate with two examples of architectural conflicts which you would find while making choices in the architecture**

Soln.

- Using large-grain components improves performance but reduces maintainability.
- Introducing redundant data improves availability but makes security / data integrity more difficult.

- Localizing safety-related features usually means more communication so degraded performance.

5. **Describe say 4 different considerations which influence the design characteristics like simplicity, maintenance.**

Soln.

Any four of the following

1. Abstraction

   Abstraction ensures focus on essential properties and ignores (abstracts) details that are not as relevant. E.g. Need sorting to happen to the data .... But don’t need to know the sorting algorithm. These could be data or procedural abstraction.

2. Modularity, coupling and cohesion

   Modularity would be a part of structuring the system. Cohesion would be the glue that keeps a module together. This needs to be made as strong as possible. Coupling would be the measure of the strength of the connection between modules. Loose coupling between components is beneficial to a system.

3. Information hiding

   Design involves a series of decisions while considering who needs to know and who can be kept in the dark

4. Limit complexity

   Look at the complexity of the dependencies between modules and the information flow between the modules and limit the same.

5. Hierarchical structure

   The structure of the modules in the design has an impact on the complexity, information hiding and of course the design.