MODULE 4

1.1 INTRODUCTION TO SOFTWARE PROJECT MANAGEMENT

1. Software Project Management is an art & Science of planning & leading software Projects from **ideas to reality.**
2. A Software Project is the complete procedure of software development from requirement gathering to testing and maintenance, carried out according to the execution methodologies, in a specified period of time to achieve intended software product
3. Project management is the discipline of defining and achieving targets while optimizing the new resources (time, money, people, materials, energy, space , etc.) over the course of a project (a set of activities of finite duration).
4. Project management involves the planning, monitoring, and control of people, process, and events that occur during software development.

Everyone manages, but the scope of each person's management activities varies according his or her role in the project.

Software needs to be managed because it is a complex undertaking with a long duration time.

Managers must focus on the fours P's to be successful (people, product, process, and project).

A project plan is a document that defines the four P's in such a way as to ensure a cost effective, high quality software product.

The only way to be sure that a project plan worked correctly is by observing that a high-quality product was delivered on time and under budget.

1.2 WHY IS SOFTWARE PROJECT MANAGEMENT IMPORTANT ?

- Large amounts of money are spent on ICT (information and communication technology) e.g. UK government in 2003-04 spent € 2.3 billions on contracts for ICT and only € 1.4 billion on road building. (1 billion =100 crore).
- Project often fail – Standish Group claim only a third of ICT projects are successful. 82 % were late and 43 % exceeded their budget. Poor project management a major factor in these failures.

- The methodology used by the Standish Group to arrive at their findings has been criticized, but the general perception of the prevalence of ICT project failure is still clear.

Software Development Life Cycle:

The Software Development life cycle is a methodology that also forms the framework for planning and controlling the creation, testing, and delivery of an information system.

The software development life cycle concept acts as the foundation for multiple different development and delivery methodologies, such as the Hardware development life -cycle and software development life -cycle . While Hardware development life -cycle deal specially with hardware and Software development life -cycle deal with software, a systems development life -cycle differs from each in that it can deal with any combination of hardware and software , as a system can be composed of hardware only , software only, or a combination of both.

Four Project Dimensions:

- People
- Process
- Product
- Technology

The 5 Variables of Project Control

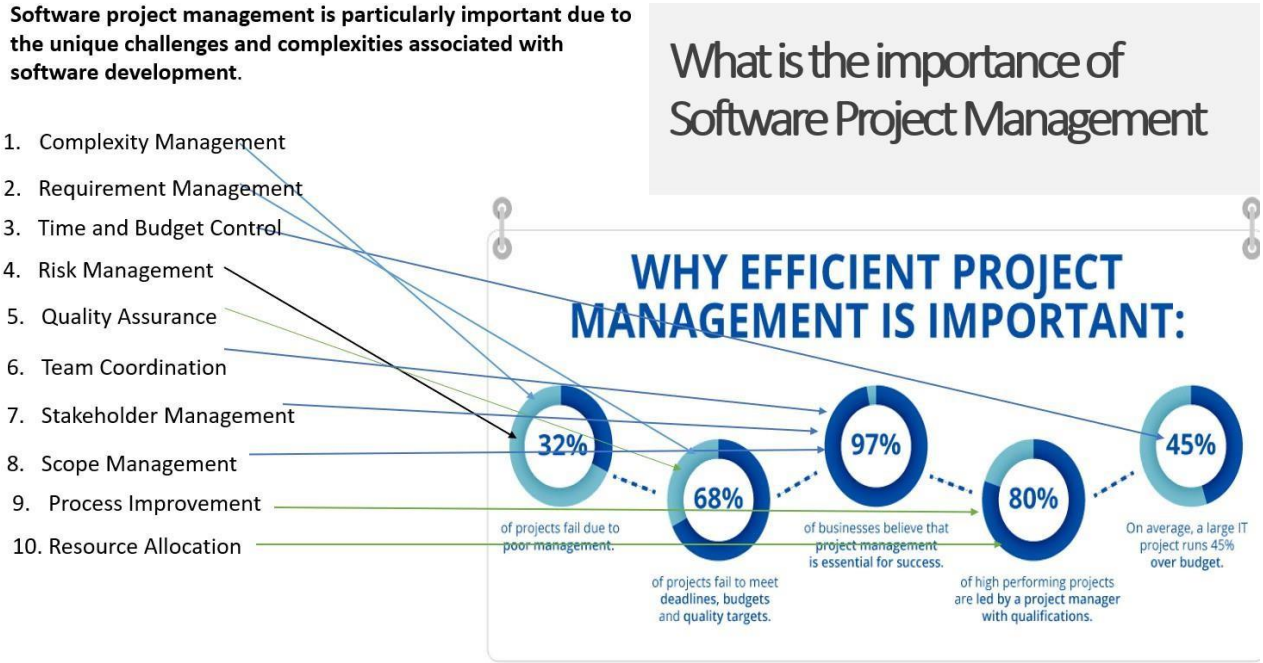
1. Time: amount of time required to complete the project.
2. Cost: calculated from the time variable
3. Quality: The amount of time put into individual tasks determines the overall quality of the project.
4. Scope: Requirements specified for the end result.
5. Risk: Potential points of failure.

Trade -off Triangle

The triangle illustrates the relationship between three primart forces in a project. Time is the available time to deliver the project. Cost represents the amount of money or resources available and quality represents the fit-to-purpose that the project must achieve to be a scuccess.

The normal situation is that one of thse factors is fixed and the other two will vary in inverse proportion to each other. For example , time is often fixed and the quality of the end product will depend on the cost and resources available. Similarly if you are working to a fixed level of quality

then the cost of the project will largely be dependable upon the time available(if you have longer you can do it with fewer people).



1. **Complexity Management**
   - Software projects often involve intricate systems and interdependencies. Effective management of this complexity ensures that the project remains coherent and manageable.
2. **Requirement Management**
   - Clear and precise requirement management is essential to ensure that the final product meets user needs and expectations. Mismanagement here can lead to scope creep and project failure.
3. **Time and Budget Control**
   - Monitoring and controlling the project timeline and budget is vital. This includes planning, estimating, and adhering to schedules and financial constraints to prevent overruns.
4. **Risk Management**

   Identifying, assessing, and mitigating risks can prevent unforeseen issues from derailing a project. This proactive approach helps in managing uncertainties effectively.
5. **Quality Assurance**
   - Ensuring that the project meets quality standards is crucial for user satisfaction and reducing post-release defects. Continuous testing and validation are key practices.
   -
6. **Team Coordination**
   - Effective communication and coordination among team members are essential for collaboration and timely problem-solving, ensuring that everyone is aligned with project goals.

7. **Stakeholder Management**
   - Engaging and managing stakeholders helps in gaining their support and addressing their concerns, which is critical for project acceptance and success.
8. **Scope Management**
   - Defining and controlling what is included in the project prevents scope creep, ensures that all necessary features are delivered, and avoids unnecessary work.
9. **Process Improvement**
   - Continuously improving processes ensures that the project is using the most efficient methods and practices, leading to better performance and outcomes.
10. **Resource Allocation**
    - Efficient allocation and management of resources (human, financial, and material) ensure that the project has what it needs to succeed without wastage.

Statistics Highlighting the Importance of Efficient Project Management

1. **32% of Projects Fail Due to Poor Management**
   - This statistic underscores the critical impact of project management on the overall success of software projects. Poor management can lead to project failures, highlighting the need for skilled project managers.
2. **68% of Projects Fail to Meet Deadlines, Budgets, and Quality Targets**
   - This indicates that a significant majority of projects struggle with time, budget, and quality control. Effective project management practices in these areas can significantly improve success rates.
3. **97% of Businesses Believe Project Management is Essential for Success**
   - This near-unanimous belief among businesses highlights the recognized value of project management. It underscores that investing in good project management practices is seen as crucial for achieving business objectives.
4. **80% of High-Performing Projects are Led by a Project Manager with Qualifications**
   - This shows a clear correlation between the qualifications of the project manager and the performance of the project. Qualified project managers bring skills and knowledge that drive project success.
5. **On Average, a Large IT Project Runs 45% Over Budget**
   - This statistic points to common budget overruns in large IT projects, emphasizing the need for rigorous budget control and efficient resource management to prevent financial overshooting.

Conclusion

Effective software project management is essential due to the inherent complexities and challenges of software development. The key areas outlined require diligent attention and management to ensure project success. The statistics provided illustrate the high stakes involved and the substantial impact that good project management can have on the success rates of software projects. By focusing on these areas, businesses can significantly improve their chances of delivering successful projects that meet deadlines, stay within budget, and satisfy quality standards.

1.3 WHAT IS A PROJECT :

The definition of a project as being planned assume that to a large extent we can determine how we are going to carry out a task before we start. There may be some projects of an exploratory nature where this might be quite hard. Planning is in essence thinking carefully about something before you do it and even in the case of uncertain projects this is worth doing as long as it is accepted that the resulting plans will have provisional and speculative elements. Other activities, concerning, for example, to routine maintenance, might have been performed so many times that everyone involved knows exactly what needs to be done. In these cases, planning hardly seems necessary, although procedures might need to be documented to ensure consistency and to help newcomers to the job.

Dictionary definitions of 'project' include:

- A specific plan or design
- A planned undertaking
- A large undertaking e.g. a public works scheme"

Key points above are planning and size of task

Here are some definitions of 'project'. No doubt there are other ones: for example,

'Unique process, consisting of a set of coordinated and controlled activities with start and finish dates, undertaken to achieve an objective conforming to specific requirements, including constraints of time, cost and resources. There is a hazy boundary between the non-routine project and the routine job. The first time you do a routine task, it will be like a project. On the other hand, a project to develop a system similar to previous ones you have developed will have a large element of the routine.

1.3.1 Characteristics of Project:

- Non-routine tasks are involved
- Planning is required
- Specific objectives are to be met or a specified product is to be created
- The project has a pre-determined time span
- Work is carried out for someone other than yourself

Fig:1.1 Activities most likely to benefit from project management.

- Work involves several specialism's
- Work is carried out in several phases
- The resources that are available for use on the project are constrained
- The project is large or complex.

The project that employs 20 developers is likely to be disproportionately more difficult than one with only 20 staff because of the need for additional coordination.

1.3.2 Software Projects versus Other Types of Project:

Many of the techniques of general project management are applicable to software project management. One way of perceiving software project management is as the process of making visible that which is invisible.

Invisibility: When a physical artifact such as a bridge or road is being constructed the progress being made can actually be seen. With software, progress is not immediately visible.
Complexity: Software products contain more complexity than other engineered artifacts.

Conformity: The 'traditional' engineer is usually working with physical systems and physical materials like cement and steel. These physical systems can have some complexity, but are governed by physical laws that are consistent. Software developers have to conform to the requirements of human clients. It is not just that individuals can be inconsistent.

Flexibility: The ease with which software can be changed is usually seen as one of its strengths. However, this means that where the software system interfaces with a physical or organizational system, it is expected that, where necessary, the software will change to accommodate the other components rather than vice versa. This means the software systems are likely to be subject to a high degree of change.

An example for infrastructure project is construction of a flyover. An example for a software project is development of a payroll management system for an organization using Oracle l0g and Oracle Forms 10G.

1.4 CONTRACT MANAGEMENT

- **In-house projects** are where the users and the developers of new software work for the same organization.
- However, increasingly organizations **contract** out ICT development to **outside developers**. Here, the client organization will often appoint a 'project manager' to supervise the contract who will delegate many technically oriented decisions to the contractors.
- Thus, the project manager will not worry about estimating the effort needed to write individual software components as long as the overall project is within budget and on time. On the supplier side, there will need to be project managers who deal with the more technical issues.

- **Contract management** is the process of managing the creation, execution, and analysis of contracts to maximize operational and financial performance and minimize risk.
- It involves various activities from the initial request for a contract, through negotiation, execution, compliance, and renewal. Effective contract management ensures that all parties to a contract fulfill their obligations as efficiently as possible.

<u>Various Stages of Contract Management</u>

1. Request and Creation:

Request: Identifying the need for a contract and gathering the necessary information to draft it.

Creation: Drafting the contract terms and conditions that align with the requirements and objectives of all parties involved.

Example: A software company needs to hire a third-party developer to work on a new project.

The project manager identifies the need for a contract and gathers details about the scope of work, timelines, payment terms, and other specifics.

2. **Negotiation**:

   Parties involved discuss and negotiate the terms of the contract to reach a mutual agreement. This stage often involves revisions and adjustments.

   Example: The software company and the third-party developer negotiate the terms. The developer might request more time or a higher payment, while the company might request milestones for progress checks. Approval and Execution:
   Approval: Obtaining necessary approvals from stakeholders and legal departments.
   Execution: Signing the contract, making it a legally binding document.

   Example: Once the terms are finalized, the contract is reviewed by both parties' legal teams. After approval, both the software company and the developer sign the contract.

3. **Obligations and Performance:**

   Ensuring that all parties adhere to the terms and conditions agreed upon in the contract. Monitoring performance and compliance.

   Example: The developer starts working on the project, adhering to the deadlines and deliverables specified in the contract. The software company provides the necessary resources and makes payments as per the contract.

4. **Modification and Renewal:**

   Making necessary amendments if any changes occur during the contract period. Reviewing and renewing contracts as needed.

   Example: Midway through the project, the software company requests additional features not covered in the original contract. An amendment is made to include these new features and adjust the payment terms accordingly. As the project nears completion, the company and developer may negotiate a renewal for ongoing maintenance.

5. **Closure:**

Completing all contractual obligations, ensuring all parties have met their requirements, and formally closing the contract.

Example: The developer finishes the project, and the software company conducts a final review to ensure all deliverables meet the agreed-upon standards. Once confirmed, the contract is closed, and a final payment is made.

Benefits of Effective Contract Management

Risk Mitigation: Identifies and manages potential risks early in the contract lifecycle.

Improved Compliance: Ensures that all parties comply with legal and regulatory requirements.

Cost Savings: Avoids unnecessary costs and penalties by managing contracts efficiently.

Performance Tracking: Monitors performance against contract terms to ensure objectives are met.

Relationship Management: Maintains positive relationships between contracting parties through clear and consistent communication.

Speed to Market: Accelerates project timelines by leveraging the vendor's expertise and resources.

COMPANY: XYZ Tech

1. **Identifying Needs:** XYZ Tech identifies a need for a mobile app to complement its existing software suite.

2. **Selecting a Vendor**: XYZ Tech shortlists several development firms based in India, known for their expertise in mobile app development.

3. **Defining Requirements**: XYZ Tech provides detailed specifications, including desired features, user interface design, and performance metrics.

4. **Contract Negotiation**: XYZ Tech negotiates terms, focusing on deliverables, timelines, and confidentiality clauses.

5. **Project Management**: XYZ Tech assigns a project manager to liaise with the vendor, ensuring regular updates and adherence to milestones.

6. **Delivery and Integration**: The vendor delivers the app, which is integrated with XYZ Tech's software suite after thorough testing.

7. **Post-Delivery Support:** The vendor provides ongoing maintenance and support, addressing any post-launch issues. By following these steps and learning from real-world examples, software companies can effectively outsource projects to thirdparty vendors, ensuring successful project completion and maximizing business value.

1.5 ACTIVITIES COVERED BY SOFTWARE PROJECT MANAGEMENT:

The activities covered by Software Project management are diagrammatically illustrated as follows:

1.5.1 The Feasibility Study:

This is an investigation into whether a prospective project is worth starting that it has a valid *business case*. Information is gathered about the requirements of the proposed application. The probable developmental and operational costs, along with the value of the benefits of the new system, are estimated. The study could be part of a strategic planning exercise examining and prioritizing a range of potential software developments.

1.5.2 Planning:

If the feasibility study produces results which indicate that the prospective project appears viable, planning of the project can take place. However, for a large project, we would not do all our detailed planning right at the beginning. We would formulate an outline plan for the whole project and a detailed one for the first stage. More detailed planning of the later stages would be done as they approached. This is because we would have more detailed and accurate information upon which to base our plans nearer to the start of the later stages.

1.5.3 Project Execution:

The project can now be executed. The execution of a project often contains *design* and

implementation subphases. The same is illustrated in **Figure 1.2** which shows the typical sequence of software development activities recommended in the international standard ISO 12207.

1.5.3.1 Requirements Analysis:

This starts with requirement elicitation or requirement gathering which establishes what the users require of the system that the project is to implement. Some work along these lines will almost certainly have been carried out when the project was evaluated, but now the original information obtained needs to be updated and supplemented.

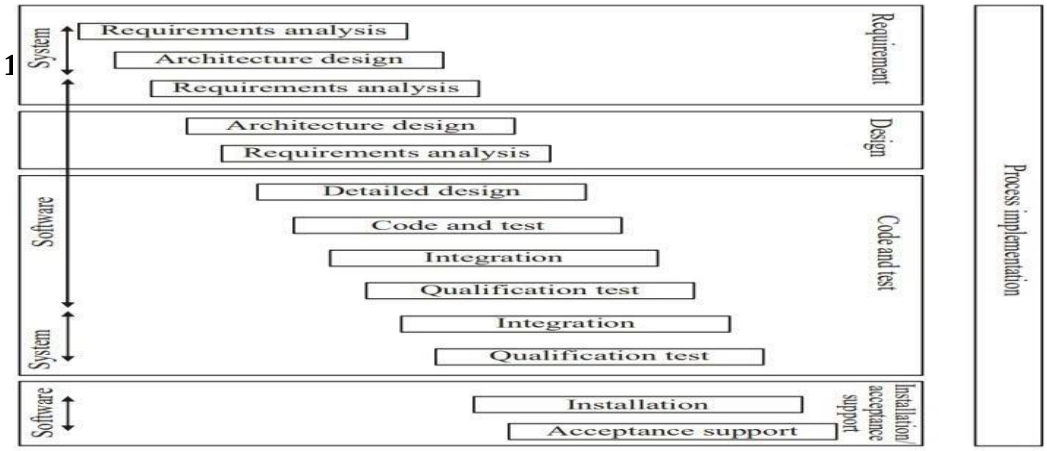


Figure 1.2: Sub Phases in Project Execution

1.5.3.3 Design:

A design has to be drawn up which meets the specification. This design will be in two stages. One will be the external or user design concerned with the external appearance of the application. The other produces the physical design which tackles the way that the data and software procedures are to be structured internally.

- **Architecture Design:** This maps the requirements to the components of the system that is to be built. At the system level, decisions will need to be made about which processes in the new system will be carried out by the user and which can be computerized. This design of the system architecture thus forms an input to the development of the software requirements. A second architecture design process then takes place which maps the software requirements to software components.

1.5.3.4 Detailed Design:

Each software component is made up of a number of software units that can be separately coded and tested. The detailed design of these units is carried out separately. Coding:This may refer to writing code in a procedural language or an object-oriented language or could refer to the use of an application-builder. Even where software is not being built from scratch,

some modification to the base package could be required to meet the needs of the new application.

1.5.3.5 Testing(Verification and Validation):

Whether software is developed specially for the current application or not, careful testing will be needed to check that the proposed system meets its requirements.

Integration: The individual components are collected together and tested to see if they meet the overall requirements. Integration could be at the level of software where different software components are combined, or at the level of the system as a whole where the software and other components of the system such as the hardware platforms and networks and the user procedures are brought together.

Qualification Testing: The system, including the software components, has to be tested carefully to ensure that all the requirements have been fulfilled.

1.5.3.6 Implementation/ Installation:

Some system development practitioners refer to the whole of the project after design as 'implementation' (that is, the implementation of the design) while others insist that the term refers to the installation of the system after the software has been developed.

1.5.3.7 Acceptance Support:

Once the system has been implemented there is a continuing need for the correction of any errors that may have crept into the system and for extensions and improvements to the system. Maintenance and support activities may be seen as a series of minor software projects.

1.6 PLANS, METHODS AND METHODOLOGIES

A plan for an activity must be based on some idea of a *method of work.* To take a simple example, if you were asked to test some software, even though you do not know anything about the software to be tested, you could assume that you would need to:

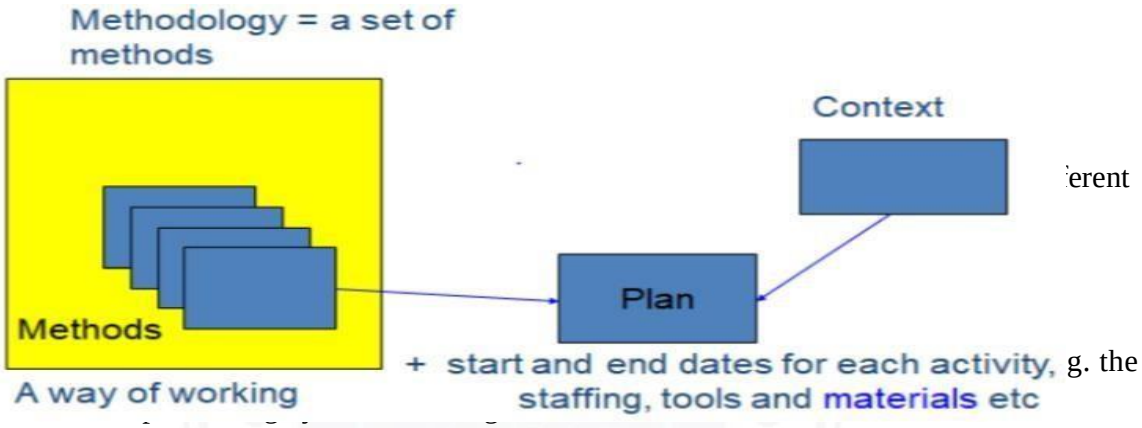
- Analyze the requirements for the software
- Devise and write test cases that will check that each requirement has been satisfied
- Create test scripts and expected results for each test case
- Compare the actual results and the expected results and identify discrepancies

While a method relates to a type of activity in general, a plan takes that method (and perhaps others) and converts it to real activities, identifying for each activity:

- Its start and end dates
- Who will carry it out?
- What tools and materials will be used?

'Materials' in this context could include information, for example a requirements document. With complex procedures, several methods may be deployed, in sequence or in parallel. The output from one method might be the input to another. Groups of methods or techniques are often referred to as methodologies.

1.7



D erent p g. the

- Information systems versus embedded systems.
- Software Products verses services
- Product-development versus outsourced.
- Object-driven development.

1.7.1 Changes to the characteristics of software projects

- Over the last few decades, the characteristics of software projects have undergone drastic changes. In earlier days of software development every software was being written from scratch and there was no code reusability. In contrast, at present almost every programming language supports ways of reusing existing code, by customizing and extending existing code, efficiently and dynamically linking library routines and support for frameworks.
- Project durations have now shrunk to only a few months compared to multi-year projects.
- In past, customer participation in software projects was largely restricted to only initial interactions, gathering and specification and taking delivery of the developed software, but at present, customer participation in almost every aspect of a project.

1.7.2 Compulsory versus voluntary users

In workplaces there are systems that staff have to use if they want to do something, such as recording a sale. However, use of a system is increasingly voluntary, as in the case of computer games. Here it is difficult to elicit precise requirements from potential users as we could with a business system. What the game will do will thus depend much on the informed ingenuity of the developers, along with techniques such as market surveys, focus groups and prototype evaluation.

1.7.3 Information Systems versus Embedded systems

A traditional distinction has been between information systems which enable staff to carry out office processes and *embedded systems* which control machines. A stock control system would be an information system. An embedded, or process control, system might control the air conditioning equipment in a building. Some systems may have elements of both where, for example, the stock control system also controls an automated warehouse.

1.7.4 Software Products verses services

All types of software projects can broadly be classified into software product development projects and software services projects. It can be further classified as shown in below Fig.1.7 A software product development concerns developing the software by keeping the requirements to the general customers in mind and developed software is usually sold-off-the helf to a large number of customers. Examples of generic software development are Microsoft's Windows operating system and Oracle Corporatism's Oracle 8i database management software. Domain-specific software targets specific segments of customers(verticals) Example BANCS from TCS. FINACLE from Infosys.

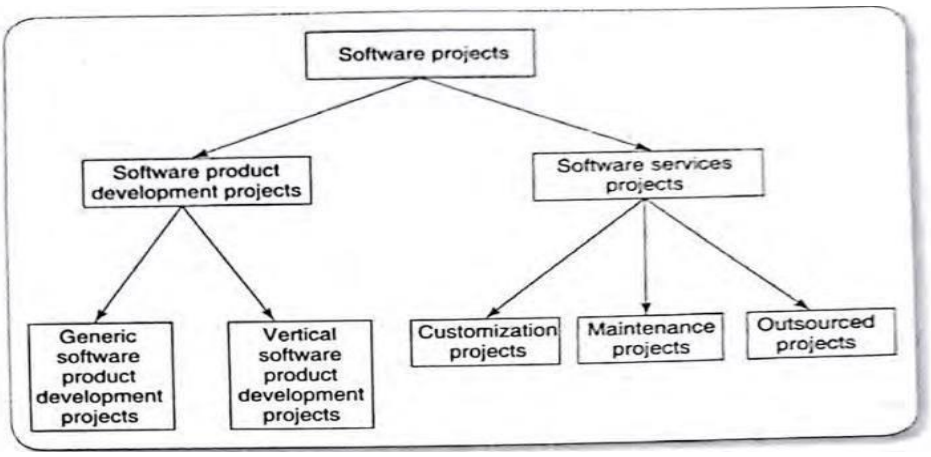


Fig: 1.7 A Classification of software projects

Software services cover a large gamut of software projects such as customization, outsourcing, maintenance, testing and consultancy.

Projects may be distinguished by whether their aim is to produce a product or to meet certain objectives.

1.7.5 Outsourced Projects

While developing a large project, it makes good commercial sense for a company to outsource some parts of its work to other companies.

For example, A company may consider outsourcing as a good option, it if feels that it does not have sufficient expertise to develop some specific parts of the product or if it determines that some parts can be developed cost-effectively another company.

1.7.6 Object-driven development

Projects may be distinguished by whether their aim is to produce or to meet certain objective. any software projects have two stages, First is an object-driven project resulting in recommendations which identify the need for a new software system and next stage is a project actually to create the software product.

1.8 STAKEHOLDERS

These are people who have a stake or interest in the project. It is important that they be identified as early as possible, because you need to set up adequate communication channels with them right from the start. The project leader also has to be aware that not everybody who is involved with a project has the same motivation and objectives. The end-users might, for instance, be concerned about the ease of use of the system while their managers might be interested in the staff savings the new system will allow.

Boehm and Ross proposed a *'Theory W'* of software project management where the manager concentrates on creating the role and format situations where all parties benefit from a project and therefore have an of communication interest in its success. (The 'W' stands for 'win-win'.)

Stakeholders might be internal to the project team, external to the project team but in the same organization, or totally external to the organization.

- ***Internal to the project team:*** This means that they will be under the direct managerial control of the project leader.
- ***External to the project team but within the same organization:*** For example, the project leader might need the assistance of the information management group in order to add some additional data types to a database or the assistance of the users to carry out systems testing. Here the commitment of the people involved has to be negotiated.
- ***External to both the project team and the organization:*** External stakeholders may be customers (or users) who will benefit from the system that the project implements or contractors who will carry out work for the project. One feature of the relationship with these people is that it is likely to be based on a legally binding contract.

Different types of Stakeholders may have different objectives and one of the jobs of the successful project leader is to recognize these different interests and to be able to reconcile them. It should therefore come as no surprise that the project leader needs to be a good communicator and negotiator.

1.9 SETTING OBJECTIVES

- The objectives should define what the project team must achieve for project success.
- Objectives focus on the desired outcomes of the project rather than the tasks within it-they are the 'post-conditions' of the project.
- Objectives could be set of statements following the opening words ***'the project will be a success if ….***' .
- To have a successful software project, the manager and the project team members must know what will constitute success. This will make them concentrate on what is essential to project success.
- There may be several sets of users of a system and there may be several different groups of specialists involved its development. There is a need for well-defined objectives that are accepted by all these people. Where there is more than one user group, a project authority needs to be identified which has overall authority over what the project is to achieve.
- This authority is often held by a *project steering committee* (or project board or project management board) which has overall responsibility for setting, monitoring and modifying objectives. The project manager still has responsibility for running the project on a day-to-day basis, but has to report to the steering committee at regular intervals. Only the steering committee can authorize changes to the project objectives and resources.

1.9.1 Sub-objectives and Goals:

Setting objectives can guide and motivate individuals and groups of staff. An effective objective for an individual must be something that is within the control of that individual. An objective might be that the software application to be produced must pay for itself by reducing staff costs over two years. As an overall business objective this might be reasonable. For software developers it would be unreasonable as, though they can control development costs, any reduction in operational staff costs depends not just on them but on the operational management after the application has 'gone live'. What would be appropriate would be to set a goal or sub-objective for the software developers to keep development costs within a certain budget.

Thus, objectives will need be broken down into goals or sub-objectives. Here we say that in order to achieve the objective we must achieve certain goals first. These goals are steps on the way to achieving an objective, just as goals scored in a football match are steps towards the objective of winning the match.

The mnemonic SMART is sometimes used to describe well defined objectives:

- **Specific**: Effective objectives are *concrete and well defined*. Vague aspirations such as 'to improve customer relations' are unsatisfactory. Objectives should be defined in such a way that it is obvious to all whether the project has been successful or not.

- **Measurable**: Ideally there should be measures of effectiveness which tell us how successful the project has been. For example, 'to reduce customer complaints' would be more satisfactory as an objective than 'to improve customer relations'. The measure can, in some cases, be an answer to simple yes/no questions, e.g. 'Can we install the new software by 1 November 2011?'

- **Achievable**: It must be within the power of the individual or group to achieve the objective.

- **Relevant**: The objective must be relevant to the true purpose of the project.

- **Time constrained:** There should be a defined point in time by which the objective should have been achieved.

1.9.2 Measures of effectiveness

Measures of effectiveness provide practical methods of ascertaining whether an objective has been met. 'Mean time between failures' (mtbf) is used to measure reliability. A measure

of effectiveness will usually be related to the installed operational system.

1.10 BUSINESS CASE

- Most projects need to have a justification or business case: the effort and expense of pushing the project through must be seen to be worthwhile in terms of the benefits that will eventually be felt.
- The quantification of benefits will often require the formulation of a business model which explains how the new application can generate the claimed benefits.

Any project plan must ensure that the business case is kept intact. For example:

- The development costs are not allowed to rise to a level which threatens to exceed the value of benefits.
- The features of the system are not reduced to a level where the expected benefits cannot be realized.
- The delivery date is not delayed so that there is an unacceptable loss benefit.

1.11 PROJECT SUCCESS AND FAILURE

- The project plan should be designed to ensure project success preserving the business case for the project.
- Different stakeholders have different interests, some stakeholders in a project might see it as a success while others do not.
- The project objectives are the targets that the project team is expected to achieve—They are summarized as delivering:
    - The agreed functionality
    - To the required level of quality
    - In time
    - Within budget
- A project could meet these targets but the application, once delivered could fail to meet the business case. A computer game could be delivered on time and within budget, but might then not sell.
- In business terms, the project is a success if the value of benefits exceeds the costs.

- A project can be a success on delivery but then be a business failure, On the other hand, a project could be late and over budget, but its deliverables could still, over time, generate benefits that outweigh the initial expenditure.
- The possible gap between project and business concerns can be reduced by having a broader view of projects that includes business issues.

- **Technical learning** will increase costs on the earlier projects, but later projects benefit as the learnt technologies can be deployed more quickly cheaply and accurately.

- **Customer relationships** can also be built up over a number of projects. If a client has trust in a supplier who has done satisfactory work in the past, they are more likely to use that company again.

1.12 MANAGEMENT AND MANAGEMENT CONTROL

1.12.1 MANAGEMENT:

Management involves following activities:

- Planning - deciding what is to be done;
- Organizing - making arrangements;
- Staffing - selecting the right people for the job etc.;
- Directing - giving instructions;
- Monitoring - checking on progress;
- Controlling - taking action to remedy hold-ups;
- Innovating - coming up with new solutions;
- Representing - liaising with clients, users, developer, suppliers and other stakeholders.

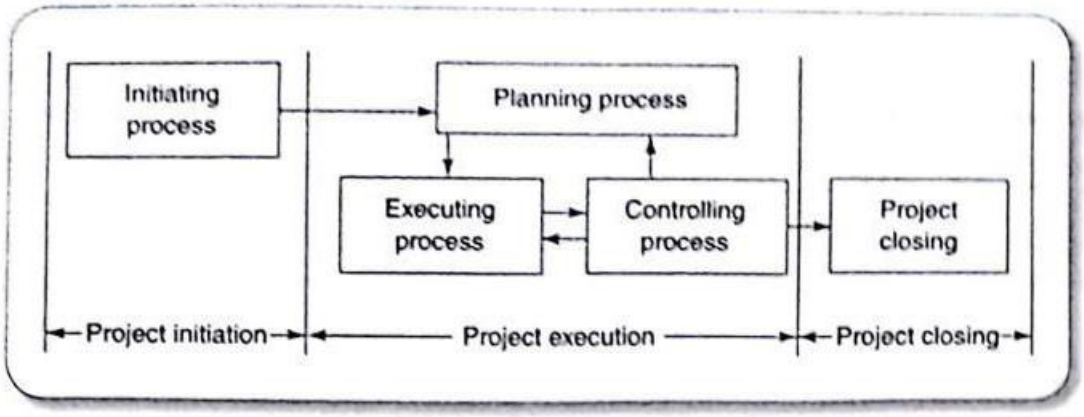


FIGURE 1.5 Principal project management processes

Much of the project manager's time is spent only in three activities , i.e. Project Planning , Monitoring and control. This time period during which these activities are carried out is indicated in Fig 1.5.

It shows that project management is carried out over three well-defined stages or processes irrespective of the methodology used.

In the **Project initiation stage**, an initial plan is made. As a project starts, **the project is monitored and controlled to process as planned**. Initial plan is revised periodically to accommodate additional details and constraints about the project as they become available. Finally, the **project is closed**.

Initial project is undertaken immediately after the feasibility study phase and before starting the requirement analysis and specification process.

Initial project planning involves estimating several characteristics of a project. Based on these estimates all subsequent project activities are planned.

The **monitoring activity** involves monitoring the progress of the project. **Control activities** are initiated to minimize any significant variation in the plan,

Project Planning is an important responsibility of the project Manager. During project planning, the project manger needs to perform a few well-defined activities that have been outlined below/ Several best practices have been proposed for software project planning activities, PRINCE2 is used extensively in UK and Europe . In USA Project management Institute's 'PMBOK' which refers to their publication "A Gude to the Project Management Body of knowledge, is used.

- **Estimation**: The following project attributes are estimated.
- **Cost**: How much is it going to cost to complete the project.
- **Duration**: How long is it going to take to complete the project.
- **Effort**: How much effort would be necessary for completing the project?

The effectiveness of all activities such as scheduling and staffing are planned at later stage.

- **Scheduling**: Based on estimations of effort and duration, the schedules for manpower and other resources are developed.
- **Staffing**: Staff organization and staffing plans are made.
- **Risk Management**: This activity includes risk identification, analysis, and abatement planning.
- **Miscellaneous Plans**: This includes making several other plans such as quality assurance plan, configuration management plan etc.

While carrying out project monitoring and control activities, a project manager may sometimes find it necessary to change the plan to cope with specific situations and make the plan more

accurate as more project data becomes available.

1.12.2 MANAGEMENT CONTROL

Management involves setting objectives for a system and monitoring the performance of the system.

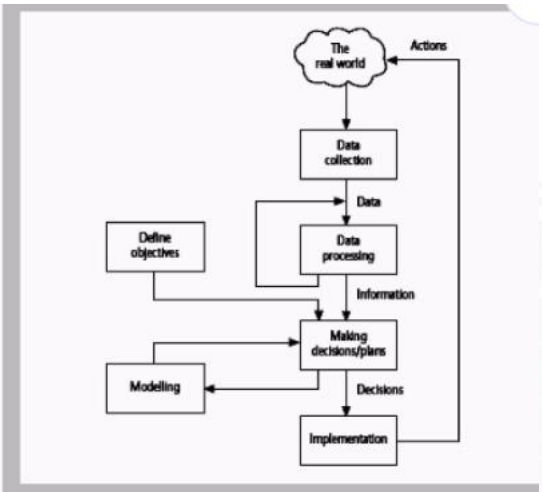


Figure 1.4 The project control cycle.

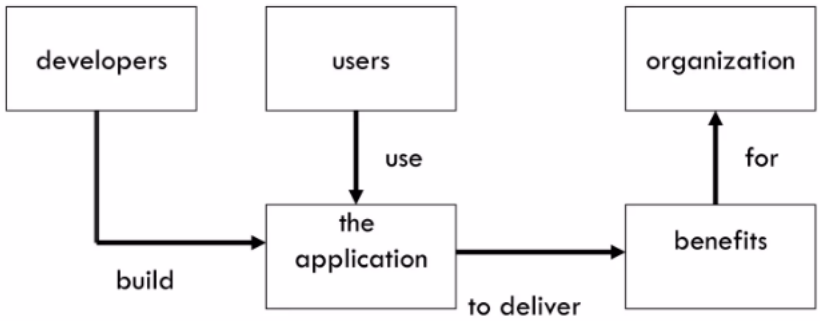
ig: The Project control cycle

- In the above Fig, local mangers involve in data collection. Bare details such as "location X has processed 2000 documents" may not be useful to higher management.
- Data processing is required to transform this raw data into useful information. This might be in such forms as "Percentage of records Processed", average documents per day per person", and estimated completion date".
- In this example , the project management might examine the "estimated completion date" for completing data transfer for each branch. They are comparing actual performance with overall project objectives.
- They might find that one or two branches will fail to complete the transfer of details in time.
- It can be seen that a project plan is dynamic and will need constant adjustment during the execution of the project. A good plan provides a foundation for a good project, but is nothing without intelligent execution.

1.13 PROJECT MANAGEMENT LIFE CYCLE

Software development life cycle denotes (SDLC) the stages through which a software is developed. In contrast to SDLC, the project management life cycle typically starts well before the software development activities start and continues for the entire duration of SDLC. (Fig 1.7)

In Project Management process, the project manager carries out project initiation, planning, execution, monitoring, controlling and closing.The different phases of the project management life cycle are shown in Fig: 1.8.



1. **Project Initiation**: The project initiation phase starts with project **concept development**. During concept development the different characteristics of the software to be developed are thoroughly understood, which includes, the scope of the project, the project constraints, the cost that would be incurred and the benefits that would accrue. Based on this understanding, a **feasibility study** is undertaken to determine the project would be financially and technically feasible.

   Based on feasibility study, the **business case** is developed. Once the top management agrees to the business case, the **project manager** is appointed, the **project charter** is written and finally **project team** is formed. This sets the ground for the manager to start the **project planning phase.**

2. **W5HH Principle:** Barry Boehm, summarized the questions that need to be asked and answered in order to have an understanding of these project characteristics.

- Why is the software being built?
- What will be done?
- When will it be done?
- Who is responsible for a function?
- Where are they organizationally located?
- How will the job be done technically and managerially?
- How much of these each resource is needed.

3. **Project Bidding:** Once the top management is convinced by the business case, the **project charter** is developed. For some categories of projects, it may be necessary to have formal bidding process to select suitable vendor based on some cost-performance criteria. The different types of bidding techniques are:

- **Request for quotation(RFQ) :** An organization advertises an RFQ if it has good understanding of the project and the possible solutions.
- **Request for Proposal(RFP) :** An organization had reasonable understanding of the problem to be solved, however, it does not have good grasp of the solution aspects. i.e. may not have sufficient knowledge about different features to be implemented. The purpose of RFP is to get an understanding of the alternative solutions possible that can be deployed and not vendor selection. Based on the RFP process, the requesting organization can form a clear idea of the project solutions required, based on which it can form a statement work (SOW) for requesting RFQ for the vendors.
- **Request for Information (RFI):** An organization soliciting bids may publish an RFI. Based on the vendor response to the RFI, the organization can assess the competencies of the vendors and shortlist the vendors who can bid for the work.

4. Project Planning: An importance of the project initiation phase is the project charter. During the project planning the project manger carries out several processes and creates the following documents:

- **Project plan:** This document identifies the project the project tasks and a schedule for the project tasks that assigns project resources and time frames to the tasks.
- **Resource Plan:** It lists the resources , manpower and equipment that would be required to execute the project.
- **Functional Plan:** It documents the plan for manpower, equipment and other costs.
- **Quality Plan:** Plan of quality targets and control plans are included in this document.
- **Risk Plan:** This document lists the identification of the potential risks, their prioritization and a plan for the actions that would be taken to contain the different risks.

5. Project Execution: In this phase the tasks are executed as per the project plan developed during the planning phase. Quality of the deliverables is ensured through execution of proper processes. Once all the deliverables are produced and accepted by the customer, the project execution phase completes and the project closure phase starts.

6. Project Closure: Project closure involves completing the release of all the required deliverables to the customer along with the necessary documentation. All the Project resources are released and supply agreements with the vendors are terminated and all the

pending payments are completed. Finally, a postimplementation review is undertaken to analyze the project performance and to list the lessons for use in future projects.

1.14 TRADITIONAL VERSUS MODERN MANAGEMENT PRACTICES

Over the last two decades, the basic approach taken by the software industry to develop software has undergone a radical change.

Software is not developed from scratch any more, Software development projects are based on either tailoring some existing product or reusing certain pre-built libraries both will maximize code reuse and compression of project durations.

Other goals include facilitating and accommodating client feedback and client feedbacks and customer participation in project development work and incremental delivery of the product with evolving functionality.

Some Important difference between modern management practices and traditional practices are:

- **Planning Incremental Delivery:** Earlier, projects were simpler and therefore more predictable than the present-day projects. In those days, projects were planned with sufficient detail much before the actual project execution started. After the project initiation, monitoring and control activities were carried out to ensure that the project execution proceeded as per plan, Now, the projects are required to be completed over a much shorter duration, and rapid application development and deployment are considered key strategies.
  Instead of making a long-term project completion plan, the project manger now plans all incremental deliveries with evolving functionalities. This type of project management is often called **extreme project management.** Extreme project management is highly flexible approach that concentrates on human side of project management(e.g. managing project stakeholders).

- **Quality Management:** Customer awareness about product quality has increased significantly. The key responsibility of a project manager now includes assessment of project progress and tracking the quality of all intermediate artifacts.

- **Change Management:** Earlier, when the requirements were signed off by the customer, any changes to the requirements were rarely entertained. Customer suggestions are now actively solicited and incorporated throughout the development process. To facilitate customer feedback, **incremental delivery models** are popularly being used. Product

development i being carried out through a series of product versions implementing increasingly greater fnctionalities. The Project manager plays a key role in product base lining and version control. This has made change management a crucial responsibility of the project manager. Change Management is also known as configuration management.

- **Requirement Management:** In older development methodologies , the requirements had to be identified upfront and these were 'signed off' by the customer and frozen before the development could start. At present , in most projects, the requirements change frequently during the development cycle. **Requirement management** has therefore become a systematic process of controlling changes, documenting , analyzing, tracing, prioritizing requirements and then communicating the changes to the relevant stakeholders.

- **Release Management:** Release management concerns planning, prioritizing and controlling the different releases of a software. Modern development processes such as Agile development processes advocate frequent and regular releases of the software to be made to the customer during the software development. Starting with the release of basic or core functionalities of the software, more complete functionalities are made available to the customer every couple of weeks . Hence effective Release Management has become important.

- **Risk Management:** In modern software development practices. Effective risk management is considered very important to the success of a project. A risk is any negative situation that may arise as the project progresses and may threaten the success of the

  project. Risk Management involves identification of risks, assessment of the impacts of various risks, prioritization of the risks and preparation of risk-containment plans.

- **Scope Management:** Modern software development encourages customer to come up with change requests. While accepting the requests, three critical project parameters: scope , schedule and project cost are interdependent and related.