Software project management

• Concerned with activities involved in ensuring that software is delivered on time and on schedule and in accordance with the requirements of the organizations developing and procuring the software.

• Project management is needed because software development is always subject to budget and schedule constraints that are set by the organization developing the software.

Success criteria

• Deliver the software to the customer at the agreed time.

• Keep overall costs within budget.

• Deliver software that meets the customer's expectations.

• Maintain a coherent and well-functioning development team.

Software management distinctions

• **The product is intangible.**

• Software cannot be seen or touched. Software project managers cannot see progress by simply looking at the artefact that is being constructed.

• **Many software projects are 'one-off' projects**.

• Large software projects are usually different in some ways from previous projects. Even managers who have lots of previous experience may find it difficult to anticipate problems.

• **Software processes are variable and organization specific.**

• We still cannot reliably predict when a particular software process is likely to lead to development problems.

Factors influencing project management

• Company size

• Software customers

• Software size

• Software type

• Organizational culture

• Software development processes

• These factors mean that project managers in different organizations may work in quite different ways.

Universal management activities

• **Project planning**

• Project managers are responsible for planning. estimating and scheduling project development and assigning people to tasks.

• **Risk management**

• Project managers assess the risks that may affect a project, monitor these risks and take action when problems arise.

• **People management**

• Project managers have to choose people for their team and establish ways of working that leads to effective team performance.

Management activities

- **Reporting**
- Project managers are usually responsible for reporting on the progress of a project to customers and to the managers of the company developing the software.
- **Proposal writing**
- The first stage in a software project may involve writing a proposal to win a contract to carry out an item of work. The proposal describes the objectives of the project and how it will be carried out.

Risk management

- Risk management is concerned with identifying risks and drawing up plans to minimise their effect on a project.
- Software risk management is important because of the inherent uncertainties in software development.
    - These uncertainties stem from loosely defined requirements, requirements changes due to changes in customer needs, difficulties in estimating the time and resources required for software development, and differences in individual skills.
- You have to anticipate risks, understand the impact of these risks on the project, the product and the business, and take steps to avoid these risks.

Risk classification

- There are two dimensions of risk classification

- The type of risk (technical, organizational, ..)
- what is affected by the risk:

• Project risks affect schedule or resources;

• Product risks affect the quality or performance of the software being developed;

• Business risks affect the organisation developing or procuring the software.

Examples of project, product, and business risks

Risk	Affects	Description
Staff turnover	Project	Experienced staff will leave the project before it is finished.
Management change	Project	There will be a change of organizational management with different priorities.
Hardware unavailability	Project	Hardware that is essential for the project will not be delivered on schedule.
Requirements change	Project and product	There will be a larger number of changes to the requirements than anticipated.
Specification delays	Project and product	Specifications of essential interfaces are not available on schedule.
Size underestimate	Project and product	The size of the system has been underestimated.
CASE tool underperformance	Product	CASE tools, which support the project, do not perform as anticipated.
Technology change	Business	The underlying technology on which the system is built is superseded by new technology.
Product competition	Business	A competitive product is marketed before the system is completed.

The risk management process

• **Risk identification**

Identify project, product and business risks;

• **Risk analysis**

Assess the likelihood and consequences of these risks;

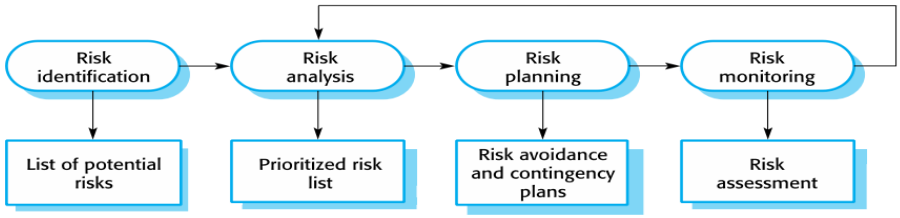
• **Risk planning**

Draw up plans to avoid or minimise the effects of the risk;

- **Risk monitoring**

Monitor the risks throughout the project;

The risk management process



Risk identification

• May be a team activities or based on the individual project manager's experience.

• A checklist of common risks may be used to identify risks in a project

- Technology risks.
- Organizational risks.
- People risks.
- Requirements risks.
- Estimation risks.

Examples of different risk types

Risk type	Possible risks
Estimation	The time required to develop the software is underestimated. The rate of defect repair is underestimated. The size of the software is underestimated.
Organizational	The organization is restructured so that different management are responsible for the project. Organizational financial problems force reductions in the project budget.
People	It is impossible to recruit staff with the skills required. Key staff are ill and unavailable at critical times. Required training for staff is not available.
Requirements	Changes to requirements that require major design rework are proposed. Customers fail to understand the impact of requirements changes.
Technology	The database used in the system cannot process as many transactions/second as expected. Reusable software components contain defects that mean they cannot be reused as planned.
Tools	The code generated by software code generation tools is inefficient. Software tools cannot work together in an integrated way.

Risk analysis

- Assess probability and seriousness of each risk.
- Probability may be very low, low, moderate, high or very high.
- Risk consequences might be catastrophic, serious, tolerable or insignificant.

Risk types and examples

Risk	Probability	Effects
Organizational financial problems force reductions in the project budget.	Low	Catastrophic
It is impossible to recruit staff with the skills required for the project.	High	Catastrophic
Key staff are ill at critical times in the project.	Moderate	Serious
Faults in reusable software components have to be repaired before these components are reused.	Moderate	Serious
Changes to requirements that require major design rework are proposed.	Moderate	Serious
The organization is restructured so that different management are responsible for the project.	High	Serious
The database used in the system cannot process as many transactions per second as expected.	Moderate	Serious
The time required to develop the software is underestimated.	High	Serious
Software tools cannot be integrated.	High	Tolerable
Customers fail to understand the impact of requirements changes.	Moderate	Tolerable
Required training for staff is not available.	Moderate	Tolerable
The rate of defect repair is underestimated.	Moderate	Tolerable
The size of the software is underestimated.	High	Tolerable
Code generated by code generation tools is inefficient.	Moderate	Insignificant

Risk planning

- Consider each risk and develop a strategy to manage that risk.
- Avoidance strategies

  The probability that the risk will arise is reduced;
- Minimization strategies

  The impact of the risk on the project or product will be reduced;
- Contingency plans

  If the risk arises, contingency plans are planned to deal with that risk;

Strategies to help manage risk

Risk	Strategy
Organizational financial problems	Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business and presenting reasons why cuts to the project budget would not be cost-effective.
Recruitment problems	Alert customer to potential difficulties and the possibility of delays; investigate buying-in components.
Staff illness	Reorganize team so that there is more overlap of work and people therefore understand each other's jobs.
Defective components	Replace potentially defective components with bought-in components of known reliability.
Requirements changes	Derive traceability information to assess requirements change impact; maximize information hiding in the design.
Organizational restructuring	Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business.
Database performance	Investigate the possibility of buying a higher-performance database.
Underestimated development time	Investigate buying-in components; investigate use of a program generator.

Risk monitoring

• Assess each identified risks regularly to decide whether or not it is becoming less or more probable.

• Also assess whether the effects of the risk have changed.

• Each key risk should be discussed at management progress meetings.

Risk indicators

Risk type	Potential indicators
Estimation	Failure to meet agreed schedule; failure to clear reported defects.
Organizational	Organizational gossip; lack of action by senior management.
People	Poor staff morale; poor relationships amongst team members; high staff turnover.
Requirements	Many requirements change requests; customer complaints.
Technology	Late delivery of hardware or support software; many reported technology problems.
Tools	Reluctance by team members to use tools; complaints about CASE tools; demands for higher-powered workstations.

Managing people

• People are an organisation's most important assets.

• The tasks of a manager are essentially people-oriented. Unless there is some understanding of people, management will be unsuccessful.

• Poor people management is an important contributor to project failure.

People management factors

• **Consistency**

Team members should all be treated in a comparable way without favourites or discrimination.

• **Respect**

Different team members have different skills and these differences should be respected.

- **Inclusion**

Involve all team members and make sure that people's views are considered.

- **Honesty**

You should always be honest about what is going well and what is going badly in a project.

Teamwork

- Most software engineering is a group activity

  The development schedule for most non-trivial software projects is such that they cannot be completed by one person working alone.

- A good group is cohesive and has a team spirit. The people involved are motivated by the success of the group as well as by their own personal goals.
- Group interaction is a key determinant of group performance.
- Flexibility in group composition is limited

  Managers must do the best they can with available people.

Group cohesiveness

- In a cohesive group, members consider the group to be more important than any individual in it.
- The advantages of a cohesive group are:
  - Group quality standards can be developed by the group members.
  - Team members learn from each other and get to know each other's work;

Inhibitions caused by ignorance are reduced.

• Knowledge is shared. Continuity can be maintained if a group member leaves.

• Refactoring and continual improvement is encouraged. Group members work collectively to deliver high quality results and fix problems, irrespective of the individuals who originally created the design or program.

The effectiveness of a team

• The people in the group

You need a mix of people in a project group as software development involves diverse activities such as negotiating with clients, programming, testing and documentation.

• The group organization

A group should be organized so that individuals can contribute to the best of their abilities and tasks can be completed as expected.

• Technical and managerial communications

Good communications between group members, and between the software engineering team and other project stakeholders, is essential.

Selecting group members

• A manager or team leader's job is to create a cohesive group and organize their group so that they can work together effectively.

• This involves creating a group with the right balance of technical

skills and personalities, and organizing that group so that the members work together effectively.

Assembling a team

• May not be possible to appoint the ideal people to work on a project

Project budget may not allow for the use of highly-paid staff;

Staff with the appropriate experience may not be available;

An organisation may wish to develop employee skills on a software project.

• Managers have to work within these constraints especially when there are shortages of trained staff.

Group composition

• Group composed of members who share the same motivation can be problematic

• Task-oriented - everyone wants to do their own thing;

• Self-oriented - everyone wants to be the boss;

• Interaction-oriented - too much chatting, not enough work.

• An effective group has a balance of all types.

• This can be difficult to achieve software engineers are often task-oriented.

• Interaction-oriented people are very important as they can detect and defuse tensions that arise.

Group organization

• The way that a group is organized affects the decisions that are made by that group, the ways that information is exchanged and

the interactions between the development group and external project stakeholders.

Key questions include:

- Should the project manager be the technical leader of the group?
- Who will be involved in making critical technical decisions, and how will these be made?
- How will interactions with external stakeholders and senior company management be handled?
- How can groups integrate people who are not co-located?
- How can knowledge be shared across the group?
- Small software engineering groups are usually organised informally without a rigid structure.
- For large projects, there may be a hierarchical structure where different groups are responsible for different sub-projects.
- Agile development is always based around an informal group on the principle that formal structure inhibits information exchange

Informal groups

- The group acts as a whole and comes to a consensus on decisions affecting the system.
- The group leader serves as the external interface of the group but does not allocate specific work items.
- Rather, work is discussed by the group as a whole and tasks are allocated according to ability and experience.
- This approach is successful for groups where all members are experienced and competent.

Group communications

• Good communications are essential for effective group working.

• Information must be exchanged on the status of work, design

decisions and changes to previous decisions.

• Good communication also strengthens group cohesion as it

promotes understanding.

Group size

• The larger the group, the harder it is for people to communicate with other group members.

Group structure

• Communication is better in informally structured groups than in hierarchically structured

groups.

Group composition

• Communication is better when there are different personality types in a group and when

groups are mixed rather than single-gender.

The physical work environment

• Good workplace organization can help encourage communications.