Project management

Objectives

- To explain the main tasks undertaken by project managers
- To introduce software project management and to describe its distinctive characteristics
- To discuss project planning and the planning process
- To show how graphical schedule representations are used by project management
- To discuss the notion of risks and the risk management process

Tracking project progress

- Do you understand customer problem and needs?
- Can you design a system to solve customer problem or satisfy customer needs?
- How long will it take you to develop the system?
- How much will it cost to develop the system?

Topics covered

- Management activities
- Project planning
- Project scheduling
- Risk management

- In many ways these are the activities that make software engineering different from programming

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.

Management commonalities

- These activities are not peculiar to software management.
- Many techniques of engineering project management are equally applicable to software project management.
- Technically complex engineering systems tend to suffer from the same problems as software systems.

Management activities

- Proposal writing.
- Project planning and scheduling.
- Project costing.
- Project monitoring and reviews.
- Personnel selection and evaluation.
- Report writing and presentations.

Project deliverables

- Documents
- Demonstrations of function
- Demonstrations of subsystems
- Demonstrations of accuracy
- Demonstrations of reliability, performance or security

Milestones and activities

- Activity: takes place over a period of time
- Milestone: completion of an activity -- a particular point in time
- Precursor: event or set of events that must occur in order for an activity to start
- Duration: length of time needed to complete an activity
- Due date: date by which an activity must be completed

Managing groups

- 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.
- 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.

Project staffing

- 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 organization 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.

Factors influencing groups

- Group composition.
- Group cohesiveness.
- Group communications.
- Group organization.

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 leadership

- Leadership depends on respect not titular status.
- There may be both a technical and an administrative leader.
- Democratic leadership is more effective that autocratic leadership.

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;
  - Group members work closely together so inhibitions caused by ignorance are reduced;
  - Team members learn from each other and get to know each other's work;
  - Egoless programming where members strive to improve each other's programs can be practised.

Developing cohesiveness

- Cohesiveness is influenced by factors such as the organizational culture and the personalities in the group.
- Cohesiveness can be encouraged through
  - Social events;
  - Developing a group identity and territory;
  - Explicit team-building activities.
- Openness with information is a simple way of ensuring all group members feel part of the group.

Group loyalties

- Group members tend to be loyal to cohesive groups.
- 'Groupthink' is preservation of group irrespective of technical or organizational considerations.
- Management should act positively to avoid groupthink by forcing external involvement with each group.

Group communications

- 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 sex.
- The physical work environment
  - Good workplace organization can help encourage communications.

Group organization

- Small software engineering groups are usually organized informally without a rigid structure.
- For large projects, there may be a hierarchical structure where different groups are responsible for different sub-projects.

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.

Extreme programming groups

- Extreme programming groups are variants of an informal, democratic organisation.
- In extreme programming groups, some 'management' decisions are devolved to group members.
- Programmers work in pairs and take a collective responsibility for code that is developed.

➔ More later with discussion of agile methods

Chief programmer teams

- Consist of a kernel of specialists helped by others added to the project as required.
- The motivation behind their development is the wide difference in ability in different programmers.
- Chief programmer teams provide a supporting environment for very able programmers to be responsible for most of the system development.

Project planning

- Probably the most time-consuming project management activity.
- Continuous activity from initial concept through to system delivery. Plans must be regularly revised as new information becomes available.
- Various different types of plan may be developed to support the main software project plan that is concerned with schedule and budget.

Types of project plan

Plan	Description
Quality plan	Describes the quality procedures and standards that will be used in a project. See Chapter 27.
Validation plan	Describes the approach, resources and schedule used for system validation. See Chapter 22.
Configuration management plan	Describes the configuration management procedures and structures to be used. See Chapter 29.
Maintenance plan	Predicts the maintenance requirements of the system, maintenance costs and effort required. See Chapter 21.
Staff development plan.	Describes how the skills and experience of the project team members will be developed. See Chapter 25.

Project planning process

```
Establish the project constraints
Make initial assessments of the project parameters
Define project milestones and deliverables
while project has not been completed or cancelled loop
        Draw up project schedule
        Initiate activities according to schedule
        Wait ( for a while )
        Review project progress
        Revise estimates of project parameters
        Update the project schedule
        Re-negotiate project constraints and deliverables
        if ( problems arise ) then
                Initiate technical review and possible revision
        end if
end loop
```

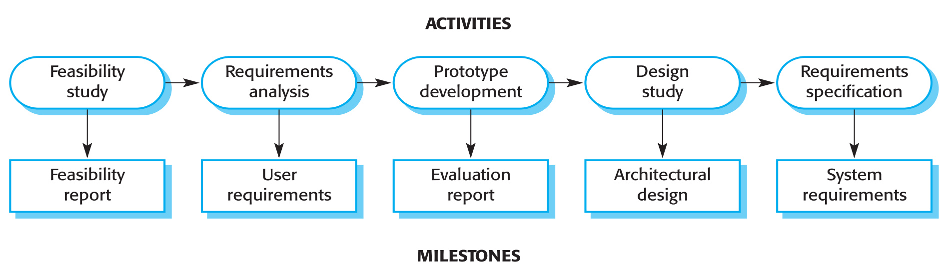

The project plan

- The project plan sets out:
  - The resources available to the project;
  - The work breakdown;
  - A schedule for the work.
- Activities
  - Introduction.
  - Project organization.
  - Risk analysis.
  - Hardware and software resource requirements.
  - Work breakdown.
  - Project schedule.
  - Monitoring and reporting mechanisms.

Activity organization

- Activities in a project should be organized to produce tangible outputs for management to judge progress.
- *Milestones* are the end-point of a process activity.
- *Deliverables* are project results delivered to customers.
- The waterfall process allows for the straightforward definition of progress milestones.

Milestones in the RE process



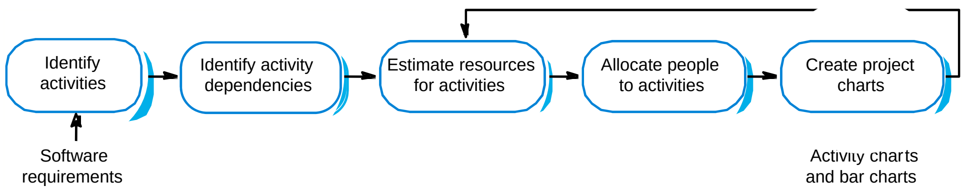
Project scheduling

- Expert judgment
  - analogy
  - proportion
  - Delphi technique
  - Wolverton model
- Algorithmic methods: $E = (a + bS^c)\ m(\mathbf{X})$
  - Walston and Felix model: $E = 5.25S^{0.91}$
  - Bailey and Basili model: $E = 5.5 + 0.73S^{1.16}$

Project scheduling

- Split project into tasks and estimate time and resources required to complete each task.
  - Organize tasks concurrently to make optimal use of workforce.
  - Minimize task dependencies to avoid delays caused by one task waiting for another to complete. (Pert or Gantt chart)
- Dependent on project managers intuition and experience.
  - WBS (Work Breakdown Structure). Make parts small enough so that effort is known
  - Tools – COCOMO, SLIM, Price S, Function Points

The project scheduling process



Software Cost Estimation

Basic method:

1. break system down into recognizable pieces
2. estimate each piece
3. sum pieces together

This works when we understand each decomposition and can estimate the pieces.

For other engineering fields (e.g., bridge and tunnel building), thousands are built each year so the basic understanding is there. Not so with software. We don't fully understand the decomposition principle to make well understood pieces.

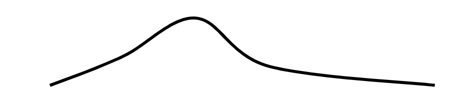
Software Cost Estimation

- However, even in other fields, estimation is not good if the product is new (e.g., although the McHenry tunnel in Baltimore was built well under budget, there were billions in overruns on the English-French "Chunnel"; also NASA and space station).

- There is still some belief in the "exchanging" of staff and time (e.g., Brook's "Mythical Man Month").
  - Research in 1970s, based upon hardware reliability theory, indicates that the "natural" growth of staff follows the Rayleigh curve.

Rayleigh estimator

$$\text{Effort} = 2\,k\,a\,t\,e^{-at2}$$



However, there is some empirical evidence that this schedule cannot be compressed more than about 25% without dire consequences.

Measurement models

Major data needed to monitor a project:

- Effort (or its related cost)
- Defects
- Schedule
- Functionality

We don't really know what "functionality" means so we use alternatives:

- Size (e.g., Lines of code)
- Data complexity (e.g., function points
- Modules
- Other measures of effort-to-build (e.g., function points)

COCOMO –COst COntainment MOdel

- Boehm – 1981. The assumption is that effort is related to size as follows: Effort= $k(LOC)^p$ where k and p depend upon application domain and environment and p>1.
- Much of COCOMO is based upon 1970s technology which underlies many of the assumptions in the model (e.g., batch environments, DoD contracting, FORTRAN and similar languages).
- One of the weaknesses of COCOMO is that other environments may not have the same underlying assumptions used in setting the parameters.

Scheduling problems

- Estimating the difficulty of problems and hence the cost of developing a solution is hard.
- Productivity is not proportional to the number of people working on a task.
- Adding people to a late project makes it later because of communication overheads.
- The unexpected always happens. Always allow contingency in planning.
  - After allowing for the expected unexpected, add time for the unexpected unexpected.

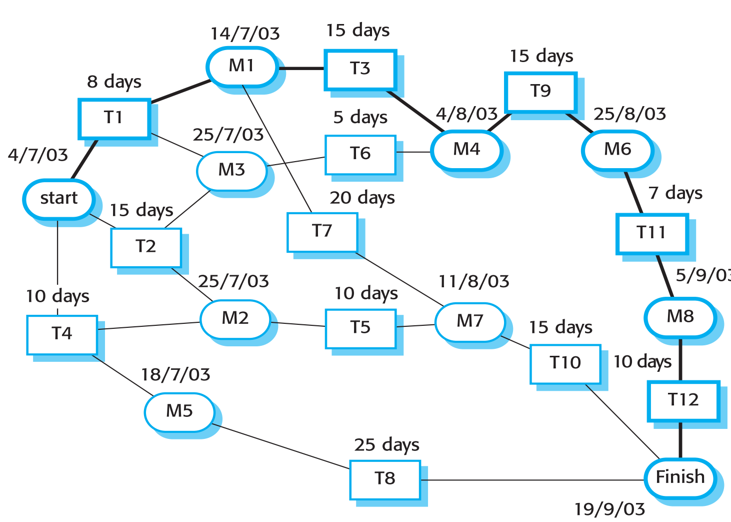
Bar charts and activity networks

- Graphical notations used to illustrate the project schedule.
- Show project breakdown into tasks. Tasks should not be too small. They should take about a week or two.
- Activity charts show task dependencies and the the critical path.
- Bar charts show schedule against calendar time.

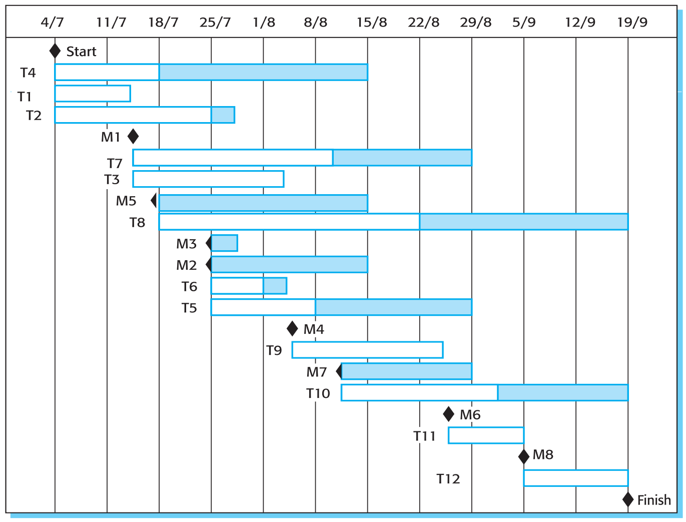
Task durations and dependencies

Activity	Duration (days)	Dependencies
T1	8	
T2	15	
T3	15	T1 (M1)
T4	10	
T5	10	T2, T4 (M2)
T6	5	T1, T2 (M3)
T7	20	T1 (M1)
T8	25	T4 (M5)
T9	15	T3, T6 (M4)
T10	15	T5, T7 (M7)
T11	7	T9 (M6)
T12	10	T11 (M8)

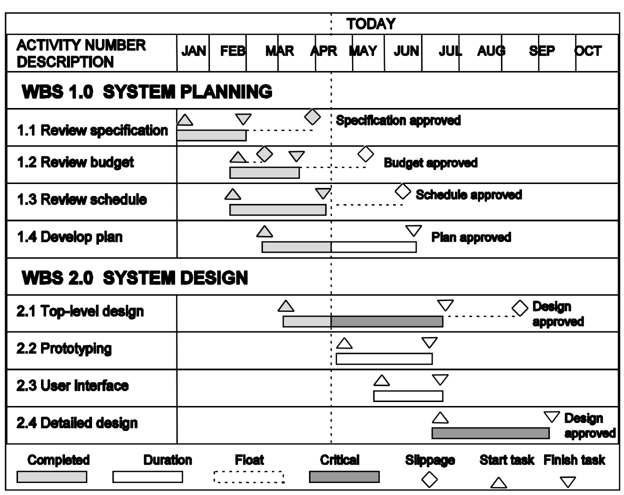
Activity network



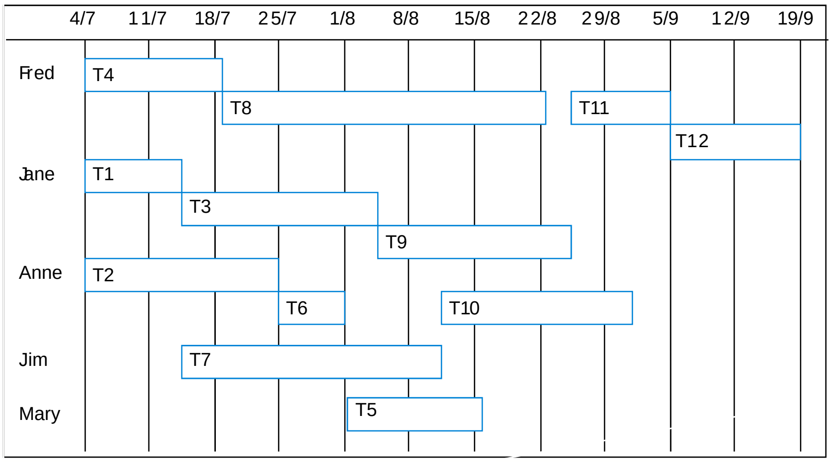
Activity timeline



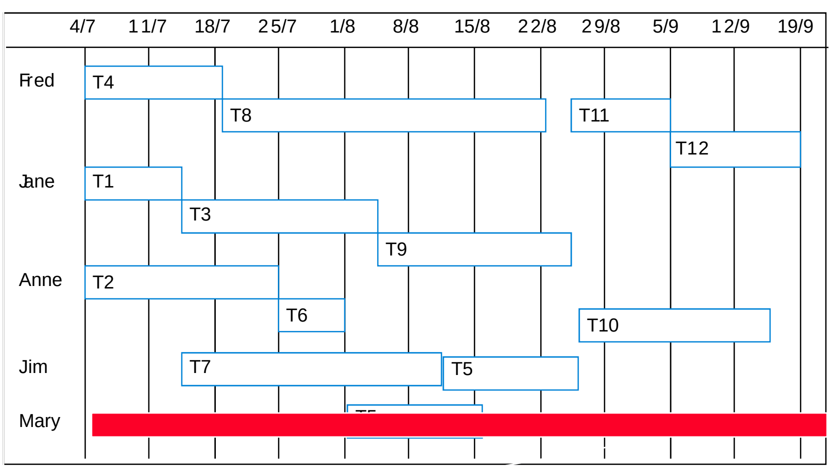
Gantt chart



Staff allocation



Reallocation saves staff



Risk management

- Risk management is concerned with identifying risks and drawing up plans to minimize their effect on a project.
- A risk is:
  1. A probability that some adverse circumstance (i.e., a cost) will occur
  2. A potential future harm that may arise from some present action
- Project risks affect schedule or resources
  - Product risks affect the quality or performance of the software being developed;
  - Business risks affect the organization developing or procuring the software.

Risk management requirements

- Risk impact: the loss associated with the event
- Risk probability: the likelihood that the event will occur
- Risk control or Risk mitigation: the degree to which we can change the outcome

Risk exposure = (risk probability) x (risk impact)

Three strategies for risk reduction

- avoiding the risk: change requirements for performance or functionality
- transferring the risk: transfer to other system, or buy insurance
- assuming the risk: accept and control it

risk leverage = difference in risk exposure divided by cost of reducing the risk

Risk

- Most managers don't understand risk or are not willing to defend their decision if a risk analysis determines a project will fail. E.g., in an engineering discipline, an engineer may not undertake a project if professional judgment says it will fail. Software managers rarely make this decision.

- Some risk analysis is subjective, so there may not be a "right" answer. But there should at least be an indication of what the risk is.

One measure of risk – Utility functions

Which of the following would you choose:

1. You can win between $0 and $1000
   - Spin a dial containing the numbers 1 through 10. If 1—9 show up, you get nothing; if 10 shows up you get $1000.
   - Win $100 for doing nothing

2. You can lose between $0 and $1000
   - Spin a dial containing the numbers 1 through 10. If 1—9 show up, you pay nothing; if 10 shows up you pay $1000.
   - Pay $100 for doing nothing

Utility function experiment

- For 1, most people choose the $100 as "found money." (E.g., in one class, 31 out of 46 chose this option.) It didn't matter that they had the potential for winning $1000.
- But for 2, most people chose spinning the dial, even though they risked losing $1000 (e.g., 16 out of 46 chose paying $100).
- But both experiments were the same (minimal gain of +/- $100 and average gain or loss of $100), yet on the chances of losing money, people were more willing to gamble than if they were guaranteed a win.
  - Understanding risk tolerance important for any project manager.

10 most important risk factors:

From Capers Jones:	From Barry Boehm:
1. Inaccurate metrics	Personnel shortfalls
2. Inadequate measurement	Unrealistic schedules and budgets
3. Excessive schedule pressure	Wrong functionality
4. Inaccurate cost estimating	Gold plating
5. Management malpractice (e.g., knowledge)	Wrong user interface
6. Silver bullet syndrome (will save the project)	Continuous requirements changes
7. Creeping user requirements	Shortfalls in externally supplied components
8. Low quality	Shortfalls in externally provided tasks
9. Low productivity	Real time performance constraints
10. Cancelled projects	Straining computer science technology

Note: Boehm's experiences working for a DoD contractor (TRW) show up in his list – depending upon subcontracts (#7 #8) – and 1970s computer technology – performance (#9)

Risk Factors (Reifer)

Risk item	Risk reduction	Cost impact
Late delivery of hardware	Acquire time on another system	Computer time costs
New operating system	Benchmark early and perform acceptance test before use	Staffing to prepare benchmark
Feasibility of requirements	Feasibility analysis and simulation	Staffing to conduct analysis
Staffing up	Start recruiting and training early	Recruiting and training costs
Feasibility of design	Peer reviews	Added effort for review preparation
Lack of management visibility	Use detailed work packaging and weekly reporting	Added effort to prepare inputs for reports
Configuration integrity of software products	Use formal change control system	Purchase price of library plus administrative costs
Lack of a test discipline	Use independent test group and get them involved early	Cost of using test group

Software risks (Sommerville)

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 organisational 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 under-performance	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.

MANAGING RISK- Risk Assessment

Risk identification – What can go wrong? Which of 10 Jones or Boehm risk factors are possible to occur?

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

- Risk analysis – For each risk, what is the cost if it occurs?
- Risk prioritization – What is the probability that the risk may occur? What is the expected value of the loss (cost of risk times its probability of occurrence)?

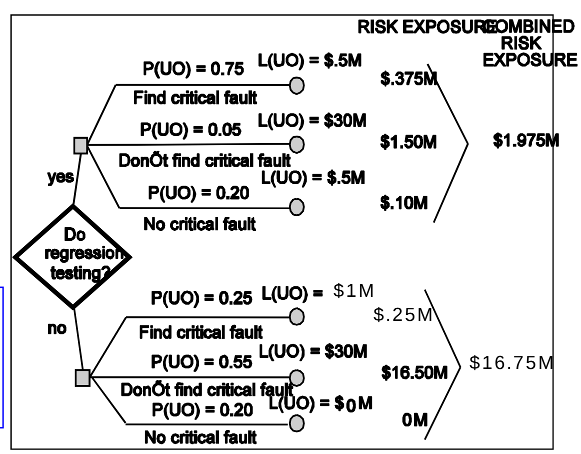
MANAGING RISK- Risk Control

- Risk management planning – Identify high cost items and develop alternative strategies to handle occurrence of risk
- Risk resolution (Mitigation) – Prototype, simulation, benchmarks, … Evaluate alternative strategies before they are needed
- Risk monitoring – Continually evaluate risks and determine new risks during project execution.

Risk exposure calculation

UO = Unwanted outcome
P(OU) = Prob(UO)
L(OU) = Loss associated with OU

Assumptions:
Cost of regression testing= $.5M
Cost to fix defect later= $1M
Cost of released defect= $30M



Risks and risk types

Risk type	Possible risks
Technology	The database used in the system cannot process as many transactions per second as expected. Software components that should be reused contain defects that limit their functionality.
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.
Organizational	The organization is restructured so that different management are responsible for the project. Organisational financial problems force reductions in the project budget.
Tools	The code generated by CASE tools is inefficient. CASE tools cannot be integrated.
Requirements	Changes to requirements that require major design rework are proposed. Customers fail to understand the impact of requirements changes.
Estimation	The time required to develop the software is underestimated. The rate of defect repair is underestimated. The size of the software is underestimated.

Risk analysis

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

Risk analysis -2

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
Software components that should be reused contain defects which limit their functionality.	Moderate	Serious
Changes to requirements that require major design rework are proposed.	Moderate	Serious
The organisation is restructured so that different management are responsible for the project.	High	Serious

Risk analysis -3

Risk	Probability	Effects
The database used in the system cannot process as many transactions per second as expec ted.	Moderate	Serious
The time required to develop the software is underestimated.	High	Serious
CASE 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
The code generated by CASE 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 plans to deal with that risk;

Risk management strategies

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.
Recruitment problems	Alert customer of 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.

Risk management strategies -2

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

Augustine* Laws

- (Law of the Core Dump) The thickness of the proposal required to win a multimillion dollar contract is about one millimeter per million dollars. If all the proposals conforming to this standard were piled on top of each other at the bottom of the Grand Canyon, it would probably be a good idea.
- One tenth of the participants produce over one third of the output. Increasing the number of participants merely reduces the average output. (A variation on Brook's Law-Adding people to speed up a late project just makes it later.)
- The last 10% of performance generates one third of the cost and two thirds of the problems.
- (Law of Unmitigated Optimism) Any task can be completed in only one-third more time than is currently estimated.
- (Law of Inconstancy of Time) A revised schedule is to business what a new season is to an athlete or a new canvas to an artist.
- Law of Propagation of Misery) If a sufficient number of management layers are superimposed on top of each other, it can be assured that disaster is not left to chance.
- VP of Boeing on 767 project: "is further ahead at the halfway point than any new airliner program in Boeing history."

* - Norman Augustine – Former CEO, Lockheed Martin Corp.

Key points

- Good project management is essential for project success.
- The intangible nature of software causes problems for management.
- Managers have diverse roles but their most significant activities are planning, estimating and scheduling.
- Planning and estimating are iterative processes which continue throughout the course of a project.

Key points

- A project milestone is a predictable state where a formal report of progress is presented to management.
- Project scheduling involves preparing various graphical representations showing project activities, their durations and staffing.
- Be proactive about managing risk or you'll constantly be in crisis-driven, firefighting mode.
- Systematically surface risks by meeting with marketing and the customer, by using checklists and taxonomies, by comparing with past projects, and by decomposing large, unwieldy risks into smaller, more manageable risks.
- All the stakeholders must communicate about risks throughout the entire development cycle. Communication is at the center of the risk management process.
- Prioritize risks by computing the risk exposure of each risk. Sort the list of risks based upon the risk exposure and proactively manage those on the top of the list.
- Develop a "Top 10" risk list for your projects. It is likely that this "Top 10" list will contain risks that will appear on your next projects as well.
- Utilize a risk-driven process for choosing between an agile and a plan-driven process, or a hybrid of the two.