Software Project Management

Engr. Madeha Mushtaq
Department of Computer Science
Iqra National University

What is Management?

- Basically, the management involves the following activities:
- **Planning**- deciding what is to be done
- **Organizing**- making arrangements
- **Staffing**- selecting the right people for the job
- **Directing**- giving instructions
- **Monitoring**- checking on progress
- **Controlling**- taking action to remedy hold-ups
- **Innovating**- coming up with new solutions
- **Representing**- liaising with users, etc.

What is a Project?

- A project can be defined as:
- A specific plan or design,
- A planned activity,
- A project is an activity with specific goals which takes place over a finite period of time.
- Another key aspect of project is that the undertaking is non-routine, a job which is repeated a number of times is not a project.

What is Project Management?

- Project Management is the art of maximizing the probability that a project delivers its goals on Time, to Budget and at the required Quality.
- Project management is the application of knowledge, skills, tools, and techniques to project activities to meet project requirements.
- Project management is accomplished through the use of the processes such as: initiating, planning, executing, controlling, and closing.
- It is important to note that many of the processes within project management are iterative in nature.

What is Project Management?

- The term project management is sometimes used to describe an organizational approach to the management of ongoing operations.
- Almost any human activity that involves carrying out a non-repetitive task can be a project.
- But there is a big difference between carrying out a very simple project involving one or two people and one involving a complex mix of people, organizations and tasks.

What is Software Project Management(SPM)?

- Software Project Management(SPM) is almost the same as general project management but the products of software projects have certain characteristics that make them different.
- Invisibility:
- Complexity:
- Flexibility:

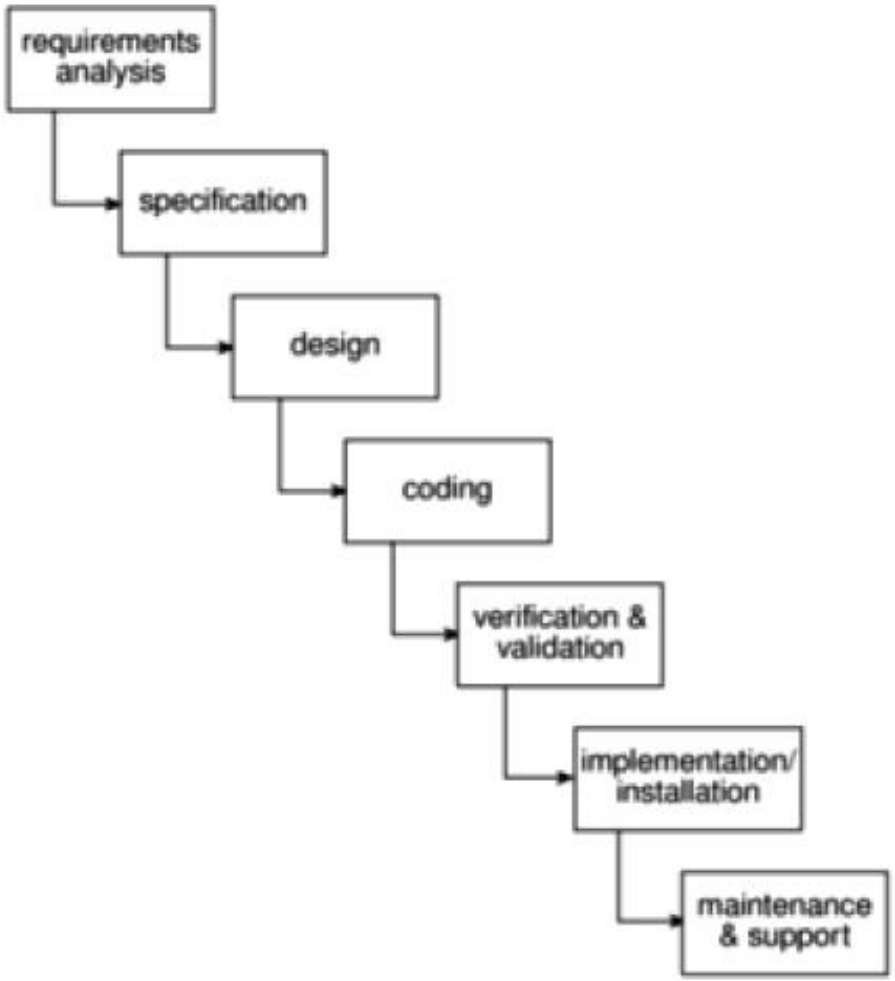
Activities covered by SPM

- Usually there are three successive processes that bring a new system into being.
- The Feasibility Study:
- This is an investigation to decide whether a prospective project is worth starting.
- Information will be gathered about the general requirements of the proposed system.
- Planning:
- If the feasibility study indicates that the prospective project is viable, then planning of the project can take place.

Activities covered by SPM

- An outline plan for the whole project and a detailed one for the first stage is formulated.
- Project Execution:
- The project can now be executed.
- Individual projects are likely to differ considerably but a classic project life cycle is as shown in the figure.

Project Life Cycle



Denver Airport Baggage System

- The New Denver Airport was to have a totally automated baggage handling system:
  - a large scale, client/server,
  - real time distributed system with 35 kilometres of track,
  - 4,000 high-speed, bag carrying, independent "telecars“ mounted on tracks to carry and deliver baggage to 20 airlines.

Denver Airport Outcomes

- Delivered 16 months late
- US$3.2m over budget
- Delivered system was mostly manual
- Cost of US$m per day in lost revenues, interest and operating costs.

Software Crisis?

- The term software crisis was coined at the first NATO SE Conference in 1968
  - Causes:
    - overall complexity of the software process
    - immaturity of software engineering as a profession
  - Effects:
    - Projects running over-budget.
    - Projects running over-time.
    - Software was of low quality.
    - Software often did not meet requirements.
    - Projects were unmanageable and code difficult to maintain.

CHAOS Report 1994

- 1994: Out of 8,380 projects in the government and private sectors in the USA:
- 31% of software projects are cancelled before they are completed.
- 53% of those that are completed cost an average of 189% of their original estimates.
- Of those 53%, only 42% have the original set of proposed features and functions.
- Only 9% of the projects were completed on time and on budget (16% counted as successes).

CHAOS Report 1994

- Project success rates have increased to 34% of all projects; more than 100% improvement from 1994
- 15% of software projects are cancelled before they are
- completed.
- 51% are *challenged* projects(over time, over budget and/or lacking critical features).
- Of those 51%, most have cost overrun *under* 20% of budget.

CHAOS Report 1994

- All projects (including failures) cost overrun 43% in 2004 180% in 1994.
- Why? “The primary reason is the projects have gotten a lot smaller.
- Doing projects with iterative processing as opposed to the waterfall method, which called for all project requirements to be defined up front, is a major step forward.” Standish Chairman Jim Johnson.(Software Magazine: 2004-01-15)

Why do software projects fail?

- People begin programming before they understand the problem
  - Everyone likes to feel that they're making progress
  - When the team starts to code as soon as the project begins, they see immediate gains
  - When problems become more complex (as they always do!), the work gets bogged down
  - In the best case, a team that begins programming too soon will end up writing good software that solves the wrong problem.

Why do software projects fail?

- The team has an unrealistic idea about how much work is involved.
  - From far away, most complex problems seem simple to solve
  - Teams can commit to impossible deadlines by being overly optimistic and not thinking through the work
  - Few people realize the deadline is optimistic until it's blown

Why do software projects fail?

- Defects are injected early but discovered late.
  - Projects can address the wrong needs
  - Requirements can specify incorrect behavior
  - Design, architecture and code can be technically flawed
  - Test plans can miss functionality
  - The later these problems are found, the more likely they are to cause the project to fail

Why do software projects fail?

- Programmers have poor habits – and they don't feel accountable for their work.
  - Programmers don't have good control of their source code
  - Code written by one person is often difficult for another person to understand
  - Programmers don't test their code, which makes diagnosing and fixing bugs more expensive
  - The team does not have a good sense of the overall health of the project.

Why do software projects fail?

- Managers try to test quality into the software.
  - Everyone assumes that the testers will catch all of the defects that were injected throughout the project.
  - When testers look for defects, managers tell them they are wasting time.
  - When testers find defects, programmers are antagonized because they feel that they are being personally criticized.
  - When testers miss defects, everyone blames them for not being perfect.

How can we make sure that our projects succeed?

- Make sure all decisions are based on openly shared information
  - It's important to create a culture of transparency, where everyone who needs information knows where to find it and is comfortable looking at it.
  - All project documents, schedules, estimates, plans and other work products should be shared with the entire team, managers, stakeholders, users and anyone else in the organization who wants them.
  - Major decisions that are made about the project should be well-supported and explained.

How can we make sure that our projects succeed?

- Don’t second-guess your team members’ expertise
  - Managers need to trust team members.
  - Just because a manager has responsibility for a project’s success, it doesn’t mean that he’s more qualified to make decisions than the team members.
  - If you don’t have a good reason to veto an idea, don’t.

How can we make sure that our projects succeed?

- Introduce software quality from the very beginning of the project
  - Review everything, test everything.
  - Use reviews to find defects – but don't expect the review to be perfect.
  - Use reviews to gain a real commitment from the team.
  - It's always faster in the long run to hold a review than it is to skip it.

How can we make sure that our projects succeed?

- Don’t impose an artificial hierarchy on the project team
  - All software engineers were created equal.
  - A manager should not assume that programming is more difficult or technical than design, testing or requirements engineering.
  - Managers should definitely not assume that the programmer is always right, or the tester is always raising false alarms.

How can we make sure that our projects succeed?

- Remember that the fastest way through the project is to use good engineering practices
  - Managers and teams often want to cut important tasks – especially estimation, reviews, requirements gathering and testing.
  - If it were faster to build the software without these practices, we would never use them.
  - Every one of these practices is about saving time and increasing quality by planning well and finding defects early.
  - Cutting them out will cost time and reduce quality.

End of Slides