Software Project Management

Lecture # 2

Outline

- The Management Spectrum
  - 4 Ps in Project Management
- W^5HH Principle

The Project Management Spectrum

- Effective software project management focuses on the four's:
  - People
  - Product
  - Process
  - Project

The People

The People

- “People factor” is very important in success of a software project.
- “Companies That sensibly manage their investment in people will prosper in the long run” *Tim & Tom.*
- Cultivation of motivated and highly skilled software people has always been important for software organizations.

The People (Contd.)

- People involved in Software Process are:
  - Stakeholders
  - Team Leaders
  - Software Team

The People - The Stakeholders

- They can be categorized into one of the following:
  - Senior Managers
    - they define business issues that often have significant influence on business
  - Project (technical) managers
    - they must plan, motivate, organize and control the practitioners who do software work
  - Practitioners
    - They deliver the technical skills necessary to engineer a product or application
  - Customers
    - They specify the requirements for the software to be engineered
  - End Users
    - They interact with the software after it is released for production use

The People - The Team Leaders

- Competent Practitioners often make poor team leaders as they lack the right mix of people skills.
- In his excellent book of technical leadership, *Jerry Weinberg* suggests a MOI model of Leadership
  - Motivation
    - encourage technical people (by “push” or “pull”) to produce
  - Organization
    - Apply , improve processes efficiently
  - Ideas or Innovation
    - Make people feel creative
    - Be Creative

The People - The Team Leaders

- Characteristics of a effective project managers:
  - Problem Solving
    - Diagnostic
    - Skill to solve
    - Ability to design solution
  - Managerial Identity
    - Take charge and control the project
    - Allow good technical people to follow their instincts
  - Achievement
    - Reward Initiative & accomplishment
    - Encourage Controlled risk taking
  - Influence and team building
    - Influence the team
    - Read people's mind and react to their needs
    - Be controlled in stress situations

The People - The Software Teams

- Organizations/Structure of teams:
  - **Democratic decentralized**
    - No permanent leader
    - Communication is horizontal
    - Suitable for small projects requiring less than 5 to 6 engineers, research-oriented projects
  - Advantages
    - At different times, different members within the team provide technical leadership.
    - High morale and job satisfaction due to autonomy, hence less employee turnover.
  - Disadvantages
    - Team members may waste time arguing about trivial points due to absence of any authority in the team.

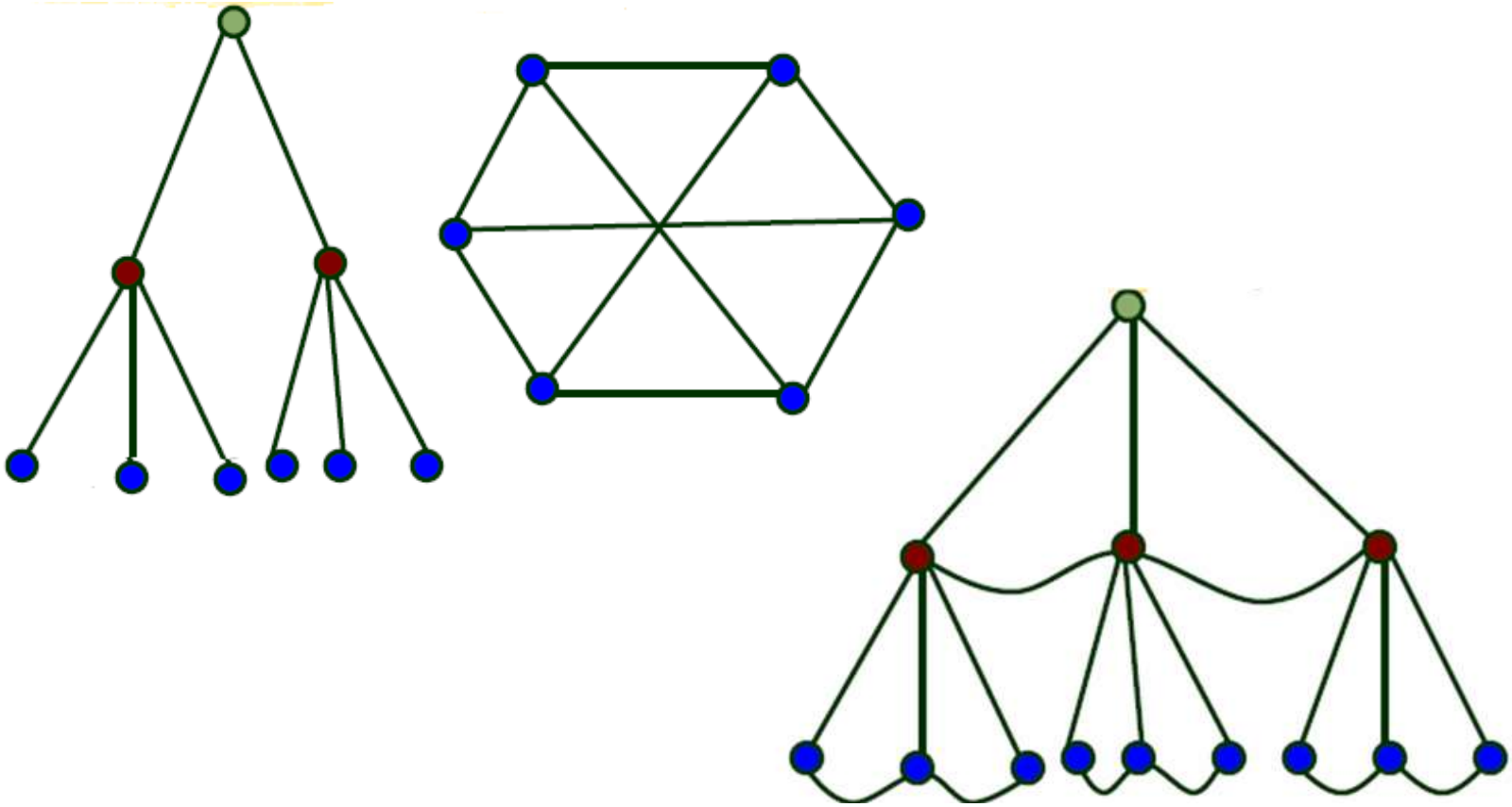
The People - The Software Teams (Contd.)

- **Controlled centralized**[Chief Programmer Team]
  - Defined team leader
  - Problem solving , communication and management by team leader
  - Communication is vertical
- Advantages
  - The senior engineer/leader partitions tasks, verifies and integrates the products developed by members.
- Disadvantages
  - Too much responsibility & authority is assigned to leader, possibility of single point of failure

The People - The Software Teams (Contd.)

- **Controlled decentralized** [Mixed Team Organization]
  - Draws upon the ideas from both earlier structures
  - Defined Leader
  - Horizontal communication
  - Problem solving is a group activity
  - Suitable for large organizations

The People - The Software Teams (Contd.)



1. Controlled Centralized

2. Democratic decentralized

3. Controlled decentralized

The People - The Software Teams

- *Mantei* describes seven factors that should be considered when planning team structure:
  - Difficulty of task
  - Size of resultant code (no. of lines)
  - Time that team will stay together
  - Degree of modularization
  - Required quality and reliability of the system being built
  - Rigidity of delivery date (schedule)
  - Degree of communication

Reading assignment

- Pg. 634 (6th edition)
  - Organizational paradigms suggested by Constantine
- Pg. 635, 636 (6th edition)
  - What is a “jelled” team?
  - Why do teams fail to jell

The People - Agile Teams

- *Agile software development encourages customer satisfaction and early incremental delivery of software with overall simplicity.*
- **Agile teams** are small, highly motivated teams.
- They adopt many characteristics of successful software project teams and avoid toxins that create problems.
- They are self organizing and do not necessarily maintain a single team structure (use elements of Constantine's paradigms)
- Agile process models give significant autonomy to agile teams.

The People - Agile Teams (Contd.)

- Planning is kept to minimum.
- The agile team is allowed to select its own approach (e.g., process, methods, tools).
- The agile team may have daily team meetings to coordinate and synchronize the day's work.
- With each passing day, this self organization and collaboration move the team towards a completed software increment.

The People - Communication & coordination Issues

- Formal approaches
  - Writings (SE documentation, Customer requests, etc.)
  - Structured meetings (e.g., Status review meetings)
  - Design and code inspections
  - Other non-interactive and impersonal comm. channels
- Informal approaches (more personal)
  - Interpersonal networking
  - Sharing of ideas on ad hoc basis
  - Seeking help from inside or outside the project team when problem arises
  - Electronic Communication (e.g., e-mail, electronic bulletin boards, video conferencing)

PM-CMM

- The “people-factor” is so important that SEI has developed People Management Capability Maturity Model (PM-CMM).

- Developed by SEI
  - “to enhance the readiness of s/w organizations to undertake increasingly complex applications by helping to attract, grow, motivate, deploy, and retain the talent needed to improve their software development capability”
  - In simple words - to enhance the people’s capabilities through personnel development

PM-CMM (Contd.)

- Key Practice Areas of PM-CMM
  - Recruiting
  - Selection
  - Performance Management
  - Training
  - Compensation
  - Career development
  - Organization and work design
  - Team/culture development

- Organizations that achieve high levels of maturity in PM-CMM have a higher likelihood of implementing effective software engineering practices

The Product

The Product

- The product and the problem it is intended to solve must be examined at very beginning of the software project.
- Before project planning
  - product objectives & scope must be established,
  - alternative solutions should be considered (to select best approach within constraints),
  - technical and management constraints should be identified.
  - Without the above, cost estimates, risk assessment, realistic work breakdown and scheduling is impossible.

The Product - Software Scope

- Scope is defined by
  - **Context** (1st step)
    - Functional location of the software product into a large system, product or business context
    - Constraints involved
  - **Information Objectives** (2nd step)
    - What data objects are required as i/p or o/p
  - **Function and Performance** (3rd step)
    - What function does the software system perform on i/p to produce o/p
    - What level of performance is required

The Product - Software Scope (Contd.)

- The **scope** of product must be established and bounded.
  - Bounded scope means
    - establishing quantitative data (e.g., no. of simultaneous users, max. allowable response time) are explicitly stated
    - constraints and limitations (e.g., product cost restricts memory size) are noted
    - and mitigating factors (e.g., algorithms are well understood) described

The Product - Problem Decomposition

- Also called partitioning OR problem elaboration
- The **problem** that the product is addressing must be decomposed
- This activity is at core of requirements analysis
- Divide and conquer policy for complex problems
  - Decompose problem in tasks
  - Decomposition in 2 major areas
    - Functionality that must be delivered
    - Process that will be used to deliver product

Problem Decomposition (Contd.)

- A complex problem is partitioned into smaller problems that are more manageable.
- Decomposition make planning easier.
- Software functions, defined in scope, are evaluated and refined to provide more detail before estimation (part of planning) begins.

The Process

The Process

- A software process provides the framework from which a comprehensive plan for software development can be established.
- Common process framework activities which are applicable to all software projects are:
  - Communication
  - Planning
  - Modeling
  - Construction
  - Deployment

The Process (Contd.)

- **Process models**
  - Linear sequential, Prototyping, RAD, Spiral, Formal …

The Process (Contd.)

- The problem is to select the process model that is appropriate for the software to be engineered by the project team.
- Project manager must decide about which model to use depending on
  - customers who have requested the product & people who will work on it
  - characteristics of the product
  - project environment in which people will work
- Project planning begins after process model is selected and later process decomposition takes place.

The Process (Contd.)

- Process decomposition
  - The way a process is decomposed depends on project complexity
  - Decomposition involves outlining of work tasks involved in each process framework activity
- Example of decomposition for ‘communication’ activity for a simple project:
  - Develop a list of clarification issues
  - Meet with customer to discuss clarification issues
  - Jointly develop statement of scope
  - Review the statement of scope with all concerned
  - Modify the statement of scope id required

The Project

The Project

- The software projects must be planned and controlled effectively to avoid complexities.
- The project managers and engineers must understand the critical success factors and develop a common sense approach for planning, monitoring and controlling the project.

The Project - Signs of Projects Risk

- *John Reel* describes ten signs that indicate that project is in jeopardy:
  - Software people don’t understand customer needs
  - Product scope is poorly defined
  - Changes are managed poorly
  - The chosen technology changes
  - Business needs change
  - Deadlines are unrealistic

The Project - Signs of Projects Risk (Contd.)

- Users are resistant
- Sponsorship is lost
- Team lacks skills
- Managers avoid best practices

The Project - How to avoid problems?

- *John Reel* suggests the following:
- Start on the right foot
  - Involves detailed understanding of project
  - setting realistic objectives & expectations
  - Selecting the right team
  - Facilitating the team
- Maintain Momentum
  - Provide incentives
  - Reduce bureaucracy and give autonomy to team members but with supervision
- Track Progress
  - Assess progress as work products are produced

The Project - How to avoid problems? (Contd.)

- Make smart decisions
  - When possible, use existing software components / COTS software
  - Choose standard approaches and keep it simple
  - Avoid risks and allocate more time than needed for complex/risky tasks
- Conduct a postmortem analysis
  - Compare planned and actual schedule
  - Collect and analyze project metrics (standards)
  - Get feedback from team and customers
  - Establish record of lessons learnt for each project

W^5HH Principle

W^5HH Principle - Introduction

- Suggested by *Barry Boehm* in one of his papers
- Excellent **planning outline** for project managers and software team
- Applicable to all sizes of software projects
- It is an approach to address
  - project objectives
  - Milestones & schedule
  - Responsibilities
  - Management & technical approaches
  - Required resources

W^5HH principle

- **W**hy is the system being develop?
  - Answer to this questions help assess validity of business reason for the software work.
  - It answers if the business purpose justifies the expenditure of people, time and money
- **W**hat will be done?
  - Answer to this question establishes the task set required for project
- **W**hen will it be done?
  - Answer to this question helps the team establish a project schedule by identifying when tasks have to be conducted and when milestones are to be reached

W^5HH principle (Contd.)

- **W**ho is responsible for a function ?
  - Answer to this question establishes roles and responsibility of each team member
- **W**here are they organizationally located ?
  - Answer to this question indicates that all roles and responsibilities are not limited to the software team itself, the customers, users and stakeholders also have responsibilities.
- **H**ow will be job done technically and managerially ?
  - Once product scope is establishes, a technical and management strategy must be defined for it.
- **H**ow much of each resource is needed ?
  - Answer to this question is derived by developing estimates based on answers to earlier questions.

Reference

- Today's lecture material was taken from
  - Chapter 21- Project Management of Software Engg. A Practitioner's Approach by Roger Pressman
  - Team Structures Portion taken from Gary Pollice lectures on Software Project Management

THE END