Chapter 31

- **Project Management Concepts**

Slide Set to accompany

Software Engineering: A Practitioner's Approach

by Roger S. Pressman

Table of Contents

31.1 The Management Spectrum

The Four P's

- People — the most important element of a successful project
  - Product — the software to be built
  - Process — the set of framework activities and software engineering tasks to get the job done
- Project — all work required to make the product a reality

31.2 The People

31.2.1 Stakeholders

- Senior managers who define the business issues that often have significant influence on the project.
- Project (technical) managers who must plan, motivate, organize, and control the practitioners who do software work.
- Practitioners (= software engineers) who deliver the technical skills that are necessary to engineer a product or application.
- Customers who specify the requirements for the software to be engineered and other stakeholders who have a peripheral interest in the outcome.
- End-users who interact with the software once it is released for production use.

31.2.2 Team Leaders

- **The MOI Model of leadership**
  - Motivation. The ability to encourage (by “push or pull”) technical people to produce to their best ability.
  - Organization. The ability to mold existing processes (or invent new ones) that will enable the initial concept to be translated into a final product.
  - Ideas or innovation. The ability to encourage people to create and feel creative even when they must work within bounds established for a particular software product or application.

31.2.3 The Software Team

When selecting a team structure, the following factors must be considered ...

- Difficulty of the problem to be solved
- Size of the resultant program(s) in lines of code or function points
- Time that the team will stay together (team lifetime)
- Degree to which the problem can be modularized
- Required quality and reliability of the system to be built
- Rigidity of the delivery date
- Degree of sociability (communication) required for the project

Organizational Paradigms

- Closed paradigm —structures a team along a traditional hierarchy of authority
- Random paradigm —structures a team loosely and depends on individual initiative of the team members
- Open paradigm —attempts to structure a team in a manner that achieves some of the controls associated with the closed paradigm but also much of the innovation that occurs when using the random paradigm
- Synchronous paradigm —relies on the natural compartmentalization of a problem and organizes team members to work on pieces of the problem with little active communication among themselves

- suggested by [Constantine 93]

Avoid Team "Toxicity"

- A frenzied work atmosphere in which team members waste energy and lose focus on the objectives of the work to be performed.
- High frustration caused by personal, business, or technological factors that cause friction among team members.
- "Fragmented or poorly coordinated procedures" or a poorly defined or improperly chosen process model that becomes a roadblock to accomplishment.
- Unclear definition of roles resulting in a lack of accountability and resultant finger-pointing.
- "Continuous and repeated exposure to failure" that leads to a loss of confidence and a lowering of morale.

31.2.4 Agile Teams (1/2)

Agile Software Development encourages:
- customer satisfaction
- early incremental delivery
- small highly motivated team (= agile team)
- informal method,
- minimal work products
- overall development simplicity

31.2.4 Agile Teams (2/2)

- Agile Team
  - Team members must have trust in one another.
  - The distribution of skills must be appropriate to the problem.
  - Mavericks (who are independent and unconventional) may have to be excluded from the team, if team cohesiveness is to be maintained.
  - Team is "self-organizing"
    - An adaptive team structure
    - Uses elements of Constantine's random, open, and synchronous paradigms
    - Significant autonomy

31.2.5 Team Coordination & Communication

Need all of these

- Formal, **impersonal** approaches
  - include documents and work products (including source code), technical memos, project milestones, schedules, and project control tools, change requests and related documentation, error tracking reports, and repository data.
- Formal, **interpersonal** procedures
  - focus on quality assurance activities applied to work products
  - include status review meetings and design and code inspections.
- Informal, **interpersonal** procedures
  - include group meetings for information dissemination and problem solving and "collocation of requirements and development staff."
- **Interpersonal** networking
  - includes informal discussions with team members and those outside the project who may have experience or insight that can assist team members.
- Electronic communication
  - encompasses electronic mail, electronic bulletin boards, and by extension, video-based conferencing systems.

31.5 The Project

- Projects get into trouble when …
  1. Software people don't understand their customer's needs.
  2. The product scope is poorly defined.
  3. Users are resistant.
  4. Deadlines are unrealistic.
  5. The project team lacks people with appropriate skills.
  6. Changes are managed poorly.
  7. The chosen technology changes.
  8. Business needs change [or are ill-defined].
  9. Sponsorship is lost [or was never properly obtained].
  10. Managers [and practitioners] avoid best practices and lessons learned.

Common-Sense Approach to Projects

1. Start on the right foot.
   - Work hard to understand the problem to solve
   - Set realistic objectives and expectations
2. Maintain momentum.
   - The PM must provide incentives to keep turnover of personnel to a minimum
   - The team should emphasize quality in every task
   - Senior management should stay out of the team's way.
3. Track progress.
   - Progress is tracked as work products (e.g., models, source code, sets of test cases) are produced and approved (using formal technical reviews) as part of a quality assurance activity.
4. Make smart decisions.
   - The decisions of the PM and the team should be to "keep it simple."
5. Conduct a postmortem analysis.
   - Establish a consistent mechanism for extracting lessons learned for each project.

31.6 The W5HH Principle [Boehm 96]

- The same principle for planning is used for management.
  - Why is the system being developed?
  - What will be done?
  - When will it be accomplished?
  - Who is responsible?
  - Where are they organizationally located?
  - How will the job be done technically and managerially?
  - How much of each resource (e.g., people, software, tools, database) will be needed?

31.7 Critical Practices

- "Consistently used by … highly successful software projects and organizations" - Airlie Council [Air 99]
  - Empirical cost and schedule estimation
  - Earned value tracking
  - Defect tracking against quality targets
  - People aware project management
  - Metrics-based project management
  - Formal risk management (See the slides for Risk Management)