Manajemen Proyek SI

(Project Management Concepts)

Oleh :

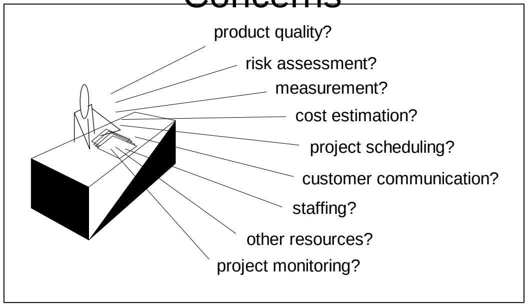
Ir. I Gede Made Karma, MT

Software Projects

Factors that influence the end result ...

- **size**
- **delivery deadline**
- **budgets and costs**
- **application domain**
- **technology to be implemented**
- **system constraints**
- **user requirements**
- **available resources**

Project Management Concerns



Why Projects Fail?

- **an unrealistic deadline is established**
- **changing customer requirements**
- **an honest underestimate of effort**
- **predictable and/or unpredictable risks**
- **technical difficulties**
- **miscommunication among project staff**
- **failure in project management**

The Management Spectrum

- Effective software project management focuses on the 4 P.
- Manager who forgets that SE work is an intensely human endeavor will never have success in project management.
- A manager who fail to encourage comprehensive customer communication early in the evolution of a project risks building an elegant solution for the wrong problem.
- The manager who pays little attention to the process runs the risk of inserting competent technical methods and tools into a vacuum.
- The manager who embarks without a solid project

The 4 P

- 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

The People

- Key Practice Areas :
  - Recruiting
  - Selection
  - Performance
  - Training
  - Compensation
  - Career Development
  - Organization and Work Design
  - Team/Culture Development

The Players

- 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.
- Practitioners 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.

Team Leaders

Project management is a people-intensive activity, and for this reason, competent practitioners often make poor team leaders. They simply don't have the right mix of people skills.

- Model of leadership :
  - Motivation. The ability to encourage (by push and 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.

Team Leaders (2)

The characteristics that define an effective project manager emphasizes :

- Problem Solving. An effective software project manager can diagnose the technical and organizational issues that are most relevant, systematically structure a solution or properly motivate other practitioners to develop the solution.
- Managerial Identity. A good manager must take charge of the project. She/He must have the confidence to assume control when necessary and the assurance to allow good technical people to follow their instincts.
- Achievement. To optimize the productivity of the project team, a manager must reward initiative and accomplishment and demonstrate through his own actions that controlled risk taking will not be punished.
- Influence and Team Building. An effective project manager must be able to "read" people; she/he must be able to understand verbal and nonverbal signals and react to the needs of the people sending these signals.

Software Teams

The following factors must be considered when selecting a software project team structure ...

- the difficulty of the problem to be solved
- the size of the resultant program(s) in lines of code or function points
- the time that the team will stay together (team lifetime)
- the degree to which the problem can be modularized
- the required quality and reliability of the system to be built
- the rigidity of the delivery date
- the 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 [CON93]

The Product

A software project manager is confronted with a dilemma at very beginning of a software engineering project :

- Quantitative estimates and an organized plan are required, but solid information is unavailable.
- A detailed analysis of software requirements would provide necessary information for estimates, but analysis often takes weeks or months to complete

Defining the Problem

- establish scope—a narrative that bounds the problem
- decomposition—establishes functional partitioning

Software Scope

- Context. How does the software to be built fit into a larger system, product, or business context and what constrains are imposed as a result of the context?
- Information objectives. What customer-visible data object are produced as output form the software? What data objects are required for input?
- Function and Performance. What function does the software perform to transform input data into output? Are any special performance characteristics to be addressed?

Problem Decomposition

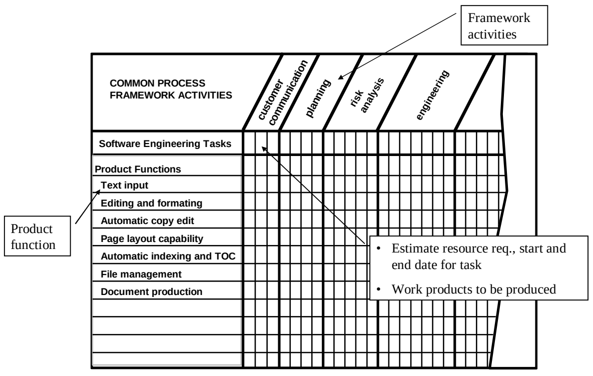
Sometimes called *partitioning* or *problem elaboration*, is an activity that sits at the core of software requirement analysis.

- The functionality that must be delivered
- The process that will be used to deliver it.

The Process

- The generic phases that characterize the software process – definition, development, and support – are applicable to all software.
- The problem is to select the process model that is appropriate for the software to be engineered by a project team.
- The project manager must decide which process model is most appropriate for :
  1. The customer who have requested the product and the people who will do the work.
  2. The characteristics of the product itself
  3. The project environment in which the software team works.

Melding Product and Process



The Project

10 signs that indicate that an information systems project is in jeopardy :

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

The Project (2)

1. Start on right foot; understand the problem, setting realistic objects, building the right team and giving the team the autonomy, authority and technology needed.
2. Maintain momentum; many projects get off to a good start and then slowly disintegrate.
3. Track progress; progress are produced and approved as part of a quality assurance activity.
4. Make smart decisions; the decisions should be to "keep it simple".
5. Conduct a postmortem analysis; establish a consistent mechanism for extracting lessons learned for each project.

To Get to the Essence of a Project

Barry Boehm : the W5HH Principle

- Why is the system being developed?
- What will be done? By when?
- Who is responsible for a function?
- 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?

Critical Practices

- Formal risk management. What are the top ten risks? The chance will become a problem and the impact.
- Empirical cost and schedule estimation. What estimated size that will delivered into operation? How?
- Metrics-based project management. Do you have a metrics program to give an early indication of evolving problems? If so, what is the current requirements volatility?
- Earned value tracking. Do you report earned value metrics? Is it computed for the entire effort to the next delivery?
- Defect tracking against quality targets. Do you track and report the defects found?
- People aware project management. What is the average staff turnover for the past three months.