投票 最多可选1项

设置

Have a good holiday?

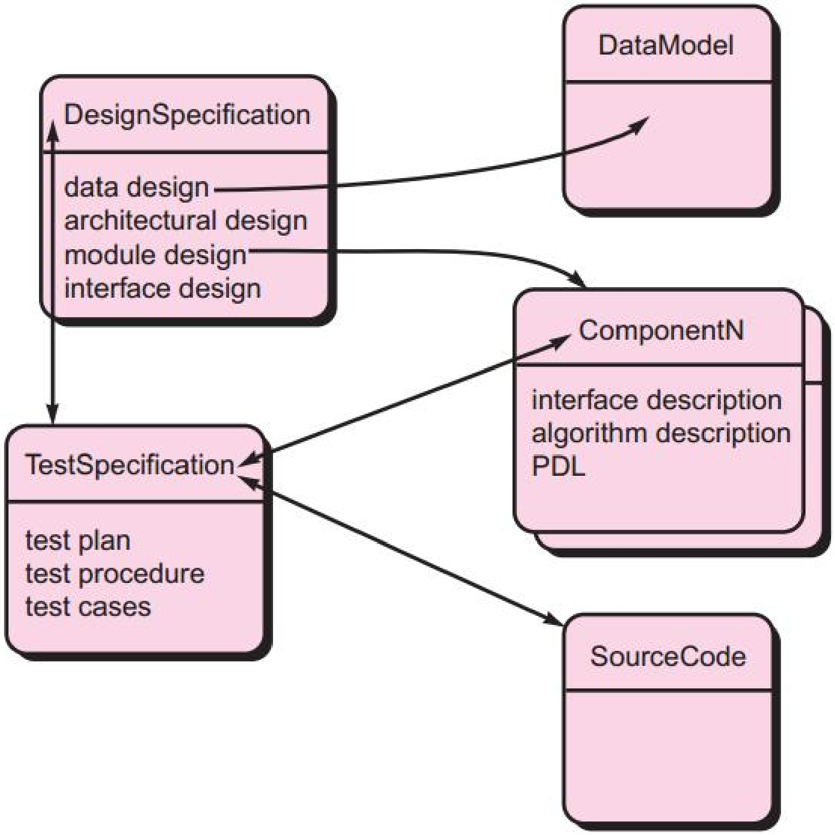
Ⓐ Yes

Ⓑ No

提交

Review – SCM1

- Software Configuration Management



DataModel

DesignSpecification

data design
architectural design
module design
interface design

ComponentN

interface description
algorithm description
PDL

TestSpecification

test plan
test procedure
test cases

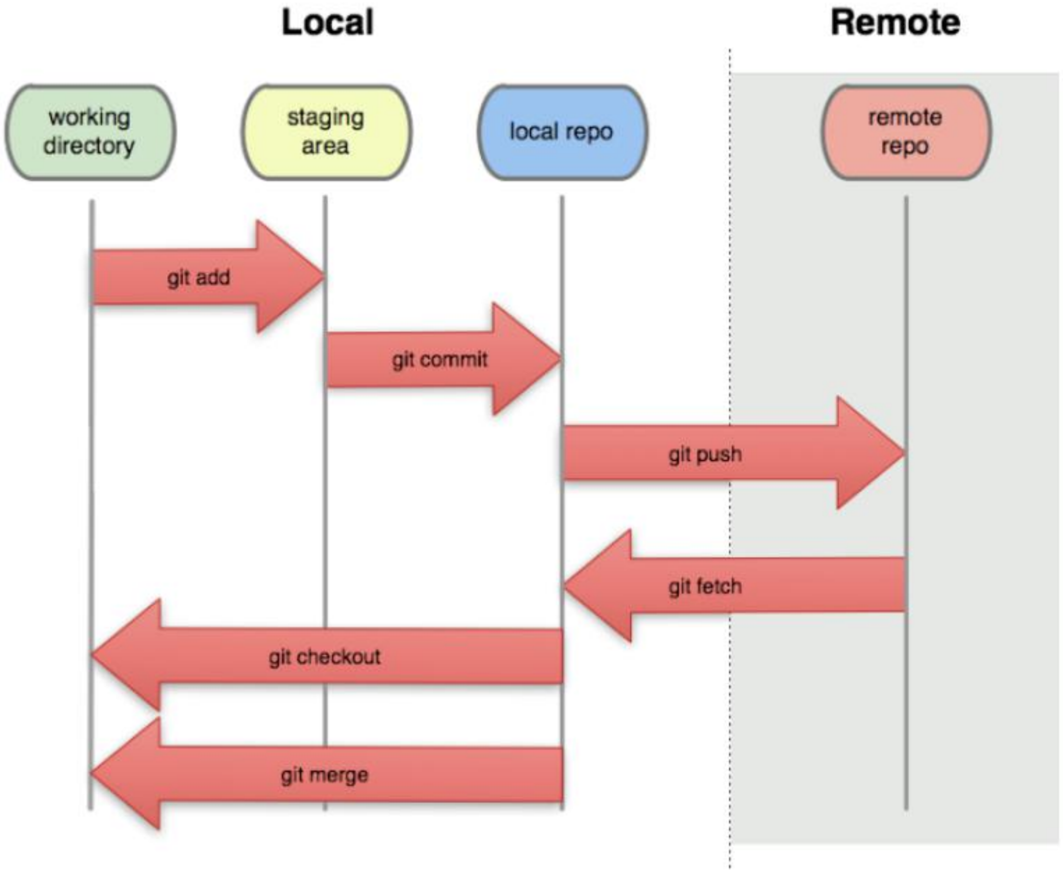
SourceCode

TortoiseSVN

github
SOCIAL CODING

Review – SCM2

- SCM tool - Git



]

Review - SCM3

- Github use flow for your team
  - Team leader (Scrum master)
    - Create a new repo (including doc/code directory) for your project
    - Invite all member for this project
      (settings->manage access -> invite a collaborator by email)
    - Deal with pull request
  - Team member
    - Accept the invitation from your project leader
    - Fetch your team repo to your local repo, and edit
    - Commit your changes to Github (create a new pull request)
    - Use “Issues” to share bug information

Change control process

Software Engineering

Part 4
Project Management

Chapter 31
Project Management Concepts

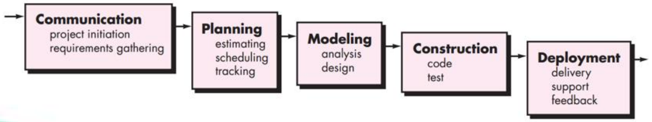
These slides are designed to accompany *Software Engineering: A Practitioner's Approach, 8/e* (McGraw-Hill 2014). Slides copyright 2014 by Roger Pressman.

Reproduced by Ning Li , 2022

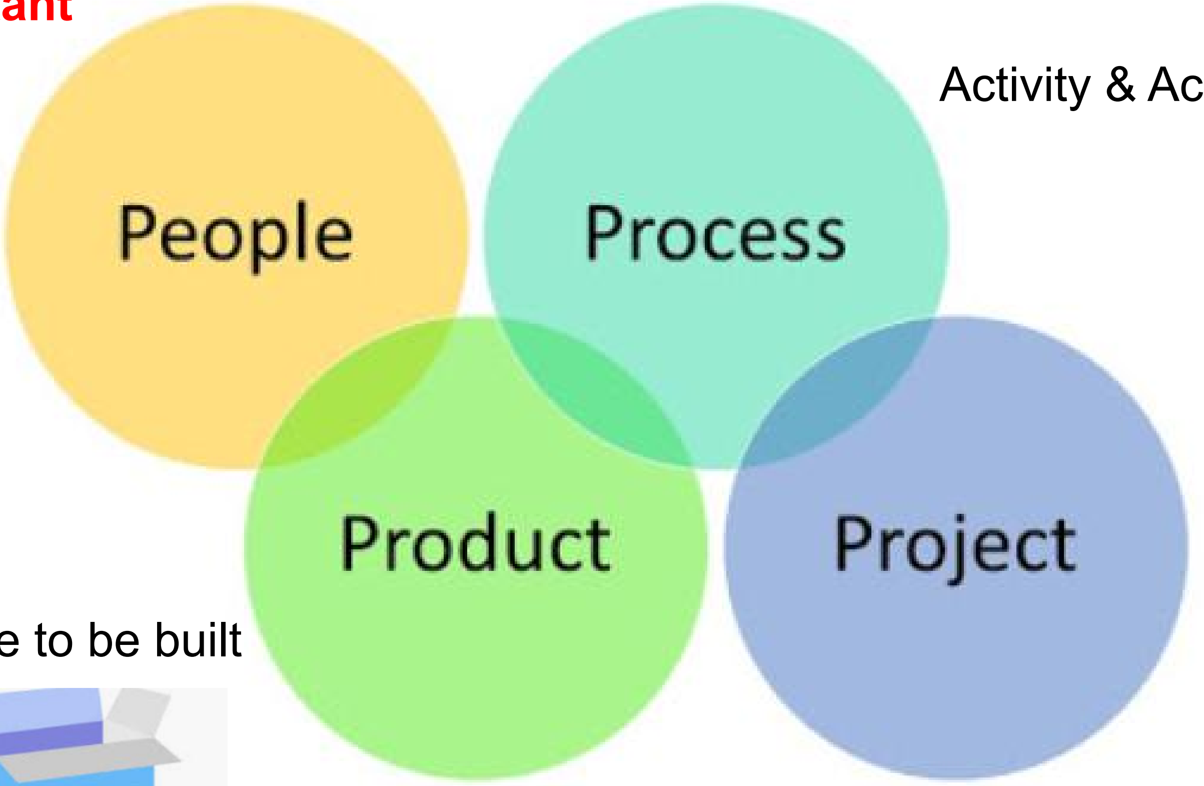
Contents

- 31.1 The Management Spectrum
- 31.2 People
  - 31.2.1 The Stakeholders
  - 31.2.2 Team Leaders
  - 31.2.3 The Software Team
  - 31.2.4 Agile Teams
  - 31.2.5 Coordination and Communication Issues
- 31.3 The product
  - 31.3.1 Software scope
  - 31.3.2 Problem Decomposition
- 31.4 The process
  - 31.4.1 Melding the product and the process
  - 31.4.2 Process Decomposition
- 31.5 The Project
- 31.6 The W^{5}HH Principles
- 31.7 Critical Practices

31.1 The Management Spectrum --- The Four P's



most important



Activity & Action & Task

the software to be built

all work required to make the product a reality

31.2.1 People --- Stakeholders

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

31.2.2 Team Leader

- The MOI Model
  - **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.

31.2.2 Team Leader

- Key traits for an effective project manager
  - **Problem solving**

    technical and organizational
  - **Managerial identity**

    confidence to control
  - **Achievement**

    controlled risk，productivity
  - **Influence and team building**

    be able to “read” people（ verbal and nonverbal）

Will you be a good manager?

31.2.2 Team Leader

How to lead?

How to organize?

How to collaborate?

How to motivate?

How to create good ideas?

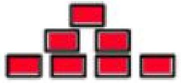
31.2.3 Software Teams

What factors should be considered when the structure of a software team is chosen?

1. the difficulty of the problem to be solved
2. the size of the resultant program(s) in lines of code or function points
3. the time that the team will stay together (team lifetime)
4. the degree to which the problem can be modularized
5. the required quality and reliability of the system to be built
6. the rigidity of the delivery date
7. the degree of sociability (communication) required for the project

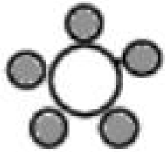
31.2.3 Organizational Paradigms

- closed paradigm— structures a team along a traditional hierarchy of authority

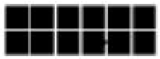


- random paradigm— structures a team loosely and depends on individual initiative of the team members

- 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



- 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



单选题 1分

设置

此题未设置答案，请点击右侧设置按钮

Which kind of organizational paradigms do you like?

- A closed paradigm
- B random paradigm
- C open paradigm
- D synchronous paradigm

提交

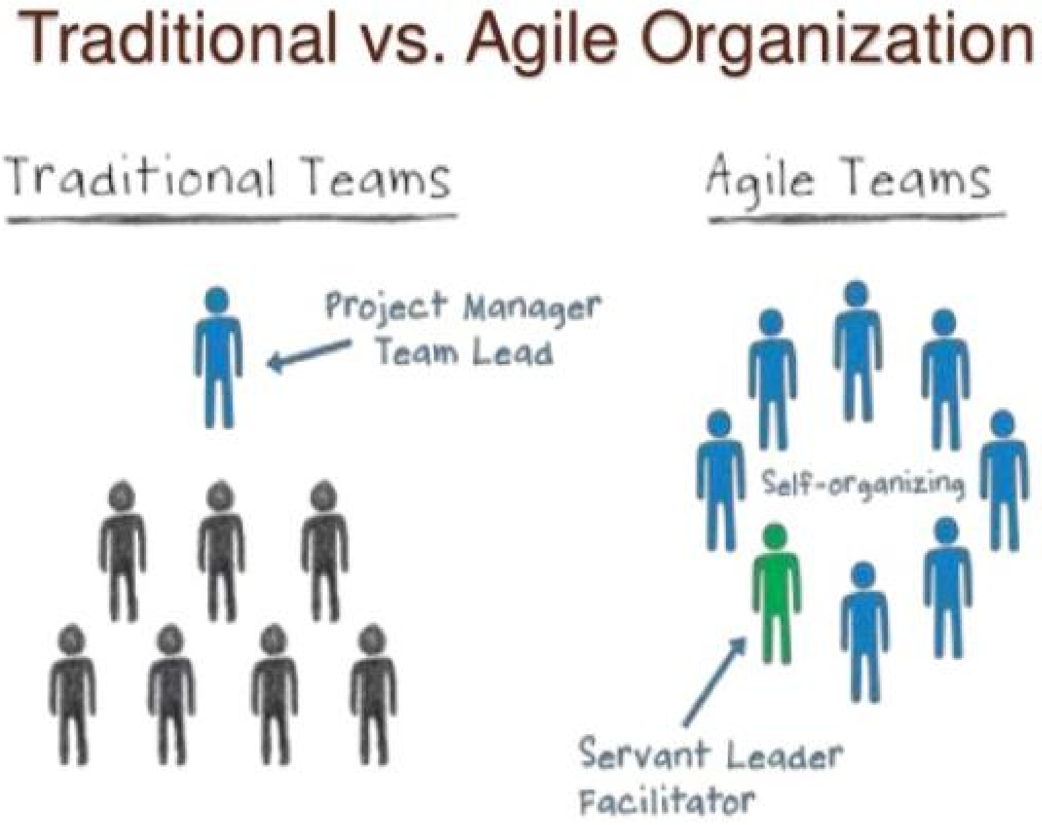
31.2.3 Avoid Team “Toxicity”

- **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** leads to a loss of confidence and a lowering of morale.

31.2.4 Agile Teams

- Team members must have **trust** in one another.
- **The distribution of skills** must be appropriate to the problem.
- **Mavericks** 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

31.2.4 Agile Teams



Recommand https://www.bilibili.com/video/av82183903?p=38

31.2 People

Reading Recommend (Cornell University)

23 people: https://www.bilibili.com/video/av82183903?p=38

31.3.1 Product – Software Scope

- Scope
  - **Context.** How does the software to be built fit into a larger system, product, or business context and what constraints are imposed as a result of the context?
  - **Information objectives.** What customer-visible data objects are produced as output from 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?

Software scope must be unambiguous and understandable at the management and technical levels.

31.3.2 Problem Decomposition

- Decomposition (*partitioning* or *problem elaboration)*
- Once scope is defined , it is …
  - decomposed into constituent functions
  - decomposed into user-visible data objects

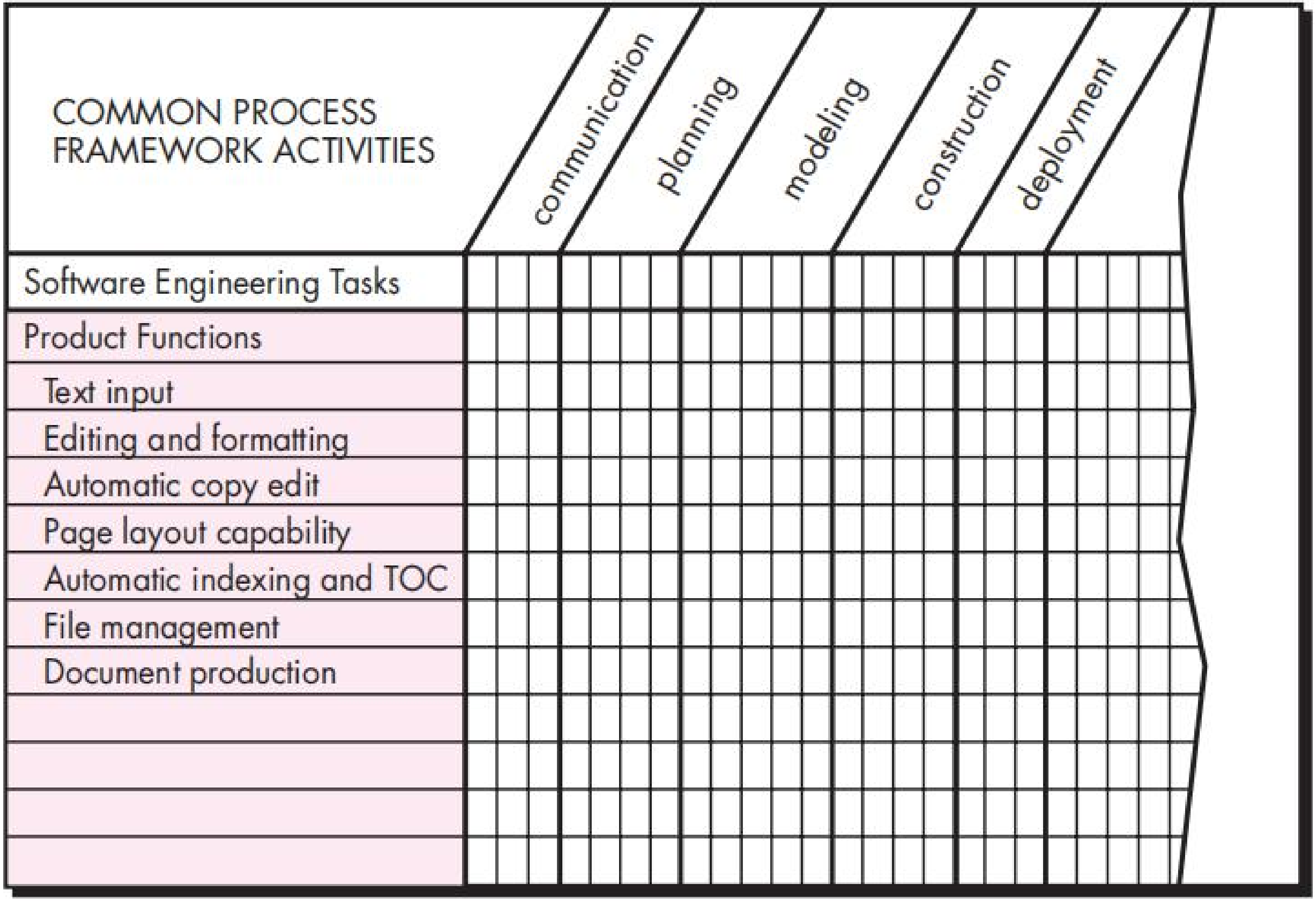
  or
  - decomposed into a set of problem classes

- Decomposition process continues until all functions or problem classes have been defined

31.4 The Process

- Decide which process model is most appropriate
  - Consider project characteristics
  - Determine the degree of rigor required
  - Define a task set for each software engineering activity
    - Task set
      - Software engineering tasks
      - Work products
      - Quality assurance points
      - Milestones

31.4.1 Melding the Problem and the Process



COMMON PROCESS FRAMEWORK ACTIVITIES	communication	planning	modeling	construction	deployment
Software Engineering Tasks					
Product Functions					
Text input					
Editing and formatting					
Automatic copy edit					
Page layout capability					
Automatic indexing and TOC					
File management					
Document production					

31.5 The Project

- *Projects get into trouble when …*
  - Software people don’t understand their customer’s needs.
  - The product scope is poorly defined.
  - Changes are managed poorly.
  - The chosen technology changes.
  - Business needs change [or are ill-defined].
  - Deadlines are unrealistic.
  - Users are resistant.
  - Sponsorship is lost [or was never properly obtained].
  - The project team lacks people with appropriate skills.
  - …

Quiz

In a study of 250 large software projects between 1998 and 2004, Capers Jones[Jon04](cited from our textbook) found about (   ) experienced major delays and overruns or were terminated without completion.

A) 50

B) 105

C) 175

31.5 Common-Sense Approach to Projects

1. ***Start on the right foot.*** Working hard to understand the problem that is to be solved and then setting realistic objectives and expectations.
2. ***Maintain momentum.*** Keep turnover of personnel to an absolute minimum, the team should emphasize quality in every task, and senior management should do everything possible to 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), and products should be produced and approved (using formal technical reviews)
4. ***Make smart decisions:*** keep it simple
5. ***Conduct a postmortem analysis.*** Establish a consistent mechanism for extracting lessons learned for each project.

31.6 the Essence of a Project (W^{5}HH)

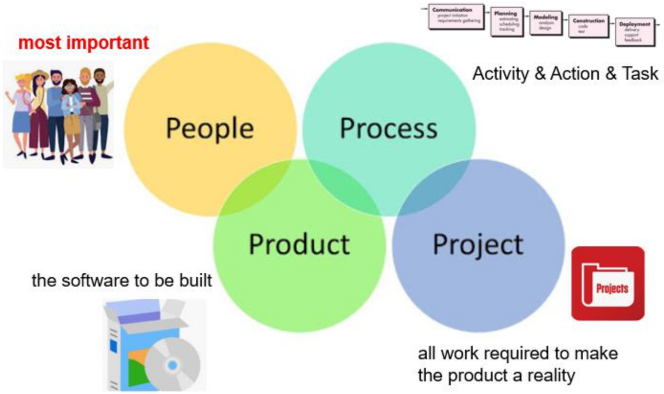
- Why is the system being developed?
- What will be done?
- When will it be accomplished?
- Who is responsible?
- Where are they organizationally located?
- How will the job be done technically and managerially?
- How much of each resource (e.g., people, software, tools, database) will be needed?

31.7 Critical Practices

- Metrics-based project management
- Cost and schedule estimation
- Earned value tracking
- Defect tracking against quality targets
- People aware project management

Summary

- Management Spectrum
  - People (stakeholders , leader , team)
  - Product (scope, decomposition)
  - Process (appropriate process model, taskset)
  - Project (W^5HH)



THE END