SWE 4603

Software Testing and Quality Assurance

Lecture 12

Prepared By Maliha Noushin Raida, Lecturer, CSE
Islamic University of Technology

Lesson Outcome

- Software Quality
- Quality Cost
- Quality Control and Quality Assurance
- Quality Management
- Software Quality Metrics
- Software Quality Assurance Model: CMM(Capability Maturity Model)

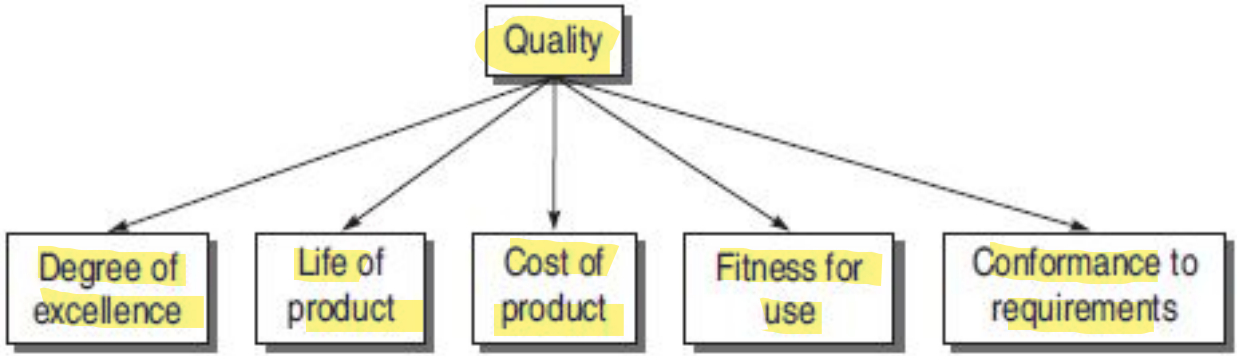
Week on:

- Chapter 13: Software Quality Management

Software Quality

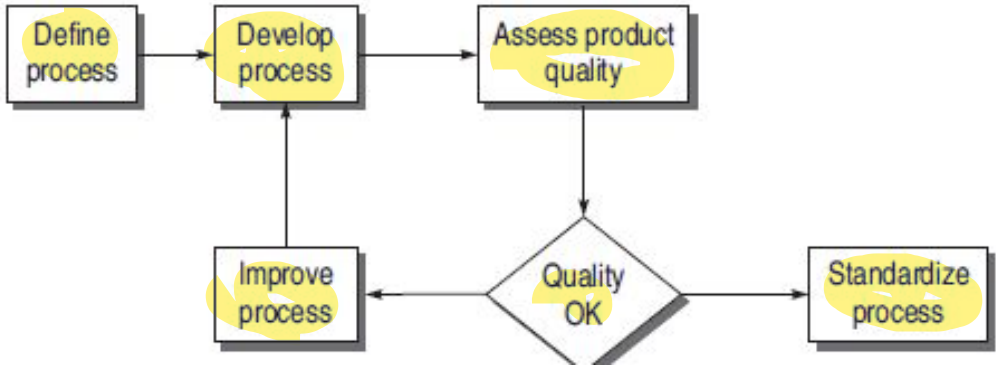
Software Quality

- Quality means whether the product is actually usable by the user or not.
- If the user is satisfied with the product, it is of good quality.
- In the basis of the multi-dimensional concept, quality can be defined as ‘the degree to which a product or service possesses a desired combination of attributes’.



Software Quality

- Software quality revolves around defects.
- It is commonly recognized as lack of bugs in the software.
- Software quality may be defined in the form of delivered defect density or defect rate, i.e. the number of defects per unit size (e.g. LOC, FP, or other unit).
- there are two views of quality.
  - **Product Quality**
    - Quality measures are taken to improve the end-product.
  - **Process Quality**
    - quality of development process directly affects the quality of delivered product



Software Quality

To improve the quality of processes and products, the following activities must be carried out:

- Choose the development process best suited for the project.
- Select and replay specific methods and approaches within the selected process.
- Adopt proper tools and technologies.
- Prepare quality metrics for products based on their desired attributes and implement software metrics program to ensure that the development process is under control and making progress.

Quality Cost

quality is never by an accident, it is always the result of an intelligent effort.

- If the efforts in the form of quality evaluation programs are not implemented, then the desired-level quality cannot be achieved.
- The cost of quality is decomposed into three major categories:

Prevention costs are related with activities that identify the cause of defects and those actions that are taken to prevent them,

Appraisal costs include the costs of evaluating the quality of software products at some level

Failure costs include the costs to analyze and remove the failures. failure costs can be divided into the following parts:

External failure costs are associated with failures that appear at the customer's site when the product has been released

Internal failure costs are those costs that appear on the developer's site prior to the release of the product

Quality Control(QC) and Quality Assurance(QA)

- **Quality control** focuses on finding and removing defects, whereas the main purpose of **quality assurance** is to verify that applicable procedures and standards are being followed.
- QC is used to verify the quality of the output;
- QA is the process of managing for quality.

Software Quality Control:

"The function of software quality that checks that the project follows its standards, processes, and procedures, and that the project produces the required internal and external (deliverable) products"

Software Quality Assurance:

"The function of software quality that assures that the standards, processes, and procedures are appropriate for the project and are correctly implemented"

Software Quality Management

- **‘Quality management’** whose scope is larger than quality control and quality assurance.
- It is an established way of managing the quality of a product.
- The major role of QM is to develop a quality culture in the organization.
- Quality culture means that every member of the team is aware and conscious about the quality and is working towards a high-quality end-product.
- The task of quality management is to
  - plan suitable quality control and quality assurance activities.
  - define procedures and standards which should be used during software development and verify that these are being followed by everyone.
  - properly execute and control activities.

Software Quality Metrics

- Software quality metrics are a subset of software metrics that focus on the quality aspects of the product, process, and project.
- In general, software quality metrics are more closely associated with process and product metrics than with project metrics.
- software quality metrics can be grouped into the following three categories in accordance with the software lifecycle:
    - product quality metrics
    - in-process quality metrics
    - Metrics for software maintenance.

Software Quality Metrics

Product Quality Metrics:

Mean-time to failure (MTTF) = total operating time of the units tested / total # of failures encountered.

Defect density = Number of defects / Size of product

Customer problem metrics: The problem metrics is usually expressed in terms of problems per user month (PUM)

$$PUM = \frac{Total\ problems\ reported\ by\ the\ customer\ for\ a\ time\ period}{Total\ number\ of\ licensed\ months\ of\ the\ software\ during\ the\ period}$$

where,
number of licensed months = number of installed licenses of the software / number of months in the calculation period

Software Quality Metrics

In-process Product Quality Metrics:

- ***Defect-density during testing***
- ***Defect-arrival pattern during testing***
- ***Defect-removal efficiency (DRE)***

$$DRE = \frac{Defects\ removed\ during\ the\ month}{Number\ of\ \ problem\ arrivals\ during\ the\ month} \times 100$$

Software Quality Metrics

Metrics for Software Maintenance:

Fix backlog and backlog management index: Fix backlog metrics is the count of reported problems that remain open at the end of a month or a week.

Backlog management index (BMI) is also the metric to manage the backlog of open unresolved problems.

$$BMI = \frac{Number\ of\ \ problems\ closed\ during\ the\ month}{Number\ of\ \ problem\ arrivals\ during\ the\ month} \times 100$$

- If BMI is larger than 100, it means the backlog is reduced
- if BMI is less than 100, the backlog is increased.
- the goal is to have a BMI larger than 100.

Software Quality Metrics

Metrics for Software Maintenance:

- *Fix response time and fix responsiveness*
- *Fix quality*
- *Percent delinquent fixes*

$$\textit{Percent delinquent fixes} = \frac{\textit{Number of fixes that exceeded the response time criteria by severity level}}{\textit{Number of fixes delivered in a specified time}} \times 100$$

Software Quality Assurance Model: CMM(Capability Maturity Model)

CMM

- Capability Maturity Model (CMM) is a framework that describes the key elements of an effective software process.
- It describes an evolutionary improvement path from an ad hoc, immature process to a mature, disciplined process.
- Covers practices for:
  - Planning
  - Engineering
  - Managing software development and maintenance.
- When followed, these key practices improve the ability of organizations to meet goals for cost, schedule, functionality, and product quality.

CMM Specification

- All definitions ae quoted from the SEI CMM v1.1 Specifications.
  - Capability Maturity Model (CMM)
  - Software process
  - Software process capability
  - Software process performance
  - Software process maturity

CMM Definition

- A description of the stages through which software organizations evolve as they define, implement, measure, control, and improve their software processes
- A guide for selecting process improvement strategies by facilitating:
  - determination of current process capabilities
  - identification of the issues most critical to software quality and process improvement

Software Process

- A *software process* can be defined as a set of activities, methods, practices, and transformations that people use to develop and maintain software and the associated products
  - E.g., project plans, design documents, code, test cases, and user manuals.
- As an organization matures, the software process becomes better defined and more consistently implemented throughout the organization.

Software Process Capability

- *Software process capability* describes the range of expected results that can be achieved by following a software process.
- The software process capability of an organization provides one means of predicting the most likely outcomes to be expected from the next software project the organization undertakes.
- The software process capability will be able to give us a reliable estimate on what the organization is “capable of” or the “latent potential” of an organization at a particular instance in time.

Software Process Performance

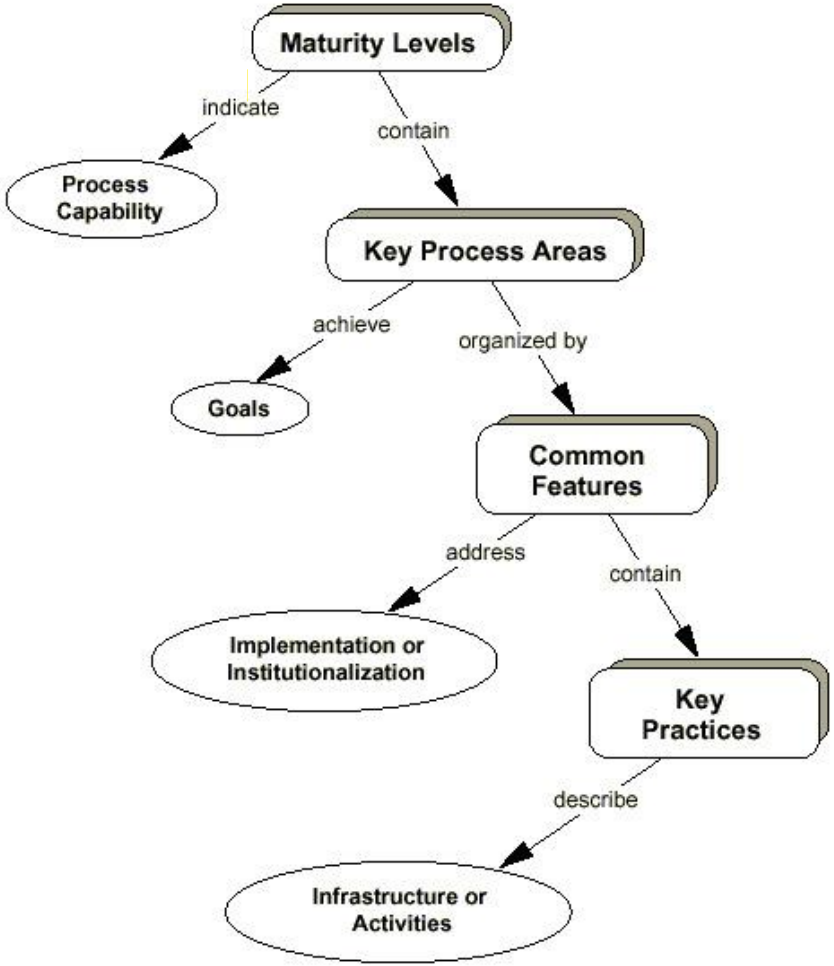
- *Software process performance* represents the actual results achieved by following a software process.
- Software process performance focuses on the results achieved, while software process capability focuses on results expected.

Software Process Maturity

- *Software process maturity* is the extent to which a specific process is explicitly defined, managed, measured, controlled, and effective.
- Maturity implies a potential for growth in capability and indicates both the richness of an organization's software process and the consistency with which it is applied in projects throughout the organization.
- The maturity of the organization is rated in the range of one to five, where "one" means that the organization is immature, or just lacks any visible process.
- Level five is the highest level of maturity, which indicates that the organization has an evolving and adapting process that learns from experience and improves on it.
- The general estimate is that nearly 50 to 60% of the software development organizations worldwide are on Level 1, which essentially means that there has been total lack of any serious process.

Structure of CMM

- The CMM is composed of five maturity levels.
- Each maturity level is composed of several key process areas (except Level 1).
- Each key process area is organized into five sections called common features.
- The common features specify the key practices that, when collectively addressed, accomplish the goals of the key process area.



CMM Maturity Level

- A maturity level is a well-defined evolutionary plateau toward achieving a mature software process.
- CMM provides for 5 top-levels:
  - Level 1- Initial
  - Level 2- Repeatable
  - Level 3- Defined
  - Level 4- Managed
  - Level 5- Optimizing

Maturity level	Process Capability
1	No process
2	Disciplined process
3	Standard consistent process
4	Predictable process
5	Continuously improving process

CMM Maturity Level - Initial

- No stable environment for developing and maintaining software
- Difficulty making commitments that the staff can meet
- Crises are common
- Projects typically abandon planned procedures and revert to coding and testing
- In spite of this ad hoc, even chaotic process, Level 1 organizations frequently develop products that work, even though they may be over the budget and schedule.
- Success depends entirely on exceptional managers and seasoned, effective software team
- Success cannot be repeated unless the same competent individuals are assigned to the next project
- Capability is a characteristic of the individuals, not the organization

CMM Maturity Level - Repeatable

- Policies for managing a software project exist
- Procedures and Standards are defined
- Planning and managing new projects is based on experience with similar projects
- Basic software management controls exist
- Realistic project commitments based prior knowledge
- The software project managers track costs, schedules, and functionality; problems in meeting commitments are identified when they arise
- The organizational requirement for achieving Level 2 is that there are policies that guide the projects in establishing the appropriate management processes.
- The software process capability of Level 2 organizations can be summarized as disciplined because planning and tracking of the software project is stable and earlier successes can be repeated.

CMM Maturity Level - Defined

- Based on a common, organization-wide understanding of the activities, roles, and responsibilities in a defined software process
- The organization exploits best practices
- Special group responsible for software process
- Projects tailor the organization's standard software process to develop their own defined software process, which accounts for the unique characteristics of a project.
- Defined software process integrating engineering and management processes like readiness criteria, inputs, standards, and procedures for performing the work, verification mechanisms, such as peer reviews, outputs, and completion criteria.
- Management has good insight into technical progress on all projects
- Cost, schedule, and functionality are under control, and software quality is tracked
- The software process is stable and reliable

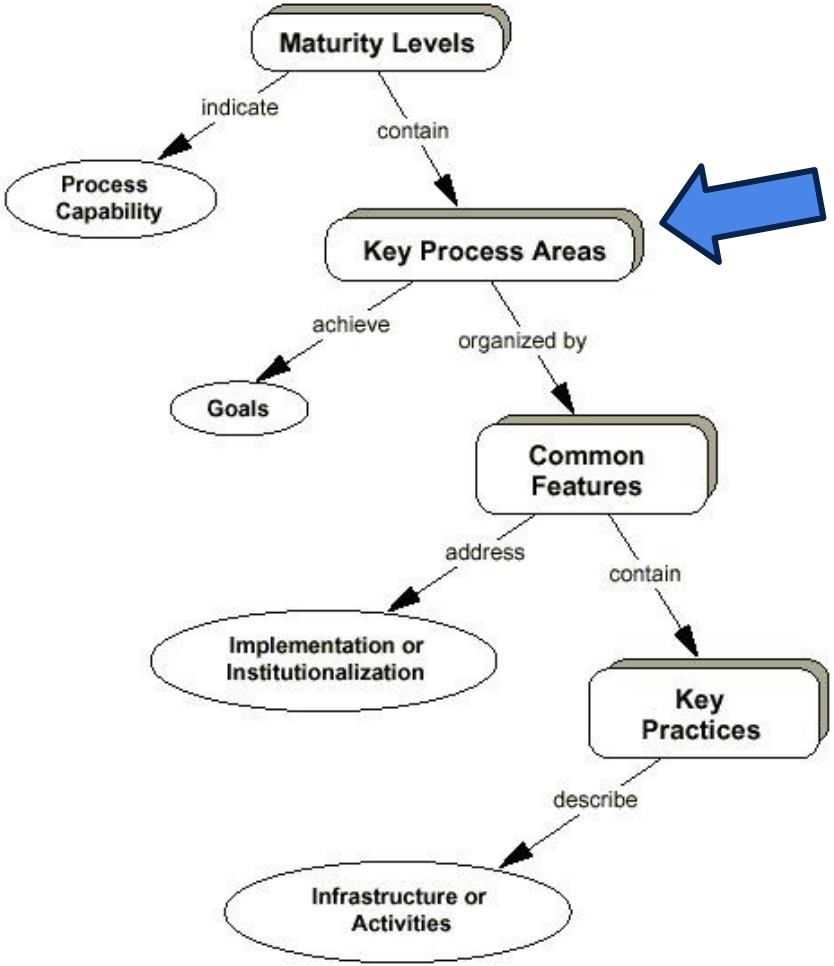
CMM Maturity Level - Managed

- Use of quantitative quality goals for both software products and processes
- Strong use of software/process metrics
- Organization-wide software process database is used
- Allows prediction of trends in process and product quality
- Process is both stable and measured
- Software products are of predictably high quality

CMM Maturity Level - Optimized

- Organization level focus on continuous process improvement
- Innovations that exploit the best software engineering practices are identified and transferred throughout the organization
- Software processes are evaluated to prevent known types of defects from recurring
- Technology and process improvements are planned and managed as ordinary business activities

Structure of CMM



Key Process Area(KPA)

- Each maturity level is composed of key process areas.
- Each key process area identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important for establishing process capability at that maturity level.
- The key process areas have been defined to reside at a single maturity level.
- For example, one of the key process areas for Level 2 is Software Project Planning.

Key Process Area(KPA)

Level 1

No process area can be observed

Level 2

- Software configuration management
- Software quality assurance
- Software project tracking and oversight
- Software project planning
- Requirements management

Level 3

- Peer reviews
- Inter-group coordination
- Software product engineering
- Integrated software management
- Training program
- Organization process definition
- Organization process focus

Key Process Area(KPA)

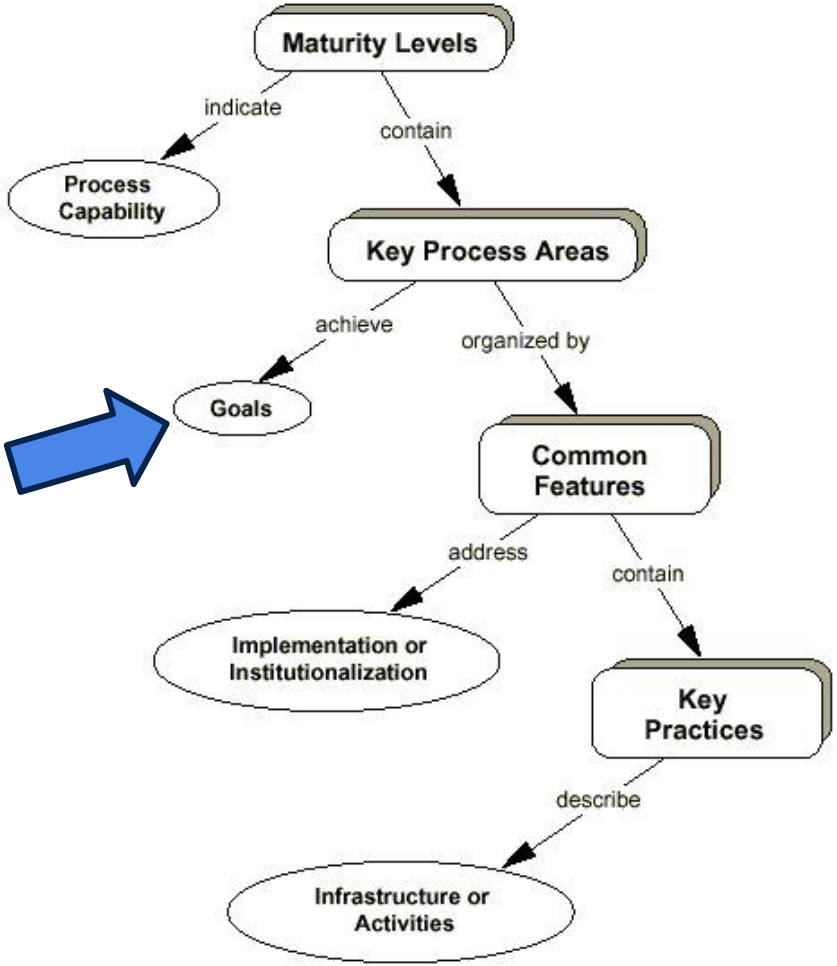
Level 4

- Quality management
- Process measurement
- Software quality management
- Quantitative process management

Level 5

- Process change management
- Technology change management
- Defect prevention

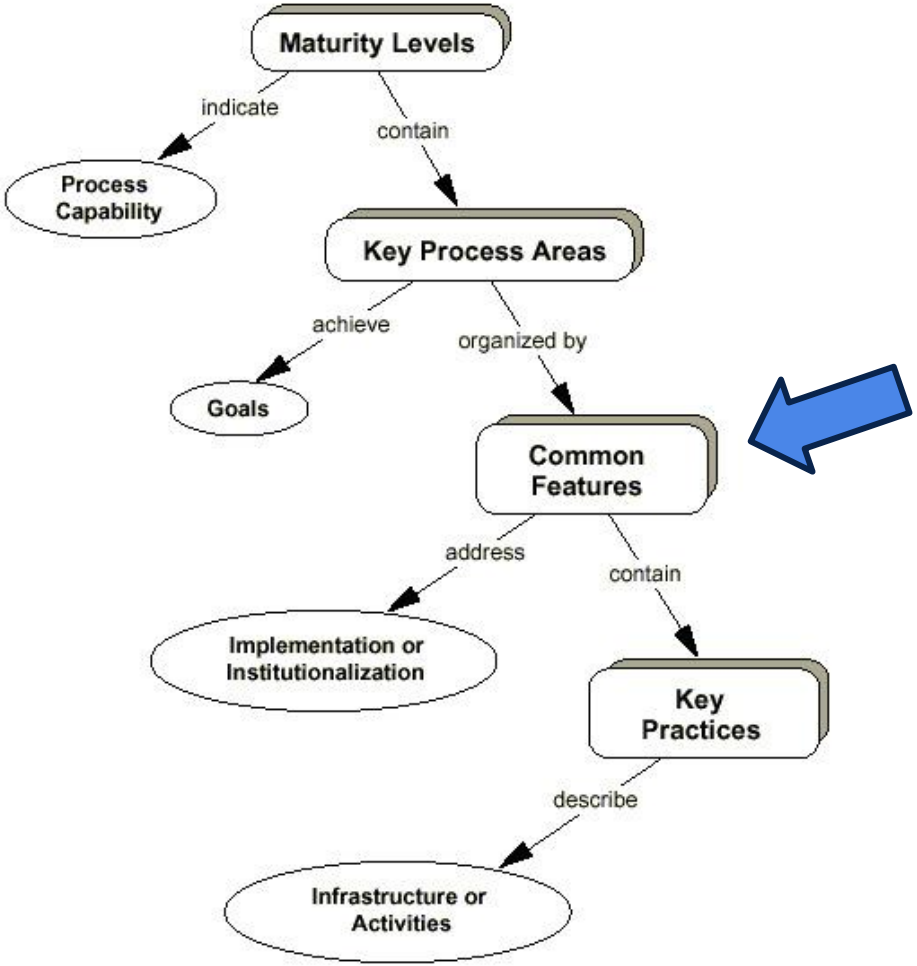
Structure of CMM



Goals

- The goals:
  - Summarize the key practices of a key process area
  - Can be used to determine whether an organization or project has effectively implemented the key process area
  - Signify the scope, boundaries, and intent of each key process area.
  - The interesting point with goals is that many software development organizations do not set measurable goals.
  - This means that the goal is never defined precisely enough to know if it has been completed.
  - Organizations that are considered mature always take time to ensure that each goal is fully and precisely defined in a way that is allows measurement
- E.g. : a goal from the Software Project Planning key process area :
  - "Software estimates are documented for use in planning and tracking the software project."

Structure of CMM



Common Features

- The common features are attributes that indicate whether the implementation and institutionalization of a key process area is effective, repeatable, and lasting.
- The key practices are divided among five Common Features sections:
  - Activities Performed (Describes Implementation Activities)
  - Commitment to Perform (Organizational culture)
  - Ability to Perform (Institutionalization factor)
  - Measurement and Analysis (Organization culture)
  - Verifying Implementation (Institutionalization factor).

Common Feature

Commitment to Perform

- Describes the actions the organization must take to ensure that the process is established and will endure.
- Typically involves establishing organizational policies and senior management sponsorship.

Ability to Perform

- Describes the preconditions that must exist in the project or organization to implement the software process competently.
- Involves resources, organizational structures, and training.

Common Feature

Activities Performed

- Describe the roles and procedures necessary to implement a key process area
- Cover what MUST be implemented to establish a process capability
- Typically involve:
  - Establishing plans
  - Procedures
  - Performing the work
  - Tracking it
  - Taking corrective actions as necessary.

Common Feature

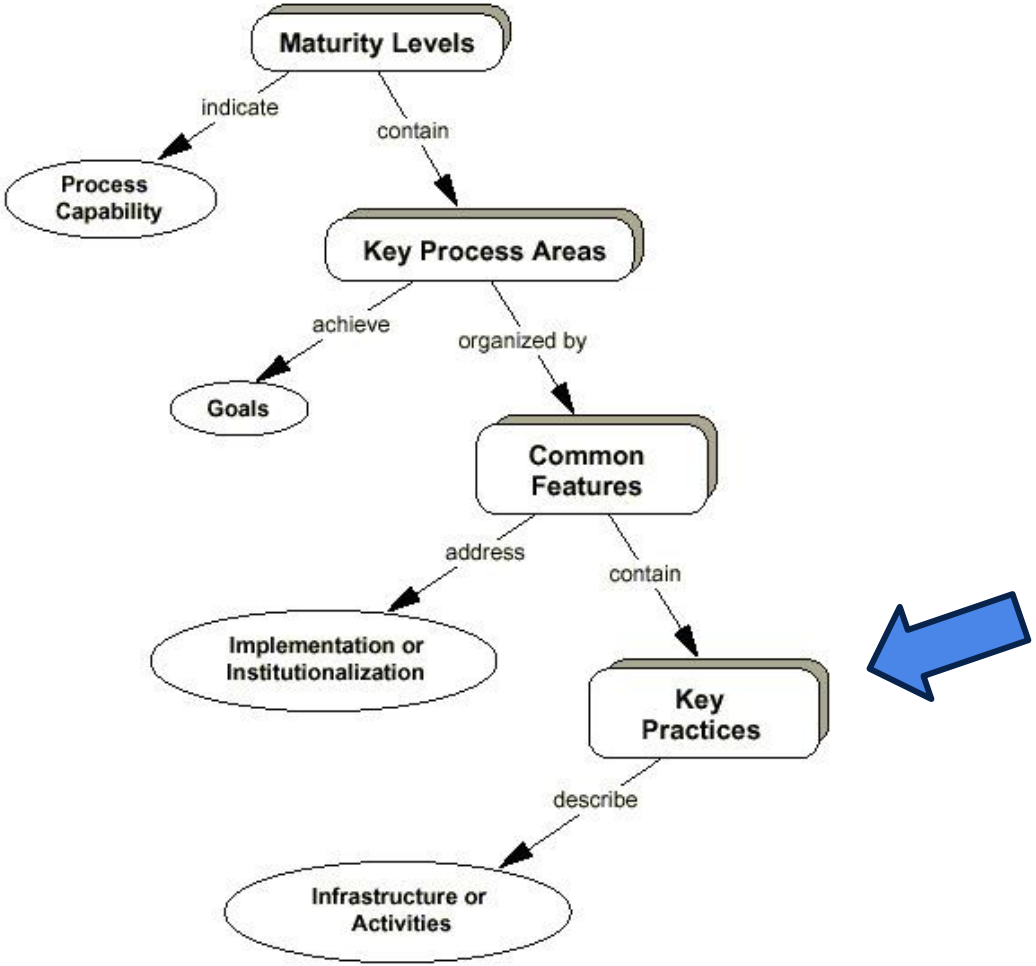
Measurement & Analysis

- Describes the need to measure the process and analyze the measurements.
- Typically includes examples of the measurements that could be taken to determine the status and effectiveness of the Activities Performed.

Verifying Implementation

- Describes the steps to ensure that the activities are performed in compliance with the process that has been established.
- Typically encompasses reviews and audits by management and software quality assurance.

Structure of CMM



Key Practices

- Each key process area is described in terms of key practices that, when implemented, help to satisfy the goals of that key process area.
- The key practices describe the infrastructure and activities that contribute most to the effective implementation and institutionalization of the key process area.
- For example, one of the practices from the Software Project Planning key process area is "The project's software development plan is developed according to a documented procedure."

End of the syllabus