

UNIT – I

SOFTWARE PROCESS MATURITY

DEFINITIONS:

Process:

A sequence of steps performed for a given purpose. The process integrates people, tools, and procedures.

Software Process:

A set of activities, methods, practices, and transformations that people employ to develop and maintain software and the associated products (documents, etc.)

Software Process Capability:

Describes the range of expected results that can be achieved by following a software process.

Software Process Performance:

The actual results achieved by following a software process.

Software Process Maturity:

The extent to which a specific process is explicitly defined, managed, measured, controlled, and effective. As a software organization matures, it needs an infrastructure and culture to support its methods, practices, and procedures so that they endure after those who originally defined them have gone.

Software Engineering:

It is a branch of both computer science and engineering focused on designing, developing, testing, and maintaining software applications.

Institutionalization:

Is the building of infrastructure and culture to support methods, practices, and procedures so that they are the ongoing way of doing business.

CMM (capability maturity model) :

The CMM focuses on the capability of software organizations to produce high- quality products consistently and predictably. Software process capability is the inherent ability of a software process to produce planned results.

SOFTWARE PROCESS MATURITY FRAMEWORK

Software maturity framework has five levels. They are

Initial:

The software process is characterized by ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort and heroics.

Repeatable (Managed):

Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

Defined:

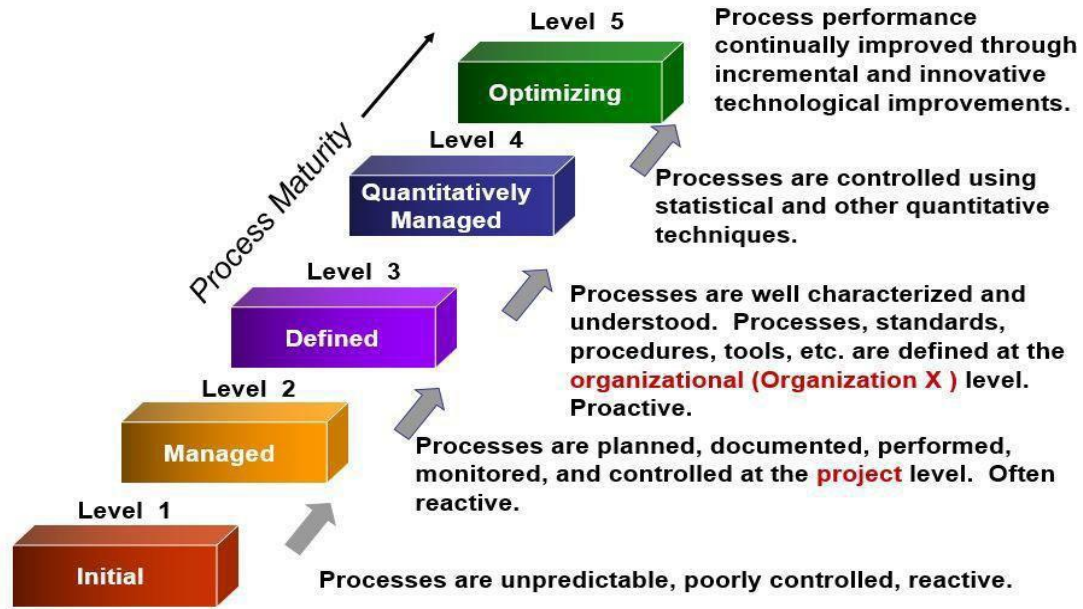
The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization's standard software process for developing and maintaining software.

Managed (Quantitatively Managed):

Detailed measures of the software process and product quality are collected. Both the software process and products are quantitative understood and controlled.

Optimizing:

Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.



PRINCIPLES OF SOFTWARE PROCESS CHANGE

Basic Principles:

- Automation of a poorly defined process will produce automation of poorly defined results
- Improvement should be made in some steps
- Educate/Train, Educate/Train, Educate/Train

Software process management has 2 key areas . They are

1. People
2. Design methods

People:

A good mix of talent is required. The best people are always in short supply. You probably have about the best team you can get right now. With proper leadership, education, training and support, most people can do better work than they are currently doing.

Design:

When domain knowledge is combined with the ability to produce a good design, a quality product will result.

Six Basic Principles of Software Process Change

1. Major changes to the software process must start at the top:

- Major changes requires leadership. Managers must provide good leadership, even though they may not do the work; they must set priorities: furnish resources and provide continuing support.

2. Ultimately, everyone must be involved:

- With an immature software process, software professionals are forced to improvise solutions. in a mature process, these individual actions are more structured, efficient and reinforcing. People are the most important aspect. It's necessary to focus on repairing the process and not the people.

3. Effective changes require the team to have common goals and knowledge of the current process:

- An effective change program requires a reasonable understanding of the current status. An assessment is an effective way to gain this understanding. Software professionals generally need most help in controlling requirements., coordinating changes, making plans. managing interdependencies and coping with system design issues. Software process and project management

4. Change is continuous:

- One of the most difficult things for a management team to recognize is that human interactive processes are never static. Both problems and people are in constant flux, and this fluidity all for periodic adjustment of tasks and relationships. In dealing with these dynamics, 3 points are important.

- Relative changes generally make things worse
- Every defect is an improvement opportunity
- Crisis prevention is more important than crisis recovery

5. Software process changes will not be retained without conscious effort and periodic reinforcements :

- Precise and accurate work is hard. Its rarely sustained for long without reinforcement.

Human adoption of new process methods involves 4 stages

- Installation, Practice , Proficiency , Naturalness

6. Software process improvement requires investment :

- While the need for dedicating resources to improvement seems self-evident it's surprising how often managers rely on exhorting their people to try harder
- To improve the aft rare process someone must work on it
- Unplanned process improvement is wishful thinking.

SOFTWARE PROCESS ASSESSMENT

A software process assessment is a disciplined examination of the software processes used by an organization, based on a process model. The assessment includes the identification and characterization of current practices, identifying areas of strengths and weaknesses, and the ability of current practices to control or avoid significant causes of poor (software) quality, cost, and schedule.

A software assessment (or audit) can be of three types.

1. A **self-assessment (first-party assessment)** is performed internally by an organization's own personnel.
2. A **second-party assessment** is performed by an external assessment team or the organization is assessed by a customer.
3. A **third-party assessment** is performed by an external party or (e.g., a supplier being assessed by a third party to verify its ability to enter contracts with a customer).

Software process assessments are performed in an open and collaborative environment. They are for the use of the organization to improve its software processes, and the results are confidential to the organization. The organization being assessed must have members on the assessment team.

Software Process Maturity Assessment

Software Process Maturity Assessment is the process of evaluating the capability and maturity of an organization's software processes. It determines how effectively the software development process is implemented and identifies areas for improvement.

Scope of Software Process Maturity Assessment

The scope of a software process assessment can cover:

1. **Entire Organization**
 - All software processes in the organization are assessed.
2. **Selected Subset (Department/Process Area)**
 - Only a particular department or selected software processes are assessed.
3. **Specific Project**
 - Assessment is limited to a single software project.

Process of Software Process Maturity Assessment

Step 1: Select the Assessment Scope

Determine what is to be assessed:

- Entire organization
- Selected department/process area
- Specific project

Step 2: Conduct the Assessment

The assessment team:

- Reviews how software processes are performed.
- Examines project documents and project results.
- Collects process data and feedback from employees.

Step 3: Analyze the Assessment Data

The collected information is analyzed to:

- Determine the maturity level of the software process.
- Identify process strengths and weaknesses.
- Evaluate process effectiveness.
- Identify areas that require improvement.

Step 4: Prepare the Assessment Report

The final report includes:

- Current maturity level
- Strengths of the process
- Weaknesses of the process
- Recommendations for process improvement

Why Do Assessment Results Differ?

When the assessment target is the organization, the results may differ even if the same assessment method is used.

Reason 1: Different Assessment Scope

Reason 2: Different Project Selection

Importance of Software Process Maturity Assessment

- Measures the maturity of software processes.
- Identifies strengths and weaknesses.
- Improves software quality and productivity.
- Supports continuous process improvement.
- Helps organizations achieve long-term process improvement goals.
- Provides an objective basis for process improvement planning.

Software Process Assessment Cycle

According to Paulk and colleagues (1995), the CMM-based assessment approach uses a six-step cycle. They are

- **Select a team :** The members of the team should be professionals knowledgeable in software engineering and management.
- **Fill the questionnaire :** The representatives of the site to be appraised complete the standard process maturity questionnaire.
- **Analyze the answers :** The assessment team performs an analysis of the questionnaire responses and identifies the areas that warrant further exploration according to the CMM key process areas.
- **Site visit :** The assessment team conducts a site visit to gain an understanding of the software process followed by the site.
- **List the findings :** The assessment team produces a list of findings that identifies the strengths and weakness of the organization's software process.
- **Prepare KPA :** The assessment team prepares a Key Process Area (KPA) profile analysis and presents the results to the appropriate audience.

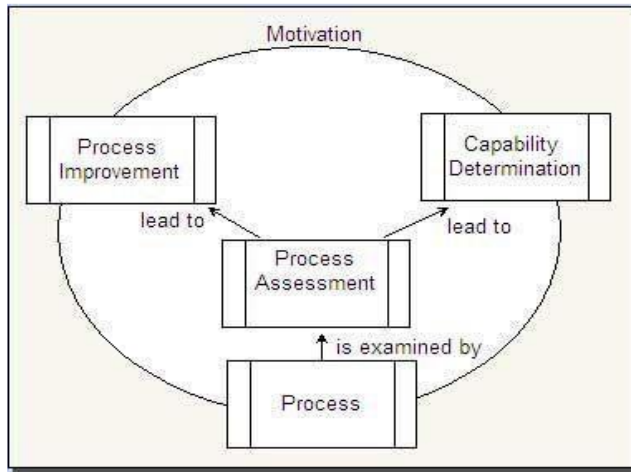
FOR EXAMPLE

The assessment team must be led by an authorized SEI Lead Assessor. The team must consist of between four to ten team members. At least, one team member must be from the organization & three members from internal organization being assessed, and all team members must complete the SEI's

Introduction to the CMM course (or its equivalent) and the SEI's CBA IPI team training course. Team members must also meet some selection guidelines.

CBAIPI : CMM based assessment method for internal process improvement

PROCESS ASSESSMENT CYCLE



Process Assessment Cycle

The **Process Assessment Cycle** is a continuous process of evaluating an organization's software process, determining its capability, identifying weaknesses, and improving the process to achieve better quality and productivity.

Process

The **process** is the software development method followed by an organization (requirements, design, coding, testing, maintenance, etc.).

Process Assessment

The existing software process is **examined by an assessment team** to evaluate its strengths, weaknesses, effectiveness, and compliance with standards.

Capability Determination

Based on the assessment results, the organization's **process capability or maturity level** is determined. It shows how effectively the organization can develop and manage software projects.

Example: CMMI Maturity Levels – **Level 1, Level 2, Level 3, Level 4, Level 5.**

Process Improvement

Based on the assessment findings, recommendations are made to remove weaknesses, improve quality, increase productivity, and enhance the software development process.

Motivation

The successful implementation of improvements and knowledge of the organization's capability motivate continuous process improvement. Therefore, the assessment cycle is repeated regularly.

SCAMPI (Standard CMMI Assessment Method for Process Improvement)

SCAMPI (Standard CMMI Appraisal Method for Process Improvement) is the official appraisal method used to evaluate an organization's software processes. It helps identify the strengths and weaknesses of the organization's processes and provides recommendations for continuous process improvement.

SCAMPI was developed by the **Software Engineering Institute (SEI)**, in **2000**. It replaced the earlier appraisal method called **CBA IPI (CMM-Based Appraisal for Internal Process Improvement)** and became the standard appraisal method for CMMI.

Need for SCAMPI

SCAMPI is used to determine whether an organization's software processes follow the best practices defined by CMMI. It helps organizations improve software quality, increase productivity, reduce project risks, achieve higher customer satisfaction, and continuously improve their development processes.

SCAMPI consist of three phases

1. Plan and preparation
2. Conduct the assessment onsite
3. Report results

The activities for the plan and preparation phase include –

PHASE-1 Plan and Preparation :

The purpose of this phase is to prepare all the activities required before conducting the actual appraisal. Proper planning ensures that the appraisal is organized and successful.

1. Identify the assessment scope
2. Develop the assessment plan
3. Prepare and train the assessment team
4. Make a brief assessment of participants
5. Administer the CMMI Appraisal Questionnaire
6. Examine the questionnaire responses
7. Conduct an initial document review

1. Identify the Assessment Scope

The first activity is to determine the scope of the assessment. The organization decides which business unit, department, project, or product will be included in the assessment.

2. Develop the Assessment Plan

After defining the scope, the assessment team prepares a detailed assessment plan. The plan includes the schedule, responsibilities, resources, documents to be reviewed, interviews to be conducted, and the overall assessment activities. This plan acts as a roadmap for the entire assessment process.

3. Prepare and Train the Assessment Team

The assessment team is selected and trained on CMMI practices, SCAMPI procedures, interview techniques, document review, and evidence collection methods.

Proper training ensures that every team member performs the assessment consistently and accurately.

4. Prepare the Participants

The organization identifies the employees who will participate in the assessment, such as project managers, software developers, testers, quality assurance engineers, and other stakeholders.

Participants are informed about the assessment objectives and are made available for interviews and discussions.

5. Administer the CMMI Appraisal Questionnaire

A standard CMMI appraisal questionnaire is distributed to the selected participants to collect information about how software processes are implemented within the organization.

The questionnaire provides preliminary evidence for the appraisal

6. Examine the Questionnaire Responses

The assessment team reviews all questionnaire responses to understand the current software processes.

The responses help identify strengths, weaknesses, and possible problem areas that require further investigation.

7. Conduct an Initial Document Review

The assessment team reviews important documents such as project plans, requirements documents, risk management plans, quality assurance reports, configuration management records, and process documents.

This review verifies whether the documented processes comply with CMMI practices.

THE ACTIVITIES FOR THE ONSITE ASSESSMENT PHASE INCLUDE –

PHASE-2 Conduct the Assessment (On-site Activities)

This phase involves collecting evidence directly from the organization through meetings, interviews, observations, and document verification.

- Conduct an opening meeting
- Conduct interviews
- Consolidate information
- Prepare the presentation of draft findings
- Present the draft findings
- Consolidate, rate, and prepare the final findings

1. Conduct an Opening Meeting

The assessment begins with an opening meeting involving management and the assessment team.

The purpose, objectives, assessment schedule, and expected outcomes are explained to all participants.

2. Conduct Interviews

The assessment team interviews project managers, developers, testers, quality engineers, and other employees to understand how software processes are actually performed.

The interviews verify whether the documented processes are followed in practice.

3. Consolidate Information

The assessment team collects information from interviews, questionnaires, observations, and document reviews.

All evidence is organized and analyzed together to obtain a complete understanding of the organization's software processes.

4. Prepare Draft Findings

Based on the collected evidence, the assessment team prepares draft findings.

These findings identify the strengths, weaknesses, and improvement opportunities but are not yet finalized.

5. Present the Draft Findings

The draft findings are presented to the organization's management.

Management reviews the findings, provides clarification, and submits additional evidence if required.

6. Consolidate, Rate, and Prepare Final Findings

After reviewing all evidence and feedback, the assessment team finalizes the findings.

Each process area is rated according to CMMI requirements, and the final assessment report is prepared.

PHASE 3: Report Results

This is the final phase of the SCAMPI assessment process.

1. Present Final Results

The assessment team presents the final report to senior management.

The report includes the appraisal results, strengths, weaknesses, ratings, conclusions, and recommendations for process improvement.

2. Conduct an Executive Feedback Session

The assessment team discusses the appraisal results with management.

During this session, management can ask questions, clarify doubts, understand the findings, and discuss future improvement plans.

3. Wrap Up the Assessment

The assessment process is formally closed after submitting the final report.

The organization begins implementing the recommended improvements to enhance its software development processes.

1. INITIAL PROCESS

At this level, the organization does not follow any standard software development process. Work is done randomly (ad hoc), so project success depends mainly on the skills of individual developers.

Characteristics of Level 1

- No standard process
- No planning
- No cost estimation
- No project tracking
- Weak management
- Poor quality
- High risk
- Success depends on individual developers

Move from Level 1 to Level 2

1. Project Management

- Plan the project.
- Assign tasks.
- Create schedules.

2. Management Oversight

- Managers regularly monitor progress.
- Solve issues quickly.

3. Quality Assurance (QA)

- Check whether work follows standards.
- Improve software quality.

4. Change Control

- Record every client change.
- Get approval before implementing changes.

2. REPEATABLE PROCESS

At **Level 2**, the organization starts following **basic project management practices**.

Projects are **planned, tracked, and controlled**, so similar projects can be repeated successfully.

The main characteristics are:

1. **Projects are planned** before development starts.
2. **Project schedules and budgets** are prepared.
3. **Roles and responsibilities** are clearly assigned.
4. **Project progress is monitored** regularly.
5. **Requirements are controlled** and changes are managed.
6. **Quality Assurance (QA)** ensures the project follows the planned process.
7. **Configuration Management** controls versions of documents and source code.
8. **Successful practices are repeated** in similar projects using previous experience.

Move from Level 2 to Level 3

1. Establish a Process Group

- Create a dedicated team to improve software processes.
- Their job is to define and improve standard processes.

2. Define a Standard Software Development Process

- Create one standard SDLC for all projects.
- Every team follows the same process.

3. Document All Processes

- Write clear procedures, guidelines, templates, and standards.
- New employees can easily understand the process.

4. Introduce Better Software Engineering Practices

- Use design reviews.
- Perform code inspections.
- Follow proper testing methods.
- Use version control tools.

5. Train Employees

- Train all team members on the organization's standard process.
- Ensure everyone follows the same way of working.

3. DEFINED PROCESS

Level 3 (Defined Process) means the organization has standard, documented processes that every project follows.

Characteristics of level-3

1. Standard Process

- The company creates one standard software development process.
- Every project follows it.

2. Process is Documented

- Every activity is written in documents.
- Employees know exactly what to do.

3. Everyone Follows the Same Process

- Even during emergencies, the team does not skip the process.

4. Focus on Process Quality

- The organization checks whether the process is being followed.

- But it does not yet measure performance using numbers.

Moving from Level 3 → Level 4

Step 1: Establish Process Measurements

Measure every process using numbers.

Measure:

- Cost
- Time
- Quality
- Number of defects

Step 2: Create a Process Database

Store all project data in one central database.

Store information like:

- Project cost
- Defects
- Time taken
- Productivity

Step 3: Assign Skilled Professionals

Experts check:

- Data accuracy
- Data quality
- Correct analysis

Usually done by the **Quality Assurance (QA)** team.

Step 4: Assess Product Quality

An **independent QA team** checks whether:

- Product quality meets the target.

- Management is informed if quality is poor.

4. MANAGED PROCESS

At **Level 4**, the organization **measures and controls the software process using data and metrics**.

Characteristics

1. **Process is measured using metrics**
 - Time, cost, defects, and productivity are measured.
2. **Decisions are based on data**
 - Management uses facts and measurements instead of assumptions.
3. **Process performance is controlled**
 - The process is monitored to keep quality and schedule under control.
4. **Quality is continuously monitored**
 - Defects and project performance are tracked throughout development.
5. **Data is used to improve the process**
 - Measurements help identify problems and make the software process better.

Moving from CMM Level 4 to Level 5

Automatic Data Collection

- Use software tools to collect process data automatically.
- Automatic collection is faster, more accurate, and reduces human errors.

Use Data for Continuous Improvement

- Analyze the collected data.
- Find the root cause of problems.
- Modify and improve the process to prevent future issues.

5.OPTIMIZING

At **Level 5 (Optimizing)**, the organization continuously improves its software development process. It uses data, feedback, and new technologies to prevent defects and make the process better, faster, and cheaper.

Characteristics of Level 5

- Continuous process improvement.
- Prevent defects instead of only fixing them.
- Root cause analysis of every major defect.
- Use of new technologies and automation.
- Decisions are based on data and measurements.
- Focus on reducing cost, time, and defects.

PROCESS REFERENCE MODELS

Capability Maturity Model(CMM):

What is CMM?

Capability Maturity Model is used as a benchmark to measure the maturity of an organization's software process.

Capability Maturity Model (CMM) is a **process improvement model** developed by the **Software Engineering Institute (SEI)** in the **mid-1980s**.

It helps software organizations **improve their software development process step by step**.

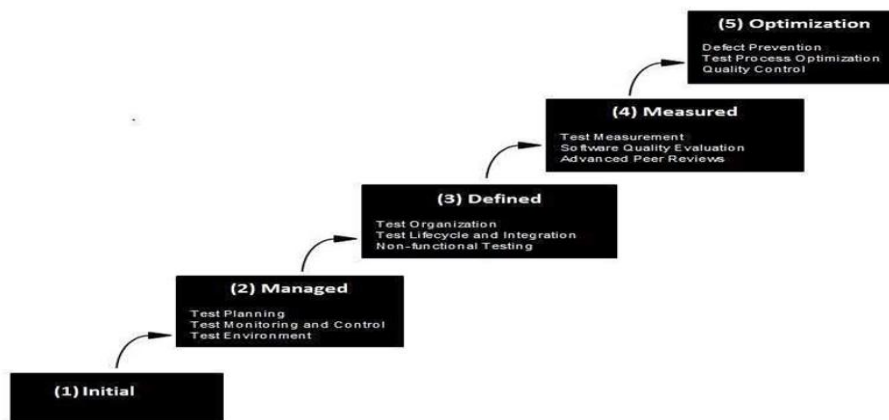
Why is CMM needed?

Without a proper process:

- Projects get delayed.
- Cost increases.
- More defects occur.
- Customer becomes unhappy.

CMM provides a standard way to improve the software process.

Levels of CMM



Level One :Initial –

The software process is characterized as inconsistent, and occasionally even chaotic. Defined processes and standard practices that exist are abandoned during a crisis. Success of the organization majorly depends on an individual effort, talent, and heroics.

• Level Two: Repeatable –

This level of Software Development Organization has a basic and consistent project management processes to track cost, schedule, and functionality. The process is in place to repeat the earlier successes on projects with similar applications. Program management is a key characteristic of a level two organization.

• Level Three: Defined –

The software process for both management and engineering activities are documented, standardized, and integrated into a standard software process for the entire organization and all projects across the organization use an approved, tailored version of the organization's standard software process for developing, testing and maintaining the application.

- **Level Four: Managed –**

Management can effectively control the software development effort using precise measurements. At this level, organization set a quantitative quality goal for both software process and software maintenance. At this maturity level, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable.

- **Level Five: Optimizing –**

The Key characteristic of this level is focusing on continually improving process performance through both incremental and innovative technological improvements. At this level, changes to the process are to improve the process performance and at the same time maintaining statistical probability to achieve the established quantitative process-improvement objectives.

Capability Maturity Model Integration (CMMI)

CMMI is a framework that combines different best practices into one model to improve quality, productivity, and customer satisfaction.

Why was CMMI introduced?

Earlier, organizations had **different CMM models** for different areas.

For example:

- Software development → SW-CMM
- Systems engineering → SE-CMM
- Product development → IPD-CMM

Using multiple models was difficult because:

- Different rules
- Different assessments
- More confusion
- More cost

So, **CMMI combined all these models into one integrated framework.**

What does CMMI do?

CMM Integration has three main purposes:

1. Builds one integrated model
2. Improves best practices
3. Supports future improvements

Objectives of CMMI

Quality products

Value for stakeholders

Customer satisfaction

CMMI DISCIPLINES

CMMI includes **four disciplines**.

1. Systems Engineering (SE)

Systems Engineering focuses on developing **complete systems**, including both hardware and software.

It converts customer needs into a complete product.

2. Software Engineering (SW)

Software Engineering focuses only on software development.

Activities include:

- Design
- Coding
- Testing
- Maintenance

3. Integrated Product and Process Development (IPPD)

IPPD encourages all departments to work together from the beginning.

Teams include:

- Developers
- Testers
- Designers
- Customers

- Managers

Everyone collaborates throughout the project.

4. Supplier Sourcing (SS)

Sometimes companies buy products or services from external suppliers.

Supplier Sourcing helps organizations:

- Select good suppliers
- Monitor supplier work
- Ensure quality before delivery

CMMI DISCIPLINE SELECTION

Organizations choose the discipline based on what they want to improve.

1. Choose Systems Engineering (SE)

If you want to improve:

- Configuration Management
- Quality Assurance
- Risk Management
- Project Monitoring
- Measurement & Analysis

2. Choose IPPD

If you want to improve:

- Team collaboration
- Process integration
- Organizational coordination

3. Choose Supplier Sourcing (SS)

If you want to improve:

- Supplier selection
- Supplier management
- Vendor quality

4. Improve Multiple Disciplines

If the organization wants overall improvement, it should implement all relevant disciplines together.

CMMI REPRESENTATIONS

A representation in CMMI is a way of organizing and improving software processes.

CMMI provides two representations:

1. Staged Representation
2. Continuous Representation

Staged Representation

Staged Representation improves the entire organization step by step using predefined maturity levels.

Organizations must follow the levels in order.

Level 1 → Level 2 → Level 3 → Level 4 → Level 5

You cannot skip levels.

Continuous Representation

Continuous Representation allows an organization to improve only the process areas it needs.

There is no fixed order.

Organizations choose which process area to improve first.

Staged Representation	Continuous Representation
Uses Maturity Levels	Uses Capability Levels
Improves the whole organization	Improves one process area
Fixed sequence (Level 1 → 5)	No fixed sequence
One maturity rating	Separate capability rating for each process area
Less flexible	More flexible

STRUCTURE OF CMMI

CMMI consists of:

- 1. Maturity Levels / Capability Levels**
- 2. Process Areas (PAs)**
- 3. Goals**
 - Specific Goals
 - Generic Goals
- 4. Practices**
 - Specific Practices
 - Generic Practices
- 5. Common Features**

1. Maturity Levels

Used in **Staged Representation**.

They measure the maturity of the **entire organization**.

There are **5 maturity levels**.

Example:

Level 1 → Initial

↓

Level 2 → Managed

↓

Level 3 → Defined

↓

Level 4 → Quantitatively Managed

↓

Level 5 → Optimizing

Capability Levels

Used in **Continuous Representation**.

They measure the capability of **one specific process area**.

Levels range from **0 to 5**.

Level	Name	Easy Meaning
0	Incomplete	Work not done properly
1	Performed	Work is completed
2	Managed	Work is planned and controlled
3	Defined	Standard process for everyone
4	Quantitatively Managed	Process controlled using data
5	Optimizing	Continuous improvement

2.Process Areas (PAs)

A **Process Area** is a group of related activities that improve one part of software development.

3. Goals

Goals tell **what should be achieved**.

There are two types.

Specific Goals (SG)

These are unique to a process area.

Generic Goals (GG)

These apply to **all process areas**.

4. Practices

Practices explain **how to achieve the goals**.

There are two types.

Specific Practices (SP)

Activities for achieving Specific Goals.

Generic Practices (GP)

Activities followed in every process area.

5. Common Features

Common Features organize the Generic Practices.

They include:

- Commitment to Perform
- Ability to Perform
- Directing Implementation

CMMI LEVELS

A Maturity Level shows how mature (well organized) a software organization's development process is.

There are 5 maturity levels.

- Each level improves the organization's process.
- An organization should move Level 1 → Level 2 → Level 3 → Level 4 → Level 5.
- Levels should not be skipped

Level	Focus	Simple Meaning
1 Initial	No process	Work depends on individuals
2 Managed	Project Management	Projects are planned and monitored
3 Defined	Standard Process	Entire organization follows the same process
4 Quantitatively Managed	Measurement	Decisions are based on metrics and statistics
5 Optimizing	Continuous Improvement	Continuously improve process and quality

PEOPLE CAPABILITY MATURITY MODEL (PCMM)

People Capability Maturity Model (P-CMM) is a framework used to **improve the skills, knowledge, performance, and management of employees** in an organization.

Why is P-CMM needed?

A company cannot succeed with technology alone.

It needs:

- Skilled People
- Good Processes
- Technology

When these three work together, the company produces **high-quality products and services**.

OBJECTIVES OF P-CMM

P-CMM helps an organization to:

- Improve employee skills
- Develop workforce capability
- Standardize HR practices
- Improve teamwork
- Increase employee motivation
- Support continuous improvement

PROCESS MATURITY RATING

The maturity rating shows **how well an organization manages its employees**.

It starts from:

Ad-hoc work



Defined HR practices



Managed workforce



Continuously improving workforce

THREE IMPORTANT FACTORS (3 PS)

According to P-CMM, success depends on three things:

1. People

Employees should have knowledge, skills, and motivation.

2. Process

Work should follow proper procedures.

3. Technology (Products)

Use appropriate tools and technology.

P-CMM consists of **five maturity levels**, and each level improves workforce practices step by step.

LEVEL 1 – INITIAL

At this level, there is no formal or standardized HR process. Workforce practices are informal, and project success mainly depends on the skills and efforts of individual employees rather than organizational processes. Recruitment, training, and performance evaluation are inconsistent.

Characteristics

- No standard HR policies.
- Success depends on individuals.
- Limited training and development.
- Employee performance is inconsistent.
- Workforce management is unpredictable.

LEVEL 2 – MANAGED (REPEATABLE)

At this level, the organization introduces basic workforce management practices. Standard procedures are established for hiring, training, performance evaluation, communication, and employee compensation. The main goal is to bring discipline and consistency into workforce management.

Key Process Areas (KPAs)

Staffing: Recruit qualified employees through a proper hiring process.

Work Environment: Provide a safe and comfortable workplace with required facilities.

Communication and Coordination: Improve communication and teamwork among employees.

Performance Management: Evaluate employee performance using regular appraisals.

Training and Development: Provide technical and professional training to improve employee skills.

Compensation: Reward employees fairly through salaries, incentives, and promotions.

characteristics

Organizations follow proper recruitment procedures, provide training, measure employee performance, and maintain a good working environment.

LEVEL 3 – DEFINED

At this level, HR practices become standardized across the organization. The focus shifts to developing employee competencies, career growth, teamwork, and organizational learning. Employees are assigned work according to their skills.

Key Process Areas (KPAs)

Competency Analysis: Identify the skills required for each job role.

Workforce Planning: Estimate future manpower requirements.

Competency Development: Improve employee skills through training and certifications.

Career Development: Provide clear career growth opportunities.

Competency-Based Practices: Assign work based on employee skills and competencies.

Participatory Culture: Encourage teamwork, knowledge sharing, and employee participation.

Characteristics

- Standard HR policies across the organization.
- Career planning and skill development.
- Better teamwork and collaboration.
- Employees work according to their competencies.

LEVEL 4 – PREDICTABLE (MANAGED)

At this level, employee and team performance is measured using quantitative data. Organizations use performance metrics to monitor workforce capability, improve productivity, and make better management decisions.

Key Process Areas (KPAs)

Mentoring: Senior employees guide junior employees.

Team Building: Develop strong and effective teams.

Team-Based Practices: Encourage collaborative work and shared responsibilities.

Organizational Competency Management: Manage employee skills across the organization.

Quantitative Performance Management: Measure productivity, quality, and performance using metrics.

Organizational Performance Alignment: Align employee goals with organizational objectives.

CHARACTERISTICS

Organizations make data-driven decisions, measure workforce performance, improve teamwork, and achieve higher productivity.

LEVEL 5 – OPTIMIZING

This is the highest maturity level. The organization focuses on continuous improvement, innovation, coaching, and lifelong learning. Employees are encouraged to develop new skills and contribute innovative ideas for organizational growth.

Key Process Areas (KPAs)

Personal Competency Development: Employees continuously learn new technologies and improve their skills.

Coaching: Experienced employees regularly mentor and support others.

Continuous Workforce Innovation: Encourage innovative ideas, automation, and process improvements.

CHARACTERISTICS

- Continuous learning and skill enhancement.
- Innovation and creativity.
- Continuous process improvement.
- High employee satisfaction and organizational growth.

SUMMARY OF P-CMM MATURITY LEVELS

Level	Name	Main Focus
1	Initial	No formal HR practices; success depends on individuals.
2	Managed (Repeatable)	Basic HR practices such as staffing, training, performance management, and compensation.
3	Defined	Standard HR processes, competency development, career growth, and teamwork.
4	Predictable (Managed)	Quantitative measurement and management of workforce performance.
5	Optimizing	Continuous improvement, innovation, coaching, and continuous learning.

PERSONAL SOFTWARE PROCESS (PSP)

Personal Software Process (PSP) is a **structured software development process** developed by **Watts S. Humphrey** at the **Software Engineering Institute (SEI)**.

Why was PSP introduced?

PSP was introduced to solve common problems faced by software developers by teaching them to:

- 1. Plan Properly**
- 2. Estimate Accurately**
- 3. Track Their Work**
- 4. Improve Continuously**

Objectives of PSP

1. Improve Planning Skills – Plan the work properly.
2. Improve Estimation Skills – Estimate time and effort accurately.
3. Make Realistic Commitments – Promise only what can be completed.
4. Reduce Software Defects – Find and fix bugs early.
5. Improve Software Quality – Deliver better software.
6. Increase Productivity – Complete more work efficiently.
7. Continuously Improve Personal Performance – Learn from every project and improve.

PSP Process

The **Personal Software Process (PSP)** is a disciplined process that helps an individual software engineer develop high-quality software by following a fixed sequence of activities.

Input to PSP

SRS (Software Requirements Specification) is the **input** to the PSP.

- **SRS** is a document that clearly explains **what the software should do**.
- It contains:
 - Functional requirements (features)
 - Non-functional requirements (performance, security, etc.)
 - Constraints
 - User needs

Flow:

SRS → Planning → Design → Coding → Compile → Testing → Postmortem

PSP LEVELS

Level	Name	Main Focus	New Features
PSP0	Baseline personal process (Process discipline)	Measure current performance	Planning, design, coding, testing, time recording, defect recording, LOC
PSP0.1	Process improvement	Improve process	Coding standards, size measurement, Personal Improvement Plan (PIP)
PSP1	Personal planning process	Better estimation	Size estimation, time estimation, effort estimation, test report
PSP1.1	Task planning and schedule planning	Better planning	Task planning, schedule planning, improved time estimation
PSP2	Personal quality management	Improve quality	Design review, code review
PSP2.1	Better software design	Better design	Design templates, design specifications, design analysis

PSP Data Collection

PSP (Personal Software Process) data collection means recording information about your work in every project. This historical data helps you estimate and improve future projects.

PSP uses 4 elements for data collection

1. Scripts
 - Step-by-step instructions for doing each activity.
 - Example: Planning → Design → Coding → Testing.
2. Measures
 - The data that is recorded.
 - Example: Time spent, program size, defects.
3. Standards
 - Rules or guidelines to follow while developing software.
 - Example: Coding standards, design standards.
4. Forms
 - Templates used to record the collected data.
 - Example: Time Log, Defect Log, Size Log.

PSP Measures

PSP measures are the values recorded to evaluate a software engineer's work and improve future performance.

Four Main PSP Measures

1. Size

- Measures how big the software is.
- Usually measured in Lines of Code (LOC).

2. Effort

- Measures how much time is spent on each activity.
- Recorded in minutes or hours.

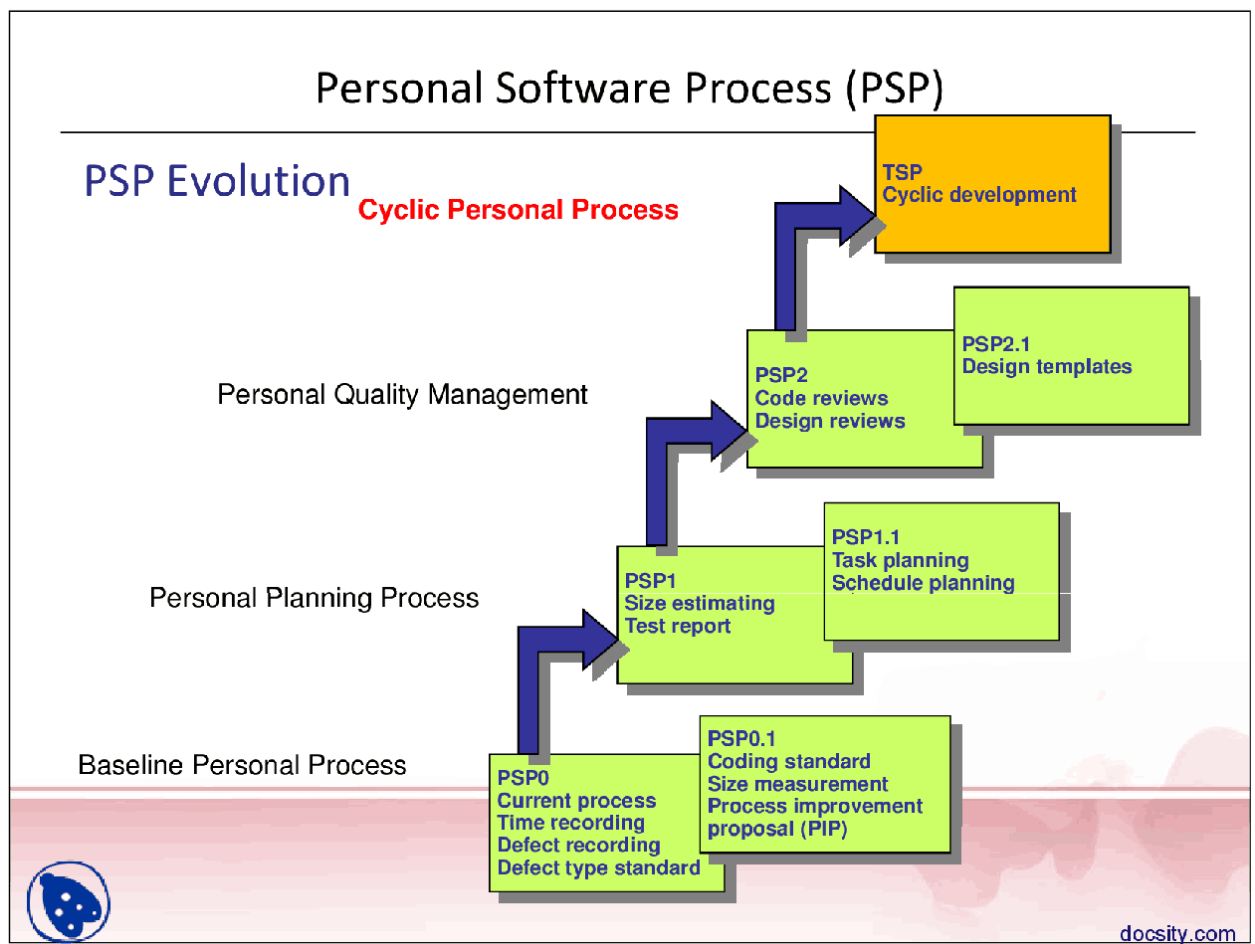
3. Quality

- Measures how good the software is.
- Measured by the number of defects (bugs).

4. Schedule

- Measures whether the project is completed on time.

PSP EVOLUTION DIAGRAM



TEAM SOFTWARE PROCESS (TSP)

Team Software Process (TSP) is a structured software development process developed by **Watts S. Humphrey (SEI)**. It helps a team of software engineers work together to develop **high-quality software on time and within budget**.

Objectives of TSP

1. Improve teamwork.
2. Improve software quality.
3. Reduce defects.
4. Increase productivity.
5. Meet project deadlines.
6. Deliver within budget.
7. Improve customer satisfaction.
8. Continuously improve the process.

Need for TSP

TSP helps solve common software project problems such as:

- Poor planning
- Communication issues
- Project delays
- High defects
- Cost overruns

Components of TSP

1. Team Building

- Assign roles
- Set goals
- Build teamwork
- Improve communication

2. Team Working

- Project planning
- Software development
- Quality control
- Defect tracking
- Process improvement

TSP Lifecycle (How TSP Works)

Step 1: PSP Training

- Every engineer first learns PSP.
- Learns planning, estimation, and defect tracking.

Step 2: Launch Phase

- Define project goals.
- Assign roles.
- Estimate effort and schedule.
- Prepare the project plan.

Step 3: Execution Phase

- Design and code the software.
- Review and test.
- Track time, defects, and schedule.
- Conduct regular team meetings.

Step 4: Postmortem Phase

- Compare planned vs actual results.
- Analyze defects.
- Record lessons learned.
- Improve future projects.

TSP FLOW DIAGRAM

