SOFTWARE PROCESS AND PROJECT MANAGEMENT
(PROFESSIONAL ELECTIVE – III)

B.Tech. IV Year I Sem. (IT)	**L**	**T**	**P**	**C**
Course Code: CS734PE	**3**	**0**	**0**	**3**

Course Objectives

- To acquire knowledge on software process management
- To acquire managerial skills for software project development
- To understand software economics

Course Outcomes

- Gain knowledge of software economics, phases in the life cycle of software development, project organization, project control and process instrumentation
- Analyze the major and minor milestones, artifacts and metrics from management and technical perspective
- Design and develop software product using conventional and modern principles of software project management

UNIT - I

Software Process Maturity

Software maturity Framework, Principles of Software Process Change, Software Process Assessment, The Initial Process, The Repeatable Process, The Defined Process, The Managed Process, The Optimizing Process. Process Reference Models, Capability Maturity Model (CMM), CMMI, PCMM, PSP, TSP).

UNIT - II

Software Project Management Renaissance

Conventional Software Management, Evolution of Software Economics, Improving Software Economics, The old way and the new way.

Life-Cycle Phases and Process artifacts

Engineering and Production stages, inception phase, elaboration phase, construction phase, transition phase, artifact sets, management artifacts, engineering artifacts and pragmatic artifacts, model-based software architectures.

UNIT - III

Workflows and Checkpoints of process

Software process workflows, Iteration workflows, Major milestones, minor milestones, periodic status assessments.

Process Planning

Work breakdown structures, Planning guidelines, cost and schedule estimating process, iteration planning process, Pragmatic planning.

UNIT - IV

Project Organizations

Line-of- business organizations, project organizations, evolution of organizations, process automation.

Project Control and process instrumentation

The seven-core metrics, management indicators, quality indicators, life-cycle expectations, Pragmatic software metrics, metrics automation.

UNIT - V

CCPDS-R Case Study and Future Software Project Management Practices

Modern Project Profiles, Next-Generation software Economics, Modern Process Transitions.

TEXT BOOKS:

1. Managing the Software Process, Watts S. Humphrey, Pearson Education
2. Software Project Management, Walker Royce, Pearson Education

REFERENCES:

1. An Introduction to the Team Software Process, Watts S. Humphrey, Pearson Education,2000 Process Improvement essentials, James R. Persse, O'Reilly,2006
2. Software Project Management, Bob Hughes & Mike Cotterell, fourth edition, TMH,2006
3. Applied Software Project Management, Andrew Stellman & Jennifer Greene, O'Reilly, 2006.
4. Head First PMP, Jennifer Greene & Andrew Stellman, O'Reilly,2007
5. Software Engineering Project Managent, Richard H. Thayer & Edward Yourdon, 2nd edition, Wiley India, 2004.
6. Agile Project Management, Jim Highsmith, Pearson education, 2004.

UNIT-1

Software Process Maturity

Software maturity Framework, Principles of Software Process Change, Software Process Assessment, The Initial Process, The Repeatable Process, The Defined Process, The Managed Process, The Optimizing Process.

Software Maturity Framework

Fundamentally, software development must be predictable. The software process is the set of tools, methods, and practices we use to produce a software product. The objectives of software process management are to produce products according to plan while simultaneously improving the organization's capability to produce better products.

The basic principles are those of statistical process control. A process is said to be stable or under statistical control if its future performance is predictable within established statistical limits. When a process is under statistical control, repeating the work in roughly the same way will produce roughly the same result. To obtain consistently better results, it is necessary to improve the process. If the process is not under statistical control, sustained progress is not possible until it is.

Lord Kelvin - "When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely in your thoughts advanced the stage of science."

SOFTWARE PROCESS IMPROVEMENT

To improve their software capabilities, organizations must take six steps;

- Understand the current status of the development processes
- Develop a vision of the required process
- Establish a list of required process improvement actions in order of priority
- Produce a plan to accomplish the required action
- Commit the resources to execute the plan
- Start over at the first step.

PROCESS MATURITY LEVELS

1. **Initial.** Until the process is under statistical control, orderly progress in process improvement is not possible. Must at least achieve rudimentary

predictability of schedules and costs.

2. **Repeatable**. The organization has achieved a stable process with a repeatable level of statistical control by initiating rigorous project management of commitments, costs, sched- ules, and changes.

3. **Defined.** The organization has defined the process as a basis for consistent implementation and better understanding. At this point, advanced technology can be introduced.

4. **Managed**. The organization has initiated comprehensive process measurements and analysis. This is when the most significant quality improvements begin.

5. **Optimizing**. The organization now has a foundation for continuing improvement and optimization of the process.

These levels are selected because they

- Represent the historical phases of evolutionary improvement of real software organizations
- Represent a measure of improvement that is reasonable to achieve from a prior level
- Suggest improvement goals and progress measures
- Make obvious a set of intermediate improvement priorities once an organization‟s status in this framework is known

The Initial Process (Level 1)

Usually ad hoc and chaotic - Organization operates without formalized procedures, cost estimates, and project plans. Tools are neither well intergrated with the process nor uni- formly applied. Change control is lax, and there is little senior management exposure or understanding of the problems and issues. Since many problems are deferred or even forgotten, software installation and maintenance often present serious problems.

While organizations at this level may have formal procedures for planning and tracking work, there is no management mechanism to insure they are used. Procedures are often abandoned in a crisis in favor of coding and testing. Level 1 organizations don‟t use design and code inspections and other techniques not directly related to shipping a product.

Organizations at Level 1 can improve their performance by instituting basic project controls. The most important ones are

- Project management
- Management oversight
- Quality assurance
- Change control

The Repeatable Process (Level 2)

This level provides control over the way the organization establishes plans and commitments. This control provides such an improvement over Level 1 that the people in the organization tend to believe they have mastered the software problem. This strength, however, stems from their prior experience in doing similar work. Level 2 organizations face major risks when presented with new challenges.

Some major risks:

•New tools and methods will affect processes, thus destroying the historical base on which the organization lies. Even with a defined process framework, a new technology can do more harm than good.

•When the organization must develop a new kind of product, it is entering new territory.

•Major organizational change can be highly disruptive. At Level 2, a new manager has no orderly basis for understanding an organization‟s operation, and new members must learn the ropes by word of mouth.

Key actions required to advance from Repeatable to the next stage, the Defined Process, are:

•Establish a process group: A process group is a technical resource that focuses heavily on improving software processes. In most software organizations, all the people are generally devoted to product work. Until some people are assigned full-time to work on the process, little orderly progress can be made in improving it.

•Establish a software development process architecture (or development cycle) that describes the technical and management activities required for proper execution of the development process. The architecture is a structural decomposition of the development cycle into tasks, each of which has a defined set of prerequisites, functional decompositions, verification procedures, and task completion specifications.

•Introduce a family of software engineering methods and technologies. These include design and code inspections, formal design methods, library control systems, and comprehensive testing methods. Prototyping and modern languages should be considered

.

The Defined Process (Level 3)

The organization has the foundation for major and continuing change. When faced with a crisis, the software teams will continue to use the same process that has been defined.

However, the process is still only qualitative; there is little data to indicate how much is accomplished or how effective the process is. There is considerable debate about the value of software process measurements and the best one to use.

The key steps required to advance from the Defined Process to the next level are:

•Establish a minimum set of basic process measurements to identify the quality and cost parameters of each process step. The objective is to quantify the relative costs and benefits of each major process activity, such as the cost and yield of error detection and correction methods.

•Establish a process database and the resources to manage and maintain it. Cost and yield data should be maintained centrally to guard against loss, to make it available for all projects, and to facilitate process quality and productivity analysis.

•Provide sufficient process resources to gather and maintain the process data and to advise project members on its use. Assign skilled professionals to monitor the quality of the data before entry into the database and to provide guidance on the analysis methods and interpretation.

•Assess the relative quality of each product and inform management where quality targets are not being met. Should be done by an independent quality assurance group.

The Managed Process (Level 4)

Largest problem at Level 4 is the cost of gathering data. There are many sources of potentially valuable measure of the software process, but such data are expensive to collect and maintain. Productivity data are meaningless unless explicitly defined. For example, the simple measure of lines of source code per expended development month can vary by 100 times or more, depending on the interpretation of the parameters.

When different groups gather data but do not use identical definitions, the results are not comparable, even if it makes sense to compare them. It is rare when two processes are comparable by simple measures. The variations in task complexity caused by different product types can exceed five to one. Similarly, the cost per line of code for small modifications is often two to three times that for new programs.

Process data must not be used to compare projects or individuals. Its purpose is too illuminate the product being developed and to provide an informed basis for improving the process. When such data are used by management to evaluate individuals or terms, the reliability of the data itself will deteriorate.

The two fundamental requirements for advancing from the Managed Process to the next level are:

- Support automatic gathering of process data. All data is subject to error and omission, some data cannot be gathered by hand, and the accuracy of manually gathered data is often poor.
- Use process data to analyze and to modify the process to prevent problems and improve efficiency.

The Optimizing Process (Level 5)

To this point software development managers have largely focused on their products and will typically gather and analyze only data that directly relates to product improvement. In the Optimizing Process, the data are available to tune the process itself.

For example, many types of errors can be identified far more economically by design or code inspections than by testing. However, some kinds of errors are either uneconomical to detect or almost impossible to find except by machine. Examples are errors involving interfaces, performance, human factors, and error recovery.

So, there are two aspects of testing: removal of defects and assessment of program quality. To reduce the cost of removing defects, inspections should be emphasized. The role of functional and system testing should then be changed to one of gathering quality data on the program. This involves studying each bug to see if it is an isolated problem or if it indicates design problems that require more comprehensive analysis.

With Level 5, the organization should identify the weakest elements of the process and fix them. Data are available to justify the application of technology to various critical tasks, and numerical evidence is available on the effectiveness with which the process has been applied to any given product.

The Principles of Software Process Change

People:

- The best people are always in short supply
- You probably have about the best team you can get right now.
- With proper leadership and support, most people can do much better than they are currently doing

Design:

- Superior products have superior design. Successful products are designed by people who understand the application (domain engineer).
- A program should be viewed as executable knowledge. Program designers should have application knowledge.

The Six Basic Principles of Software Process Change:

- Major changes to the process must start at the top.
- Ultimately, everyone must be involved.
- Effective change requires great knowledge of the current process
- Change is continuous
- Software process changes will not be retained without conscious effort and periodic reinforcement
- Software process improvement requires investment

Continuous Change:

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

Software Processes Changes Won't Stick by Themselves

The tendency for improvements to deteriorate is characterized by the term *entrophy* (Webster"s: a measure of the degree of disorder in a...system; entrophy always increases and available energy diminishes in a closed system.).

New methods must be carefully introduced and periodically monitored, or they to will rapidly decay.

Human adoption of new process involves four stages:

- Installation - Initial training
- Practice - People learn to perform as instructed
- Proficiency - Traditional learning curve
- Naturalness - Method ingrained and performed without intellectual effort.

It Takes Time, Skill, and Money!

- To improve the software process, someone must work on it
- Unplanned process improvement is wishful thinking
- Automation of a poorly defined process will produce poorly defined results
- Improvements should be made in small steps
- Train!!!!.

Some Common Misconceptions about the Software Process

- We must start with firm requirements
- If it passes test it must be OK
- Software quality can't be measured
- The problems are technical
- We need better people
- Software management is different.

FIRM REQUIEMENTS - A software perversity law seems to be the more firm the specifications, the more likely they are to be wrong. With rare exceptions, requirements change as the software job progresses. Just by writing a program, we change our perceptions of the task. Requirements cannot be firm because we cannot anticipate the ways the tasks will change when they are automated.

For large-scale programs, the task of stating a complete requirement is not just difficult; it is impossible. Generally, we must develop software incrementally. However, we must have stability long enough to build and test a system. However, if we freeze requirements too early, later retrofits are expensive.

SOFTWARE MANAGEMENT IS DIFFERENT - Management must not view it as black art. Must insist on tracking plans and reviews.

Champions, Sponsors, and Agents

- Champions - Ones who initiate change. They bring management's attention to the subject, obtain the blessing of a sponsor, and establish the credibility to get the change program launched. The champion maintains focus on the goal, strives to overcome obstacles, and refuses to give up when the going gets tough.
- Sponsors - Senior manager who provides resources and official backing. Once a sponsor is found, the champion's job is done; it is time to launch the change process.
- Agents - Change agents lead the planning and implementation. Agents must be enthusiastic, technically and politically savvy, respected by others, and have management's confidence and support.

Elements of Change

- Planning
- Implementation
- Communication

3: Software Process Assessment

Process assessments help software organizations improve themselves by identifying their crucial problems and establishing improvement priorities. The basic assessment objec- tives are:

- Learn how the organization works
- Identify its major problems
- Enroll its opinion leaders in the change process

The essential approach is to conduct a series of structured interviews with key people in the organization to learn their problems, concerns, and creative ideas.

ASSESSMENT OVERVIEW

A software assessment is not an audit. Audit are conducted for senior managers who suspect problems and send in experts to uncover them. A software process assessment is a review of a software organization to advise its management and professionals on how they can improve their operation.

The phases of assessment are:

- Preparation - Senior management agrees to participate in the process and to take actions on the resulting recommendations or explain why not. Concludes with a training program for the assessment team
- Assessment - The on-site assessment period. It takes several days to two or more weeks. It concludes with a preliminary report to local management.
- Recommendations - Final recommendations are presented to local managers. A local action team is then formed to plan and implement the recommendations.

Five Assessment Principles:

- The need for a process model as a basis for assessment
- The requirement for confidentiality
- Senior management involvement
- An attitude of respect for the views of the people in the organization be assessed
- An action orientation

Start with a process model - Without a model, there is no standard; therefore, no measure of change.

Observe strict confidentiality - Otherwise, people will learn they cannot speak in confidence. This means managers can‟t be in interviews with their subordinates.

Involve senior management - The senior manager (called *site manager* here) sets the organizations priorities. The site manager must be personally involved in the assessment and its follow-up actions. Without this support, the assessment is a waste of time because lasting improvement must survive periodic crises.

Respect the people in the assessed organization - They probably work hard and are trying to improve. Do not appear arrogant; otherwise, they will not cooperate and may try to prove the team is ineffective. The only source of real information is from the workers.

Assessment recommendations should highlight the three or four items of highest priority. Don‟t overwhelm the organization. The report must always be in writing.

Implementation Considerations - The greatest risk is that no significant improvement actions will be taken. Superficial changes won‟t help. A small, full-time group should guide the implementation effort, with participation from other action plan working groups. Don‟t forget that site managers can change or be otherwise distracted, so don‟t rely on that person solely, no matter how committed.

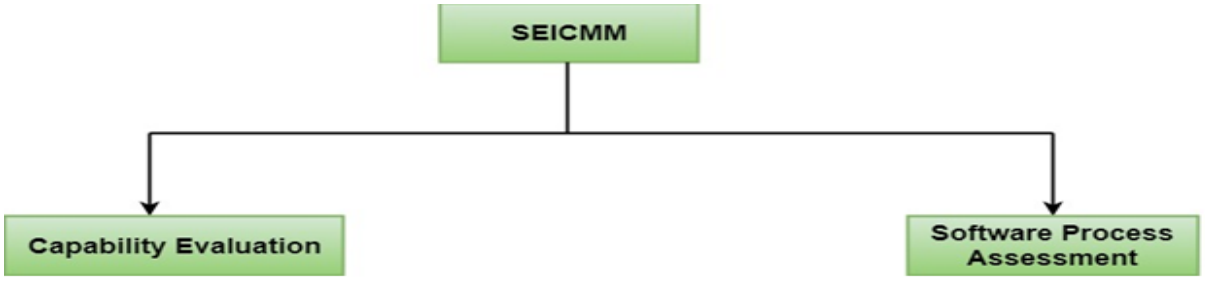
Process Reference Models:

The Optimizing Process. Process Reference Models, Capability Maturity Model (CMM), CMMI, PCMM, PSP, TSP).

1. Capability Maturity Model :CMM

The Software Engineering Institute (SEI) Capability Maturity Model (CMM) specifies an increasing series of levels of a software development organization. The higher the level, the better the software development process, hence reaching each level is an expensive and time- consuming process.

- Capability Maturity Model is used as a benchmark to measure the maturity of an organization's software process.
- Methods of SEICMM
- There are two methods of SEICMM:

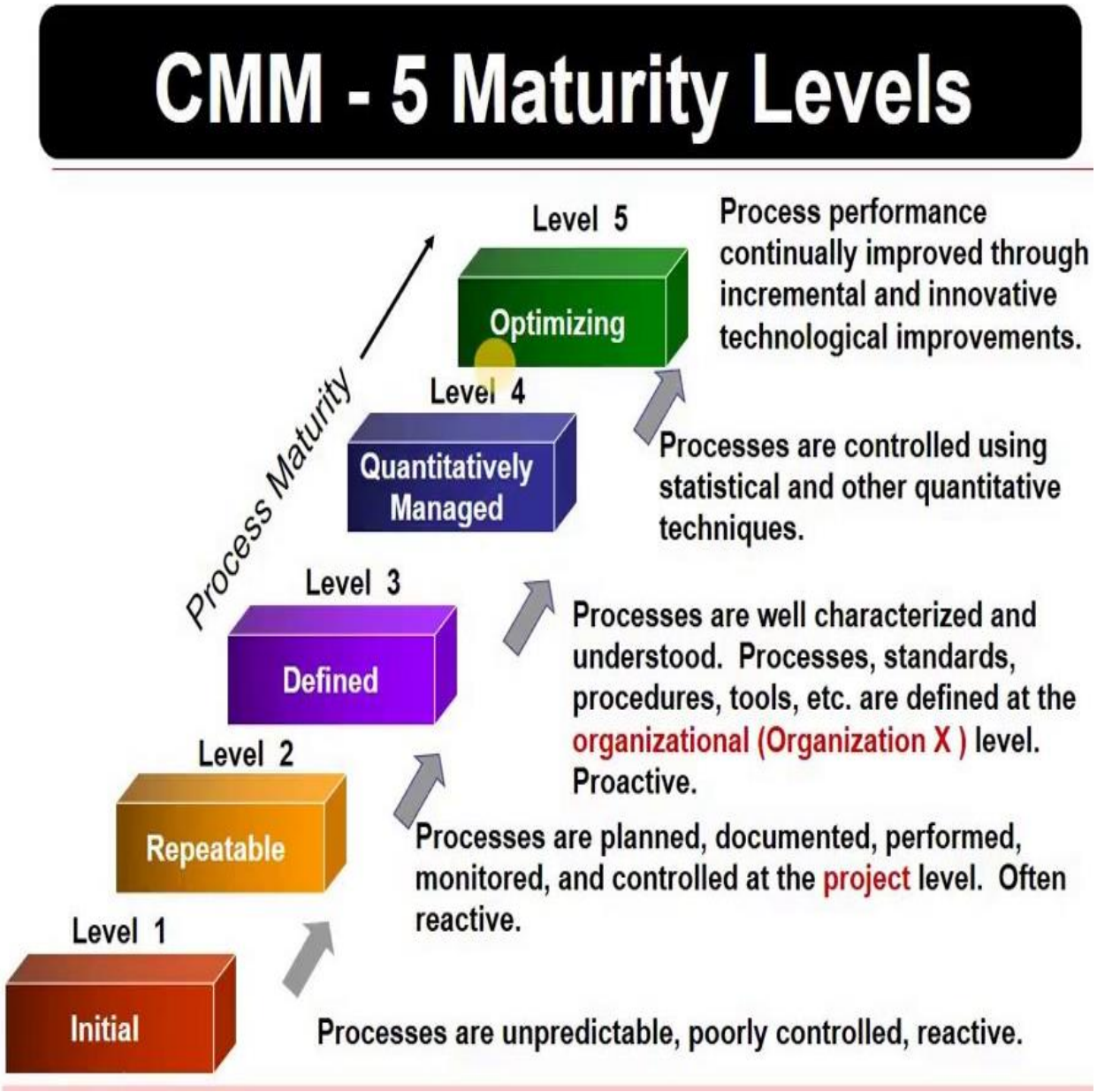


- **Capability Evaluation:** Capability evaluation provides a way to assess the software process capability of an organization.
- **Software Process Assessment:** Software process assessment is used by an organization to improve its process capability. Thus, this type of evaluation is for purely internal use
-

SEI CMM categorized software development industries into the following five maturity levels.

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. The heroes eventually move on to other organizations taking their wealth of knowledge or lessons learnt with them.
- **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.

Key Process Areas (KPA) of a software organization

- Except for SEI CMM level 1, each maturity level is featured by several Key Process Areas (KPAs) that contains the areas an organization should focus on improving its software process to the next level. The focus of each level and the corresponding key process areas.

CMM Level	Focus	Key Process Areas
1. Initial	Competent People	NO KPA'S
2. Repeatable	Project Management	Software Project Planning software Configuration Management
3. Defined	Definition of Processes	Process definition Training Program Peer reviews
4. Managed	Product and Process quality	Quantitiative Process Metrics Software Quality Management
5. Optimizing	Continuous Process improvement	Defect Prevention Process change management Technology change management

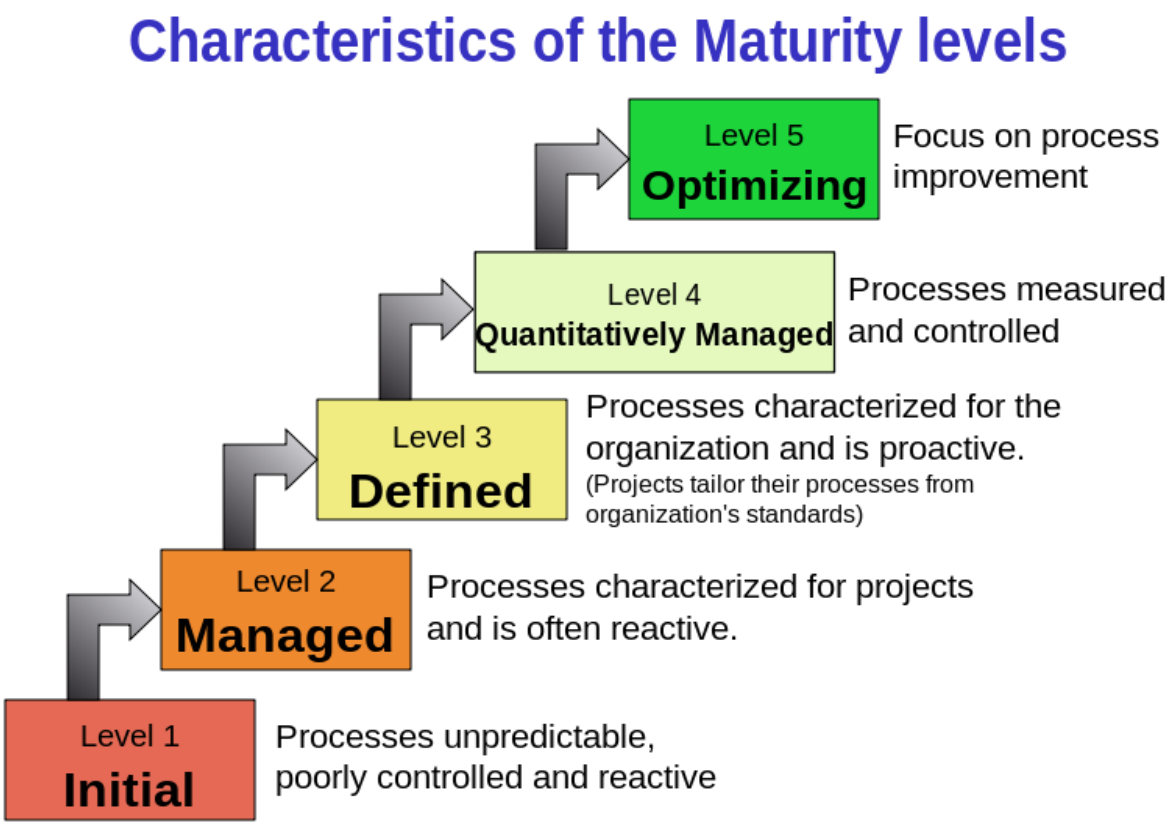
The focus of each SEI CMM level and the Corresponding Key process areas.

CMMI

- CMMI (Capability Maturity Model Integration) is a proven industry framework to improve product quality and development efficiency for both hardware and software.
- Sponsored by US Department of Defence in cooperative with Carnegie Mellon University and the Software Engineering Institute(SEI)
- CMMI, staged, uses 5 levels to describe the maturity of the organization.
- CMMI is an evolutionary improvement path for software organizations from immature process to a mature, disciplined one.
- Provides guidance on how to gain control of processes for developing and maintaining software.
- CMMI describes the key elements of an effective software process.

In CMMI models with a staged representation, there are five maturity levels designated by the numbers 1 through 5

1. Initial
2. Managed
3. Defined
4. Quantitatively Managed
5. Optimizing



Maturity Level 1: Initial

- Maturity level 1 deals with Performed Processes.
- Processes are unpredictable, poorly controlled, and reactive.
- The process performance may not be stable and may not meet specific objectives such as quality, cost and schedule, but useful work can be done.

Maturity Level 2: Managed

- At maturity level 2, an organization has achieved all the **specific** and **generic** goals of the maturity level 2 process areas. In other words, the projects of the organization have ensured that requirements are managed and that processes are planned performed, measured, and controlled.
- In the managed level basic project management is in place.
- But the basic project management and practices are followed only in the project level
- The managed process comes closer to achieving the specific objectives such as quality, cost, and schedule.

Maturity Level 3: Defined

- At maturity level 3, an organization has achieved all **specific** and **generic goals** of the process areas assigned to maturity levels 2 and 3.
- In the previous level all good practices and processes were done at project level.
- In this level all good practices and processes are brought to the organizational level.
- At maturity level 3, processes are well characterized and understood, and are described in standards, procedures, tools, and methods.
- An important distinction between maturity level 2 and 3 is that at level 3, processes are described in more detail and more rigorously than at level 2 and are at an organizational level.

Maturity Level 4: Quantitatively Managed

- At maturity level 4, an organization has achieved all the **specific goals** of the process areas assigned to maturity levels 2, 3, and 4.
- At this level processes are controlled by using statistical and other quantitative techniques.

- Product quality, processes performance and service quality are understood in statistical terms and are managed throughout the life of the processes.
- Maturity level 4 concentrate on using metrics to make decisions and to truly measure whether progress is happening and the product is becoming better.
- The main difference between level 3 and 4 is that at level 3, processes are qualitatively predictable.
- Level 4 addresses cause of process variation and take corrective actions.

Maturity Level 5: Optimizing

- At maturity level 5, an organization has achieved all the **specific goals** of the process areas assigned to maturity levels 2, 3, 4, and 5.
- Maturity level 5 focuses on continually improving process performance through both incremental and innovative technological improvements.
- In this level processes are continually improved based on an understanding of common causes of variation within the process.
- This is like the final level, defects are minimized, and products are delivered on time and within the budget boundary.

How can CMMI help?

CMMI provides a way to focus and manage hardware and software development from product inception through deployment and maintenance.

Behavioral changes are needed at both management and staff levels. Examples:

- Increased personal accountability.
- Tighter links between Product Management, Development.

Initially a lot of investment required – but, if properly managed, we will be more efficient and productive while turning out products with consistently higher quality.

Behaviors at the Five Levels

Maturity Level	Process Characteristics	Behaviors
5 Optimizing	Focus is on continuous quantitative improvement	Focus on "fire prevention"; improvement anticipated and desired, and impacts assessed.
4 Quantitatively Managed	Process is measured and controlled	Greater sense of teamwork and inter-dependencies
3 Defined	Process is characterized for the organization and is proactive	Reliance on defined process. People understand, support and follow the process.
2 Managed	Process is characterized for projects and is often reactive	Over reliance on experience of good people – when they go, the process goes. "Heroics."
1 Initial	Process is unpredictable, poorly controlled, and reactive	Focus on "fire fighting"; effectiveness low – frustration high.

14

CMMI COMPONENTS

- Within each of the 5 Maturity Level, there are basic functions that need to be performed these are called Process Areas (Pas).
- All Process areas that must be completely satisfied.
- Within each PA there are Goals to be achieved and within each Goal there are Practices, work products, etc. to be followed that will support each of the Goal.

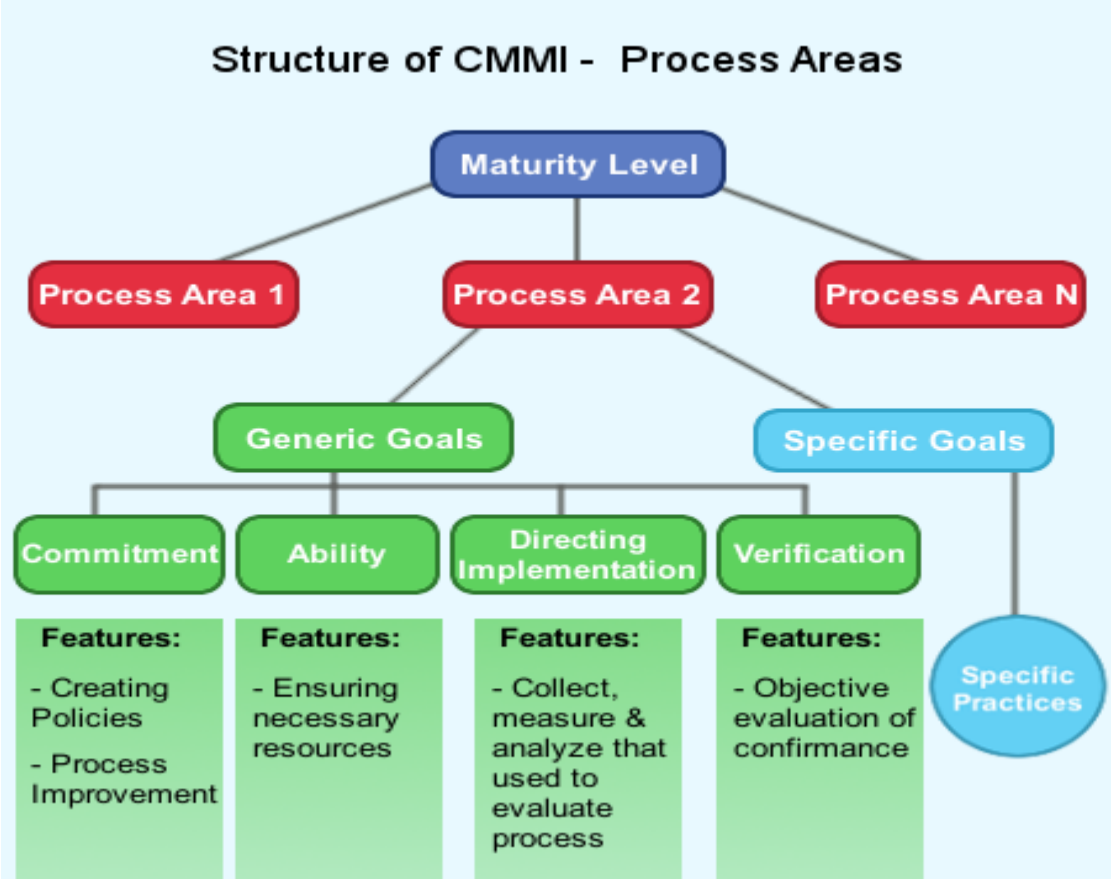
Maturity Levels and Process Areas:

Here is a list of all the corresponding process areas defined for a S/W organization. These process areas may be different for different organization.

Level	Focus	Key Process Area	Result
5 Optimizing	Continuous Process Improvement	• Organizational Innovation and Deployment • Causal Analysis and Resolution	Highest Quality / Lowest Risk
4 Quantitatively Managed	Quantitatively Managed	• Organizational Process Performance • Quantitative Project Management	Higher Quality / Lower Risk
		• Requirements Development • Technical Solution • Product Integration • Verification • Validation • Organizational Process Focus • Organizational Process Definition • Organizational Training	Medium

3 Defined	Process Standardization	• Integrated Project Mgmt (with IPPD extras) • Risk Management • Decision Analysis and Resolution • Integrated Teaming (IPPD only) • Org. Environment for Integration (IPPD only) • Integrated Supplier Management (SS only)	Quality / Medium Risk
2 Managed	Basic Project Management	• Requirements Management • Project Planning • Project Monitoring and Control • Supplier Agreement Management	Low Quality / High Risk
		• Measurement and Analysis • Process and Product Quality Assurance • Configuration Management	

1 Initial	Process is informal and Adhoc		Lowest Quality / Highest Risk



Difference between CMM and CMMI:

CMM is a reference model of matured practices in a specified discipline like Systems Engineering CMM, Software CMM, People CMM, Software Acquisition CMM etc. But they were difficult to integrate as and when needed.

CMMI is the successor of the CMM and evolved as a more matured set of guidelines and was built combining the best components of individual disciplines of CMM (Software CMM, People CMM etc). It can be applied to product manufacturing, People management, Software development etc.

CMM describes about the software engineering alone where as CMM Integrated describes both software and system engineering. CMMI also incorporates the Integrated Process and Product Development and the supplier sourcing.

People Capability Maturity Model (PCMM)

PCMM is a maturity structure that focuses on continuously improving the management and development of the human assets of an organization.

It defines an evolutionary improvement path from Adhoc, inconsistently performed practices, to a mature, disciplined, and continuously improving the development of the knowledge, skills, and motivation of the workforce that enhances strategic business performance.

The People Capability Maturity Model (PCMM) is a framework that helps the organization successfully address their critical people issues. Based on the best current study in fields such as human resources, knowledge management, and

organizational development, the PCMM guides organizations in improving their steps for managing and developing their workforces.

The People CMM defines an evolutionary improvement path from Adhoc, inconsistently performed workforce practices, to a mature infrastructure of practices for continuously elevating workforce capability.

The PCMM subsists of five maturity levels that lay successive foundations for continuously improving talent, developing effective methods, and successfully directing the people assets of the organization. Each maturity level is a well-defined evolutionary plateau that institutionalizes a level of capability for developing the talent within the organization.

People Capability Maturity Model: Primary Objective

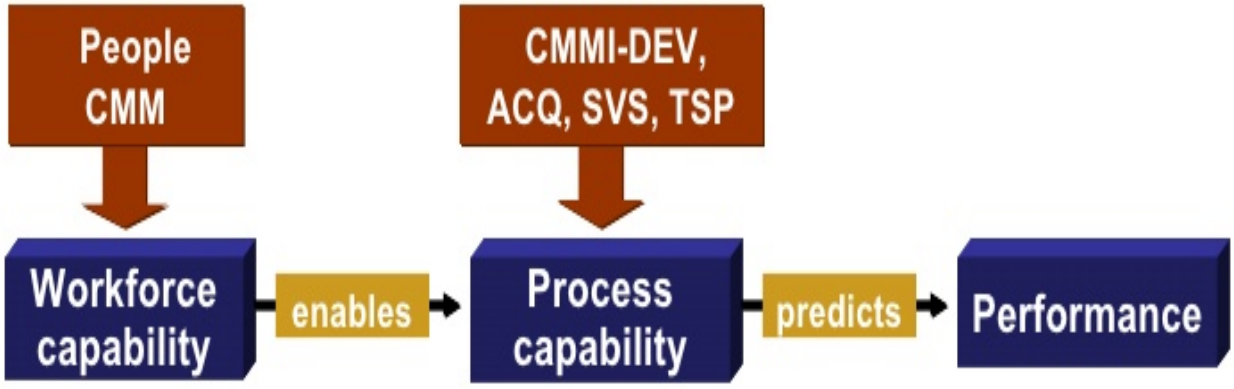
The primary objective of:

a **CMM** is to improve the **capability** of an organization.

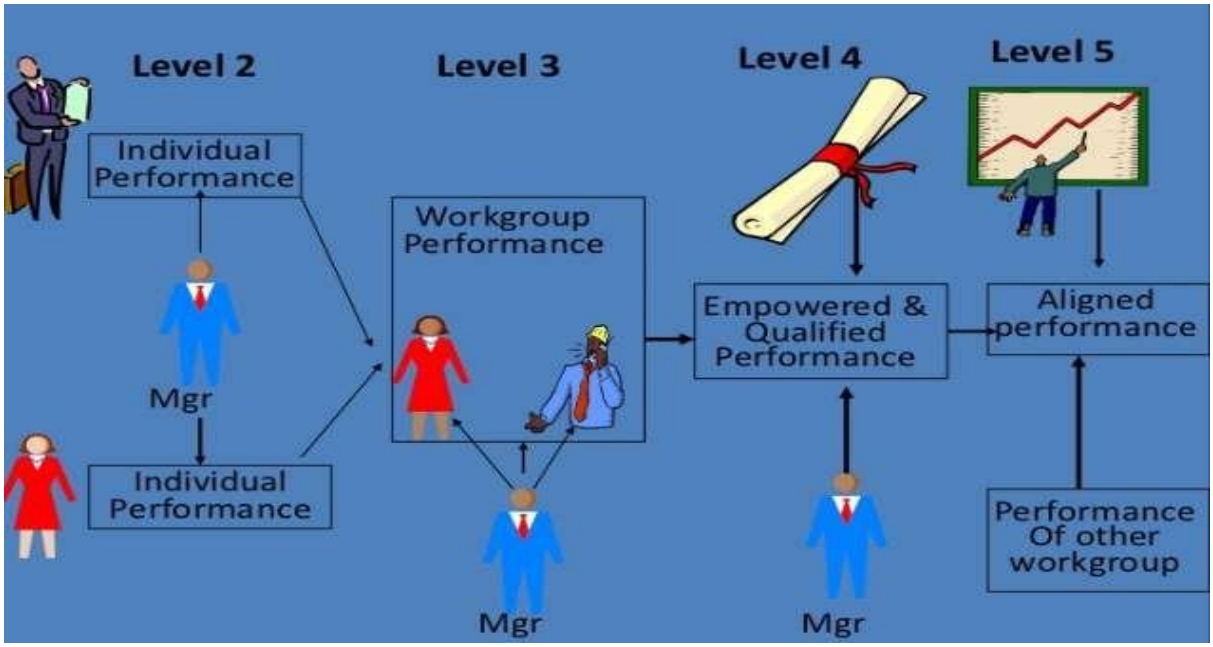
the **CMMI** (**DEV**, **ACQ**, **SVC**), is to improve the **capability** of an organizations processes.

the **People CMM** is to improve the **capability** of an organization's workforce.

The People CMM, defines **capability** as the **level** of **knowledge**, **skills**, and **process abilities** available within each workforce competency of the organization to build its products or deliver its services.



Maturity Framework



Initial Level: Maturity Level 1

The Initial Level of maturity includes no process areas. Although workforce practices implement in Maturity Level, 1 organization tend to be inconsistent or ritualistic, virtually all of these organizations perform processes that are defined in the Maturity Level 2 process areas.

Managed Level: Maturity Level 2

To achieve the Managed Level, Maturity Level 2, managers starts to perform necessary people management practices such as staffing, operating performance, and adjusting compensation as a repeatable management discipline. The organization establishes a culture focused at the unit level for ensuring that person can meet their work commitments. In achieving Maturity Level 2, the organization develops the capability to handle skills and performance at the unit level. The process areas at Maturity Level 2 are Staffing, Communication and Coordination, Work Environment, Performance Management, Training and Development, and Compensation.

Defined Level: Maturity Level 3

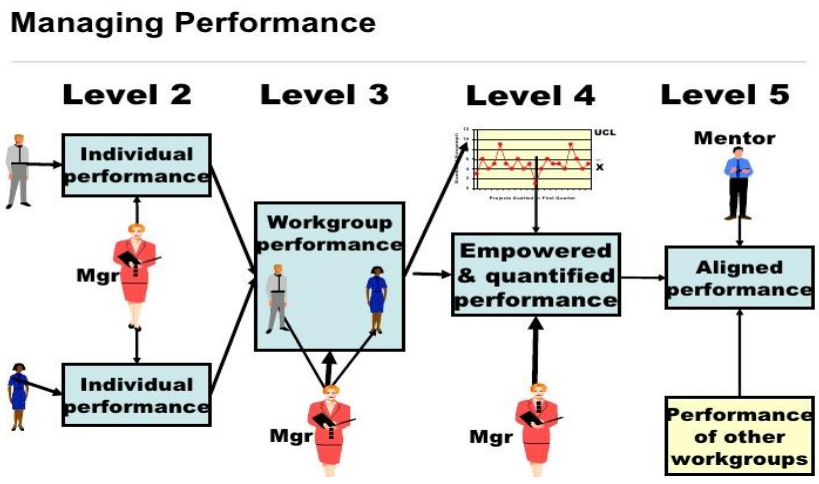
The fundamental objective of the defined level is to help an organization gain a competitive benefit from developing the different competencies that must be combined in its workforce to accomplish its business activities. These workforce competencies represent critical pillars supporting the strategic workforce competencies to current and future business objectives; the improved workforce practices for implemented at Maturity Level 3 become crucial enablers of business strategy.

Predictable Level: Maturity Level 4

At the Predictable Level, the organization handles and exploits the capability developed by its framework of workforce competencies. The organization is now able to handle its capacity and performance quantitatively. The organization can predict its capability for performing work because it can quantify the ability of its workforce and of the competency-based methods they use performing in their assignments.

Optimizing Level: Maturity Level 5

At the Optimizing Level, the integrated organization is focused on continual improvement. These improvements are made to the efficiency of individuals and workgroups, to the act of competency-based processes, and workforce practices and activities.



- The P-CMM is a practical tool for improving the management of people in an organization because it provides a framework for motivating, recognising, standardizing and improving good practice. However, like all capability models created by the SEI, it is designed for large rather than small companies.
- It reinforces the need to recognize the importance of people as individuals and to develop their capabilities. Of course, the complete application of this model is very expensive and probably unnecessary for most organizations.
- However, it is a helpful guide that can lead to significant improvements in the capability of organizations to produce high-quality software.

Structural Components of a CMM

It means in PCMM,there are :-

5 maturity levels.

4 process area thread.

22 processes with related goals

246 practices

249 institutionalization practices

However, goals and practices may vary as per requirement of the organization.

Defining Workforce Competency

Knowledge represents the comprehension acquired by experience and or study.

Skills represents the proficiency or ability in techniques or tools that an individual must be able to demonstrate.

Process abilities is the capacity to perform individual skills in the sequencing or method used in the organization.

Knowledge + **Skills** + **Process abilities** = **Workforce Competency**

The Personal Software Process (PSP)

The **software process** is about making software engineering groups/teams work to the best of their abilities

The **personal software process** is about making individual engineers work to the best of their abilities

The PSP is individual oriented: it is possible for an individual to succeed within a project that fails.

Objectives of Personal Software Process

The aim of PSP is providing software developers with disciplined methods and strategies for improving personal software development processes. PSP assists software engineers to:

- Improve their planning and estimating skills.
- Make commitments and schedules they can keep and meet.
- Reduce defects in their projects.
- Manage the quality of their plans.

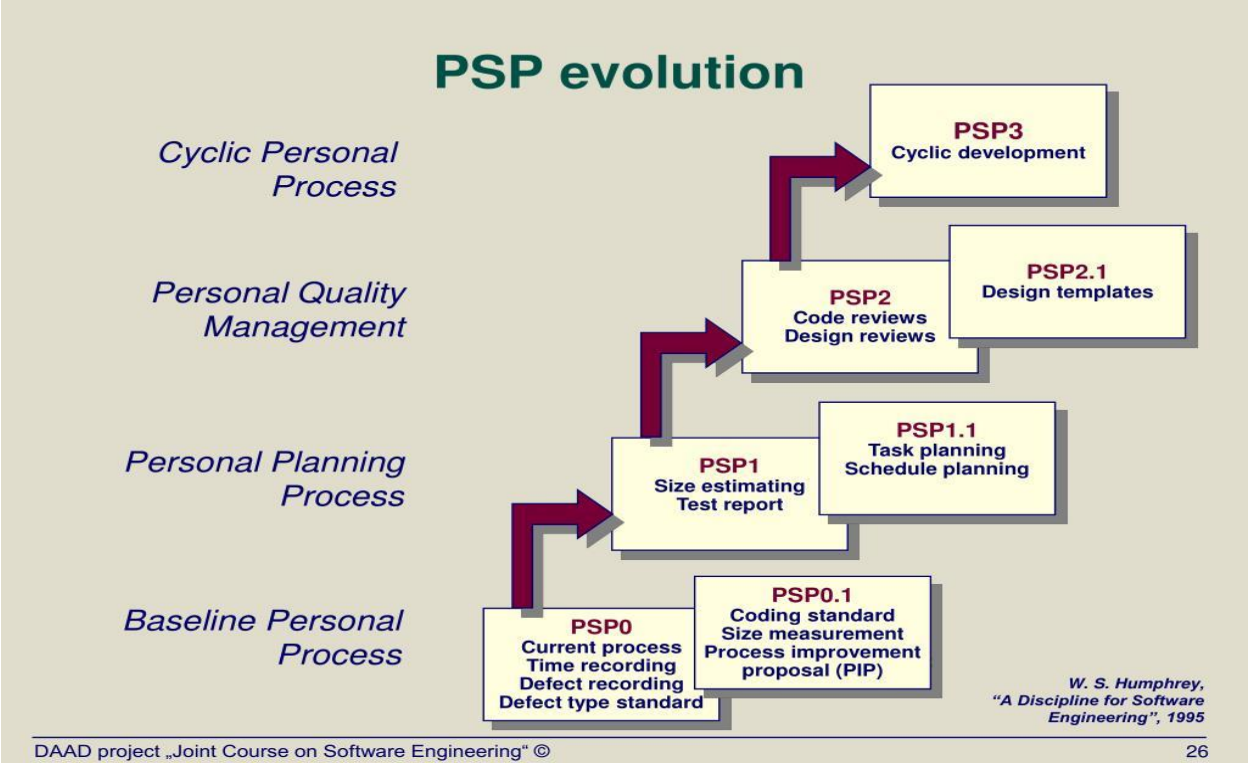
PSP Benefits:

- Increases personal commitment by investing each engineer with process responsibility
- Assists engineers in making accurate plans
- Provides steps engineers can take to improve personal and project quality
- Sets benchmarks to measure personal process improvements
- Demonstrates the impact of process changes on an engineer's performance

Principles of Personal Software Process

The design of PSP is based on these planning and quality principles:

- Every engineer is different. For software engineers to become more active, they should plan their work and base these plans on their personal data.
- To improve their performance, software engineers should personally use regular and well-defined processes.
- For software engineers to produce quality software products, they should feel personally responsible for the quality of the products they are making. Superior software products are never created by mistake but by striving to do quality work.
- It's cheaper to trace and fix defects earlier than later.
- It's easier to prevent errors than finding and fixing them.
- The cheapest and fastest way to do any task is doing it in the right direction.

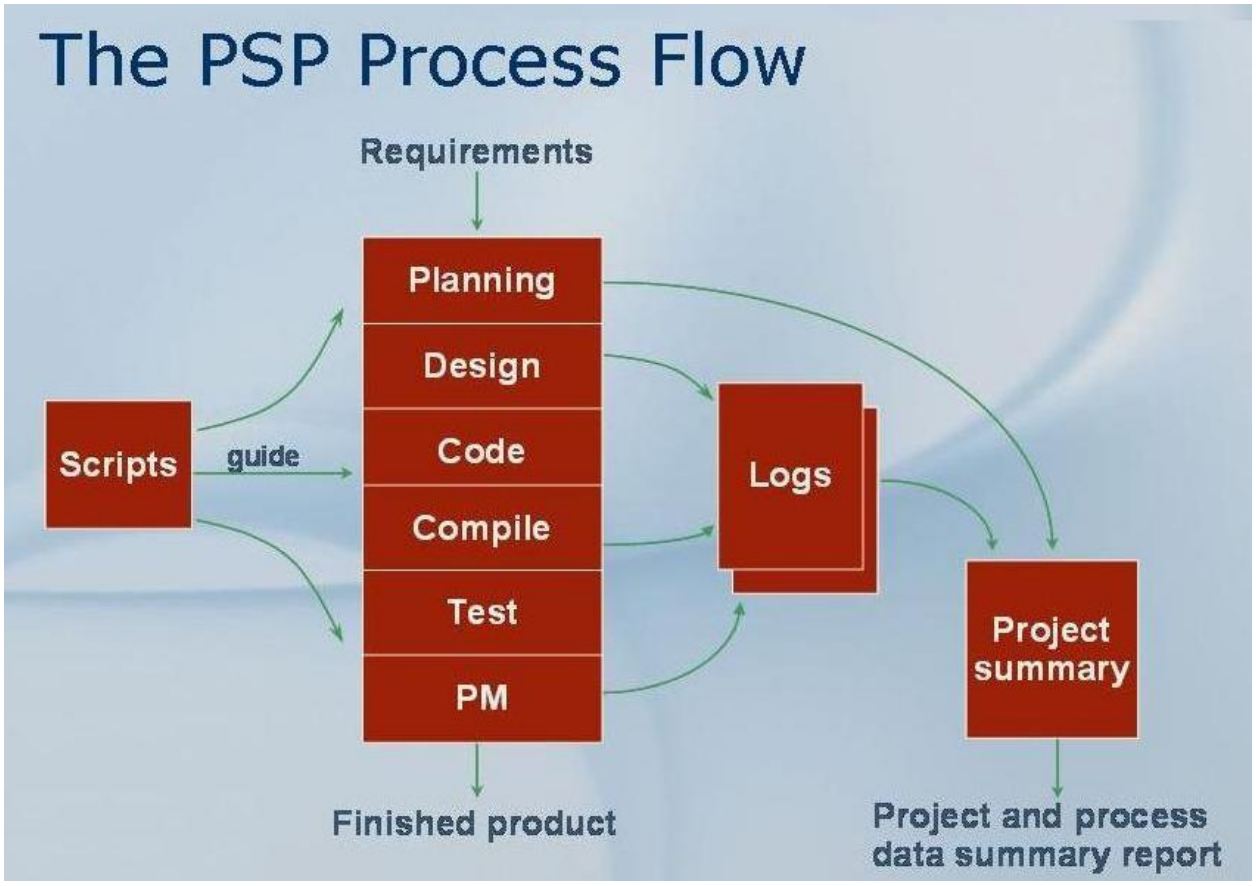


The engineers follow methods from PSP0 to PSP3 while working on the software with each level they are introduced to different methods which let them know and measure their performance. The PSP consists of following processes

- Base line personal process (PSP0)
- Personal planning process (PSP1)
- Personal quality management (PSP2)
- Cyclic personal process (PSP3)

Baseline personal process (PSP0)

- It establishes the baseline for measuring the progress and to define a foundation on which to improve
- Psp0 has three phases that is planning, development and postmortem
- The planning and development phase includes plan, design, code, compile and test
- In postmortem a comparison of actual performance with the plan is made. This provides a basis for measuring progress and defines a foundation on which to improve
- Psp0.1 defines coding standard, size measurement and personal improvement proposal



PSP Planning:

- Problem definition
- Estimate max, min, and total LOC
- Determine minutes/LOC
- Calculate max, min, and total developments times
- Enter the plan data in project plan summary form
- Record the planned time in log

PSP Design:

- Design the program.

- Record the design in specified format.
- Record the design time in time recording log.

PSP Code:

- Implement the design.
- Use a standard format for code text.
- Record the coding time in time recording log.

PSP-Test/Post mortem:

- Test

- TEST the Program

- Fix all the defects found

- Record testing time in time recording log.

- Post Mortem

-Complete project plan summary from with actual time and size data.

-Record post mortem time in time record log .

Personal planning process (psp1):

- Psp1 and psp1.1 focus on personal project management techniques, introducing size and effort estimating, schedule planning and schedule tracking methods
- Size and effort estimates are made using PROBE (PROXY Based Estimating) method
- With PROBE engineers use the relative size of proxy to make their initial estimate

Personal quality management (psp2):

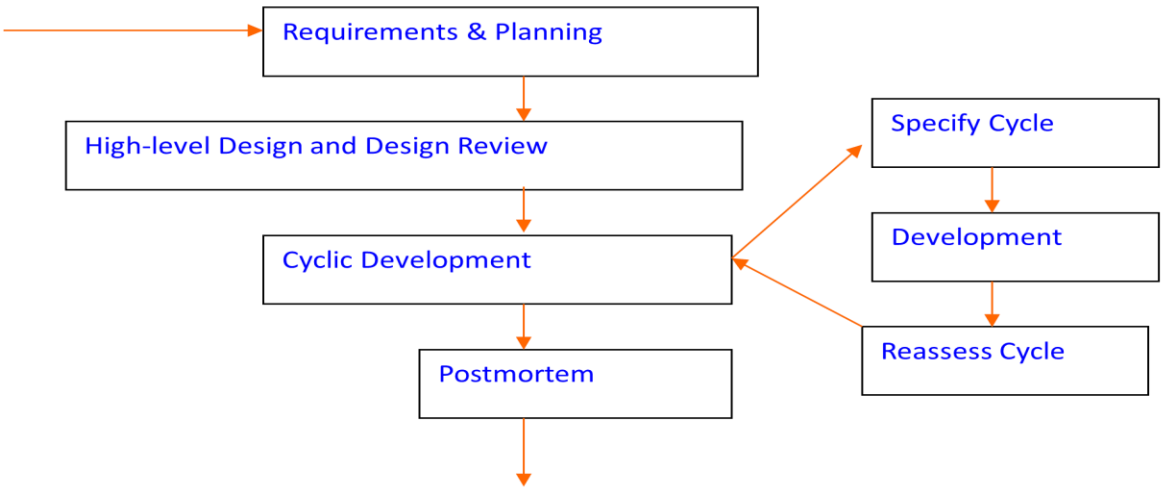
- Psp2 introduces defect management. This phase adds personal design and code reviews and quality management and evaluation to psp1
- There are two tasks involve in psp2 they are code review and design review
- Code review: whose goal is to improve the quality of the software by reviewing the program and its associated documentation

- Design review: whose task is to verify program logic, verify function use, ensure that requirements, specifications are completely covered by the design
- The psp2.1 introduces design completeness and design verification
- The psp2.1 adds four design templates and design verification systems. This improves the design process and design verification.

Cyclic personal process (psp3):

- This is the final step of PSP. Till now what we have discussed is about related to small programs
- Psp3 is used for large programs the technique is to divide the larger programs into small programs of psp2.1 size then psp3 combines multiple psp2.1 process into large scale software development.
- This is a iterative process

Cyclic personal process

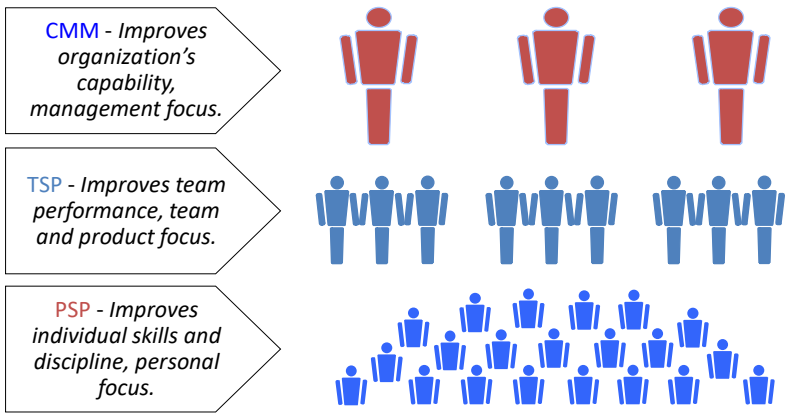

Requirements & Planning
High-level Design and Design Review
Cyclic Development
Postmortem
Specify Cycle
Development
Reassess Cycle

The Team Software Process (TSP)

Team software process (TSP): The primary goal of TSP is to create a team environment for establishing and maintaining a self-directed team, and supporting disciplined individual work as a base of PSP framework. Self-directed team means that the team manages itself, plans and tracks their work, manages the quality of their work, and works proactively to meet team goals. TSP has two principal components: team-building and team- working. Team-building is a process that defines roles for each team member and sets up teamwork through TSP launch and periodical relaunch. Team-working is a process that deals with engineering processes and practices utilized by the team. TSP, in short, provides engineers and managers with a way that establishes and manages their team to produce the high-quality software on schedule and budget.

TSP along with the Personal Software Process helps the high-performance engineer to:

- Ensure quality software products
- Create secure software products
- Improve process management in an organization.



Differences between PSP and TSP:

- **Personal software process is focused on individuals to improve their performance.**

- PSP process consists of methods, forms and tricks to guide software engineers in doing their development work.

- Team software process depends on a group of individuals and aimed at improving the performance of the team.

- TSP process consists of programming strategies which will help a software engineering team to build better quality products.

- Team Software Process (TSP) is a process for PSP-trained software engineering teams with 2 to 20 members.
- TSP supports
    - development, enhancement, and repair
    - self-directed teams
    - interdisciplinary teams
    - isolated software teams
    - statistical process control

- SDWT: A **self-directed** work **team** (SDWT) is a group of people, usually employees in a company, who combine different skills and talents to work without the usual managerial supervision toward a common purpose or goal. Typically, an SDWT has somewhere between two and 25 members.

- InterdisciplinaryTeam: **Interdisciplinary** approach involves **team** members from different disciplines working collaboratively, with a common **purpose**, to set goals, make decisions and share resources and responsibilities.

- A team is a group of people who
    - are working together

- have a common end objective
- do interdependent work
- depend on and support each other
- act, feel, and think like a close-knit group

Not all working groups are teams

TSP Objectives:

TSP was developed to

- help software engineering teams build quality products within cost and schedule constraints
- accelerate software process improvement
- make Level 5 behavior normal and expected

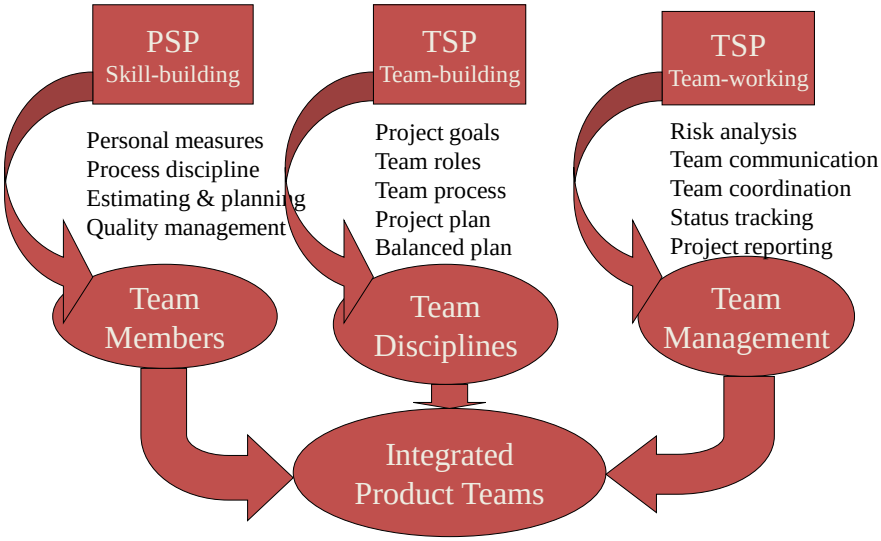
A principal TSP design goal was to create a process that builds effective teams and optimizes team performance throughout the project.

Effective Teams:

- Teams must be properly skilled and be able to work as cohesive units. Effective teams have certain common characteristics:
- The members are skilled.
- The team's goal is important, defined, visible, and realistic.
- The team's resources are adequate for the job.
- The members are motivated and committed to meeting the team's goal.
- The members cooperate and support each other.

- The members are disciplined in their work.

Building Effective Teams -3



Team Member Roles -1

- Being a team-directed project means the team has to manage itself.
  - plan and track work
  - manage the quality of the work
  - responsibly manage the project risks
  - work aggressively to meet team goals
- The team must also show management and the customer that they are managing themselves.
  - frequently report status and progress

- anticipate, plan for, and report on project risks

Team Member Roles -2

- The self-management responsibilities are shared among the team members.
- The eight team member roles are:
  - Customer Interface Manager
  - Design Manager
  - Implementation Manager
  - Planning Manager
  - Process Manager
  - Quality Manager
  - Support Manager
  - Test Manager

The Team Leader's Role

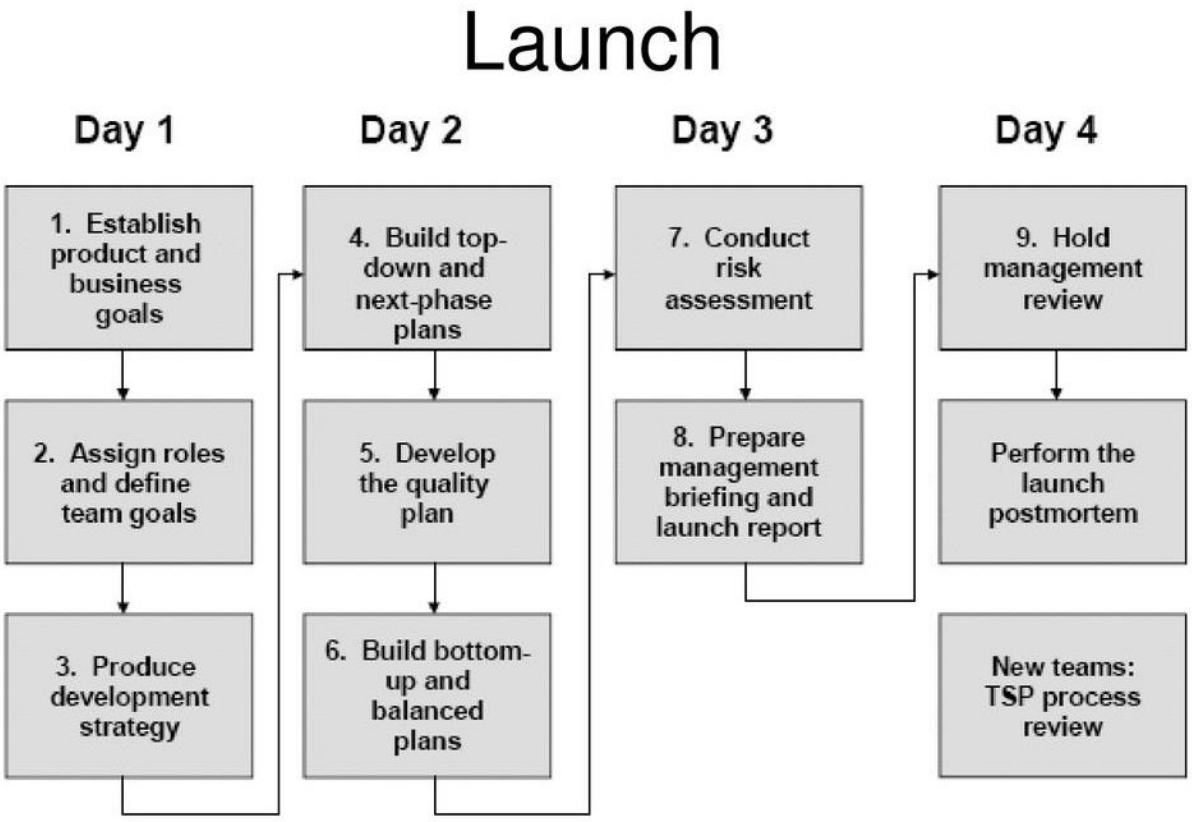
- The team leader does not take one of these team roles.
- The team leader's job is to
  - coach the team
  - provide support
  - guide the team in doing their work
  - establish and maintain high standards for the work

TSP Base Measures

- TSP uses the same base measures as the PSP
  - product size in pages or lines of code

- time in minutes per phase or task
- defects injected and removed by phase
- schedule planning/tracking with earned value

- All the other TSP measures are derived from these basic measures.



The TSP Launch:

- A 3-day TSP launch or a 2-day TSP re-launch workshop is used to start each project phase.
- The launch workshops are part of the project.
- They are planned and tracked.
- The supervisor and all team members participate.

Purpose of the TSP Launch

- The purpose of the launch process is to establish a common team understanding of the project.
  - the development work to be done
  - management's goals for the project
  - the team and team members' goals
  - the processes the team will use
  - the roles the team members will perform
  - the plan for doing the work
  - the management and customer reporting system
  - the ongoing team communication process

Selecting Roles

- Team Leader
- Development Manager
- Planning Manager
- Quality/Process Manager
- Support Manager

- Customer interface manager
- Design manager
- Test manager
- Safety manager
- Security manager
- Performance manager

Team Leader Responsibilities

- Motivating team members
- Handling customer issues
- Interaction with management
- Day-to-day direction of the work
- Protecting team resources
- Resolving team issues
- Conducting team meetings
- Reporting on the work status

Planning Manager

- Supports and guides the team in planning and tracking their work
  - Lead the team in producing the task plan and schedule for each development cycle
  - Lead the team in producing the balanced team development plan
  - Track the team's progress against their plan

Quality / Process Manager

- Supports the team in defining their process needs, in making the quality plan and in tracking process and product quality
  - Lead the team in producing and tracking their quality plan
  - Identify where quality performance falls short of objectives.
  - Lead the team in defining, documenting, and maintaining their processes and development standards
  - Act as moderator and lead all team reviews and inspections

Support Manager

- Supports the team in determining, obtaining, and managing the tools needed to meet its technology and administrative support needs
  - Lead the team in determining their support needs and obtaining the needed tools and facilities
  - Lead the development and management of Change/Configuration Management System
  - Handle the team's issue and risk tracking system
  - Act as the team's reuse advocate

TSP Quality Guidelines

- Percent (of modules) Defect Free (PDF) at entrance to
  - Compile > 10%
  - Unit Test > 50%
  - Integration Test > 70%
  - System Test > 90%
- Defects/KLOC:
  - Total defects injected 75 - 150; If not PSP trained, use 100 to 200.
  - Compile < 10
  - Unit Test < 5
  - Integration Test < 0.5
  - System Test < 0.2
- Defect Ratios
  - Detailed design review defects /unit test defects > 2.0
  - Code review defects/compile defects > 2.0