UNIT I

Software Process Maturity 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." (But, your numbers must be reasonably meaningful.)

The mere act of measuring human processes changes them because of people's fears, and so forth. Measurements are both expensive and disruptive; overzealous measurements can disrupt the process under study.

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

Software Process Assessment

Process assessments help software organizations improve themselves by identifying their crucial problems and establishing improvement priorities. The basic assessment objectives 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 (the "disappearing problem" syndrome). 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.

The Initial Process(Level1)

Usually ad hoc and chaotic - Organization operates without formalized procedures, cost estimates, and project plans. Tools are neither well integrated with the process nor uniformly 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. Prototying 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.

Process reference models

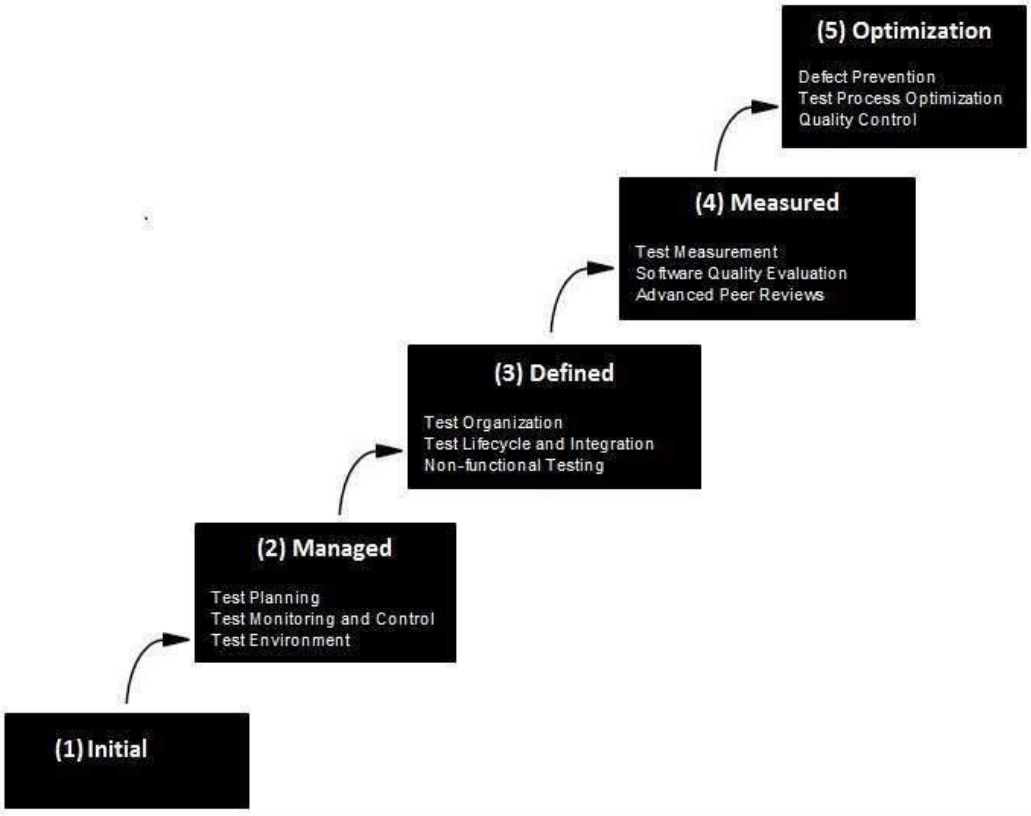
The process framework or reference model acts as an interface between the way the content is organized and the way work is performed. A uniform process model organized under a process reference model makes business modeling and systems designing much easier.

Capability Maturity Model (CMM)

Broadly refers to a **process** improvement approach that is based on a **process model**. **CMM** also refers specifically to the first such **model**, developed by the Software Engineering Institute (SEI) in the mid-1980s, as well as the family of **process models** that followed.

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 timeconsuming 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. 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.

What is CMMI ?

CMM Integration project was formed to sort out the problem of using multiple CMMs. CMMI Product Team's mission was to combine three Source Models into a single improvement framework to be used by the organizations pursuing enterprise-wide process improvement. These three Source Models are :

- Capability Maturity Model for Software (SW-CMM) - v2.0 Draft C
- Electronic Industries Alliance Interim Standard (EIA/IS) - 731 Systems Engineering
- Integrated Product Development Capability Maturity Model (IPD-CMM) v0.98 **CMM Integration:**
- - builds an initial set of integrated models.
- - improves best practices from source models based on lessons learned.
- - establishes a framework to enable integration of future models.

Following are obvious objectives of CMMI:

- **Produce quality products or services:** The process-improvement concept in CMMI models evolved out of the Deming, Juran, and Crosby quality paradigm: Quality products are a result of quality processes. CMMI has a strong focus on qualityrelated activities including requirements management, quality assurance, verification, and validation.
- **Create value for the stockholders:** Mature organizations are more likely to make better cost and revenue estimates than those with less maturity, and then perform in line with those estimates. CMMI supports quality products, predictable schedules, and effective measurement to support management in making accurate and defensible forecasts. This process maturity can guard against project performance problems that could weaken the value of the organization in the eyes of investors.
- **Enhance customer satisfaction:** Meeting cost and schedule targets with high-quality products that are validated against customer needs is a good formula for customer satisfaction. CMMI addresses all of these ingredients through its emphasis on planning, monitoring, and measuring, and the improved predictability that comes with more capable processes.

The CMM Integration is a model that has integrated several disciplines/bodies of knowledge. Currently there are four bodies of knowledge available to you when selecting a CMMI model.

Systems Engineering

Systems engineering covers the development of complete systems, which may or may not include software. Systems engineers focus on transforming customer needs, expectations, and constraints into product solutions and supporting these product solutions throughout the entire lifecycle of the product.

Software Engineering

Software engineering covers the development of software systems. Software engineers focus on the application of systematic, disciplined, and quantifiable approaches to the development, operation, and maintenance of software.

Integrated Product and Process Development

Integrated Product and Process Development (IPPD) is a systematic approach that achieves a timely collaboration of relevant stakeholders throughout the life of the product to better satisfy customer needs, expectations, and requirements. The processes to support an IPPD approach are integrated with the other processes in the organization.

If a project or organization chooses IPPD, it performs the IPPD best practices concurrently with other best practices used to produce products (e.g., those related to systems engineering). That is, if an organization or project wishes to use IPPD, it must select one or more disciplines in addition to IPPD.

Supplier Sourcing

As work efforts become more complex, project managers may use suppliers to perform functions or add modifications to products that are specifically needed by the project. When those activities are critical, the project benefits from enhanced source analysis and from monitoring supplier activities before product delivery. Under these circumstances, the supplier sourcing discipline covers the acquisition of products from suppliers.

Similar to IPPD best practices, supplier sourcing best practices must be selected in conjunction with best practices used to produce products.

CMMI Discipline Selection

Selecting a discipline may be a difficult step and depends on what an organization wants to improve.

- If you are improving your systems engineering processes, like Configuration Management, Measurement and Analysis, Organizational Process Focus, Project Monitoring and Control, Process and Product Quality Assurance, Risk Management, Supplier Agreement Management etc., then you should select Systems engineering (SE) discipline. The discipline amplifications for systems engineering receive special emphasis.
- If you are improving your integrated product and process development processes like Integrated Teaming, Organizational Environment for Integration, then you should select IPPD. The discipline amplifications for IPPD receive special emphasis.
- If you are improving your source selection processes like Integrated Supplier Management then you should select Supplier sourcing (SS). The discipline amplifications for supplier sourcing receive special emphasis.

- If you are improving multiple disciplines, then you need to work on all the areas related to those disciplines and pay attention to all of the discipline amplifications for those disciplines.

The CMMI is structured as follows −

- Maturity Levels (staged representation) or Capability Levels (continuous representation)
- Process Areas
- Goals: Generic and Specific
- Common Features
- Practices: Generic and Specific

This chapter will discuss about two CMMI representations and rest of the subjects will be covered in subsequent chapters.

A representation allows an organization to pursue different improvement objectives. An organization can go for one of the following two improvement paths.

Staged Representation

The staged representation is the approach used in the Software CMM. It is an approach that uses predefined sets of process areas to define an improvement path for an organization. This improvement path is described by a model component called a Maturity Level. A maturity level is a well-defined evolutionary plateau towards achieving improved organizational processes.

CMMI Staged Representation

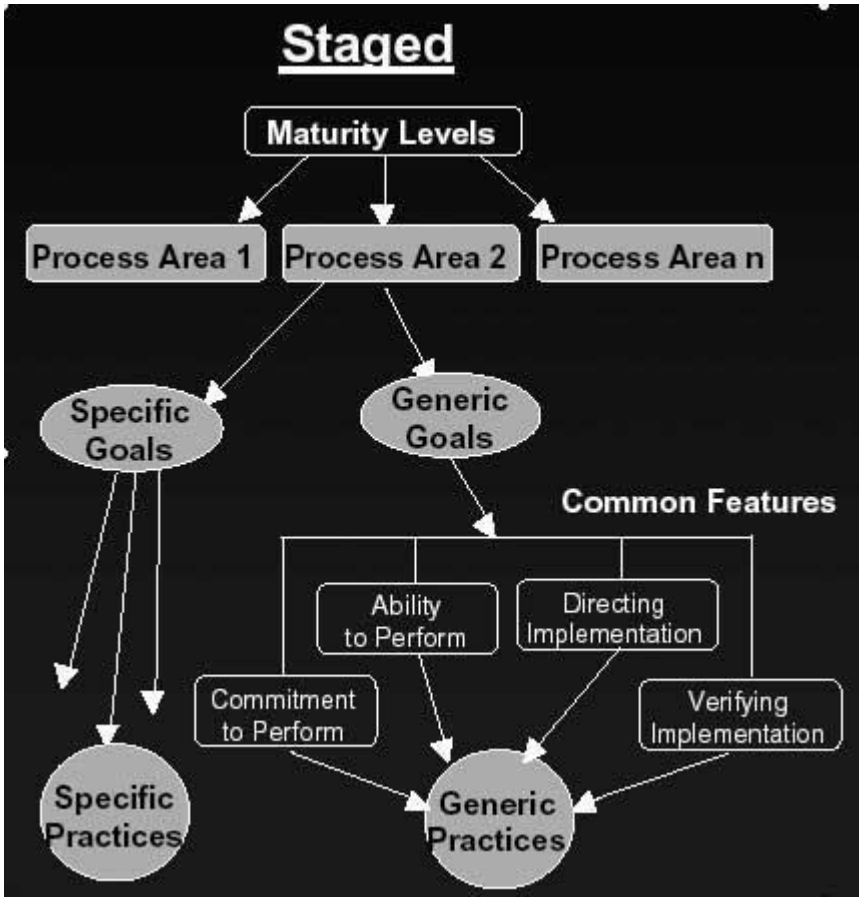
- Provides a proven sequence of improvements, each serving as a foundation for the next.
- Permits comparisons across and among organizations by the use of maturity levels.
- Provides an easy migration from the SW-CMM to CMMI.
- Provides a single rating that summarizes appraisal results and allows comparisons among organizations.

Thus Staged Representation provides a pre-defined roadmap for organizational improvement based on proven grouping and ordering of processes and associated organizational relationships. You cannot divert from the sequence of steps.

CMMI Staged Structure

Following picture illustrates CMMI Staged Model Structure.

rit



Continuous Representation

Continuous representation is the approach used in the SECM and the IPD-CMM. This approach allows an organization to select a specific process area and make improvements based on it. The continuous representation uses Capability Levels to characterize improvement relative to an individual process area.

CMMI Continuous Representation

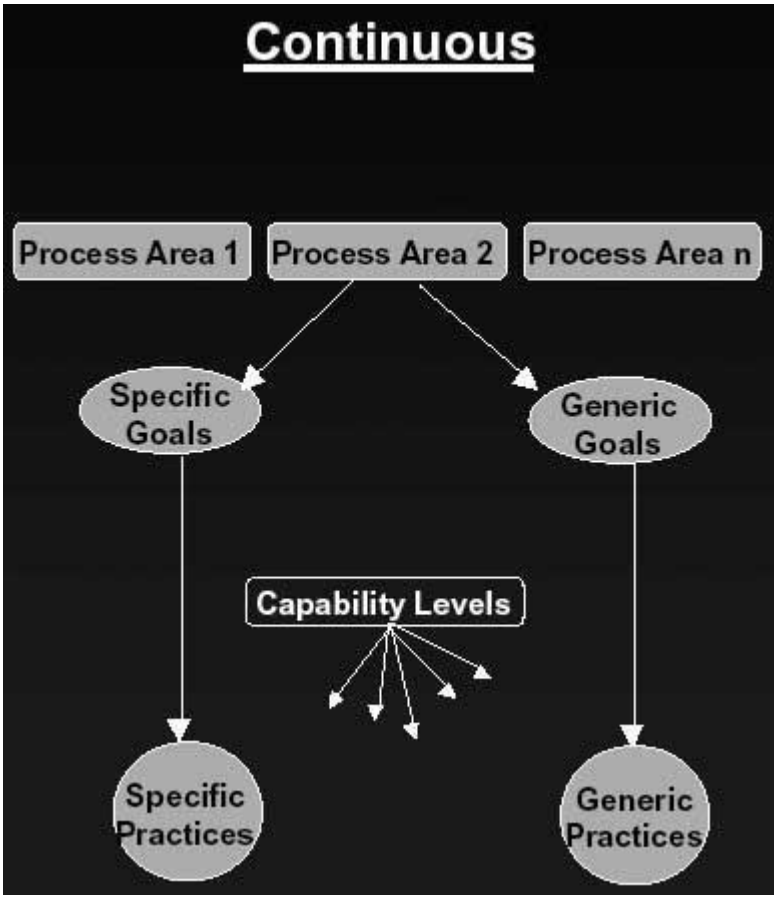
- Allows you to select the order of improvement that best meets your organization's business objectives and mitigates your organization's areas of risk.
- Enables comparisons across and among organizations on a process-area-by-processarea basis.
- Provides an easy migration from EIA 731 (and other models with a continuous representation) to CMMI.

Thus Continuous Representation provides flexibility to organizations to choose the processes for improvement, as well as the amount of improvement required.

CMMI Continuous Structure

The following picture illustrates the CMMI Continuous Model Structure.

rit



Continuous vs Staged Representations

Continuous Representation	Staged Representation
Process areas are organized by process area categories.	Process areas are organized by maturity levels.
Improvement is measured using capability levels. Capability levels measure the maturity of a particular process across an organization; it ranges from 0 through 5.	Improvement is measured using maturity levels. Maturity levels measure the maturity of a set of processes across an organization: it ranges from 1 through 5.
There are two types of specific practices: base and advanced. All specific practices appear in the continuous representation.	There is only one type of specific practice. The concepts of base and advanced practices are not used. All specific practices appear in the staged representation except when a related base-advanced pair of practices appears in the continuous representation, in which case only the advanced practice appears in the staged representation.

Capability levels are used to organize the generic practices.	Common features are used to organize generic practices.
All generic practices are included in each process area.	Only the level 2 and level 3 generic practices are included.
Equivalent staging allows determination of a maturity level from an organization's achievement profile.	There is no need for an equivalence mechanism to back the continuous representation because each organization can choose what to improve and how much to improve using the staged representation.

- **Increase market share:** Market share is a result of many factors, including quality products and services, name identification, pricing, and image. Customers like to deal with suppliers who have a reputation for meeting their commitments.
- **Gain an industry-wide recognition for excellence:** The best way to develop a reputation for excellence is to consistently perform well on projects, delivering quality products and services within cost and schedule parameters. Having processes that conform to CMMI requirements can enhance that reputation.

A maturity level is a well-defined evolutionary plateau toward achieving a mature software process. Each maturity level provides a layer in the foundation for continuous process improvement.

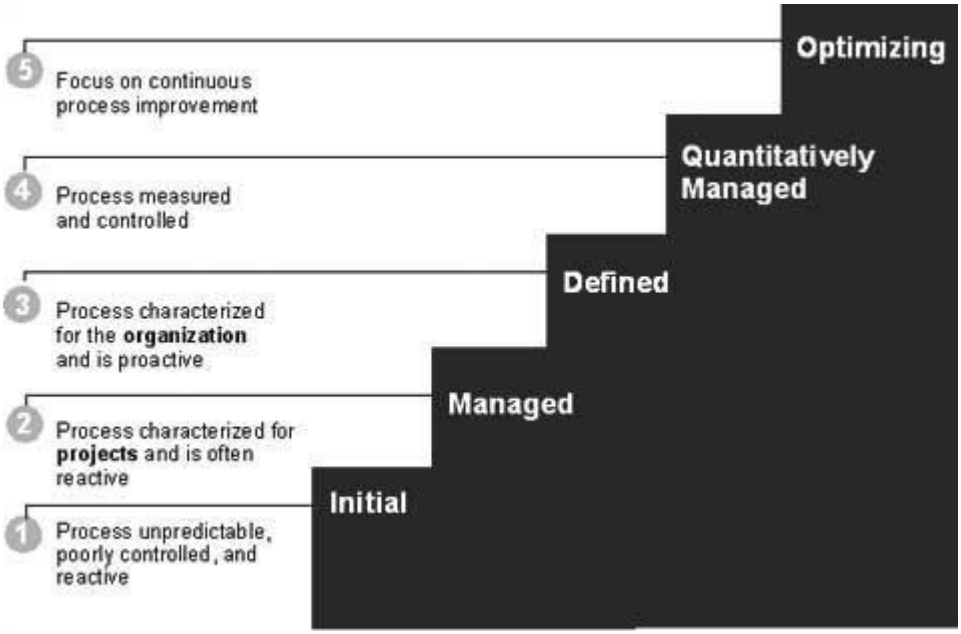
CMMI models with staged representation, have five maturity levels designated by the numbers 1 through 5. They are −

- Initial
- Managed
- Defined
- Quantitatively Managed
- Optimizing

CMMI Staged Representation Maturity Levels

The following image shows the maturity levels in a CMMI staged representation.

rit



Now we will learn the details about each maturity level. Next section will list down all the process areas related to these maturity levels.

Maturity Level Details

Maturity levels consist of a predefined set of process areas. The maturity levels are measured by the achievement of the **specific** and **generic goals** that apply to each predefined set of process areas. The following sections describe the characteristics of each maturity level in detail.

Maturity Level 1 Initial

At maturity level 1, processes are usually ad hoc and chaotic. The organization usually does not provide a stable environment. Success in these organizations depend on the competence and heroics of the people in the organization and not on the use of proven processes.

Maturity level 1 organizations often produce products and services that work; however, they frequently exceed the budget and schedule of their projects.

Maturity level 1 organizations are characterized by a tendency to over commit, abandon processes in the time of crisis, and not be able to repeat their past successes.

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.

The process discipline reflected by maturity level 2 helps to ensure that existing practices are retained during times of stress. When these practices are in place, projects are performed and managed according to their documented plans.

At maturity level 2, requirements, processes, work products, and services are managed. The status of the work products and the delivery of services are visible to management at defined points.

Commitments are established among relevant stakeholders and are revised as needed. Work products are reviewed with stakeholders and are controlled.

The work products and services satisfy their specified requirements, standards, and objectives.

Maturity Level 3 Defined

At maturity level 3, an organization has achieved all the **specific** and **generic goals** of the process areas assigned to maturity levels 2 and 3.

At maturity level 3, processes are well characterized and understood, and are described in standards, procedures, tools, and methods.

A critical distinction between maturity level 2 and maturity level 3 is the scope of standards, process descriptions, and procedures. At maturity level 2, the standards, process descriptions, and procedures may be quite different in each specific instance of the process (for example, on a particular project).

At maturity level 3, the standards, process descriptions, and procedures for a project are tailored from the organization's set of standard processes to suit a particular project or organizational unit. The organization's set of standard processes includes the processes addressed at maturity level 2 and maturity level 3. As a result, the processes that are performed across the organization are consistent except for the differences allowed by the tailoring guidelines.

Another critical distinction is that at maturity level 3, processes are typically described in more detail and more rigorously than at maturity level 2. At maturity level 3, processes are managed more proactively using an understanding of the interrelationships of the process activities and detailed measures of the process, its work products, and its services.

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 and the **generic goals** assigned to maturity levels 2 and 3.

At maturity level 4, sub-processes are selected that significantly contribute to the overall process performance. These selected sub-processes are controlled using statistical and other quantitative techniques.

Quantitative objectives for quality and process performance are established and used as criteria in managing the processes. Quantitative objectives are based on the needs of the customer, end users, organization, and process implementers. Quality and process performance are understood in statistical terms and are managed throughout the life of the processes.

For these processes, detailed measures of process performance are collected and statistically analyzed. Special causes of process variation are identified and, where appropriate, the sources of special causes are corrected to prevent future occurrences.
Quality and process performance measures are incorporated into the organization's measurement repository to support fact-based decision making in the future.

A critical distinction between maturity level 3 and maturity level 4 is the predictability of process performance. At maturity level 4, the performance of processes is controlled using statistical and other quantitative techniques, and is quantitatively predictable. At maturity level 3, processes are only qualitatively predictable.

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 and the **generic goals** assigned to maturity levels 2 and 3.

Processes are continually improved based on a quantitative understanding of the common causes of variation inherent in processes.

This level focuses on continually improving process performance through both incremental and innovative technological improvements.

The quantitative process-improvement objectives for the organization are established, continually revised to reflect changing business objectives, and used as criteria in managing process improvement.

The effects of deployed process improvements are measured and evaluated against the quantitative process-improvement objectives. Both the defined processes and the organization's set of standard processes are targets of measurable improvement activities.

Optimizing processes that are agile and innovative, depends on the participation of an empowered workforce aligned with the business values and objectives of the organization. The organization's ability to rapidly respond to changes and opportunities is enhanced by finding ways to accelerate and share learning. Improvement of the processes is inherently a role that everybody has to play, resulting in a cycle of continual improvement.

A critical distinction between maturity level 4 and maturity level 5 is the type of process variation addressed. At maturity level 4, processes are concerned with addressing special causes of process variation and providing statistical predictability of the results. Though processes may produce predictable results, the results may be insufficient to achieve the established objectives. At maturity level 5, processes are concerned with addressing common causes of process variation and changing the process (that is, shifting the means of the process performance) to improve process performance (while maintaining statistical predictability) to achieve the established quantitative process-improvement objectives.

Maturity Levels Should Not be Skipped

Each maturity level provides a necessary foundation for effective implementation of processes at the next level.

- Higher level processes have less chance of success without the discipline provided by lower levels.
- The effect of innovation can be obscured in a noisy process.

Higher maturity level processes may be performed by organizations at lower maturity levels, with the risk of not being consistently applied in a crisis.

Maturity Levels and Process Areas

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

rit

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
3 Defined	Process Standardization	Requirements Development Technical Solution Product Integration Verification Validation Organizational Process Focus Organizational Process Definition Organizational Training 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)	Medium Quality / Medium Risk

rit

2 Managed	Basic Project Management	Requirements Management Project Planning Project Monitoring and Control Supplier Agreement Management Measurement and Analysis Process and Product Quality Assurance Configuration Management	Low Quality / High Risk
1 Initial	Process is informal and Adhoc		Lowest Quality / Highest Risk

A capability level is a well-defined evolutionary plateau describing the organization's capability relative to a process area. A capability level consists of related specific and generic practices for a process area that can improve the organization's processes associated with that process area. Each level is a layer in the foundation for continuous process improvement.

Thus, capability levels are cumulative, i.e., a higher capability level includes the attributes of the lower levels.

In CMMI models with a continuous representation, there are six capability levels designated by the numbers 0 through 5.

- 0 − Incomplete
- 1 − Performed
- 2 − Managed
- 3 − Defined
- 4 − Quantitatively Managed
- 5 − Optimizing

A short description of each capability level is as follows −

Capability Level 0: Incomplete

An "incomplete process" is a process that is either not performed or partially performed. One or more of the specific goals of the process area are not satisfied and no generic goals exist for this level since there is no reason to institutionalize a partially performed process.
This is tantamount to Maturity Level 1 in the staged representation.

Capability Level 1: Performed

A Capability Level 1 process is a process that is expected to perform all of the Capability Level 1 specific and generic practices. Performance may not be stable and may not meet specific objectives such as quality, cost, and schedule, but useful work can be done. This is only a start, or baby-step, in process improvement. It means that you are doing something but you cannot prove that it is really working for you.

Capability Level 2: Managed

A managed process is planned, performed, monitored, and controlled for individual projects, groups, or stand-alone processes to achieve a given purpose. Managing the process achieves both the model objectives for the process as well as other objectives, such as cost, schedule, and quality. As the title of this level indicates, you are actively managing the way things are done in your organization. You have some metrics that are consistently collected and applied to your management approach.

Note − metrics are collected and used at all levels of the CMMI, in both the staged and continuous representations. It is a bitter fallacy to think that an organization can wait until Capability Level 4 to use the metrics.

Capability Level 3: Defined

A capability level 3 process is characterized as a "defined process." A defined process is a managed (capability level 2) process that is tailored from the organization's set of standard processes according to the organization's tailoring guidelines, and contributes work products, measures, and other process-improvement information to the organizational process assets.

Capability Level 4: Quantitatively Managed

A capability level 4 process is characterized as a "quantitatively managed process." A quantitatively managed process is a defined (capability level 3) process that is controlled using statistical and other quantitative techniques. Quantitative objectives for quality and process performance are established and used as criteria in managing the process. Quality and process performance is understood in statistical terms and is managed throughout the life of the process.

Capability Level 5: Optimizing

An optimizing process is a quantitatively managed process that is improved, based on an understanding of the common causes of process variation inherent to the process. It focuses on continually improving process performance through both incremental and innovative improvements. Both the defined processes and the organization's set of standard processes are the targets of improvement activities.

Capability Level 4 focuses on establishing baselines, models, and measurements for process performance. Capability Level 5 focuses on studying performance results across the organization or entire enterprise, finding common causes of problems in how the work is done (the process[es] used), and fixing the problems in the process. The fix would include updating the process documentation and training involved where the errors were injected.

Organization of Process Areas in Continuous Representation

rit

Category	Process Area
Project Management	• Project Planning • Project Monitoring and Control • Supplier Agreement Management • Integrated Project Management(IPPD) • Integrated Supplier Management (SS) • Integrated Teaming (IPPD) • Risk Management Quantitative Project Management
Support	• Configuration Management • Process and Product Quality Assurance • Measurement and Analysis Causal Analysis and Resolution • Decision Analysis and Resolution • Organizational Environment for Integration (IPPD)
Engineering	• Requirements Management • Requirements Development • Technical Solution • Product Integration • Verification • Validation
Process Management	• Organizational Process Focus • Organizational Process Definition • Organizational Training • Organizational Process Performance • Organizational Innovation and Deployment
PCMM:	•

The People Capability Maturity Model (People CMM, P-CMM) is part of the CMMI product family of process maturity models. It is a framework to guide organisations in improving their processes for managing and developing human workforces. It helps organisations to characterize the maturity of their workforce practices, establish a program of continuous workforce development, set priorities for improvement actions, integrate workforce development with Process Improvement, and establish a culture of excellence. PCMM is
based on proven practices in fields of human resources, knowledge management, and organisational development.

rit

P-CMM is part of the CMMI product family of process maturity models. It describes a progression for continuous improvement and process improvement of the HR processes for managing and developing human workforces.

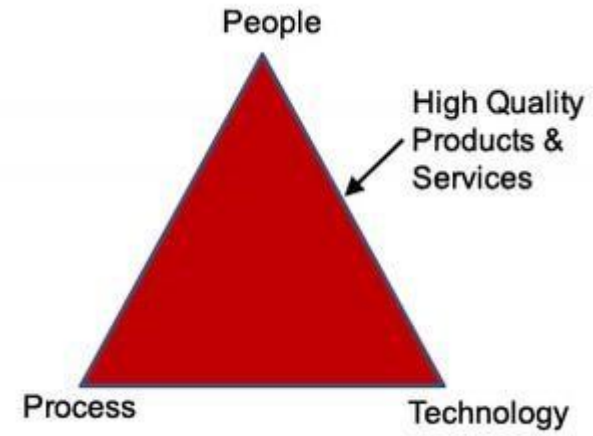
The P-CMM framework enables organisations to incrementally focus on key process areas and to lay foundations for improvement in workforce practices. Unlike other HR models, P-CMM requires that key process areas, improvements, interventions, policies, and procedures are institutionalised across the organisation — irrespective of function or level.

Therefore, all improvements have to percolate throughout the organisation, to ensure consistency of focus, to place emphasis on a participatory culture, embodied in a team-based environment, and encouraging individual innovation and creativity. Process Maturity Rating

The process maturity rating is from *ad hoc and inconsistently performed* practices, to a mature and disciplined development of the knowledge, skills, and motivation of the workforce.

Traditionally, process maturity models like ISO/IEC 15504 or CMMI focus on organisational improvement with respect to operational (Product) Development Processes. PCMM in contrast focus instead on the three prominent factors for operational capability to deliver successful products and services:

1. People
2. Process
3. Products, Technology



Thus, these 3Ps of PCMM are comparable to ITIL's 4Ps: ***P**eople*, ***P**rocesses*, ***P**roducts* (tools and technology) and ***P**artners* (suppliers, vendors, and outsourcing organisations). P-CMM is characterised by a holistic approach to people-related issues. Instead of looking at traditional Human Resource interventions in a reactionary scrappy fashion. The P-CMM framework enables organisations to incrementally focus on key process areas and to lay foundations for improvement in workforce practices.

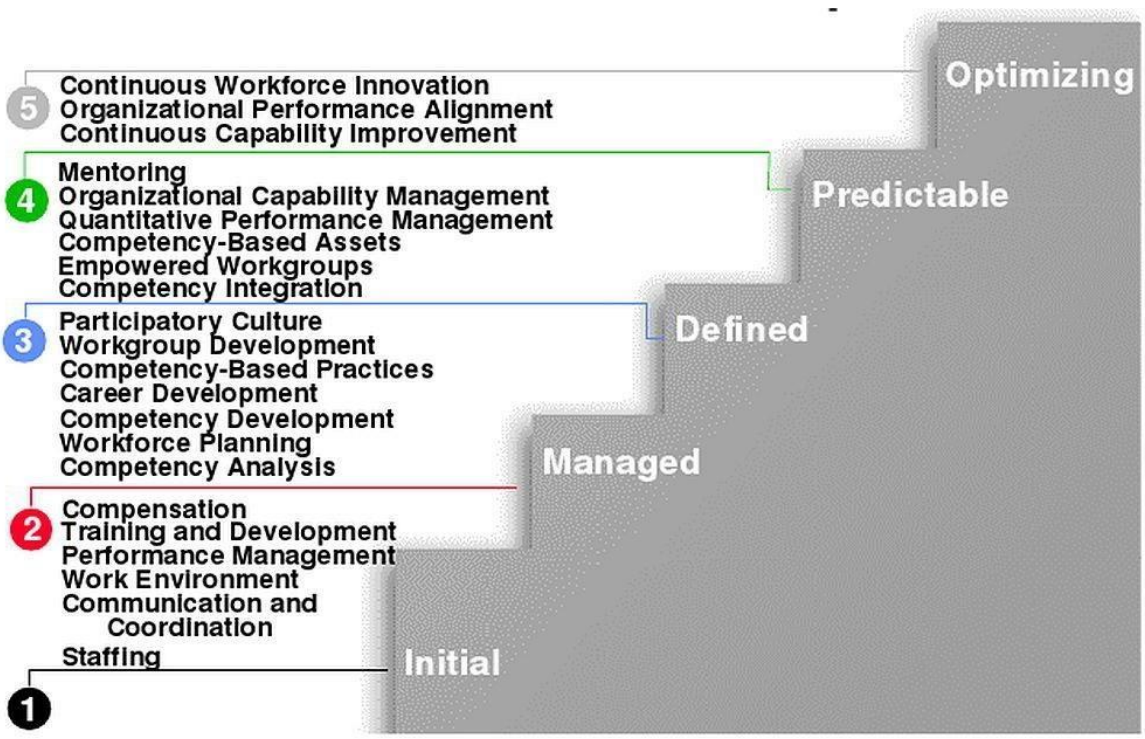
rit

Unlike other HR models, P-CMM requires that key process areas, improvements, interventions, policies, and procedures are institutionalized across the organisation — irrespective of function or level. Therefore, all improvements have to percolate throughout the organisation, to ensure consistency of focus, to place emphasis on a participatory culture, embodied in a team-based environment, and encouraging individual innovation and creativity.

P-CMM Maturity Levels

Like other staged maturity models of the CMMI product family developed at the Software Engineering Institute (SEI) at Carnegie-Mellon University, the P-CMM consists of maturity levels that establish successive foundations for continuously improving workforce competencies.

These range from initial (Level 1), where workforce practices are performed inconsistently or ritualistically and frequently fail to achieve their intended purpose, to the optimised level (Level 5), where everyone in the organisation is focused on continuously improving their Capability and the organisation's workforce practices.



The five maturity levels of the People CMM are:

rit

Levels	People CMM Threads			
	Developing competency	Building workgroups& culture	Motivating & managing performance	Shaping the workforce
5 Optimizing	Continuous Capability Improvement		Organisational Performance Alignment	Continuous Workforce Innovation
4 Predictable	Competency Based Assets Mentoring	Competency Integration Empowered workgroups	Quantitative Performance Management	Organisational Capability Management
3 Defined	Competency development Competency Analysis	Workgroup Development Participatory Culture	Competency Based Practices Career Development	Workforce Planning
2 Managed	Training and Development	Communication & Coordination	Compensation Performance Management Work Environment	Staffing

1. **Initial**.
2. **Repeatable**. The key process areas at Level 2 focus on instilling basic discipline into workforce activities. They are: ○ Work Environment ○ Communications ○ Staffing
   - ○ Performance Management ○ Training ○ Compensation
3. **Defined**. The key process areas at Level 3 address issues surrounding the identification of the organisation's primary competencies and aligning its people management activities with them. They are:
   - ○ Knowledge and Skills Analysis
   - ○ Workforce Planning ○ Competency Development ○ Career Development ○ Competency-Based Practices ○ Participatory Culture
4. **Managed**. The key process areas at Level 4 focus on quantitatively managing organisational growth in people management capabilities and in establishing competency-based teams. They are:
   - ○ Mentoring ○ Team Building
   - ○ Team-Based Practices
   - ○ Organisational Competency Management ○ Organisational ○ Performance Alignment

5. **Optimising**. The key process areas at Level 5 cover the issues that address continuous improvement of methods for developing competency, at both the organisational and the individual level. They are:
    - Personal Competency Development
    - Coaching
    - Continuous Workforce Innovation

PSP

The **Personal Software Process** (**PSP**) is a structured software development process that is designed to help software engineers better understand and improve their performance by bringing discipline to the way they develop software and tracking their predicted and actual development of the code. It clearly shows developers how to manage the quality of their products, how to make a sound plan, and how to make commitments. It also offers them the data to justify their plans. They can evaluate their work and suggest improvement direction by analyzing and reviewing development time, defects, and size data. The PSP was created by Watts Humphrey to apply the underlying principles of the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) to the software development practices of a single developer. It claims to give software engineers the process skills necessary to work on a team software process (TSP) team.

The PSP aims to provide software engineers with disciplined methods for improving personal software development processes. The PSP helps software engineers to:

- Improve their estimating and planning skills.
- Make commitments they can keep.
- Manage the quality of their projects.
- Reduce the number of defects in their work.
- PSP training follows an evolutionary improvement approach: an engineer learning to integrate the PSP into his or her process begins at the first level – PSP0 – and progresses in process maturity to the final level – PSP2.1. Each Level has detailed scripts, checklists and templates to guide the engineer through required steps and helps the engineer improve their own personal software process. Humphrey encourages proficient engineers to customize these scripts and templates as they gain an understanding of their own strengths and weaknesses.
- **Process**

  The input to PSP is the requirements; requirements document is completed and delivered to the engineer.
- **PSP0, PSP0.1 (Introduces process discipline and measurement)**
- PSP0 has 3 phases: planning, development (design, code, compile, test) and a post mortem. A baseline is established of current process measuring: time spent on programming, faults injected/removed, size of a program. In a post mortem, the engineer ensures all data for the projects has been properly recorded and analysed. PSP0.1 advances the process by adding a coding standard, a size measurement and the development of a personal process improvement plan (PIP). In the PIP, the engineer records ideas for improving his own process.
- **PSP1, PSP1.1 (Introduces estimating and planning)**

- Based upon the baseline data collected in PSP0 and PSP0.1, the engineer estimates how large a new program will be and prepares a test report (PSP1). Accumulated data from previous projects is used to estimate the total time. Each new project will record the actual time spent. This information is used for task and schedule planning and estimation (PSP1.1).
- **PSP2, PSP2.1 (Introduces quality management and design)**
- PSP2 adds two new phases: design review and code review. Defect prevention and removal of them are the focus at the PSP2. Engineers learn to evaluate and improve their process by measuring how long tasks take and the number of defects they inject and remove in each phase of development. Engineers construct and use checklists for design and code reviews. PSP2.1 introduces design specification and analysis techniques
- (PSP3 is a legacy level that has been superseded by TSP.)

One of the core aspects of the PSP is using historical data to analyze and improve process performance. PSP data collection is supported by four main elements:

- Scripts
- Measures
- Standards
- Forms

The PSP scripts provide expert-level guidance to following the process steps and they provide a framework for applying the PSP measures. The PSP has four core measures:

- Size – the size measure for a product part, such as lines of code (LOC).
- Effort – the time required to complete a task, usually recorded in minutes.
- Quality – the number of defects in the product.
- Schedule – a measure of project progression, tracked against planned and actual completion dates.

Applying standards to the process can ensure the data is precise and consistent. Data is logged in forms, normally using a PSP software tool. The SEI has developed a PSP tool and there are also open source options available, such as Process Dashboard.

The key data collected in the PSP tool are time, defect, and size data – the time spent in each phase; when and where defects were injected, found, and fixed; and the size of the product parts. Software developers use many other measures that are derived from these three basic measures to understand and improve their performance. Derived measures include:

- estimation accuracy (size/time)
- prediction intervals (size/time)
- time in phase distribution
- defect injection distribution
- defect removal distribution
- productivity
- reuse percentage
- cost performance index
- planned value
- earned value
- predicted earned value
- defect density

rit

- defect density by phase
- defect removal rate by phase
- defect removal leverage
- review rates
- process yield
- phase yield
- failure cost of quality (COQ)
- appraisal COQ
- appraisal/failure COQ ratio

Planning and tracking

Logging time, defect, and size data is an essential part of planning and tracking PSP projects, as historical data is used to improve estimating accuracy.

The PSP uses the PROxy-Based Estimation (PROBE) method to improve a developer's estimating skills for more accurate project planning. For project tracking, the PSP uses the earned value method.

The PSP also uses statistical techniques, such as correlation, linear regression, and standard deviation, to translate data into useful information for improving estimating, planning and quality. These statistical formulas are calculated by the PSP tool.

Using the PSP

The PSP is intended to help a developer improve their personal process; therefore PSP developers are expected to continue adapting the process to ensure it meets their personal needs.

PSP and the TSP

In practice, PSP skills are used in a TSP team environment. TSP teams consist of PSP-trained developers who volunteer for areas of project responsibility, so the project is managed by the team itself. Using personal data gathered using their PSP skills; the team makes the plans, the estimates, and controls the quality.

Using PSP process methods can help TSP teams to meet their schedule commitments and produce high quality software. For example, according to research by Watts Humphrey, a third of all software projects fail,[3] but an SEI study on 20 TSP projects in 13 different organizations found that TSP teams missed their target schedules by an average of only six percent.[4]

Successfully meeting schedule commitments can be attributed to using historical data to make more accurate estimates, so projects are based on realistic plans – and by using PSP quality methods, they produce low-defect software, which reduces time spent on removing defects in later phases, such as integration and acceptance testing.

PSP and other methodologies[edit]

The PSP is a personal process that can be adapted to suit the needs of the individual developer. It is not specific to any programming or design methodology; therefore it can be used with different methodologies, including Agile software development.

Software engineering methods can be considered to vary from predictive through adaptive. The PSP is a predictive methodology, and Agile is considered adaptive, but despite their differences, the TSP/PSP and Agile share several concepts and approaches – particularly in regard to team organization. They both enable the team to:

- Define their goals and standards.
- Estimate and schedule the work.
- Determine realistic and attainable schedules. Make plans and process improvements.

Both Agile and the TSP/PSP share the idea of team members taking responsibility for their own work and working together to agree on a realistic plan, creating an environment of trust and accountability. However, the TSP/PSP differs from Agile in its emphasis on documenting the process and its use of data for predicting and defining project schedules.

Quality[edit]

High-quality software is the goal of the PSP, and quality is measured in terms of defects. For the PSP, a quality process should produce low-defect software that meets the user needs.

The PSP phase structure enables PSP developers to catch defects early. By catching defects early, the PSP can reduce the amount of time spent in later phases, such as Test.

The PSP theory is that it is more economical and effective to remove defects as close as possible to where and when they were injected, so software engineers are encouraged to conduct personal reviews for each phase of development. Therefore, the PSP phase structure includes two review phases:

- Design Review
- Code Review

To do an effective review, you need to follow a structured review process. The PSP recommends using checklists to help developers to consistently follow an orderly procedure.

The PSP follows the premise that when people make mistakes, their errors are usually predictable, so PSP developers can personalize their checklists to target their own common errors. Software engineers are also expected to complete process improvement proposals, to identify areas of weakness in their current performance that they should target for improvement. Historical project data, which exposes where time is spent and defects introduced, help developers to identify areas to improve.

PSP developers are also expected to conduct personal reviews before their work undergoes a peer or team review.

TSP

The **team software process** (**TSP**) provides a defined operational process framework that is designed to help teams of managers and engineers organize projects and produce software the principles products that range in size from small projects of several thousand lines of code (KLOC) to very large projects greater than half a million lines of code. The TSP is intended to improve the levels of quality and productivity of a team's software development project, in order to help them better meet the cost and schedule commitments of developing a software system

The initial version of the TSP was developed and piloted by Watts Humphrey in the late 1990s and the Technical Report for TSP sponsored by the U.S. Department of Defense was published

in November 2000. The book by Watts Humphrey, Introduction to the Team Software Process, presents a view of the TSP intended for use in academic settings, that focuses on the process of building a software production team, establishing team goals, distributing team roles, and other teamwork-related activities.

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.

How TSP works

Before engineers can participate in the TSP, it is required that they have already learned about the PSP, so that the TSP can work effectively. Training is also required for other team members, the team lead and management. The TSP software development cycle begins with a planning process called the launch, led by a coach who has been specially trained, and is either certified or provisional. The launch is designed to begin the team building process, and during this time teams and managers establish goals, define team roles, assess risks, estimate effort, allocate tasks, and produce a team plan. During an execution phase, developers track planned and actual effort, schedule, and defects meeting regularly (usually weekly) to report status and revise plans. A development cycle ends with a Post Mortem to assess performance, revise planning parameters, and capture lessons learned for process improvement.

The coach role focuses on supporting the team and the individuals on the team as the process expert while being independent of direct project management responsibility. The team leader role is different from the coach role in that, team leaders are responsible to management for products and project outcomes while the coach is responsible for developing individual and team performance.

UNIT 2

Software Project Management Renaissance

Conventional Software Management :

Conventional software management practices are sound in theory, but practice is still tied to archaic (outdated) technology and techniques.

Conventional software economics provides a benchmark of performance for conventional software manage- ment principles.