Syllabus: To Understand the importance of Software Project Management

1.1.1 Explain Software Project Management Framework 1.1.2 Describe methods to Estimate project time and cost 1.1.3 Describe about Resource Management 1.1.4 Describe how Project Risks can be identified, analyzed, mitigated, and monitored 1.1.5 Describe how project quality can be ensured and managed 1.1.6 Describe about Configuration Management 1.1.7 Describe change management 1.1.8 Explain about CMMI, different levels and need of accreditation

---

1.1.1 Explain Software Project Management Framework

SOFTWARE PROJECT MANAGEMENT

- The main goal of software project management is to enable a group of software developers to work efficiently towards the successful completion of the project.

RESPONSIBILITIES OF A PROJECT MANAGER

- Software project managers take the overall responsibility of steering project to success.
- Classified into two:
  1. Project planning
  2. Project monitoring and control activities

Project planning:

- Estimating several characteristics of the project and then planning the project activities based on the estimates made.
- Project planning is undertaken after the feasibility study and before the requirement analysis and specification phase.
- The initial project plans that are made are revised from time to time.

Project monitoring and control activities:

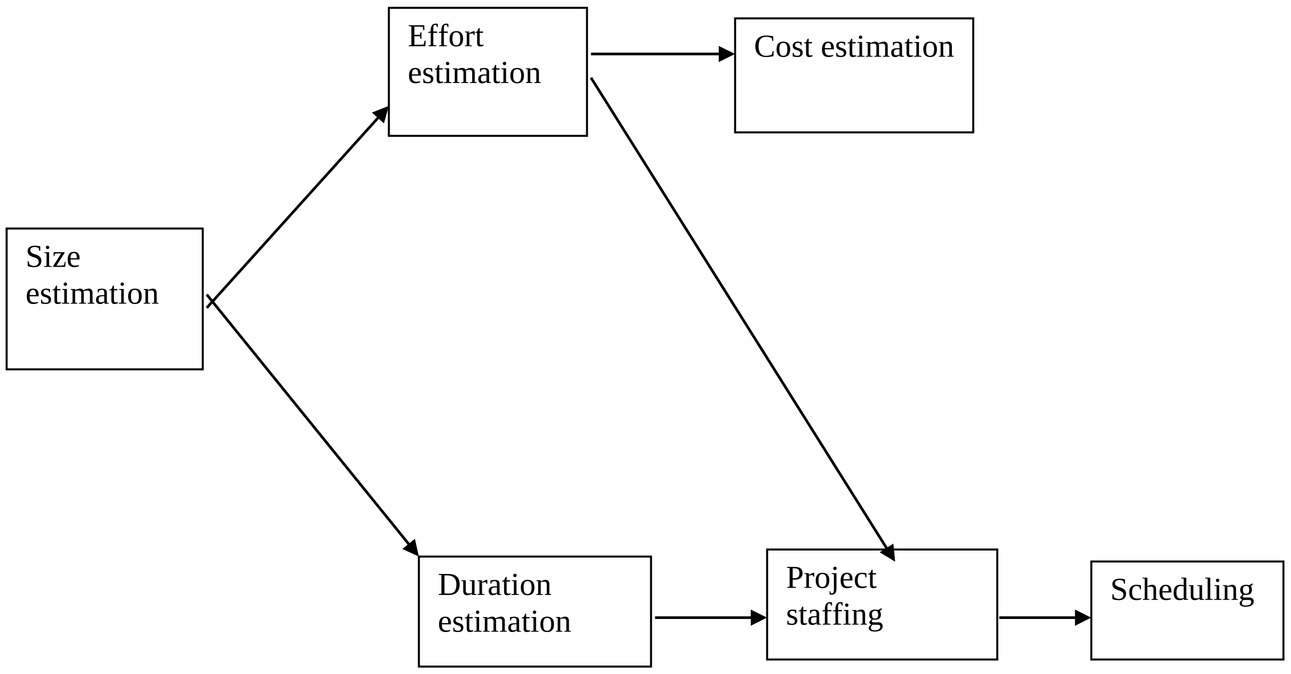
- Are undertaken once the development activities start.
- Ensures that the development proceeds as per plan.
- The plan is changed whenever required to cope up with the situation at hand.

PROJECT PLANNING

- Project planning is undertaken and completed even before any development activity starts.
- Consist of following activities
  1. Estimation: The following project attributes have to be estimated.
     i. *Cost* How much is it going to cost to develop the software?
     ii. *Duration* How long is it going to take to develop the product?
     iii. *Effort* How much effort would be required to develop the product?
  2. Scheduling: Schedules for manpower and resources were developed.
  3. Staffing: Staff organization and staff plans have to be made.
  4. Risk Management: Risk identification, analysis and planning have to be done.
  5. Miscellaneous Plans: Other plans such as quality assurance, configuration management, etc. have to be done.
- Observe that size estimation is the first activity.

- Size is the most fundamental parameter based on which all other estimates are made.
- Based on size estimation, the effort required to complete the project and duration are estimated.
- Based on the effort estimation the cost of the project is computed.
- The estimated cost forms the negotiations with customer made.

Precedence ordering among planning activities



- Other planning activities such as staffing, scheduling etc, are undertaken based on the estimates made.
- For large projects the managers plan over a number of stages this is called as *Sliding Window Planning*.

1.1.2 <u>Describe methods to Estimate project time and cost</u>

<u>Project Estimation</u>

For an effective management accurate estimation of various measures is a must. With correct estimation managers can manage and control the project more efficiently and effectively.

Project estimation may involve the following:

- ✓ Software size estimation
- ✓ Effort estimation
- ✓ Time estimation

  The sum of time required to complete all tasks in hours or days is the total time invested to complete the project.
- ✓ Cost estimation

- **Software size estimation** Software size may be estimated either in terms of KLOC (Kilo Line of Code) or by calculating number of function points in the software. Lines of code depend upon coding practices and Function points vary according to the user or software requirement.

- **Effort estimation** The managers estimate efforts in terms of personnel requirement and man-hour required to produce the software. For effort estimation software size should be known. This can either be derived by managers' experience, organization's historical data or software size can be converted into efforts by using some standard formulae.
- **Time estimation** Once size and efforts are estimated, the time required to produce the software can be estimated. Efforts required is segregated into sub categories as per the requirement specifications and interdependency of various components of software. Software tasks are divided into smaller tasks, activities or events by Work Breakthrough Structure (WBS). The tasks are scheduled on day-to-day basis or in calendar months.
  The sum of time required to complete all tasks in hours or days is the total time invested to complete the project.
- **Cost estimation** This might be considered as the most difficult of all because it depends on more elements than any of the previous ones. For estimating project cost, it is required to consider -
  - Size of software
  - Software quality
  - Hardware
  - Additional software or tools, licenses etc.
  - Skilled personnel with task-specific skills
  - Travel involved
  - Communication
  - Training and support

Metrics for Project Size Estimation

- The project size is a measure of the problem complexity in terms of the effort and time required to develop a product.
- Currently 2 metrics are used:
  1. Lines of Code(LOC)
  2. Function Point Metric

1) **Lines of Code (LOC)**

- Simplest among all metrics.
- Used to estimate project size.
- This is extremely popular metric.
- Measures size by counting the number of source instructions in the developed program.
- Commenting and header lines are ignored.
- LOC count at the beginning of a project is very difficult one would have to guess.

- In a problem that was divided into modules, need to predict the LOC count at different sub modules.

Shortcomings

- LOC gives numerical value of problem size that vary with individual coding style.
- Nothing to do with quality and efficiency of the code.
- Measures only the lexical complexity of the program and not the logical or structural complexity.
- Difficult to estimate LOC in the final product from the product specification.

2) Function Point metrics

- Proposed by Albrecht(1983).
- Can estimate size of a software product from the problem specification.
- In function point metric size of a s/w is directly depend on the number of different functions or features it support.
- The size also depends on the number of files and number of interfaces.
- Function point computed in 3 steps.
- First step to compute Un adjusted function Point(UFP).
- Next step, UFP is refined to reflect the differences.
- Third step, FP is computed by further refining.

UFP = (Number of inputs)*4 + (Number of outputs)*5 + (Number of inquiries)*4 + (Number of files)*10 + (Number of Interfaces)*10

Number of inputs : Eachdata item input by the user is counted.

Number of outputs : Refers toreports printed, error messages etc

Number of inquiries : Number of distinctinteractive queries which can be made by users.

Number of files : Each logical file is counted.A logical file includesdata structures and physical files

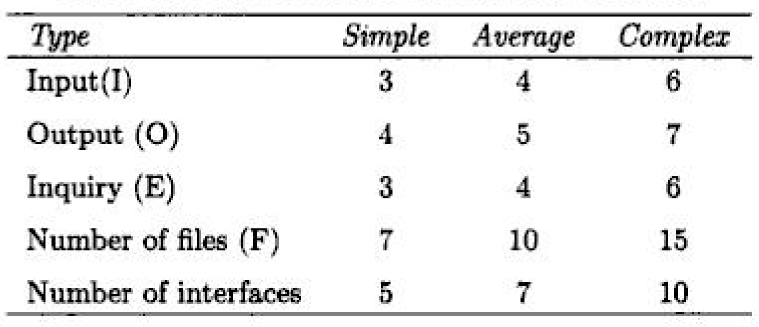
Number of Interfaces : Interfaces used to exchange information with other systems. Eg: disks, communication links.

- The complexity levels of each parameter are graded as simple, average or complex.
- Technical complexity factor is calculated (TCF)
- FP = UFP* TCF
- There is a chance that different project managers arrive at different function point for the same problem.

Feature Point Metrics

- An extension of FP metric called feature point metric.
- Feature point metric incorporates algorithm complexity as an extra parameter.

Table 3.1: Refinement of function point entities



Type	*Simple*	*Average*	*Complex*
Input(I)	3	4	6
Output (O)	4	5	7
Inquiry (E)	3	4	6
Number of files (F)	7	10	15
Number of interfaces	5	7	10

PROJECT ESTIMATION TECHNIQUES

Estimations of various project parameters like project size, effort required, duration and cost are done. It forms the basis for cost, resource planning and scheduling.

- There are 3 board categories of estimation techniques:

1. Empirical Estimation Technique
2. Heuristic Technique
3. Analytical Estimation Technique

I) Empirical Estimation Technique

- Based on making an educated guess of the project parameters.
- Past experience is considered with development of similar products.
- Techniques are based on common sense.
- 2 popular empirical Estimation techniques are :*Expert judgement and Delphi Estimation.*

1. **Expert Judgement Technique**

- Is one of the most widely used estimation technique.
- The expert estimates the cost of different components that would make up the system and then combines the individual modules to arrive at overall estimate.
- Expert makes guess of problem size after analysing problem thoroughly.
- Subject to human errors and individual favor.
- More refined form of expert judgment is the estimation by group of experts.
- Estimation by group of experts minimizes factors such as individual oversight, personal favor etc.

2. **Delphi Cost Estimation**

- Tries to overcome the shortcomings of expert judgement approach.
- Carried out by team – group of experts and a coordinator
- Coordinator provides each estimator with a copy of SRS and a form for recording his cost estimate.
- Estimators complete their estimations secretly and submit them to the coordinator.
- Coordinator prepares summary of responses and include any unusual validations noted by any estimators and is distributed to the estimators again to re estimate.

- Based on the summary estimators re-estimate.
- This process is repeated for several rounds.
- No discussion is allowed between estimators.
- After completion of several iterations of estimations, final estimation is done by the co-ordinator.

II) Heuristic Technique

- Relationship among the different project parameters can be modeled using suitable mathematical notations.
- Once the basic(independent) parameters are known , the other(dependent) parameter can be easily determined by submitting the value of the basic parameters in the mathematical expression.
- Different heuristic estimation model can be divided into 2 classes.: *Single variable model and multivariable model.*
- Single variable model provide a means to estimate the desired characteristic of a problem, using some previously estimated basic.
- multivariable model give more accurate estimates compared to single variable.
- COCOMO model is an example of multivariable estimation model.

COCOMO – A Heuristic Estimation Technique

- COCOMO (COnstructive COst estimation Model) was proposed by Boehm.
- Bohem postulate that any software project can be classified into any one of the following 3 categories based on their complexity .: *organic, semidetached and embedded.*
- Considers not only the characteristic of the product but also of the development team and development environment.

  1.Organic: Project to be organic if it deals with well understood application program, size of team is reasonably small and team members are experienced in developing similar types of projects.

  2. Semidetached: A development project can be considered to be of semidetached type, if the development team consists of a mixture of experienced and inexperienced staff. Team members may have limited experience on related systems but may be unfamiliar with some aspects of the system being developed.

  3.Embedded: A development project is considered to be of embedded type, if the softwarebeing developed has complex hardware, or if stringth regulations on the operational procedures exist.
- Bohem provides different sets of expressions to predict the effort (in unit of person-months) and development time from the size estimation given in KLOC(Kilo Lines of Source Code).
- One person-month is the effort an individual can typically put in a month.
- Effort estimation is expressed in units of person-months(PM).

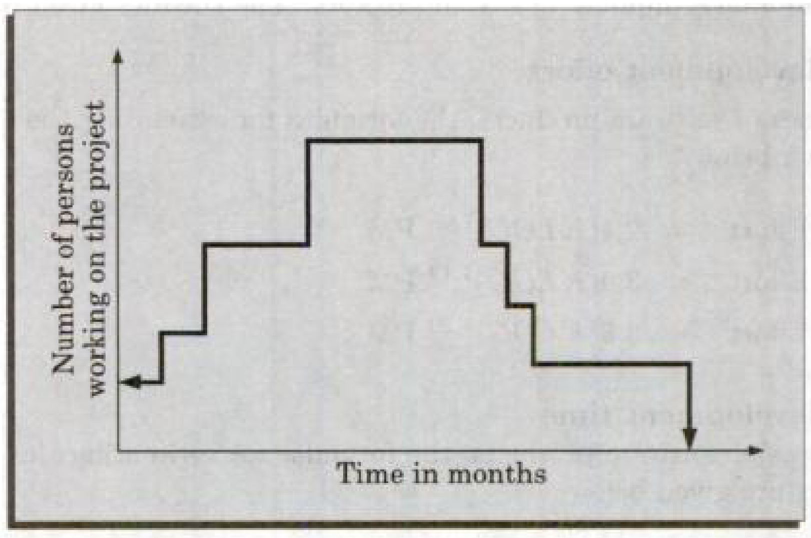


Figure 3.3: Person-month curve.

- PM is considered as an appropriate unit because developers are assigned to a project for a certain number of months.
- The effort estimation simply denotes the area under the person-month curve for the project. The plot shows that the different number of personnel may work at different point in the project development.
- The number of personal working on the project usually increases and decreases by an integral number, resulting in the sharp edges in the plot.
- Software cost estimation should be done through three stages : *basic COCOMO, intermediate COCOMO and complete COCOMO*.

Basic COCOMO MODEL

- Gives an approximate estimate of project parameters

$$\text{Effort} = a_1 \times (KLOC)^{a_2} \text{ PM}$$

$$\text{Tdev} = b_1 \times (Effort)^{b_2} \text{ Months}$$

Where

(a) KLOC is the estimated size of the software product expressed in Kilo Lines of Code,

(b) a_1, a_2, b_1, b_2 are constants for each category of software products,

(c) Tdev is the estimated time to develop the software, expressed in months,

(d) Effort is the total effort required to develop the software product, expressed in person months (PMs).

- Every source text should be calculated as one LOC .
- Thus, if a single instruction spans several lines (say n lines) it is considered to be n LOC.
- The values of a1,a2,b1,b2for different categories of products as given by Bohem

- The theories given by Bohem were:

Estimation of development effort

For the three classes of software products, the formulas for estimating the effort based on the code size are shown below:

Organic	: Effort	$= 2.4(KLOC)^{1.05}$ PM
Semidetached	: Effort	$= 3.0(KLOC)^{1.12}$ PM
Embedded	: Effort	$= 3.6(KLOC)^{1.20}$ PM

Estimation of development time

For the three classes of software products, the formulas for estimating the development time based on the effort are given below:

Organic	: Tdev	$= 2.5(Effort)^{0.38}$ Months
Semidetached	: Tdev	$= 2.5(Effort)^{0.35}$ Months
Embedded	: Tdev	$= 2.5(Effort)^{0.32}$ Months

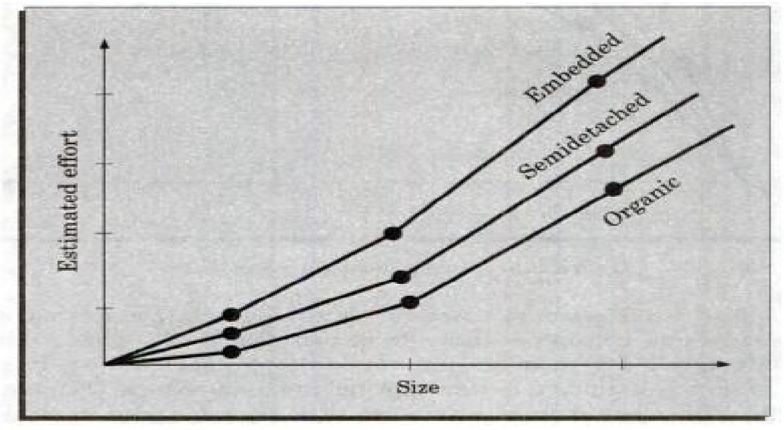


Figure 3.4: Effort versus product size.

- From this diagram the effort required to develop a product increases rapidly with project size.

> The effort and duration values computed by COCOMO are the values for doing the work in the shortest time without unduly increasing manpower cost.

- When the size of the product increases by two times the time to develop the product does not double but rises moderately.
- COCOMO assumes that a project is carried out not by a single person but by team of developers.

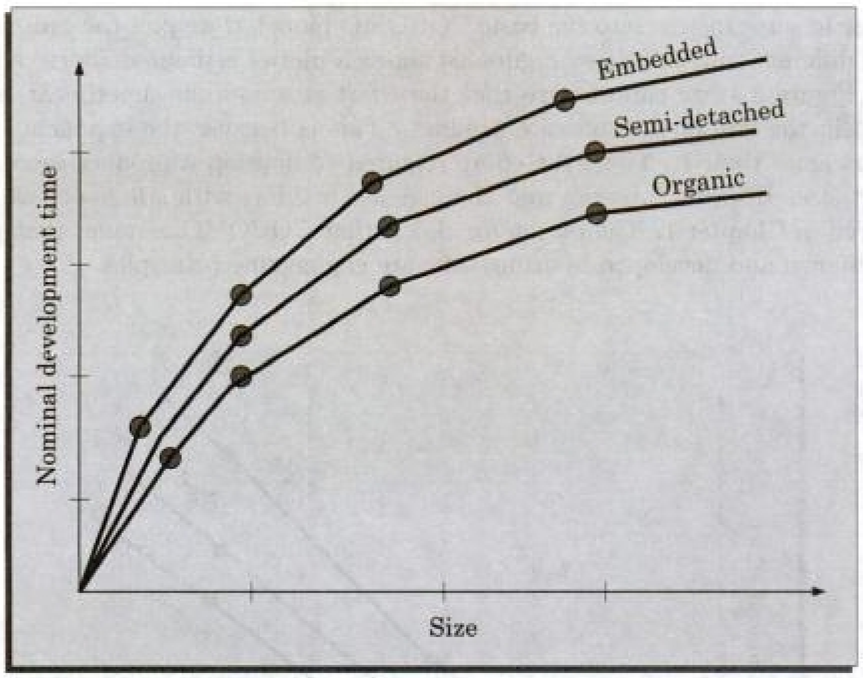
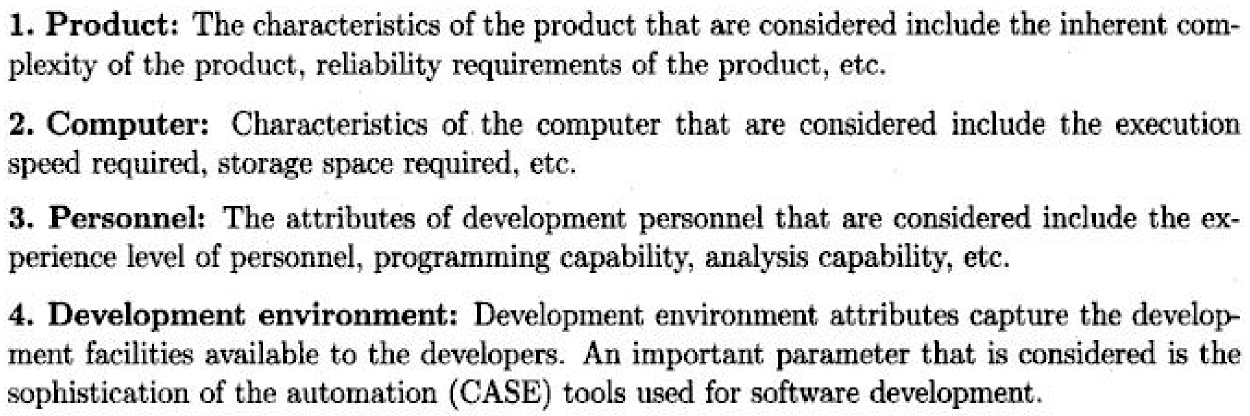


Figure 3.5: Development time versus size.

- Note that the effort and duration estimations obtained using the COCOMO model imply that if you try to complete the project in a time shorter than estimated duration, then the cost will increase drastically.
- If you complete the project over a longer period of time than that estimated, then there is almost no decrease in the estimated cost value.
- An optimum sized team is one in which developers do not have to wait idle waiting for work but at the same time employs as many developers as possible.
- That is why the duration given by COCOMO is called the nominal duration or optimal duration.

Intermediate COCOMO

- The basic COCOMO model assumes that effort and development time are functions of the product size alone.
- To obtain an accurate estimation of the effort and project duration the effect of all relevant parameters take into account.
- The intermediate COCOMO model recognizes this fact refines the initial estimate obtained using the basic COCOMO expressions by using a set of 15 cost drivers (multipliers) based on various attributes of software development.
- Bohem requires the project manager to rate these 15 different parameters for a particular project on a scale of one to three.
- Depending on these ratings, he suggests appropriate cost driver values which should be multiplied with initial estimate obtained using basic COCOMO.
- The cost drivers can be classified as being attributes of the following items:



1. Product: The characteristics of the product that are considered include the inherent complexity of the product, reliability requirements of the product, etc.

2. Computer: Characteristics of the computer that are considered include the execution speed required, storage space required, etc.

3. Personnel: The attributes of development personnel that are considered include the experience level of personnel, programming capability, analysis capability, etc.

4. Development environment: Development environment attributes capture the development facilities available to the developers. An important parameter that is considered is the sophistication of the automation (CASE) tools used for software development.

Complete COCOMO Model

- A major shortcoming of both the basic and intermediate COCOMO models is that they consider a software product as a single homogeneous entity.
- Most large systems made up of several smaller subsystems and these subsystems have widely different characteristics.
- Complete COCOMO model considers the differences in the characteristics of the subsystems and estimate the effort and development time as the sum of the estimates for the individual sub systems.
- Cost of each sub system is estimated separately.
- This approach reduces the margin of error in the final estimate.
- Eg: Consider a MIS product it *has database part(semidetached s/w), GUI part (organic s/w),Communication part (embedded s/w)*as subcomponents.
- The cost of these 3 components estimated separately and summed up to give the overall cost of the system.

III) Analytical Estimation Technique

- Derive the required results starting with basic assumptions regarding the project.
- Unlike empirical and heuristic techniques analytical techniques do have certain scientific basis.
- Eg: Halstead's software science –is an analytical technique to measure size, development effort, and development cost of software products.

Attributes of good software engineer

1. Exposure to systematic techniques, ie, experience with software engineering principles.
2. Good technical knowledge of the project areas. (domain knowledge)
3. Good programming abilities.
4. Good communication skill like oral, written and interpersonal skills.
5. High motivation
6. Sound knowledge of computer fundamentals
7. Intelligence
8. Discipline

9. Ability to work in a team

1.1.3 Describe about Resource Management

- In project management terminology, resources are required to carry ou+t the project tasks. They can be people, equipment, facilities, funding, or anything else capable of definition (usually other than labour) required for the completion of a project activity.
- Resource allocation is the process of assigning and scheduling available resources in the most effective and economical manner. Projects will always need resources and resources are scarce. The task therefore lies with the project manager to determine the proper timing of those resources within the project schedule.
- All elements used to develop a software product may be assumed as resource for that project. This may include human resource, productive tools and software libraries.
- The resources are available in limited quantity and stay in the organization as a pool of assets. The shortage of resources hampers the development of project and it can lag behind the schedule. Allocating extra resources increases development cost in the end. It is therefore necessary to estimate and allocate adequate resources for the project.

Resource management includes -

- Defining proper organization project by creating a project team and allocating responsibilities to each team member
- Determining resources required at a particular stage and their availability
- Manage Resources by generating resource request when they are required and de-allocating them when they are no more needed.

1.1.4 Describe how Project Risks can be identified, analyzed, mitigated, and monitored

RISK MANAGEMENT

- Risk is any predictable unfavourable event or circumstances that can occur while a project is underway.
- If a risk becomes real, it can cause the failure of project.
- Risk management consists of 3 essential activities : risk identification, risk assessment and risk containment.

Risk Mitigation

- ✓ The ultimate purpose of risk identification and analysis is to prepare for risk mitigation.
- ✓ Mitigation includes reduction of the likelihood that a risk event will occur and/or reduction of the effect of a risk event if it does occur.

RISK MITIGATION PLANNING

- ✓ Risk management planning needs to be an ongoing effort that cannot stop after a qualitative risk assessment, or a Monte Carlo simulation, or the setting of contingency levels.

- ✓ Risk management includes front-end planning of how major risks will be mitigated and managed once identified.
- ✓ Therefore, risk mitigation strategies and specific action plans should be incorporated in the project execution plan, or risk analyses are just so much wallpaper.
- ✓ Risk mitigation plans should

- Characterize the root causes of risks that have been identified and quantified in earlier phases of the risk management process.
- Evaluate risk interactions and common causes.
- Identify alternative mitigation strategies, methods, and tools for each major risk.
- Assess and prioritize mitigation alternatives.
- Select and commit the resources required for specific risk mitigation alternatives.

Communicate planning results to all project participants for implementation.

- ✓ Although risk mitigation plans may be developed in detail and executed by contractors, the owner's program and project management should develop standards for a consistent risk mitigation planning process.
- ✓ Owners should have independent, unbiased outside experts review the project's risk mitigation plans before final approval.
- ✓ This should be done prior to completing the project design or allocating funds for construction.
- ✓ Risk mitigation planning should continue beyond the end of the project by capturing data and lessons learned that can benefit future projects.

Risk Identification

- Project manager needs to predict the risks in project as early as possible so that impact of the risk can be minimized.
- The risks can be categorized into 3: project risk, technical risk, business risk.

Project Risks

- Project risks concern various forms budgetary, schedule, personnel, resource and customer related problems.
- An important project risk is schedule slippage.
- The invisibility of the product being developed is the reason for schedule slippage.

Technical Risks

- Technical risks concern potential design, implementation, interfacing, testing and maintenance problems.
- Technical risks also include incomplete specification, changing specification, technical uncertainty.

Business Risks

- Risks include risks of building an excellent product that no one wants, losing budgetary or personnel commitments.

Risk Assessment

- The objective of risk assessment is to rank the risk in terms of their damage causing potential.
- Risks rated in 2 ways:
  - ✓ The possibility of a risk coming true (r)
  - ✓ The consequence of the problem associated with that risk (s)
- Based on these two factors, priority is calculated as
  *P=r*s*

Risk Containment

- After all the identified risks of project are assessed, plans must be made to first contain the most damaging and most possible risks.
- Different risks require different containment procedures.
- There are three main strategies to plan for risk containment:

Avoid the risk

- Risks can be avoided in several ways, such as discussing with the customer to change the requirements to reduce the scope of the work, giving incentives to the developers to avoid the risk of manpower turnover.

Transfer the risk

- This strategy involves getting the risky component developed by a third party, buying insurance cover, and so on.

Risk reduction

- This involves planning ways to contain the damage due to a risk.
- The most to do in addressing technical risk s is to build a prototype.
- For choosing different strategies of handling risks the risk leverage is computed.

$$\text{Risk leverage} = \frac{\text{Risk exposure before reduction} - \text{Risk exposure after reduction}}{\text{Cost of reduction}}$$

1.1.5 Describe how project quality can be ensured and managed

A quality management system (often referred to as quality system) is the principal methodology used by organizations to ensure that the products they develop has the desired quality.

SOFTWARE QUALITY

A good quality product does exactly what the users want it to do

Software products has several quality factors such as following :

- ➢ Portability: Product can be easily made to work in different hardware and operating system environments.
- ➢ Usability :If different categories of users can easily invoke the functions of the product.
- ➢ Reusability :If different modules of the product can be easily reused.
- ➢ Correctness :The requirements in the SRS document have been correctly implemented.

- Maintainability :If errors can be easily corrected, new functions can be added

Important issues associated with a quality system

- A quality system is the responsibility of the organization as whole.
- The quality system activities encompass the following:
- Auditing of projects
- Review of quality system

1.1.6 Describe about Configuration Management

SOFTWARE CONFIGURATION MANAGEMENT

- ✓ Software configuration management deals with effectively tracking and controlling the configuration of a software product during its life cycle.
- ✓ The state of all objects (source code, design document, SRS document etc) at any point of time is called the configuration of the software product.

Necessity of software configuration management

- ✓ The most important reason for configuration management is to control the access to the different deliverable objects.
- ✓ Following are the some problems that appear, if configuration management is not used.
  1. Inconsistency problem when the objects are replicated :
  2. Problems associated with concurrent access :
  3. Providing a stable development environment :
  4. System accounting and maintain status information :
  5. Handling variants

Configuration Management Activities

Configuration management is carried out through two principal activities.

1. Configuration identification : involves deciding which parts of the system should be keep track of.
2. Configuration control : ensures that changes to a system happen smoothly

Configuration identification

- Project manager classifies the objects associated with software development into 3 categories.
- Controlled, pre controlled and uncontrolled
- Controlled objects are those already put under the configuration tool.
- Eg: design documents, test cases, source code.
- Pre controlled objects are not yet under the configuration control but will eventually be under the control.
- Uncontrolled objects are not and will not be subject to configuration control.

Configuration Control

- Configuration control is the process of managing changes to controlled objects.
- Configuration control allows only authorized changes and prevents unauthorized changes.

- In order to change a controlled object such as module a developer can get a private copy of the module by a reserve operation.
- Configuration management tool allow only one person to reserve a module at any time.

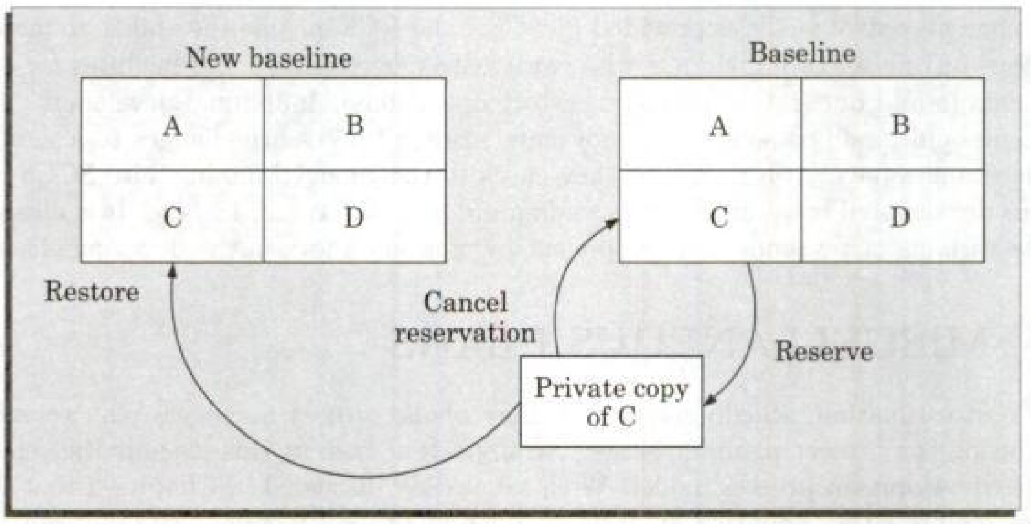


Figure 3.15: Reserve and restore operation in configuration control.

- Once an object is reserved, it does not allow anyone else to reserve this module until the reserved module is restored.
- Restoring the changed module to the system configuration requires the permission of a change control board (CCB).
- For every change that needs to be carried out, CCB reviews the changes made to the controlled object and certifies several things about the change as follows:
  1. Change is well motivated.
  2. Developer has considered and documented the effects of the change.
  3. Changes interact well with changes made by other developers.
  4. Appropriate people (CCB) have validated the change.

SOURCE CODE CONTROL SYSTEM (SCCS) AND RCS

- SCCS and RCS are two popular configuration management tools available on most UNIX systems.
- SCCS or RCS can be used for controlling and managing different version of text files.
- SCCS and RCS provide an efficient way of storing versions that minimizes the amount of occupied disk space.
- Changes needed to transform each base lined file to the next version are stored in deltas, for reducing disk space

1.1.7 Describe change management

- Change is a fact of life for large software system.

- To ensure that the changes are applied to the system in a controlled way, need a set of tools – supported , change management procedures.

Change management procedures

- Change management procedures are concerned with analyzing the costs and benefits of proposed changes, approving those changes that are worthwhile and tracking which components of the system have been changed.
- The change management process should come into effect when the software or associated documentation is base lined by the configuration management team.
- The first stage in the change management process is to complete a change request form(CRF) describing the change required to the system.
- As well as recording the change required, the CRF records the recommendations regarding the change, the estimated cost of the change and the dates when the change was requested, approved, implemented and validated.
- The CRF may also include a section where as analyst outlines how the change is to be implemented.
- The CRF is usually defined during the CM planning process.
- The engineer making the changes decides how to implement that change in these situation.
- Once a change request form has been submitted, it should be registered in the configuration database.
- The change request is then analysed to check that the change requested is necessary.
- If the analysis discovers that a change requests is invalid, duplicated or has already been considered, the change is rejected.
- For valid changes, the next stage of the process is change assessment and costing.
- The impact of the change on the rest of the system must be checked.
- This involves identifying all of the components affected by the change using information from the configuration database and the source code of the software.

1.1.8 Explain about CMMI, different levels and need of accreditation

Capability Maturity Model Integration (CMMI) is a process level improvement training and appraisal program. Administered by the CMMI Institute, a subsidiary of ISACA, it was developed at Carnegie Mellon University (CMU). It is required by many United States Department of Defence (DoD) and U.S. Government contracts, especially in software development. CMU claims CMMI can be used to guide process improvement across a project, division, or an entire organization. CMMI defines the following maturity levels for processes: Initial, Managed, Defined, Quantitatively Managed, and Optimizing. Version 1.3 was published in 2010. CMMI is registered in the U.S. Patent and Trademark Office by CMU.

CMMI Certification

- ✓ The Capability Maturity Model Integration, or CMMI, is a procedure model that gives a clear definition of what an organization should do to encourage behaviours that lead to enhanced performance.
- ✓ With five "Maturity Levels" or three "Capability Levels," the CMMI defines the most significant elements that are necessary to build great products, or deliver great services, and wraps them all up in a comprehensive model.
- ✓ The CMMI describes the principles and practices underlying software procedure maturity and is intended to support software organizations develop the maturity of their software procedures in terms of an evolutionary path from ad hoc, chaotic processes to mature, disciplined software processes.
- ✓ CMMI is organized into five maturity levels.
- ✓ A maturity level is a well – defined evolutionary plateau toward achieving a mature software procedure.
- ✓ Each maturity level offers a layer in the foundation for continuous procedure development.

Benefits of CMMI Certification

- ➢ CMMI can allow organizations to openly align management and engineering activities with their business objectives.
- ➢ Some of the other benefits reported by organization who have taken on CMMI are:
- Focus on measurement based decision making, project and organizational management
- Facilitates constant quality at all locations
- Focus on people and procedure as integrated assets to meet organizational objectives.
- Enhanced quality of the delivered products / services

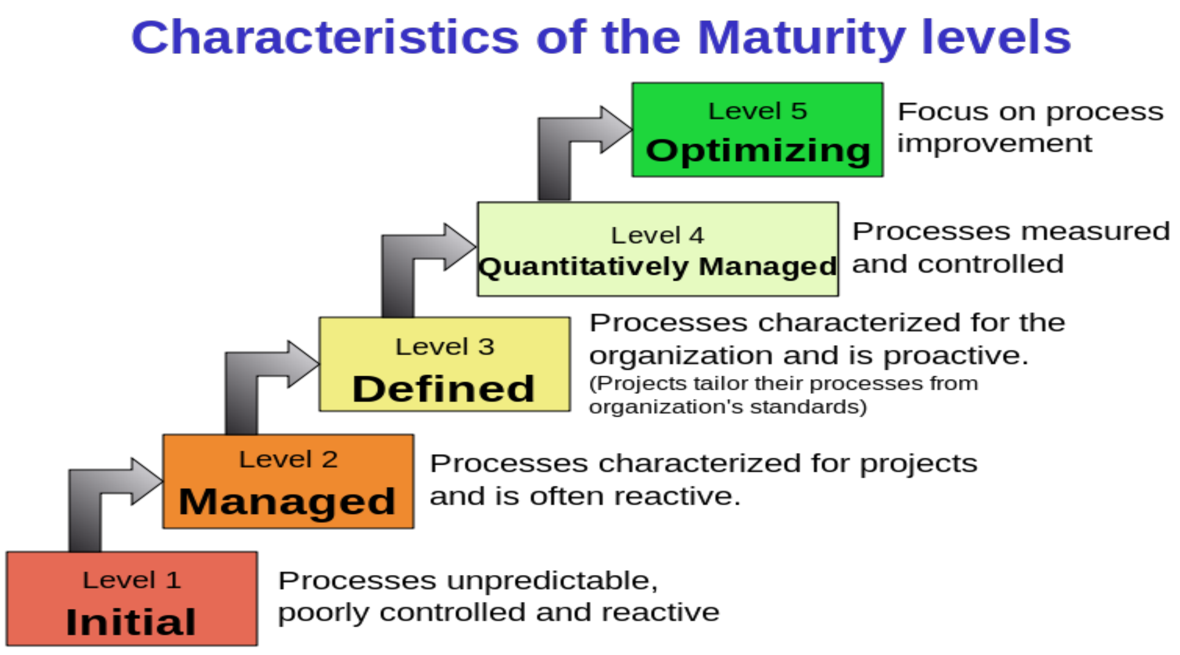
CMMI

- ✓ A maturity level is a well-defined evolutionary plateau toward achieving a mature software process.

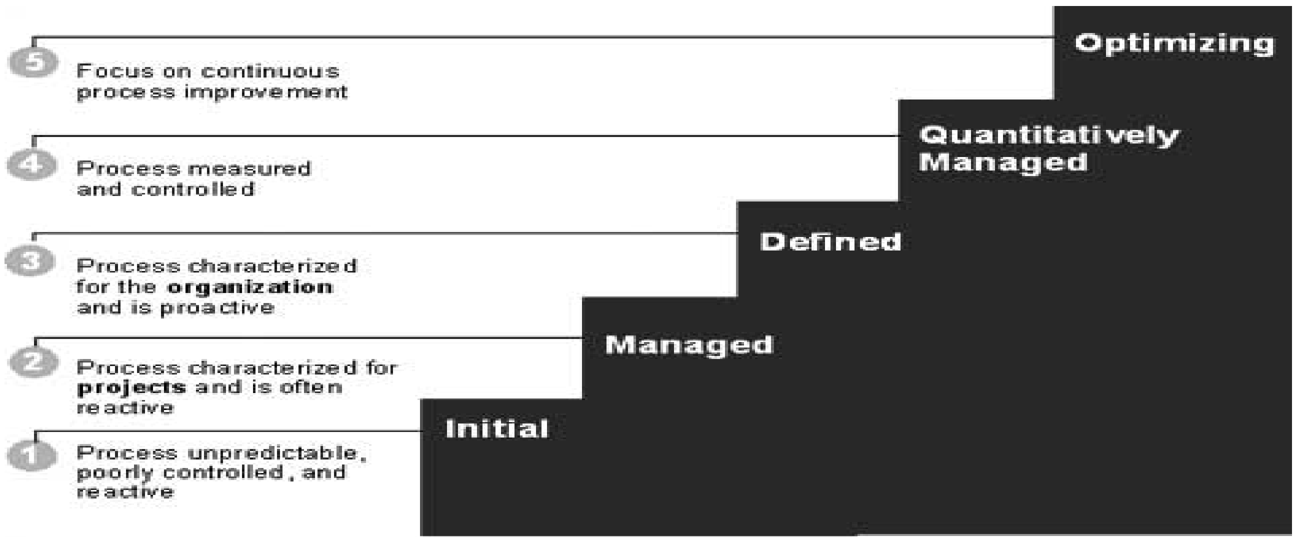
- ✓ Each maturity level provides a layer in the foundation for continuous process improvement.

- ➢ 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


Characteristics of the Maturity levels
Level 5
Optimizing
Focus on process improvement
Level 4
Quantitatively Managed
Processes measured and controlled
Level 3
Defined
Processes characterized for the organization and is proactive.
(Projects tailor their processes from organization's standards)
Level 2
Managed
Processes characterized for projects and is often reactive.
Level 1
Initial
Processes unpredictable, poorly controlled and reactive

<u>**CMMI Staged Represenation- Maturity Levels**</u>


5
Focus on continuous process improvement
4
Process measured and controlled
3
Process characterized for the organization and is proactive
2
Process characterized for projects and is often reactive
1
Process unpredictable, poorly controlled, and reactive
Optimizing
Quantitatively Managed
Defined
Managed
Initial

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 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 depends 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 Subprocesses are selected that significantly contribute to overall process performance.
- These selected subprocesses are controlled using statistical and other quantitative techniques.
- Quantitative objectives for quality and process performance are established and used as criteria in managing 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.
- Maturity level 5 focuses on continually improving process performance through both incremental and innovative technological improvements.

- 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 part of everybody's role, 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 mean 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:

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

This section is just giving names of the related process areas:-

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

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

- ❖ It 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

❖ 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.

Remember: 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 in 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 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:

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