Software Quality Engineering

V. Imp.

Q: In your opinion what are the names of three components of CMMI Process Model? (2)

Q: Write the names of standardized IT system criticality levels (3)

Q: List down the names of all six general steps of Audit process.

by Sabahat Saad
0326-1557791

Module-01: Software Quality Engineering Discipline

A. Quality Engineering Basics

What is software quality? What are the characteristics of high quality software solutions? What defines quality? These are some of the subjective question in the field of Software Quality Engineering. Modern Software Systems are usually interconnections of multiple underlying software and due to lack of standardization and varied nature it's really difficult to define quality. Software Quality Engineering involves complete software development process just to ensure that that any agreed-upon processes, standards and procedures are being followed to get desired results and there should be no cherry picking of standards.

B. Roles and Responsibility

People may have different expectations related to software quality assurance based on their roles and responsibility. The stakeholders for software development are divided into two and their expectations are as follows:

C. Consumer

Consumers of a software product are further categorized into the following:

- ***Users*** are the group which use the services acquired by the customer: The quality expectations on the side of users are as follows:
  - It performs all the functions as specified in the software requirements, which fits/meets the user's needs.
  - Performs all the specified functions correctly over repeated use or over a long period of time, or performs its functions reliably.
- ***Customer*** usually acquire the Software and Services: The quality expectations on the side of consumer are as follows:
  - Basic expectations of the consumer are similar to that of users with additional concentration on the cost of the software solution.

D. Producer

Producer of the software solutions includes person involved in the development, management, maintenance and service of the software product. It also includes third party software product and organizations. For producers, the expectations are as follows:

- Their biggest concern is to fulfill their contractual obligations by producing software products that conform to product specifications.
- Proper choice of software methodologies, languages, tools, software usability and modifiability and other factors are closely related to quality for this category of stakeholders.

E. Off the Shelf Products

These are plug-and-play products and are usually known as Plugins. They are developed and tested independently of Software Solutions. Their main purpose is to provide reusable functionality. Off-the-shelf (OTS) software products can be defined as "software product(s) available for any user, at cost or not, and used without the need to conduct development activities". Proper analysis should be perform while making decision regarding selection OTS as one solution does not fit all.

ISO-9126 Quality Framework

ISO-9126 is International Standard for Software Evaluation, it provides hierarchical framework for quality definition, organized into quality characteristics. There are six top-level quality characteristics that are summarized below:

F. Functionality

Functionality is the essential purpose of any product or service. The functionality characteristic allows drawing conclusions about how well software provides and performs desired functions. The functions are those that satisfy stated or implied needs. The more functions a product has, e.g. a sale order processing system, then the more complicated it becomes to define it's functionality. Continuing with the same example, the sales order system must be able to record sales, price, quantity, tax, shipping and inventory details. The software product may have multiple functions, but functionality is expressed as a totality of essential functions that the software product provides.

G. Reliability

The set of attributes related to the capability of software to maintain its level of performance under stated conditions for a stated period of time. The reliability characteristic tells the stakeholders about how effectively and efficiently a software solution maintains the level of performance if used under specified/stated conditions. Reliability can be used to evaluate the performance of whole or part of software and based on that suggest corrective measures to ensure continued software performance.

H. Usability

Usability can be defined as the ease to use any function especially from user view-point. Usability refers to the set of attributes of any software solution related to the individual assessment of different function by the stated users. The usability characteristic allows the stakeholders to conclude about how easily the solutions can be learned, understood and used. A good example to understand the concept is the revolutionary switch from Keyboard to touch-screen in 2007, and that makes Steve Jobs quote "Machines can be user Friendly too" a reality.

Quality Characteristics

I. Efficiency

Efficiency is a set of attributes concerning with the relationship between the level of software performance and the amount of resources used, under stated conditions. This characteristic is concerned with the system resources (amount of disk space, memory, network etc.) used when providing the required functionality. This attribute examines how well the software provides required level of performance relative to the amount of resources used. For example, Good UI Design can take several minute to load due to bad internet connection and it may happen that Heavy weight UI might take more time to load in presence of good internet connection.

J. Maintainability

Maintainability refers to the set of attributes that bear on the effort needed to make specified modifications. In other words, the ability to identify and repair a fault within a software solution or any part of it is what the maintainability characteristic tackles. In simple words, the maintainability characteristic allows to conclude about how well software can be maintained. The analyzability, changeability, testability and stability are subcomponents of maintainability. This feature is easier said than done because it is directly related to how well or bad software is designed, documented and reviewed periodically.

K. Portability

Portability refers to the set of attributes related to the ability of software to be transferred from one environment to another. The portability characteristic tells about how well and easily software can be ported from one environment to another. Presence of functionality is required to measure. This attribute also refers to how well the software can adopt to changes in its requirements as well. Due to available to multiple platforms these days in 2017 this feature is very critical for the success because it might happen that one feature might work in one version of OS but fails to work properly in another version of OS of same platform

L. What is Error?

Error is a human action that produces an incorrect result and/or the mistakes made by programmer is known as an Error. Error is usually some syntax mistakes by developer but it can be both syntax and semantic error. This could happen because of the following reasons: some confusion in understanding the requirement of the software; some miscalculation of the values; or/and misinterpretation of any value, etc. Cost of fixing the logical error increases with line of codes to be analyzed.

M. Example of Error

Examine the following lines of code:

Semantic Error	Corrected Version

Exit();?>

N. What is Defect? Q Define Defect?

Defect refers to the deviation from customer requirement. Mostly Defects are found in the Software after Software is shipped to the customer at production site. Defect is the departure of a quality characteristic from its specified value that results in a product not satisfying its normal usage requirements.

O. Example of Defect

Let's assume a software solution for online payments. Following table would explain the user expectation vs. defect.

User Expectations	Software Defect
The software will allow me to make online payments using debit/credit cards	The option of selecting the debit card for making payments is missing in production Software

P. What is Bug? Q: Define bug?

Bugs are the errors found before the software is shipped into production. **Famously the defects accepted by developers are bugs** and software is shipped with known bugs. The ugly fact in the software development is that there is nothing like **Bug Free Software**. Most bugs results from mistakes and errors made in either a program's source code or its design, or in components and operating systems used by such programs. Bug is rarely traceable by Compiler to its nearest place.

Q. Example of Bug

July 28, 1962 -- Mariner I space probe. A bug in the flight software for the Mariner 1 causes the rocket to divert from its intended path on launch. Mission control destroys the rocket over the Atlantic Ocean. The investigation into the accident discovers that a formula written on paper in pencil was improperly transcribed into computer code, causing the computer to miscalculate the rocket's trajectory.

⇒ prerequist can not be properly complete.
Some prerequist steps in SDLC are missing
that's why fault occur.

Kissi chez ka after effect.

Q: What is fault: List down two activities where ignorance leads to fault:

R. What is Fault?

An incorrect step, process, or data definition in a computer program is known as fault. Faults are fundamental condition within software that causes certain failure(s) to occur. Faults are known to be result of errors. In simple terms, Fault is an incorrect step or process due to which unanticipated result arises. SDLC → Phase → wrong output → due to error.

UAT → User acceptance Test (end user) called fault

output of error.

S. Example of Fault

Let's assume that the requirement is to write a program to add two numbers. In order to meet the requirement, the developer writes the following code:

```
#include<stdio.h>
int main ()
{
int value1, value2, ans;
Value1 = 5;
value2 = 3;
ans= value1 - value2;
printf("The addition of 5 + 3 = %d.", ans);
return 0;
}
```

error.

Q: diff b/w fault & failure

Due to wrong sign there is deviation from expected result

Chain of things that are link together.

T. What is Failure?

Failure is a result of fault; failure is inability of the program to behave as expected within given performance requirement. According to Laprie "a system failure occurs when the delivered service no longer complies with the specifications, the latter being an agreed description of the system's expected function and/or service". As mentioned above that failure is the result of fault, the following example would help understand this concept.

U. Example of Failure

```
#include<stdio.h>
int main ()
{
int value1, value2, ans;
Value1 = 5;
value2 = 3;
ans= value1 - value2;
```

By Sabahat Saad
0326-1557791

```
printf("The addition of 5 + 3 = %d.", ans);
return 0;
}
```

Imp

Fault: Due to wrong sign there is deviation from expected result

Failure: Due to Fault there is failure in the output. Instead of adding the two numbers it; subtracting the two numbers.

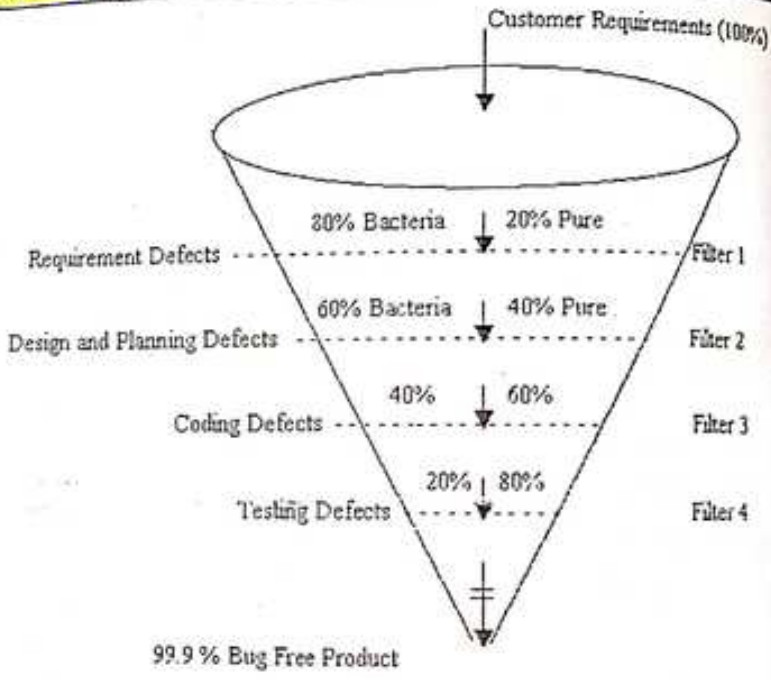
Imp & Subjective

V. Defect Prevention

Recurring defects are very costly by nature and mere wastage of time and budget and on the same hand the challenge in any software product development lies in minimizing the number of defects. Defect Prevention is strategy to identify root causes of defect and prevent them from recurring. Defect prevention is one of the important activities in any software project. It is QA process to identify the root causes of defects and improve the process to avoid introducing defects, which help to improve the quality of the software product.

occuring again again

On a macro level defects can be classified and filtered as depicted in the figure. But still there is no bug free product i.e. 99.99% does not mean 100%



W. Defect Detection or Reduction Explain defect Detection?

Defect Detection and Reduction is process to minimize defects but in a real scenario it is very unrealistic to expect project or product with 0 bug count. Defect prevention and defect reduction activities directly deal with the competing processes of defect injection and removal during the software development process (Humphrey, 1995). It is unrealistic to expect the defect prevention activities to be 100% effective in preventing accidental fault injections.

Therefore, we need effective techniques to remove as many of the injected faults as possible under project constraints.

X. Defect Removal or Containment

Due to nature of Software there are some defects which are produced under rare conditions. Defect Containment aims to reduce the chance of passing of defects from one phase to another. Due to large size and highly complex software systems, the defect reduction techniques only reduce the numbers of faults, though, to a very low level but this is not enough. The remaining faults may be triggered under certain and rare conditions. Thus it is necessary to prevent failures by breaking the causal relations between these faults and the resulting failures, thus "tolerating" these faults, or to contain the failures by reducing the resulting damage.

Module 02: Cost of Software Quality

Quality is always hard to define and in the case of software quality, it's more difficult. For any software application, the term quality may have different perception and definition among the developer, users, clients, managers, software quality engineers and other related stakeholders. Definition of quality often becomes even more complicated when quality depends upon the circumstances/environment in which it is being used. Literature reveals that software has the highest failure rate in the history of all the products resulting in loss of millions of dollars and this is one reason that makes quality important.

A. Economics of Software Quality Engineering

High concerns and challenges in the software quality engineering, one must realize the following facts in order to cope with the quality task:

MCQs

- Everything in the process of software development ends up in the user's satisfaction
- Satisfaction of the user is dependent on the overall behavior of the system, and software product comes at first
- The behavior of any software product is defined and comprehended through features and quality
- Features and quality of the software product are defined/determined through requirements

MCQs

- Any behavior related requirement of the software product can only be actualized through code that execute the behavior

associated with it.

B. Function-Quality-Cost (FCQ)

The discussion on financial ramification of engineering quality into any software product can be summarized through the following statement:

In most development projects, functionality and quality (QA precisely) are natural enemies.

Projects with open budgets are very rare, usually the budget is fixed and here the functionality and quality compete with each other in order to get a bigger share from budget. The Function-Quality-Cost comes out to be:

$$= \quad +$$

Where

A & B = Level of investment

F = Features/Functions

Q = Quality

It is very much clear that increasing feature in a closed-budget project will certainly decrease the budget share for quality of the product. The following example will elaborate the concept more clearly.

C. Quality vs. Pre-defined Budget

Let's take the example of project with fixed budget, say 100,000

decided to purchase Yahoo for $4.8 billion and this deal was to be closed in March 2017. But later in February 2017, Verizon and Yahoo announced that the deal will still go forward, but dropping the sale price by **$350 million and new offer was $4.48 Billion**. On the other side, user's confidential information including email, credit card details, bank account details and many others hit the market putting millions of users on stake.

F. Cost Analysis Based Approach

Missing Quality in Software Application has direct impact on People and Organizations as seen by the example mentioned in the above lines. Measuring such cost is critical to calculate impact and proceed with damage control otherwise the conditions will turn worst. Along with financial cost, there are other costs as well. According to **Eppler and Helfert principles** the costs are classified in two categories: direct and indirect.

Types of costs.
Q: Write down Types of costs?

G. Direct Cost of missing Quality

Direct Costs, as the name suggest, are directly linked to the missing quality. The direct costs are effects that are easily observable/measureable and they occur immediately after any unfortunate event. Examples includes; financial loss & physical injury and related. In short, direct costs are tangible, visible and measureable.

H. Indirect Cost of missing Quality

Indirect Costs are invisible cost of missing quality and hence difficult to calculate. It is also, sometime, difficult to realize or identify as they occur after a long time of the incident. Example includes: Loss of market share or reputation, loss of market and shareholders trust and

won't take appropriate action on right time then fail

J. Risk Analysis Approach

Risk analysis approach is essential in determining the cost of missing quality. As in many cases, the time and place of missing quality events is difficult to determine, a better method of cost evaluation is risk analysis approach. The risk is defined by its probability (p) and its impact or potential loss (L). Risk exposure (RE) is the product of the risk probability and its potential loss. The equation could be:

(Risk exposure ( ) = Risk Probability ( ) × Potential loss ( ))

The probability and loss are directly and strongly related to the level of criticality of the software solution under observation. The different levels of risks are elaborated below.

K. Level of Risk?

Write down level of risk?

The IEEE Standard for Software Verification and Validation has published the most broadly known scale of criticality in the IT domain. The standardized IT system criticality levels are as follows:

- **Level A: Catastrophic** / e.g earthquack (تباہ کن)
  - Continuous usage (24 hours per day)
  - Irreversible environmental damages
  - Loss of human lives

V. Imp Subj + objective

catastrophic
critical
Marginal
Negligible.

- Disastrous economic or social impact

(2) • **Level B: Critical** تنقیدی
  - Continuous usage (version change interruptions)
  - Environmental damages
  - Serious threats to human lives
  - Permanent injury or severe illness
  - Important economic or social impact.

(3) • **Level C: Marginal** برائے نام!
  - Continuous usage with fix interruption periods
  - Property damages
  - Minor injury or illness
  - Significant economic or social impact.

(4) • **Level D: Negligible** so small / unimportant
  - Time-to-time usage
  - Low property damages
  - No risks on human lives
  - Negligible economic or social impact.

BY Sabahat Saad
0326-1557791

Module 03: Standards and Models

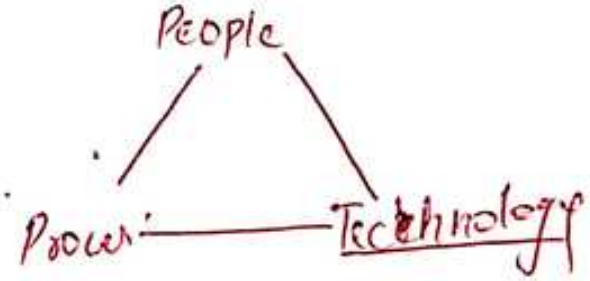
Q: What are toxic requirements?

A. Rationale for Quality Management System

A-quality management system is a formalized system to achieve Quality and the absence of which may lead to tragic situation or even product/system failure. A quality management system ensures documentation of Processes, Policies and work flows required to achieve desired standard of quality. One of the famous quality definitions - conformance to requirements - is a very unfortunate one because requirements are sometimes fill with defects, normally known as **toxic requirement.** It is for sure that conformance to those toxic requirements is not equivalent to quality. So the software engineering community has a moral obligation to eliminate such requirements.

B. Quality Leverage Points

One such framework to implement quality mindset is the concept of People - Process - Technology. It has also been referred to as the 'golden triangle'. It reveals that finest Talent is



organization.

C. Why Process is needed?

Before the discussion of why a process is needed, let's understand process first. A process is a set of practices performed to achieve a given purpose more importantly practices are uniform and same across organization to perform a specific task. A process serves as an integration point which ensures synergy. Process doesn't work as a magic stick; it needs time to realize the results. Process provides a constructive, high-leverage focus on quality. The skills and training of the workforce is not always enough and working hard is not the optimal solution. A well-defined and implemented process can provide the means to work smarter, utilizing people and technology at optimal level. Technology, by itself, will most likely not be used effectively. Technology, in the context of an appropriate process roadmap, can provide the most benefit.

D. Process Benchmarking

Process benchmarking is a very important part of process improvement initiatives and it offers a variety of benefits including very critical and empirical data related to the organization's current processes and open room for improvements. Benchmarking is comparing existing processes and performance metrics to industry's best processes practices from other companies. But this should be kept in mind that there is not good or bad process; internal limitation, circumstances and resources must be evaluated before adopting any external process that seems to be optimal because what works in one situation might not work in others.

F. Mature vs. Immature Organization

Mature organizations are system oriented and they ensure stability. They rely on documented processes with clear sense of roles and responsibility at all levels. On the opposite side, immature organizations rely on gut feelings. Even if they have processes in place, they do not follow or implement them rigorously. Following table identifies major difference between mature and immature organizations:

Immature Organization	Mature Organization
Process improvised during project	Inter-group communication and coordination
Approved processes being ignored	Work accomplished according to plan
Reactive, not proactive	Practices consistent with processes
Unrealistic budget and schedule	Processes updated as necessary
Quality sacrificed for schedule	Well-defined roles/responsibilities
No objective measure of quality	Management formally commits

G. Process Model Overview of CMMI

Capability Maturity Model Integration (CMMI) is a collection or a model of best practices in systems, product and software development. CMMI is not a process and it does not tell how to do your work rather it tell what to do to achieve high quality. CMMI is based on the premise of Process Management. The CMMI provides a framework for organizing small steps into five maturity levels that lay successive foundations for continuous process improvement. The maturity levels have associated process areas. CMMI holds the following beliefs:

- Change must come with future in mind, crisis prevention is better than recovering from crisis.

What are different CMMI Maturity levels

H. Behavior of Different Levels of CMMI

Each maturity level comes with set of best practices for implementation. When those best practices are implemented, each behavior is evaluated and appraised to measure its effectiveness. The results are compared with Metric (quantitative) based evaluation criteria which pre-defined for every behavior. Both the software process and products are quantitatively understood and controlled and the quantitative feedback enables continuous improvements. Further details of CMMI levels are given below.

I. CMMI Maturity Level 1 – Initial

At this level, the organization's environment is unstable for software development and maintenance. The processes - if any - well imperfectly defined and are reactive in nature. The organization, in overall is, unstable and unpredictable at this stage because the software process is constantly changed or modified as the work progresses. There is no roadmap for software development i.e., the process is ad hoc. Such organizations do face difficulties in retaining talented resources because of unstructured work and/or uncertainty in the organization. Let's examine a scenario:

J. Example CMMI Maturity Level 1

In a Software House, there are multiple projects in progress and projects can be assigned to single or multiple Project Manager, Assume there is new project which is assigned to two Project Managers, Client asks for what are next steps to proceed?

Answer:

K. CMMI Maturity Level 2 – Managed

At this stage, the policies and related frameworks are established for software development projects. Organizations at this level define a service strategy, create work plans and monitor and control the work to ensure the service is delivered as planned. Besides work activities and processes are managed and ensured that they are planned in accordance with the policy. Organization defines responsibilities to avoid situation mentioned above and also provide adequate resources and training to the workforce so they can smoothly execute the process. This is still not the optimal stage as the process here are often reactive and organizations rely heavily on Heroes and when they are gone, process and performance are gone. Read the following scenario.

L. Example CMMI Maturity Level 2

COMPANY IS WAITING FOR HERO.

Result: Client is shouting at Marketing Team and eventually stops using the Product

This results in supplier's credibility level going down to zero, leading to failure ultimately. Usually lack of documentation is justified with the intelligence and that's actually not true.

M. CMMI Maturity Level 3 - Defined

At the third Level, the standard process for developing and maintaining software are established and documented. The processes including both software engineering and management processes and they help workforce to perform more effectively. The reliance is on the defined process instead of Heroes. This stage can be considered as standard and consistent and people understand, support and follow the process and they are well aware of their roles and responsibilities. The major difference in Level 2 and 3 is as follows:

Level 2	Level 3
The process, standards and procedures are quite different for each instance of the process. The process can be different for a project or specific organizational unit.	The process, standards and procedures for a project are tailored from the organization's set of standard processes to suit a particular project or organizational unit.

deliver the project.

4. CMMI Maturity Level 4 – Quantitatively Managed

At this level, Organizations quantitatively manage their process and software products. Quantitative objectives are established to evaluate the quality and process performance and hence they statistically analyzed. Management can measure different valuable metrics like software process, quality and productivity and they can also tune them as required. Quantitative boundaries are decided for the processes and organizations achieve control over their products and processes by narrowing the variation in their process performance to fall within acceptable range. During the evaluation, special variation points are identified for further improvements.

A critical distinction between maturity levels 3 and 4 is the predictability of process performance. At maturity level 4, the performance of processes is controlled using statistical and other quantitative techniques and predictions are based, in part, on a statistical analysis of fine-grained process data.

		Total Time in Hours	9

Moral of the story is: ***"Process without Stats can't be improved"***

Q. CMMI Maturity Level 5 – Optimized Subj + Obj

This is the optimal level where the focus is on continuous process improvement. The organization at this stage earns the ability to proactively evaluate the process in order to avoid the defects. Continuous process improvement is based on the quantitative understanding of the variation in the process performance. This level is all about striving for continuous improvements in the process capability and process performance. Such improvements occur in incremental changes in the existing process and by adopting new technologies and methods. The difference between level 4 and 5 is that: Q: What is the diff b/w CMMI level 4 & 5

- Level 4 focus on two things: addressing special causes of variation and providing statistical predictability of the results.
- Level 5 address common causes of variations and changing the process to improve performance and maintain the statistical predictability.

by Sabahat Saad
0326-1557791

R. Example CMMI Maturity Level 5

Continuing with the example stated in level 4, there can be two options for further

Kick-Off (PM, Team Lead, Client)			
		Total Time Spend on Kick-Off (Minutes)	180
		Total Time in Hours	3

Explain capability level (V. Imp)

S. Capability Level

Capability level is part of CMMI that is concerned with the capability of the organization relative to the process area. The capability level reflects on how well an organization is aligned to a specific process area. In CMMI there are different process areas and each process area have

different processes. The capability level is consisting of specific and generic practices for a process area. Organizations can adopt those practices if they want to improve their processes associated with any process area. There are six capability levels designated by the numbers 0 through 5 and each level is a next step to the continuous improvement.

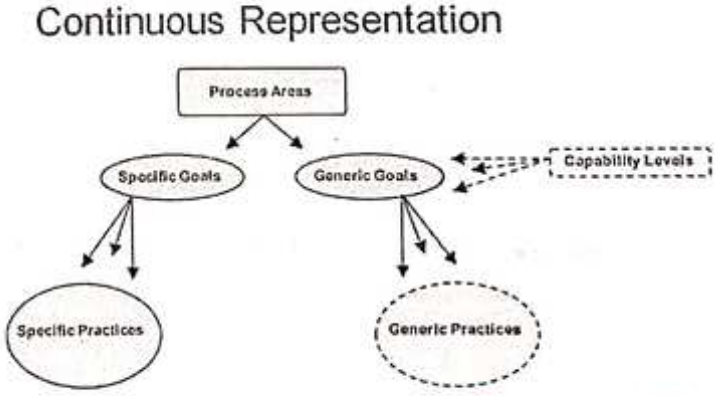
T. Component of CMMI Process Model

There are three components of CMMI process model through which maturity and capability are derived. These are as follows along with their actual definition and explanation as per CMMI:

1. ***Process Area:*** A cluster of related practices in an area that, when implemented collectively, satisfies a set of goals considered important for making improvement in that area.
2. ***Generic Practices:*** An expected model component that is considered important in achieving the associated generic goal. The generic practices associated with a generic goal describe the activities that are expected to result in achievement of the generic goal and contribute to the institutionalization of the processes associated with a process area.
3. ***Specific Practice:*** An expected model component that is considered important in achieving the associated specific goal. The specific practices describe the activities expected to result in achievement of the specific goals of a process area.

U. Process Area, Goal and Practices

There are 24 process areas in total and each process area is associated with a maturity level. The optimal level in each process area is achieved in increments. Each process area has a set of standards, processes and guidelines that an organization must need to follow in order to achieve higher maturity level. Process areas are viewed differently in the two representations; continuous and staged.



1. ***Continuous:*** the organization chose the processes that are critical to its business and achieve high capability levels.
2. ***Staged:*** Organization using this approach achieve the goals of the process areas associated each maturity level.

Q. What is the major purpose of EPG?

EPG stands for

A. Process Improvement Frameworks

The major purpose of engineering process group (EPG) is to improve the process throughout the organization. EPG first evaluate the existing process, define what a process should be and then provide suggestion for improvement. EPG also manage multiple process action teams with the purpose of improving different process areas simultaneously. PATs are individual teams created to address specific process improvement and PAT teams are consisting of technical staff from throughout the organizations. How EPG work is shown in the figure below followed by the details about process action teams.

* PAT stands for



B. Different Process Areas and Goal

There are in total 22 process areas but in this course only engineering related process areas will be discussed. As per CMMI definition, engineering process areas cover the development and maintenance activities that are shared across engineering disciplines. These are as follows:

1. Requirement Management Process Area
2. Requirement Development Process Area
3. Technical Solution Process Area
4. Product Integration Process Area
5. Software Validation Process Area
6. Software Verification Process Area

C. Process Action Teams (PAT)

As mentioned above, PAT is responsible for implementation of improvement initiatives activities in Specific Process Areas. In other words, each process area has associated process action team. The PATs are also known as the "worker bees." with immediate focus on weakness fund in process during the evaluation stage, their mandate is to write the procedures, pilot them, and update them as needed. Members of PAT belong to different domains and department of the organizations and they may include project managers. There tasks list is given below.

D. Task List of PAT

Process Action Teams (PAT) is mainly tasked to generate the process improvement documentation, policies, processes, procedures, charters, and Action Plans. For the improvement initiatives, PAT need to take care of different stakeholders for different process areas. One important task of PAT is to bring consistency in the documents throughout the organization in order to improve quality so they may need to work on drafting templates first. This will help in bringing same document structure for all processes and avoid rewriting of documents. This is also referred to standardization of artifacts.

E. Process Area: Requirement Management

This process area is concerned with the management of the entire requirement received or generated by the project, either technical or non-technical. The major purpose behind this is to ensure alignment between the requirements, project plans and the final output. One part of requirement management is to document the entire requirement, any changes in requirement along with their rationale. Change in requirements can take 2 forms, either change and/or update in the existing requirement or new requirement added to the project. Motivations behind requirement management process area are as follows:

- To manage inconsistencies between products and Requirements
- To manage different versions of Requirements

- To manage correlation between different project deliverable and requirements
- Traceability Matrix to be used to manage cross referencing

Action Item for Requirement Management

The goals of requirement management and practices to be followed are mentioned below:

Q. What practices should be followed in order to achive goals in requirement management.

Goal: Management Requirement

Practice: In order to achieve the goal, following practices are to be followed:

Subjective

- ***Understanding Requirement:*** Develop an understanding with the requirements providers on the meaning of the requirements.
- ***Obtain Commitment to Requirements:*** Obtain commitment to requirements from project stakeholders. In other words, this specific practice deals with agreements and commitments among those who carry out activities necessary to implement requirements.
- ***Manage Requirements Changes:*** Manage changes to requirements as they evolve during the project using Change Management Process by performing Impact Analysis.
- ***Maintain Bidirectional Traceability of Requirements:*** When requirements are managed well, traceability can be established from a source requirement to its lower level requirements and from those lower level requirements back to their source requirements.
- ***Identify Inconsistencies:*** Ensure that project plans and work products remain aligned with requirements.

BY: Sabahat Saad
0326-1557791

F. Example of Requirement Management

Requirement Management

Consider a real-time scenario below:

- 13-Mar-2016: Client and Project Manager agree on Requirements and Client approves it
- 14-Mar-2017: Requirements are passed on to Technical Team by Project Manager so they can work further
- 28-Mar-2017: Demo to be given to client and it was communicated to client
- 24-Mar-2017: Client and Project Manager agree on new set of requirements
- **28-Mar-2017: Client Reject the Demo by saying that Demo was not what was committed and rejected the Demo**

Root Cause Analysis

Client, Project Manager, Technical Team and QA were looking at different version of Requirements.

G. Process Area: Requirement Development

The purpose of this process area is to analyze and establish customer, product and product component requirements. Customer requirements are further divided into Product and Project Requirements. Requirements are identified and refined throughout the phases of the product lifecycle so all the requirements should be documented, analyzed and approved by the client and the source trace should be maintained.

Major artifact for this process area is Development of Software Requirement Specification (SRS).

Action Item for Requirement Management

The goals of requirement development and practices to be followed are mentioned below:

- ***Develop Customer Requirements:*** Stakeholder needs, expectations, constraints, and interfaces are collected and translated into customer requirements.
- ***Develop Product Requirements:*** Customer requirements are refined and elaborated to develop product and product component requirements.
- ***Analyze and Validate Requirements:*** The requirements are analyzed and validated.

H. Example of Requirement Development

The most important thing is that SRS should explicitly be approved by Client otherwise it will cause problem later in the Project.

The following images serve as good example of Requirement Development





Notepad:

dasdشسیسشیشسیشسی۳۲۲۴۴۴۴۲۲۴

Farsi font is applied to the whole text.

3. Process Area: Technical Solutions

Subjective

This process area is all about selection, design and implementation of solutions to the requirement of the product/project. As a good practice, design and solution should be aligned and traceable with the requirements. In other words, the selected solution should produce the required output (requirement) and the solution must also tell that which requirement it is going to fulfill. This process area focuses on evaluating different solution or design approaches that satisfy functional and quality requirement. WireFrame are also considered, through partially, to visualize the technical design. The main goals for these process areas are as follows:

- ***Select Product Component Solutions:*** Product or product component solutions are selected from alternative solutions.
- ***Develop the Design:*** Product or product component designs are developed.
- ***Implement the Product Design:*** Product components, and associated support documentation, are implemented from their designs.

The main artifact is Technical Design Document. All the details of adopted project plans and final output (product). All the details of adopted documented in this artifact which ultimately gives a picture of product architecture along with the traceability with the requirement. Sample Technical Design Document is attached in

Appendix - I

(4) K. **Process Area: Product Integration**

Software products are made of different components and this process area is all about assembling the product from multiple product components and ensuring that the product (as a whole) behaves properly and satisfy all the functional and quality requirements. Major failure occurs when the product components are either failed to integrate with each other or partially integrate which results in defects due to misaligned interfaces. So, heterogeneous Development environment is a major risk in this area. Product integration is not one-time assembling of the product components; in fact it can be done incrementally. In other words, instead of simultaneous integration of all the components, only few components are integrated and tested first and then more components are assembled. Usually Sanity is performed to ensure that integration is successfully completed no further issues/defects are introduced due to it.

The main goals for this process area are: product Integration ?

1. ***Prepare for Product Integration:*** Preparation for product integration is conducted.
2. ***Ensure Interface Compatibility:*** The product component interfaces, both internal and external, are compatible.
3. ***Assemble Product Components and Deliver the Product:*** Verified product components are assembled and the integrated, verified, and validated product is delivered.

L. Example of Product Integration

As per the example, components developed by three different teams. These components are integrated by software configuration team and the merged code based is forwarded for sanity testing. Product integration also includes removal of issues on merged codebase. Sanity testing on merged codebase will evaluate the integration and check that no defects are introduced due to integration.

Subjective

7 **M. Process Area: Software Validation**

This process area has the purpose of ensuring that the final product or its component(s) fulfill the requirements and its intended use when deployed. The product or its components are validated in the intended environment be it manufacturing, operations or any other. The major goal is to capture client requirements correctly from client and then meeting that requirement i.e. building the right thing. No code is required for software validation as it is just to check whether the product is doing what it should be doing (as per requirements) in the intended environment. Once again, Proof of Concept, WireFrames, and Requirement Modeling are key to validation. The major goal for these process areas includes the following:

Major Goal

Goals of Software Valida

1. ***Prepare for Validation:*** Preparation for validation is conducted by selecting the product, validation environment and validation criteria
2. ***Validate Product or Product Components:*** The product or product components are validated to ensure they are suitable for use in their intended operating environment.

By: Sabahat Saad
0326-1557791

N. Example of Software Validation

Sample Requirements

- Admin (Employee) should be able to login
- Employee should be able to register another employee in the organization
- Employee should be able to mark the attendance on daily basis





As per the requirement; employee is able to login, register another employee and mark attendance i.e. developer build the right product. One good strategy for software validation is that prototypes are shown to the customer and after approval the actual product is built and delivered.

6 0. Process Area: Software Verification Subjective

Software Verification process area is more concerned with the engineering/programming aspects of the project with the purpose to ensure that the final product is error free and selected work products/components meet their specified requirements. Verification does not

evaluate usefulness of the system instead verification is concerned with whether the system is well-engineered, error-free, and so on. So verification is more concerned with building the product right way. Software verification includes testing, design analysis, inspections and code reviews. The major goals for these process areas are as under:

1. ***Prepare for Verification:*** Preparation for verification is conducted.
2. ***Verify Selected Work Products:*** Selected work products are verified against their specified requirements.

P. Example of Software Verification

In verification QA team will execute each step after receiving the shipment from Development team

Q. Engineering Process Group

Engineering process group is organization's focal point to implement software processes to ensure compliance with quality standards. EPG also Act as oversight committee to monitor, evaluate and improve processes and it is a major player in coordinating process activity throughout the organization. Members of this group belongs to technical and management sides of the organization and they are responsible to assess the existing process, provide and implement suggestion for improvement and measure the effectiveness of the improved processes.

R. What are Audits?

Audit is tool to measure the organizational compliance level with the established process. Assessment and then improvements in any processes won't do any good for the organization if the process is not being followed. Through audits, organizations not only check the level of compliance but the reason behind the nonconformance. The reasons can come up in many shapes like people are not provided with required resources or training to follow the process or the process is misaligned with the working model and so on. Audits are conducted by independent auditor who first study the process, evaluate the conformance level and then assign ratings to the process after the audit.

S. Rationale for Audits

Audits are required to keep check and balance on organizational process and practices. Such audits become more important in volatile working environments. Results of audits become a starting point for process improvement and it gives valuable insights related to conformance. It can also tell where improvement is needed; in process design, process implementation, working conditions or the staff who are required to follow the process.

T. Audit Process

Define Audit? Also explain its types

Figure 1: Continuous Audit Implementation Steps

U. Audit Types

V- Imp

There are three (3) types of audits and these, along with brief explanation, are as follows:

1. ***First Party Audits:*** These are often described as internal audits. Someone from the organization itself audits a process to measure compliance and/or effectiveness.
2. ***Second Party Audits:*** This is an external audit where the audit is being performed on supplier by a customer or by a contracted organization on behalf of a customer with the intention to ensure that the supplier is meeting contract specification.
3. ***Third Party Audit:*** this is also an external audit and it's performed by an audit organization independent of supplier-customer relationship.

BY: Sabahat Saad
0326-1557791

V. Audit Roles and Responsibilities

Audit team includes a certified lead auditor who leads the audit activities and a team of 2 to 3 members supporting the lead in performing audit. Each member is equipped with right attitude and skills to measure the process results and performing the process audit to ensure compliance. Further in tough scenarios, domain experts become a part of this team to deal with the technicalities of such scenarios.

W. CMMI Appraisals

Explain CMMI Appraisals

Appraisal is defined as a process to collect, review and analyzes data to measure performance or compliance level. The collected data is then compared with the desired or standard data to identify the gap between the actual and desired, if any with the main purpose of measuring the effectiveness of the framework or process. CMMI appraisals provide ratings that accurately reflect the capability level or maturity level of the processes in use.

X. Process Reviews

Process reviews are frequently carried out in the organizations to measure the effectiveness of the process and to ensure that it is being completely followed. The frequency of process audit depends upon many factors but usually it's biannual or quarterly or on need basis. The process reviews also helps in identifying the required actions to improve the process results. Those required action may vary based on the results and it can be related to change/update in process objectives or design, training of stakeholders, technological advancement and many others.

Module 05: Process Management Process Area

A. Need for Project Improvement (PM) Framework

Besides technical aspects, there are certain management facets of software development and these are managed with the frameworks of project management. Project management covers all the management related concerns of software development and sales. The main purpose is to standardize every step of software development life cycle (SDLC) and this can be achieved by developing company-wide consistent artifacts and frameworks. Instead of arbitrary software development, Project management suggests a systematic and consistent approach to be adopted throughout SDLC to ensure desired results.

B. Project Management (PM) Framework

Q: Explain Project Management Framework?

Project Management Framework is a bridge between Development, Sales, Finance and Management. A project contains handsome number of factors contributing for desired results and none of them can be put in isolation as it may harm the project. Project management is an approach to connect technical and management facets of software development so the product is delivered with the desired quality and time span while reducing costs. The project management involves project planning (1) and execution of plans (2), management of software development teams (3), project documentation (4) and project monitoring (5). Auditable data is incorporated in all phases of project management.

C. Component of Project Management Framework ?

Q

The following document sample would help a lot to get a fair and practical idea about the project management framework. 1) objectives (2) Audiance (3) Project Name (4) Scope Guidlines (5) Budget effort (6) RoadMap (7) Demo plan (8) Updates Plan

Project Management Framework

a. **Objectives:** To ensure visibility of Project Progress to be Audited any time during Project lifecycle
b. **Audience:** Higher Management
c. **Project Name:**

In case there is deviation from plan or if a deadline is missed following steps are to be done:

iii. Updated Plan to be shared by PM

iv. Updated deadline to be shared with Client and Finance by PM

Note: For Invite detail Date / time is needed to be mentioned.

D. Artifacts of Project Management Framework

There are following artifacts of PM Framework:

- Signed Contract
- Specification Documents
- Estimations
- Gap Analysis
- Project Plan
- Demo Plan
- Invoice Plan

Q. What are the artifacts of PM Framework?

BY- Sabahat Saad
0326-1557791

Mids

Final

Q: Activities in Project planning?

E. Project Planning

Project planning includes a wide range of activities including development efforts, quality assurance and demo dates. Planning for development activities requires estimation containing