UNIT V

CMM

Capability Maturity Model is a bench-mark for measuring the maturity of an organization's software process. It is a methodology used to develop and refine an organization's software development process. CMM can be used to assess an organization against a scale of five process maturity levels based on certain Key Process Areas (KPA). It describes the maturity of the company based upon the project the company is dealing with and the clients. Each level ranks the organization according to its standardization of processes in the subject area being assessed.

A maturity model provides:

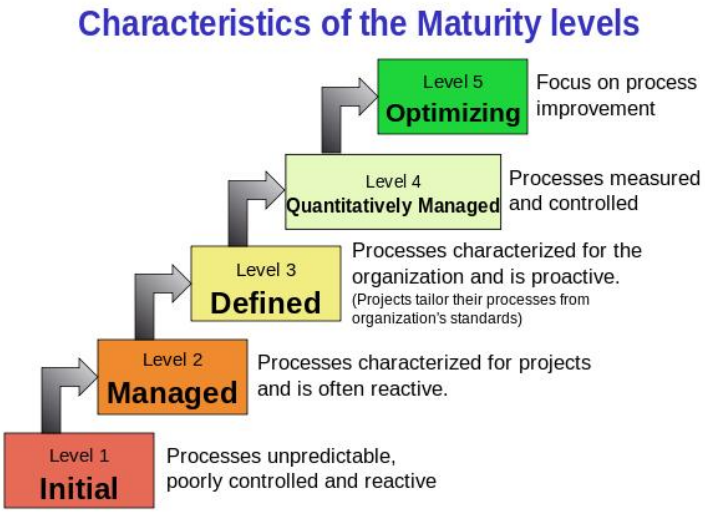
- A place to start
- The benefit of a community's prior experiences
- A common language and a shared vision
- A framework for prioritizing actions
- A way to define what improvement means for your organization

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

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

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

Maturity Level 1 – Initial: *Company has no standard process for software development. Nor does it have a project-tracking system that enables developers to predict costs or finish dates with any accuracy.*



Maturity Level 2 – Managed: *Company has installed basic software management processes and controls. But there is no consistency or coordination among different groups.*

Maturity Level 3 – Defined: *Company has pulled together a standard set of processes and controls for the entire organization so that developers can move between projects more easily and customers can begin to get consistency from different groups.*

Maturity Level 4 – Quantitatively Managed: *In addition to implementing standard processes, company has installed systems to measure the quality of those processes across all projects.*

Maturity Level 5 – Optimizing: *Company has accomplished all of the above and can now begin to see patterns in performance over time, so it can tweak its processes in order to improve productivity and reduce defects in software development across the entire organization.*

Difference between ISO and CMM

ISO	CMM
ISO certification is usually prompted because certification is needed to get contracts	CMM review is usually done to improve and involves a more detailed study than does an ISO review
With ISO 9001, once you are certified, your challenge is only to maintain certification	The challenge here is to maintain and continuously improve
For an organization that develops and manufactures embedded software products, an ISO 9001 certification tells very little about its software development capability. Certification means only that some basic practices are in place	CMM is a more comprehensive model to measure software development capability. It covers more processes and has a five-level rating system that emphasizes continuous improvement
ISO 9001 certification requires auditors, which places emphasis on opinions of outsiders whose capabilities may be unknown or marginal	CMM can be used as a self-assessment tool

CASE (Computer power-assisted software package Engineering)

A CASE (Computer power-assisted software package Engineering) tool could be a generic term accustomed denote any type of machine-driven support for software package engineering. in a very additional restrictive sense, a CASE tool suggests that any tool accustomed automatize some activity related to software package development.

Several CASE tools square measure obtainable. A number of these CASE tools assist in part connected tasks like specification, structured analysis, design, coding, testing, etc.; and other to non-phase activities like project management and configuration management.

Reasons for using case tools:
The primary reasons for employing a CASE tool are:

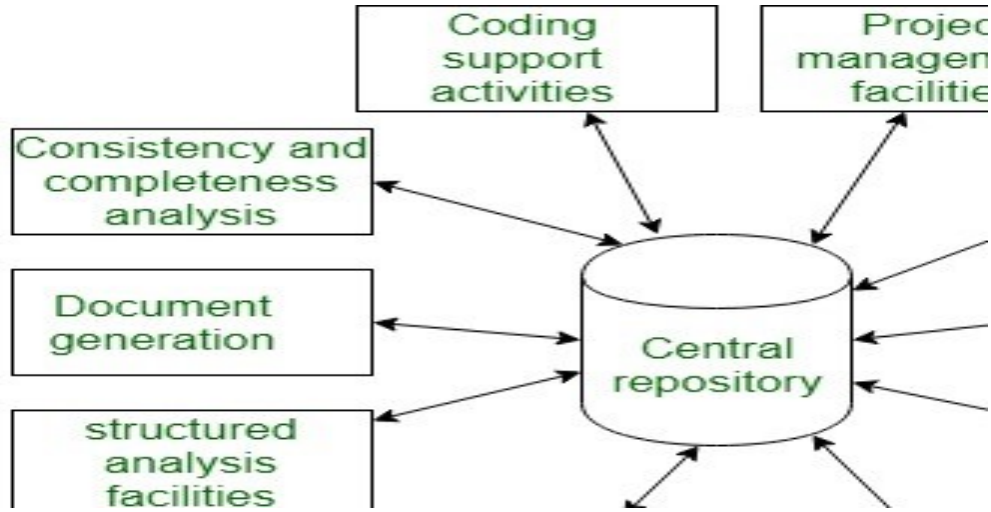
- *to extend productivity*
- *to assist turn out higher quality code at a lower price*

CASE environment:

Although individual CASE tools square measure helpful, the true power of a toolset is often completed only this set of tools square measure integrated into a typical framework or setting. CASE tools square measure characterized by the stage or stages of package development life cycle that they focus on. Since totally different tools covering

different stages share common data, it's needed that they integrate through some central repository to possess an even read of data related to the package development artifacts. This central repository is sometimes information lexicon containing the definition of all composite and elementary data things.

Through the central repository, all the CASE tools in a very CASE setting share common data among themselves,therefore a CASE setting facilities the automation of the step-wise methodologies for package development. A schematic illustration of a CASE setting is shown in the below diagram:

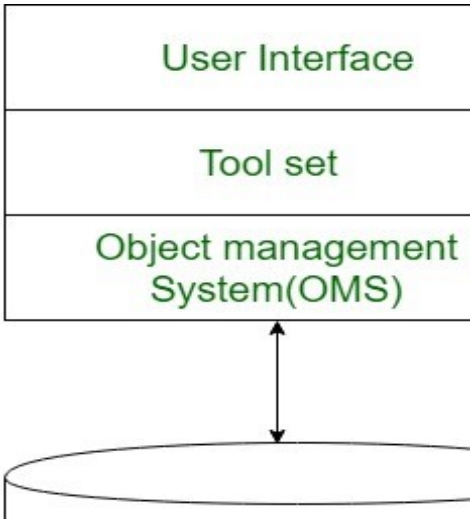


Note: *CASE environment is different from programming environment.*

A CASE environment facilitates the automation of the in small stages methodologies for package development. In distinction to a CASE environment, a programming environment is an Associate in a Nursing integrated assortment of tools to support solely the cryptography part of package development.

Architecture of a CASE environment

The design of a typical trendy CASE (Computer power-assisted software package Engineering) atmosphere is shown graphically below. The vital elements of a contemporary CASE atmosphere are a computer program, toolset, object management system (OMS), and a repository. The characteristics of a toolset are mentioned earlier.



User Interface:
the user interface provides a regular framework for accessing the various tools so creating it easier for the users to act with the different tools and reducing the overhead of learning however the different tools are used.

Object Management System (OMS) and Repository:
Different case tools represent the product as a group of entities like specification, design, text data, project arrange, etc. the thing management system maps these logical entities such into the underlying storage management system (repository).

The industrial on-line database management systems are meshed towards supporting giant volumes of data structured as straightforward comparatively short records. There are some forms of entities however sizable amount of instances. in contrast, CASE tools produce an oversized range of entity and relation varieties with maybe some instances of every. so the thing management system takes care of befittingly mapping into the underlying storage management system.