Capability Maturity Model

What is CMM?

- CMM: Capability Maturity Model
- Developed by the Software Engineering Institute of the Carnegie Mellon University
- Framework that describes the key elements of an effective software process.

What is CMM?

- Describes an evolutionary improvement path for software organizations from an ad hoc, immature process to a mature, disciplined one.
- Provides guidance on how to gain control of processes for developing and maintaining software and how to evolve toward a culture of software engineering and management excellence.

Process Maturity Concepts

- Software Process
  - set of activities, methods, practices, and transformations that people use to develop and maintain software and the associated products (e.g., project plans, design documents, code, test cases, user manuals)
- Software Process Capability
  - describes the range of expected results that can be achieved by following a software process
  - means of predicting the most likely outcomes to be expected from the next software project the organization undertakes

Process Maturity Concepts

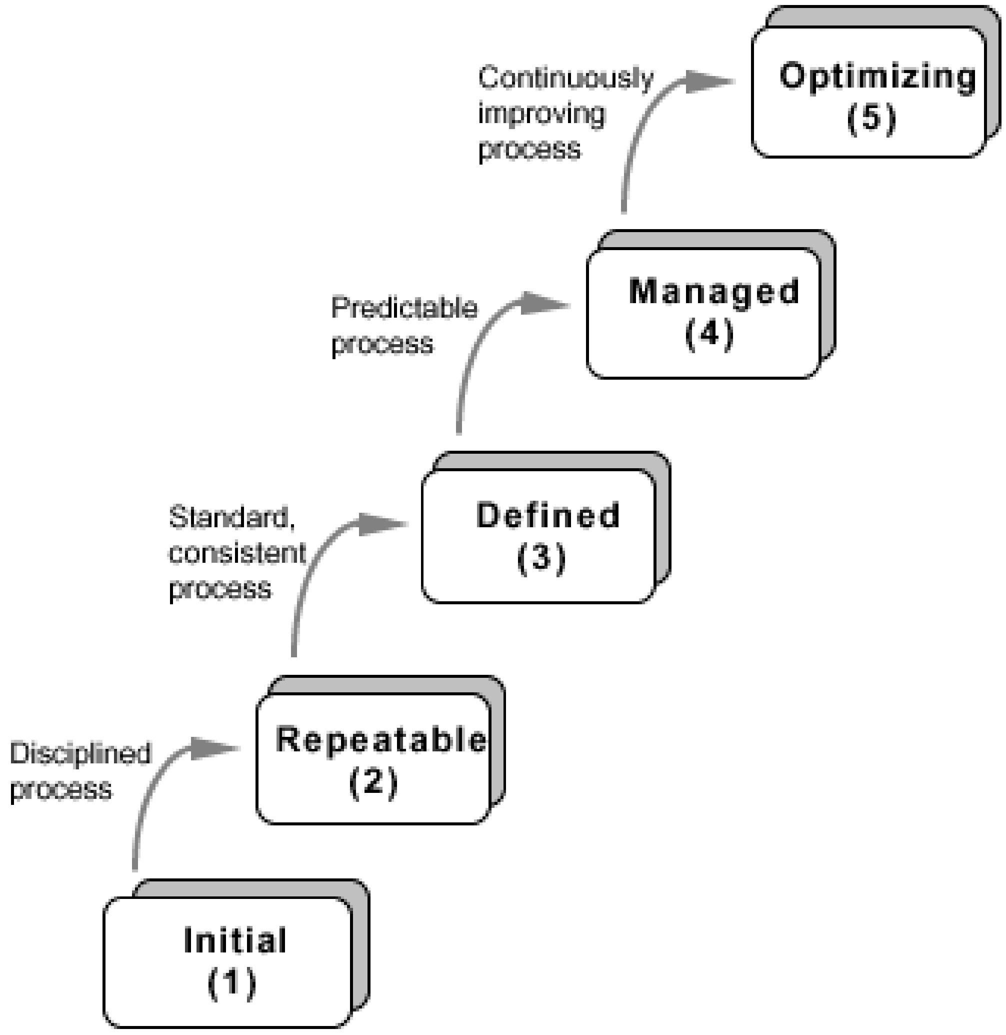
- Software Process Performance
  - actual results achieved by following a software process
- Software Process Maturity
  - extent to which a specific process is explicitly defined, managed, measured, controlled and effective
  - implies potential growth in capability
  - indicates richness of process and consistency with which it is applied in projects throughout the organization

What are the CMM Levels?

(The five levels of software process maturity)

Maturity level indicates level of process capability:

- Initial
- Repeatable
- Defined
- Managed
- Optimizing


Continuously improving process
Optimizing (5)
Predictable process
Managed (4)
Standard, consistent process
Defined (3)
Disciplined process
Repeatable (2)
Initial (1)

Level 1: Initial

- Initial : The software process is characterized as ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort.
  - At this level, frequently have difficulty making commitments that the staff can meet with an orderly process
  - Products developed are often over budget and schedule
  - Wide variations in cost, schedule, functionality and quality targets
  - Capability is a characteristic of the individuals, not of the organization

Level 2: Repeatable

- Basic process management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.
  - Realistic project commitments based on results observed on previous projects
  - Software project standards are defined and faithfully followed
  - Processes may differ between projects
  - Process is disciplined
  - earlier successes can be repeated

Level 3: Defined

- The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization’s standard software process for developing an maintaining software.

Level 4: Managed

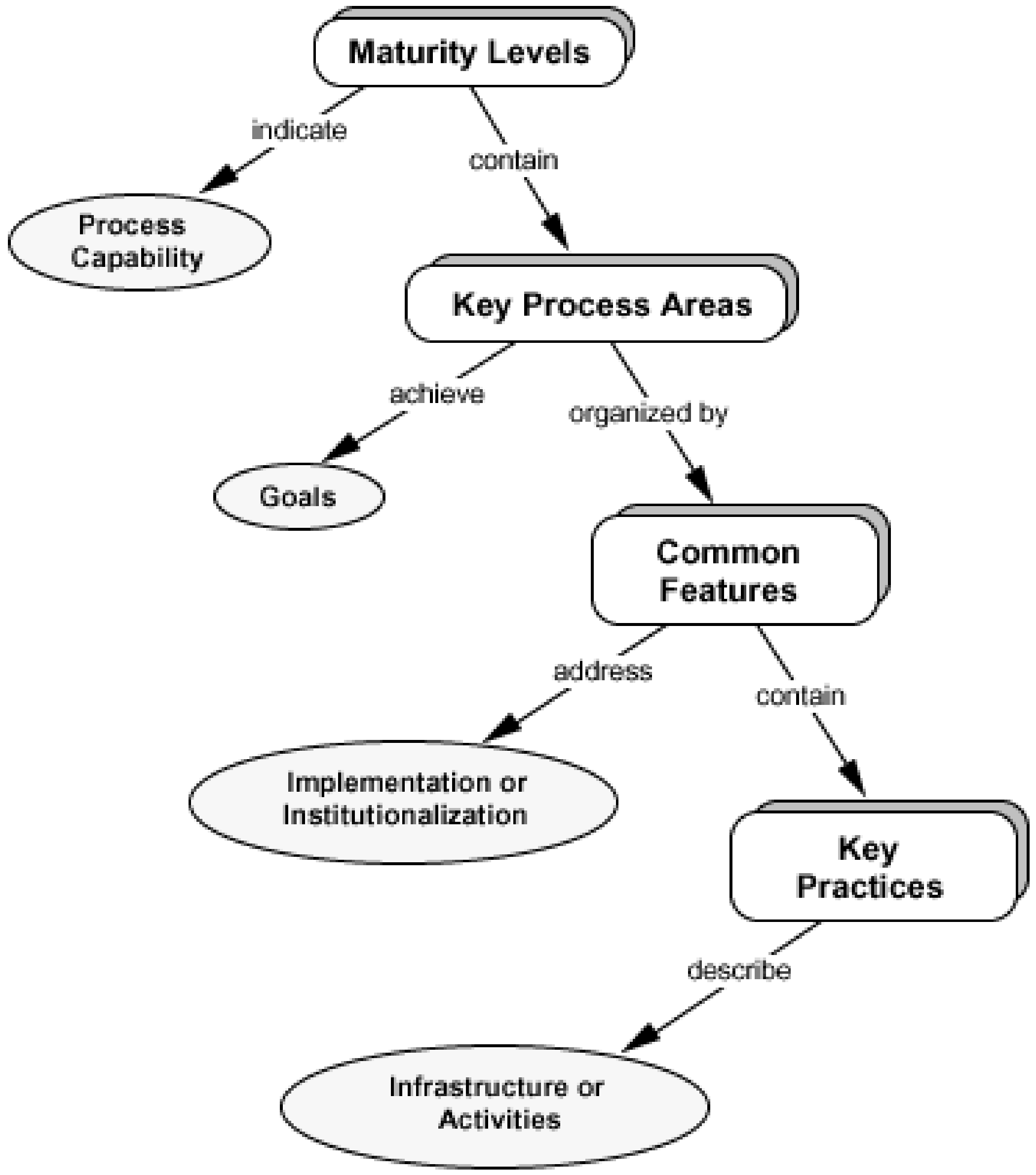
- Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.
  - Narrowing the variation in process performance to fall within acceptable quantitative bounds
  - When known limits are exceeded, corrective action can be taken
  - Quantifiable and predictable
    - predict trends in process and product quality

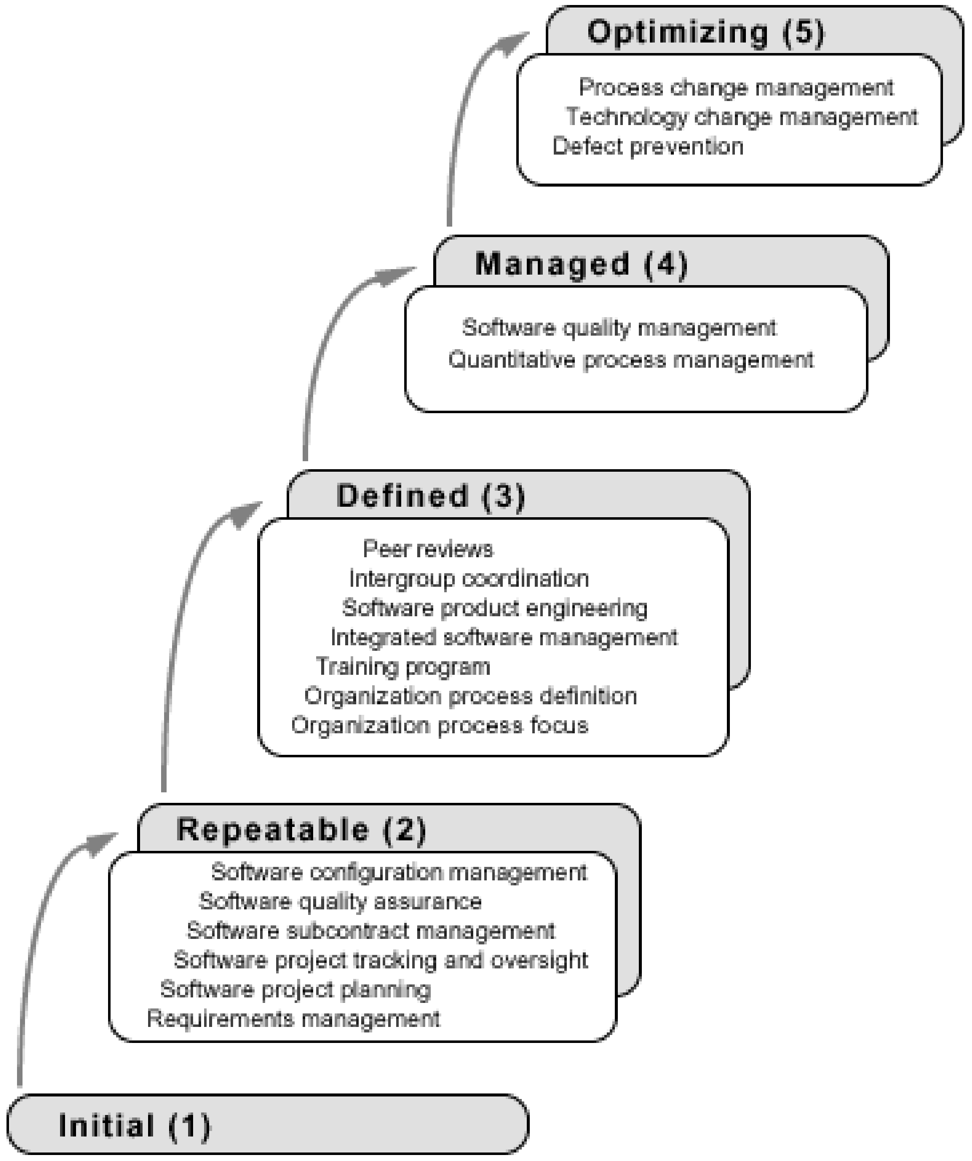
Level 5: Optimizing

- Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.
- Goal is to prevent the occurrence of defects
  - Causal analysis
- Data on process effectiveness used for cost benefit analysis of new technologies and proposed process changes

Internal Structure to Maturity Levels

- Except for level 1, each level is decomposed into key process areas (KPA)
- Each KPA identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important for enhancing software capability.
  - commitment
  - ability
  - activity
  - measurement
  - verification


Maturity Levels
indicate
contain
Process Capability
Key Process Areas
achieve
organized by
Goals
Common Features
address
contain
Implementation or Institutionalization
Key Practices
describe
Infrastructure or Activities


Optimizing (5)
Process change management
Technology change management
Defect prevention
Managed (4)
Software quality management
Quantitative process management
Defined (3)
Peer reviews
Intergroup coordination
Software product engineering
Integrated software management
Training program
Organization process definition
Organization process focus
Repeatable (2)
Software configuration management
Software quality assurance
Software subcontract management
Software project tracking and oversight
Software project planning
Requirements management
Initial (1)

The Key Process Areas by Maturity Level

Level 2 KPAs

- Requirements Management
  - Establish common understanding of customer requirements between the customer and the software project
  - Requirements is basis for planning and managing the software project
  - Not working backwards from a given release date!
- Software Project Planning
  - Establish reasonable plans for performing the software engineering activities and for managing the software project

Level 2 KPAs

- Software Project Tracking and Oversight
  - Establish adequate visibility into actual progress
  - Take effective actions when project's performance deviates significantly from planned
- Software Subcontract Management
  - Manage projects outsourced to subcontractors
- Software Quality Assurance
  - Provide management with appropriate visibility into
    - process being used by the software projects
    - work products

Level 2 KPAs

- Software Configuration Management
  - Establish and maintain the integrity of work products
  - Product baseline
  - Baseline authority

Level 3 KPAs

- Organization Process Focus
  - Establish organizational responsibility for software process activities that improve the organization's overall software process capability
- Organization Process Definition
  - Develop and maintain a usable set of software process assets
    - stable foundation that can be institutionalized
    - basis for defining meaningful data for quantitative process management

Level 3 KPAs

- Training Program
  - Develop skills and knowledge so that individual can perform their roles effectively and efficiently
  - Organizational responsibility
  - Needs identified by project
- Integrated Software Management
  - Integrated engineering and management activities
  - Engineering and management processes are tailored from the organizational standard processes
  - Tailoring based on business environment and project needs

Level 3 KPAs

- Software Product Engineering
  - technical activities of the project are well defined (SDLC)
  - correct, consistent work products
- Intergroup Coordination
  - Software engineering groups participate actively with other groups
- Peer Reviews
  - early defect detection and removal
  - better understanding of the products
  - implemented with inspections, walkthroughs, etc

Level 4 KPAs

- Quantitative Process Management
  - control process performance quantitatively
  - actual results from following a software process
  - focus on identifying and correcting special causes of variation with respect to a baseline process
- Software Quality Management
  - quantitative understanding of software quality
    - products
    - process

Level 5 KPAs

- Process Change Management
  - continuous process improvement to improve quality, increase productivity, decrease cycle time
- Technology Change Management
  - identify and transfer beneficial new technologies
    - tools
    - methods
    - processes
- Defect Prevention
  - causal analysis of defects to prevent recurrence

What are the benefits ?

- Helps forge a shared vision of what software process improvement means for the organization
- Defines set of priorities for addressing software problems
- Supports measurement of process by providing framework for performing reliable and consistent appraisals
- Provides framework for consistency of processes and product

Why measure software and software process?

Obtain data that helps us to better control

- schedule
- cost
- quality of software products

Consistent measurement provide data for:

- Quantitatively expressing requirements, goals, and acceptance criteria
- Monitoring progress and anticipating problems
- Quantifying tradeoffs used in allocating resources
- Predicting schedule, cost and quality

Measurements

- Historical
- Plan
- Actual
- Projections

SEI Core Measures

Unit of Measure	**Characteristics Addressed**
Physical source lines of code Logical source lines of code	Size, reuse, rework
Staff hours	Effort, cost, resource allocations
Calendar dates for process milestones Calendar dates for deliverables	Schedule, progress
Problems and defects	Quality, improvement trends, rework, readiness for delivery

Examples of measurements for size of work products

- Estimated number of requirements
- Actual number of requirements
- Estimated source lines of code (SLOC)
- Actual SLOC
- Estimated number of test cases
- Actual number of test cases

Example of measurements of effort

- Estimated man-hours to design/code a given module
- Actual man-hours expended for designing/coding the module
- Estimated number of hours to run builds for a given release
- Actual number of hours spent running builds for the release

Examples of measurements of quality of the work product

- Number of issues raised at requirements inspection
- Number of requirements issues open
- Number of requirements issues closed
- Number of issues raised during code inspection
- Number of defects opened during unit testing

Examples of measurements of quality of the work product

- Number of defects opened during system testing
- Number of defects opened during UAT
- Number of defects still open
- Number of defects closed
- Defect age

Examples of measurements of quality of the work product

- Total number of build failures
- Total number of defects fixed for a given release
- Total number of defects verified and accepted
- Total number of defects verified and rejected