SE345

Atılım University
Dept of Software Engineering

Asst. Prof. Dr. Aylin AKCA-OKAN

Tentative Course Schedule

Wk	Subjects	Chapter
1.	Introduction to Software Quality and Assurance	Chapter 1
2.	Introduction to Software Quality and Assurance	Chapter 1
3.	Software Quality Factors	Chapter 3
4.	Overview of Components of the SQA System	Chapter 4
5.	Overview of Components of the SQA System	Chapter 4
6.	Integrating Quality Activities in Project Life Cycle	Chapter 7
7.	Midterm	
8.	Reviews, Inspection and Audits	Chapter 8
9.	Software Quality Metrics	Chapter 21
10.	Procedures and Work Instructions	Chapter 14
11.	Software Change Process	Chapter 18
12.	Presentations	
13.	SQA Process Standards	Chapter 23

Standards and Models

What is a standard?

- A **standard** is a repeatable, harmonised, agreed and documented way of doing something.
- Standards contain technical specifications or other precise criteria designed to be used consistently as a rule, guideline, or definition.
- They help to make life simpler and increase the reliability and effectiveness of many of the goods and services we use.

Why use a model?

- **Improvement efforts need a model** that describes how your organisation works, which functions it requires, and how those functions interact. A model gives you a shared language and a structure for discussing what to improve.

A model provides several concrete benefits:

- A common framework and language that improves communication.
- Decades of experience captured as guidance.
- A way to keep the larger picture in mind while focusing on improvement work.
- Trainer and consultant support in many markets.
- An external reference that helps resolve disagreements by using agreed standards.

How about benefits?

- The ability to apply methodologies and procedures of the **highest professional level.**
- **Better mutual understanding and coordination** among development teams but especially between development and maintenance teams.
- **Greater cooperation** between the software developer and external participants in the project (stakeholders), based on the adoption of standards as part of the contract.

Contributions

- Provision of **superior professional methodologies** for use in the development process and for its management.
- Provision of **SQA certification services** based on independent professional quality audits.
- Provision of **tools for "self-assessment"** of achievements in planning and operating an organisation's SQA system.

Standard Categories

Quality management standards: Software quality assurance management standards, including certification and assessment methodologies

Project process standards: Software project development process standards.

Characteristics	Quality Management Standards	Project Process Standards
The target unit	**Management** of software development and/or maintenance and the specific **SQA units**	A software development and/or maintenance project **team**
The main focus	Organization of SQA systems, infrastructure and requirements	Methodologies for carrying out software development and maintenance projects
Standard's objective	"What" to achieve	"How" to perform
Standard's goal	Assuring supplier's software quality and assessing its software process capability	Assuring the quality of a specific software project's products

Quality Management Standards

- Focus is on the **software quality assurance system**, its organisation, infrastructure and requirements
- They leave the choice of the methods and tools to be used in the hands of the organisation (focus on the **"what" of SQA** and not its "how")
- Compliance to these standards supports the organisation's steady efforts **to assure an acceptable quality level for its software products.**

CMM, CMMI, ISO 9001, 9000-3

Project Process Standard

- Focus on the methodologies for carrying out software development and maintenance projects **(“how” a software project is to be implemented)**.
- Defines the steps to be taken, design documentation requirements, the contents of design documents, design reviews and review issues, software testing to be performed and testing topics, and so forth.

IEEE 1012, ISO/IEC 12207

Certification

Enable a software development organisation to **demonstrate consistent ability to assure acceptable quality of its software products or maintenance services.**

An external body grants certification.

Serve as an agreed-upon **basis for customer and supplier evaluation of the supplier's quality management system**.

Accomplished by the performance of a quality audit by the customer.

Support the organisation's efforts **to improve its quality management system** through compliance with the standard's requirements.

Assessment

Serve organisations as a tool for self-assessment of their ability to carry out software development projects.

Serve for improvement of development and maintenance processes by application of the standard directions

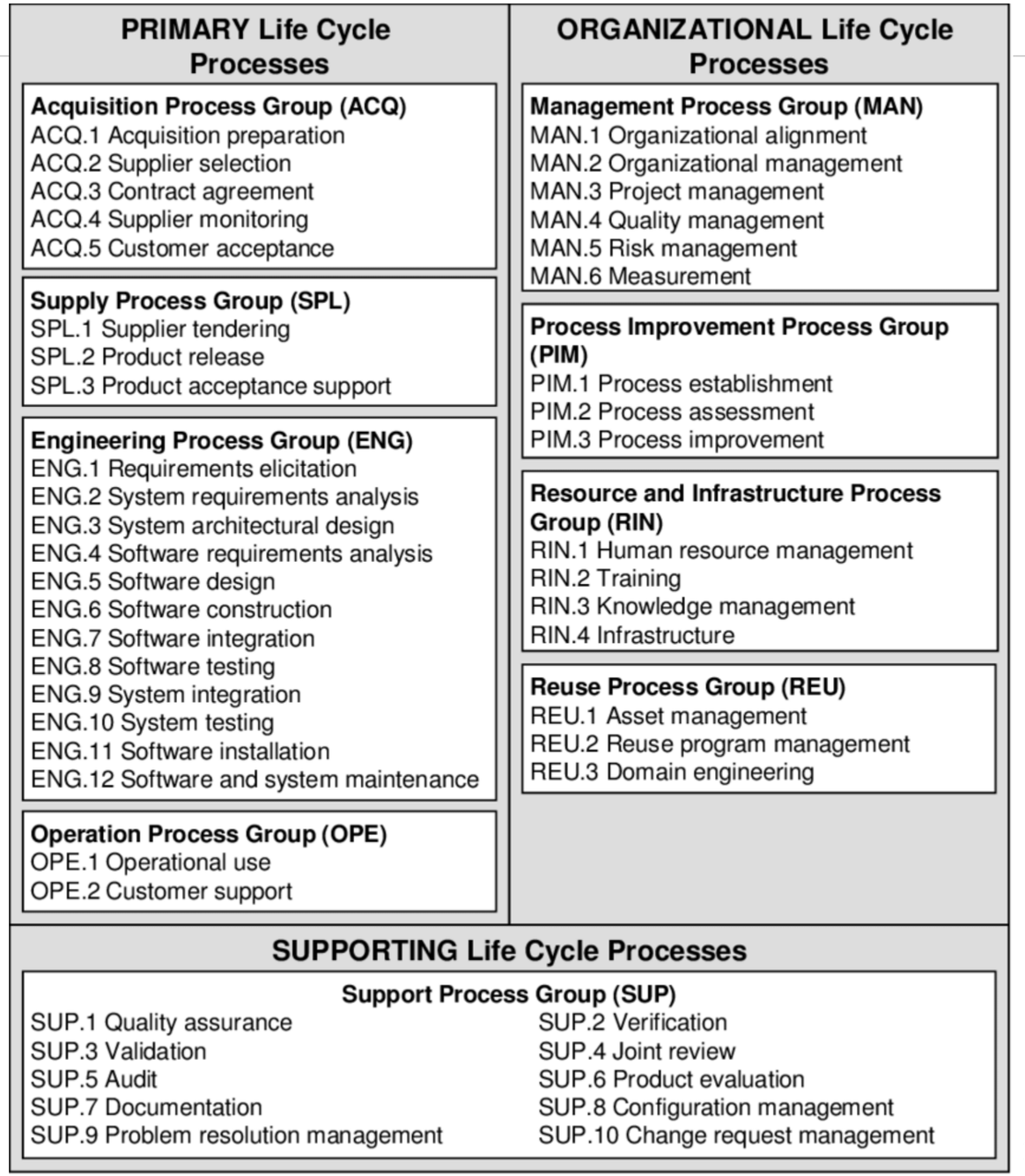
Guide training of assessor by delineating qualifications and training program curricula.

Organisations in SQA Standards Development Prominent Ones

- IEEE (Institute of Electric and Electronic Engineers) Computer Society
- ISO (International Standards Organisation)
- DOD (US Department of Defense)
- ANSI (American National Standards Institute)
- IEC (International Electrotechnical Commission)
- EIA (Electronic Industries Association)
- Carnegie Mellon Software Engineering Institute (SEI)
- ISACA (Information Systems Audit and Control Association)

ISO/IEC 12207 Process Standard

To establish an internationally recognised model of common software life cycle processes that can be referenced by the software industry worldwide.



ISO 9000

The seven quality management principles are:

QMP 1 – Customer focus

QMP 2 – Leadership

QMP 3 – Engagement of people

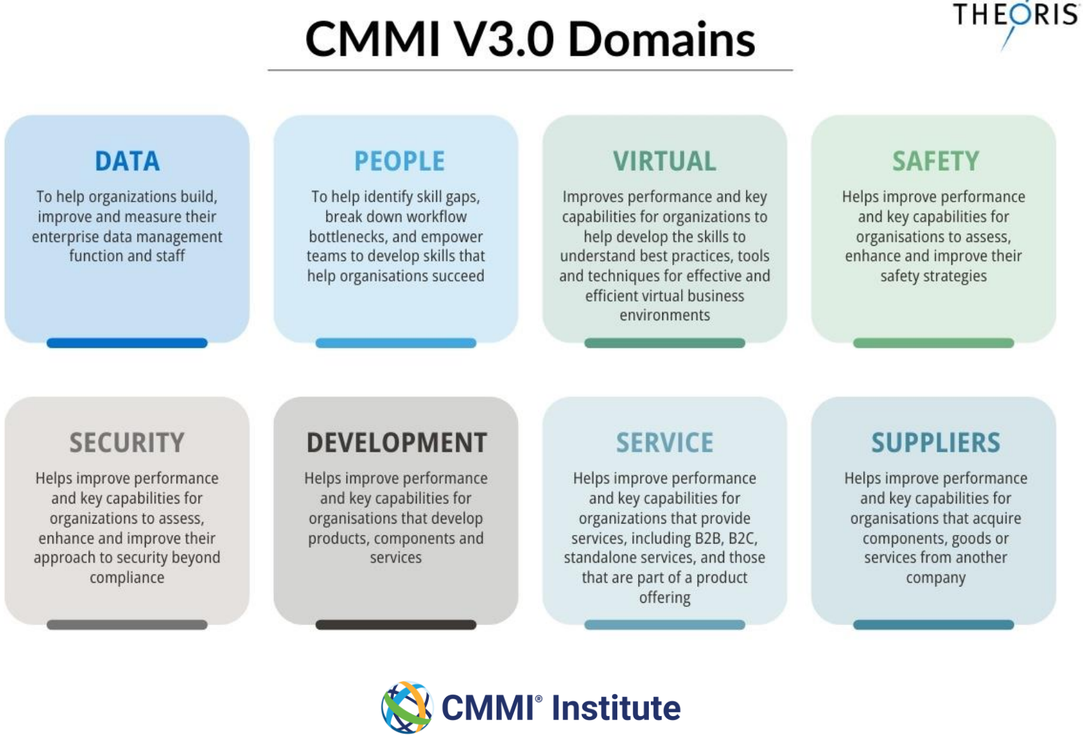
QMP 4 – Process approach

QMP 5 – Improvement

QMP 6 – Evidence-based decision making

QMP 7 – Relationship management

Capability Maturity Model - Integrated

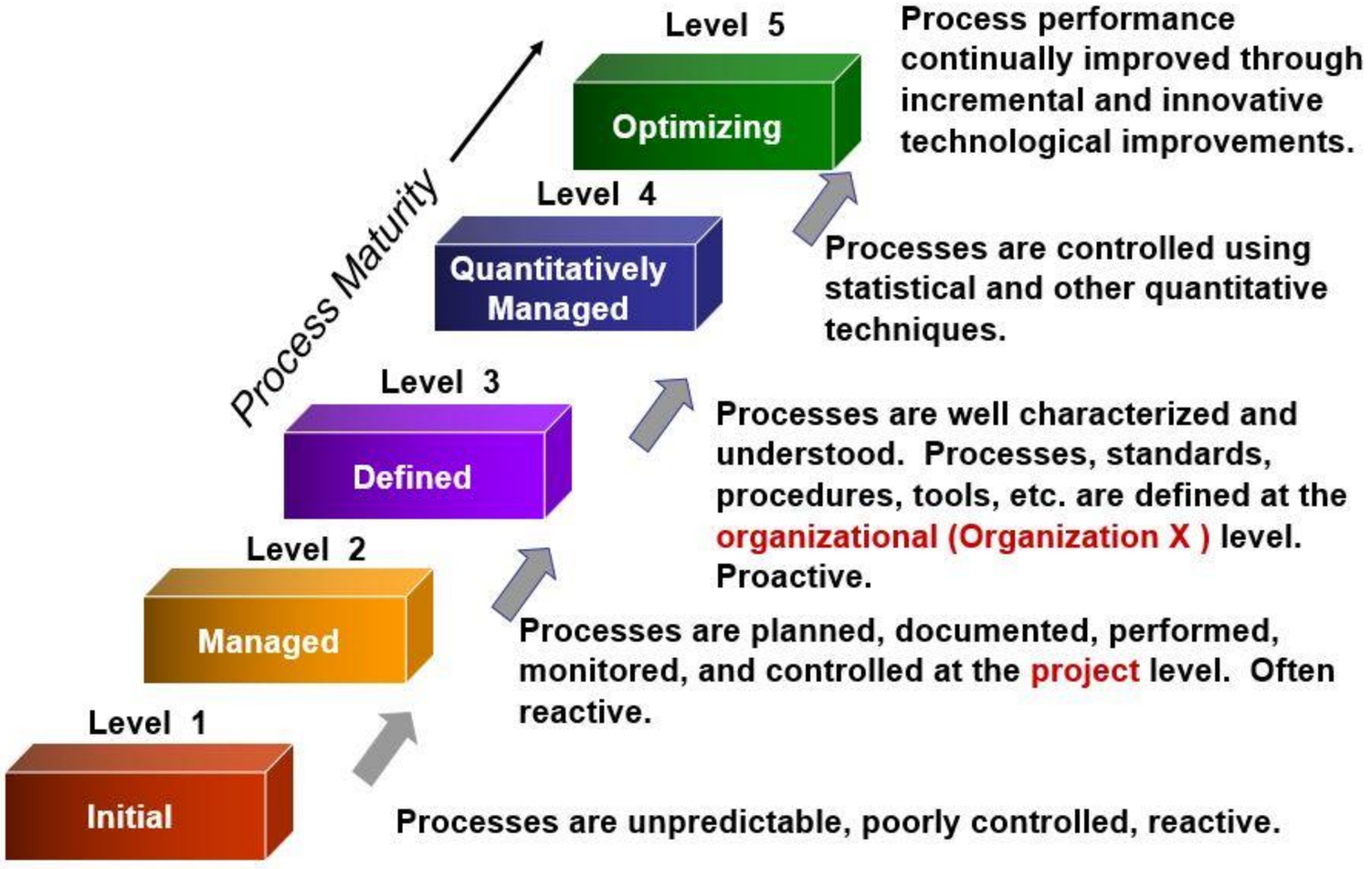


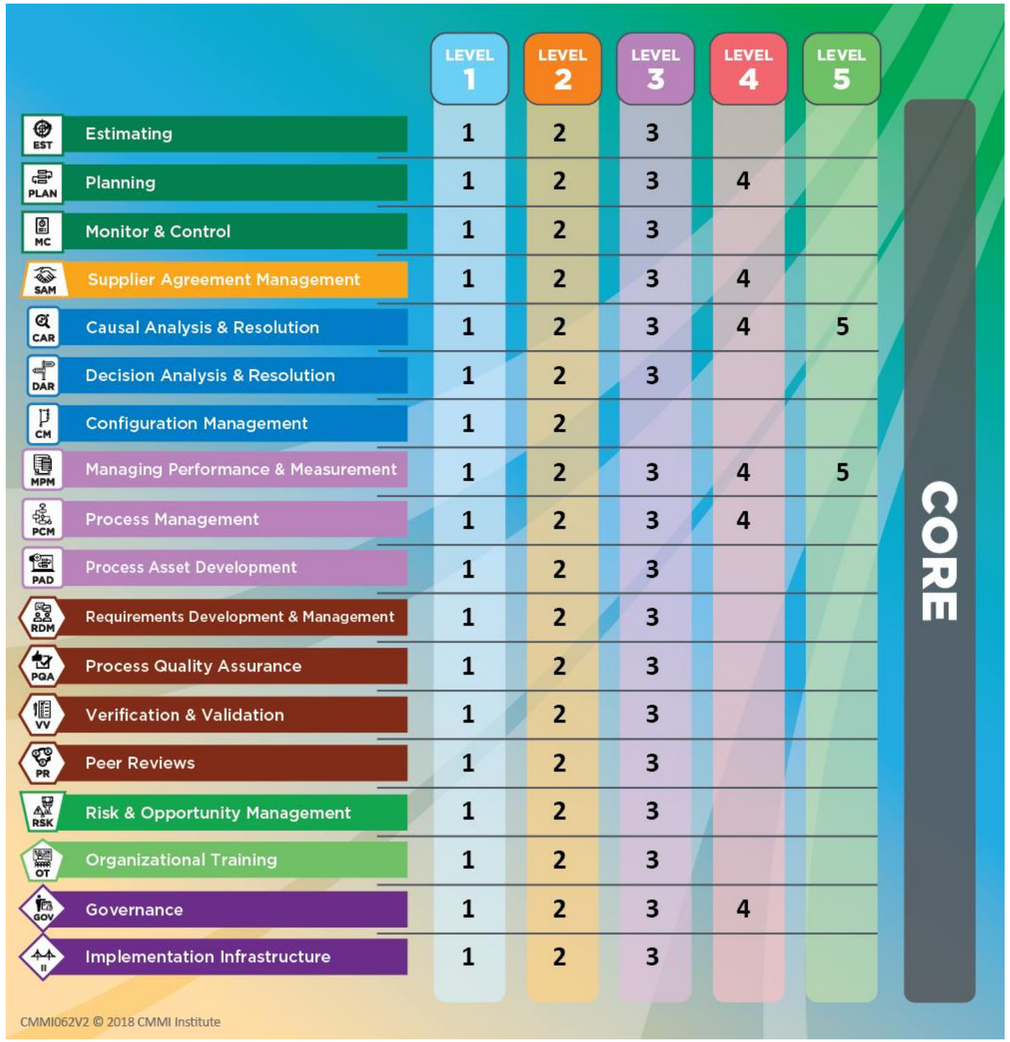
CMMI is a model rather than a prescriptive process. It defines organisational behaviours and goals that commonly lead to better outcomes in software and systems engineering.

What is the purpose of the CMMI model?

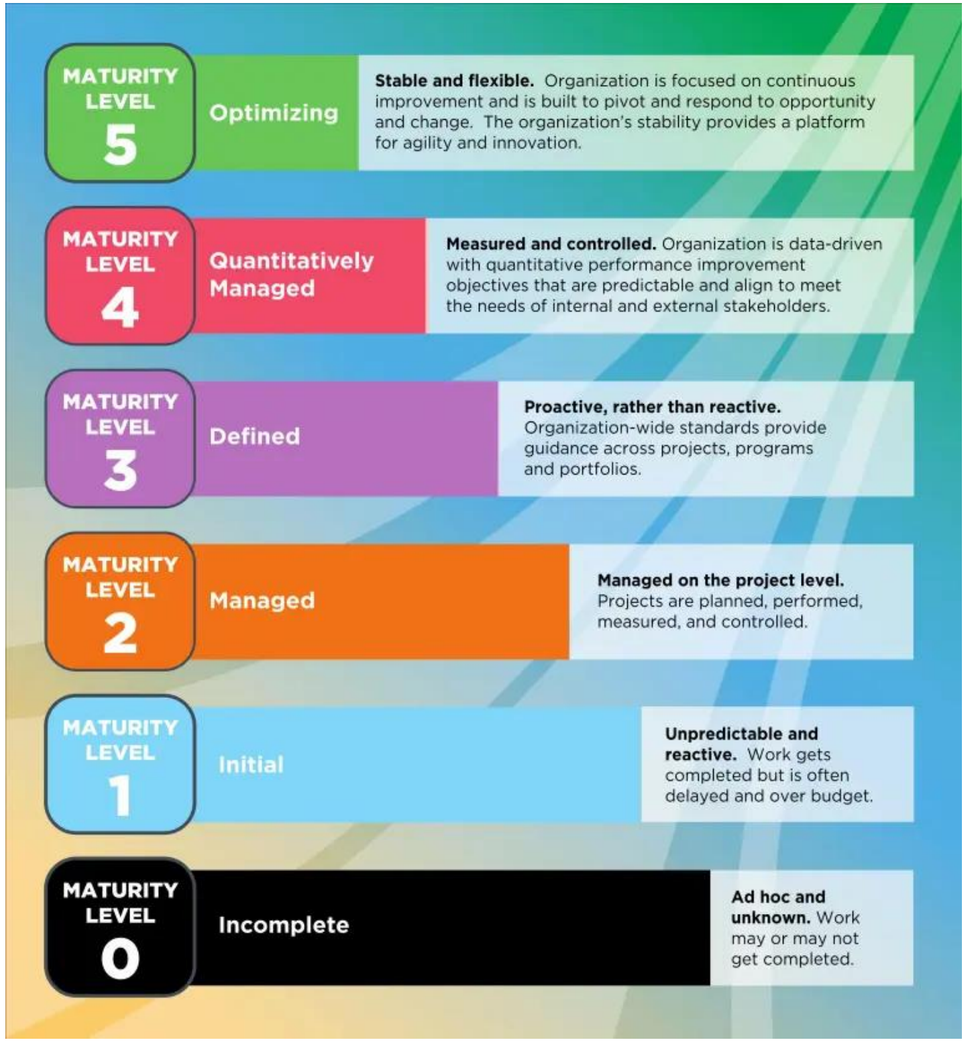
- CMMI helps you assess process maturity and guides process improvement to produce more predictable results and higher-quality products.
- It also provides a structured approach to risk management and to measuring how well an organisation manages risk. The ability to manage risk contributes directly to an organisation’s capacity to deliver high-quality results.
- CMMI doesn't guarantee economic performance.
  - Higher-maturity organisations might manage risk better and become more predictable, but they can also become risk-averse or bureaucratic, which can slow innovation and increase lead times.
  - Lower-maturity organisations sometimes innovate more but operate in a more chaotic, less predictable way; successes there can depend on individual heroics rather than repeatable processes.

CMMI - Organisational





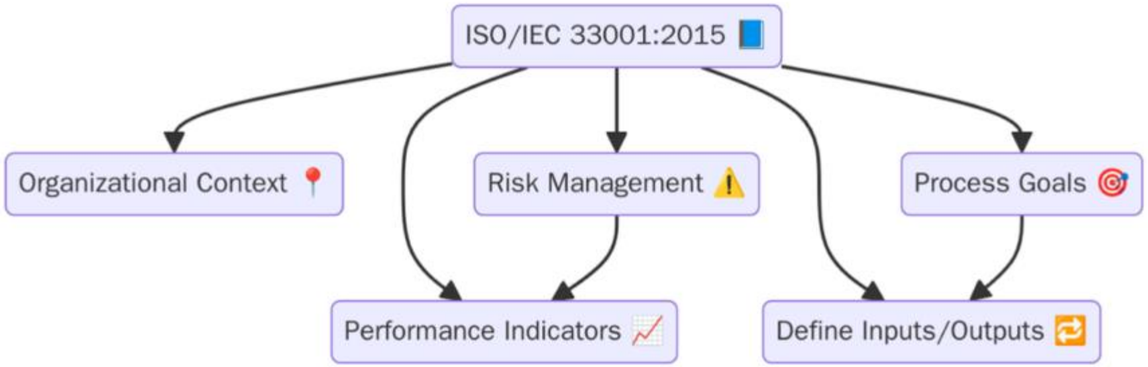
CMMI - Practice



ISO/IEC 33001 (earlier SPICE (Software Process Improvement and Capability dEtermination) ISO/IEC 15504)

ISO/IEC 33001:2015 is an international standard that defines requirements for process assessment within the realm of information technology. It is part of the ISO/IEC 33000 family, which provides guidelines and standards for evaluating and enhancing IT processes. The primary goal of ISO/IEC 33001 is to deliver a structured methodology for assessing process capability to identify improvement opportunities and ensure processes meet high-quality standards.

This standard replaces the earlier ISO/IEC 15504, known as SPICE. It provides a streamlined and adaptable framework suitable for various sectors, including software development, IT services, systems integration, and business process outsourcing.



Quality Improvement Methodologies

What Are These Methodologies?

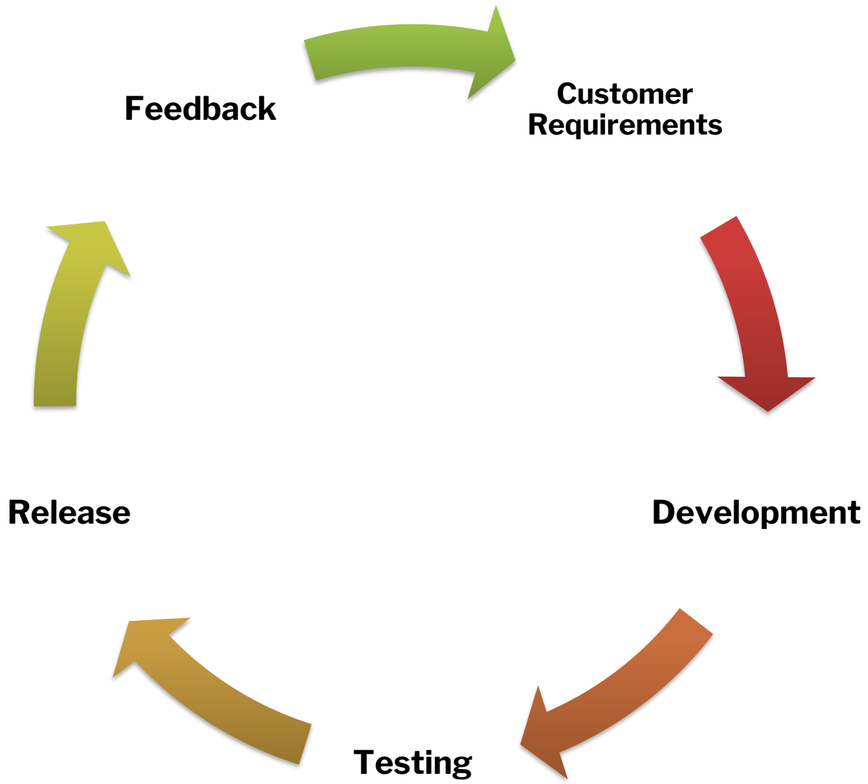
- **Continuous improvement frameworks** for managing quality and reducing waste
- Each has a **unique focus**: culture, incremental change, flow, data-driven analysis, value
- Can be **combined together** for maximum impact
- Originated in manufacturing, now widely used in software and services

TQM (Total Quality Management)

TQM is a company-wide management philosophy where everyone, at every level, is responsible for continuously improving quality with strong customer focus.

Key ideas (software context)

- **Customer focus:** Product decisions driven by user needs and feedback.
- **Total employee involvement:** Developers, testers, support, sales all contribute to quality.
- **Process approach:** Standardised SDLC, coding standards, test processes.
- **Continuous improvement & data:** Defect metrics, customer complaints, uptime, lead time.

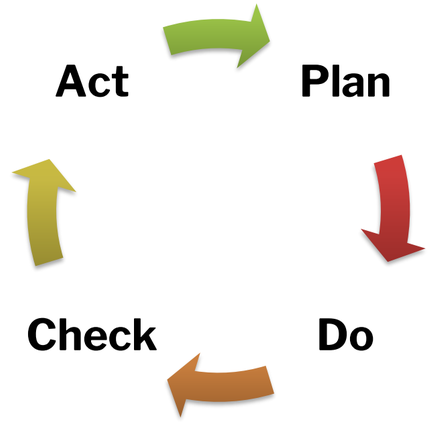


Kaizen

Kaizen is continuous, small, incremental improvements driven by the people doing the work.

Key ideas (software context)

- **Frequent small changes:** Update coding guidelines, tweak CI pipeline, refine code review template.
- **Team-owned:** Developers and testers propose and implement improvements.
- **Regular reflection:** Daily/weekly Kaizen suggestions; retrospectives leading to concrete actions.



Kanban

Kanban is a visual workflow method that limits work in progress (WIP) to improve flow and predictability.

Key ideas (software context)

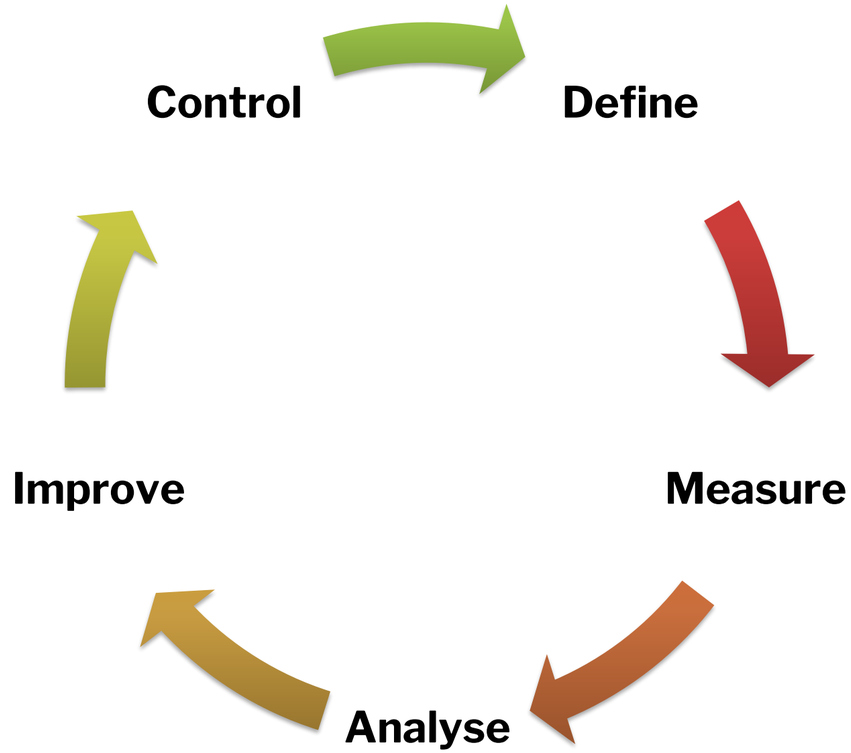
- **Visual board with columns:** To Do → In Progress → Code Review → Test → Done.
- WIP limits per column, e.g., max 3 items in “In Progress”, 2 in “Code Review”.
- Pull system: New work is pulled only when capacity is free.

ASCII Kanban board (very simplified)

To Do	In Progress (WIP=2)	Code Review (WIP=2)	Done
Task A	Task C	Task E	Task B
Task D			

Six Sigma

Six Sigma is a data-driven methodology to reduce defects and variation, typically via the DMAIC cycle: Define–Measure–Analyse–Improve–Control.



Key ideas (software context)

- **Focus on measurable defects** (e.g., escaped defects per release, DPMO in requirements, design, code).
- **Root-cause analysis** with statistical tools (Pareto charts, control charts, regression).
- **Structured projects** led by trained **practitioners** (Green Belts, Black Belts).
  - Black Belt – project leader
  - Master Black Belt - a teacher and mentor for Black Belts
  - Green Belts - project team members

Lean

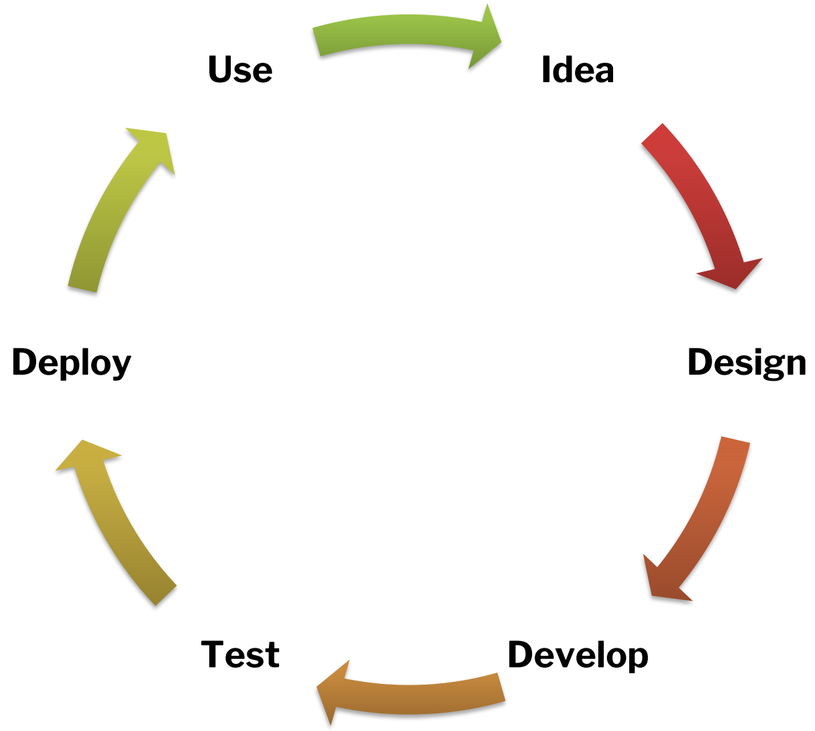
Lean focuses on maximising value and eliminating waste in the software value stream from idea to delivery.

Typical Lean software principles include: eliminate waste, build quality in, create knowledge, defer commitment, deliver fast, respect people, and optimise the whole.

Common wastes in software

- **Extra features** not used by customers (overproduction).
- **Waiting** (e.g., for approvals, environments, other teams).
- **Defects and rework**.
- **Handoffs and motion** (too many transitions and context switches).
- **Overprocessing** (excess documentation not used).

(Lean asks: where is time spent that adds no value?)



Comparison: Focus & Approach

TQM
Organisation culture. Everyone involved. Long-term.

Kaizen
Team-driven small changes. PDCA cycle. Continuous.

Kanban
Visual flow. WIP limits. Pull system.

Six Sigma
Data-driven. DMAIC. Defect reduction.

Lean
Value stream. Waste elimination. Flow.

Strengths & Weaknesses

✓ Kaizen

Low risk, team ownership, fits Agile well

✗ May miss structural issues

✓ Kanban

Improves flow, makes bottlenecks clear

✗ Needs WIP discipline, doesn't define methods

✓ Six Sigma

Quantifies improvements, fact-based

✗ Heavyweight, statistical complexity

✓ Lean

Improves speed and value flow

✗ "Waste" can be misinterpreted

When to Use Each Methodology

TQM
Company policy, executive buy-in needed, multi-year rollout

Kaizen
1–2 improvements per sprint, low-risk, fast feedback

Kanban
Support/ops team, manage flow, visible bottlenecks

Six Sigma
Chronic defect/incident problem, needs data analysis

Lean
End-to-end pipeline, value-stream redesign, speed focus

Combining Methodologies

These frameworks are not competing—they can work together:

TQM + Kaizen

- Company quality culture drives daily team improvements

Kanban + Lean

- Visualise flow + eliminate waste ("Leanban")

Six Sigma + Lean

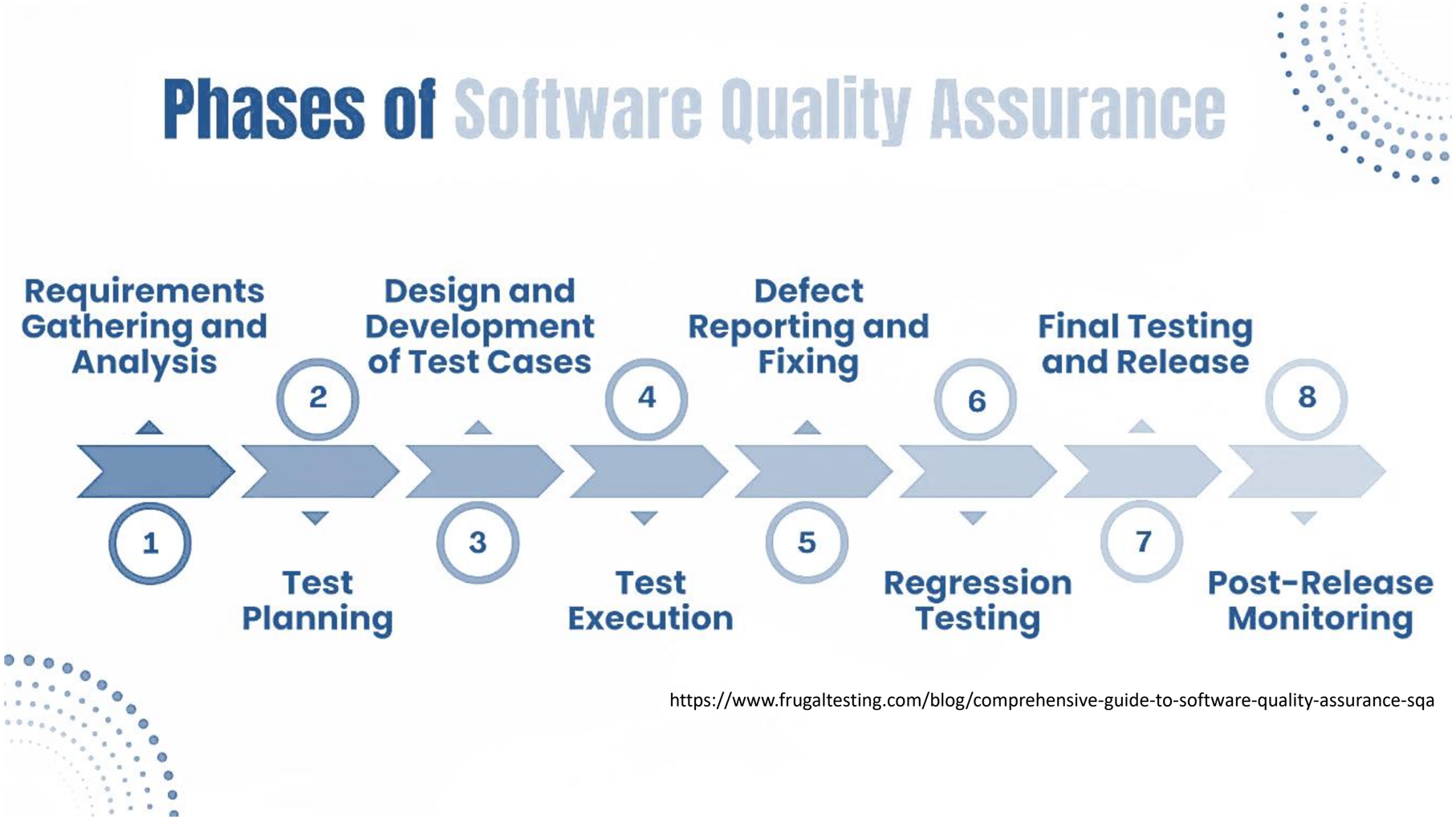
- Data-driven root cause analysis for large Lean initiatives

Scrum + Kanban

- Sprints with Kanban flow ("Scrumban")

SQA Tools

SQA…

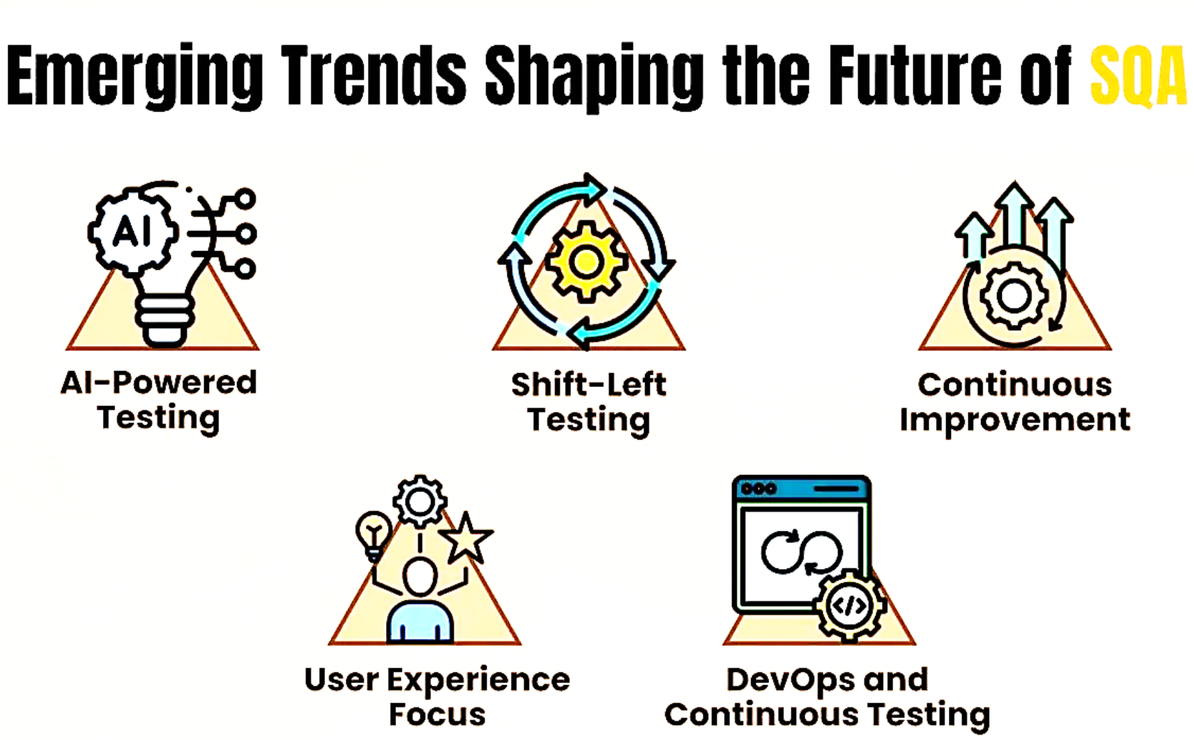


https://www.frugaltesting.com/blog/comprehensive-guide-to-software-quality-assurance-sqa

Tools in SQA

Phase / Process	Tools
Documentation and Management (inc. Test Management)	• Zephyr • Office tools • TestRail
Bug tracking	• Bugzilla • Redmine • Jira • Trello • MantisBT
Unit Testing	• JUnit • Compiler/IDE platforms
Automated functional testing tool	• Selenium • Appium • Cucumber • Junit • TestComplete
Performance	• LoadRunner • Apache JMeter
Security	• OWASP Zed Attack Proxy • Fortify • Nessus
API	• Postman

Future of SQA



https://www.frugaltesting.com/blog/comprehensive-guide-to-software-quality-assurance-sqa

1. AI-Powered Testing: The application of AI in testing automates repetitive processes and improves accuracy, hence increasing overall testing efficiency.

2. Shift-Left Testing: By performing testing early in the development cycle, teams can detect and address issues before they become more expensive to fix.

3. Continuous Improvement: Constantly refining testing procedures and processes guarantees that quality criteria are regularly satisfied and that software quality improves with time.

4. User Experience Focus: Early testing helps detect any issues that may affect the end user, resulting in a favourable product experience.

5. DevOps and Continuous Testing: Integrating testing into DevOps processes enables continuous testing, maintaining software quality throughout the development process.