CHAPTER 4
SOFTWARE QUALITY STANDARDS AND MODELS

Dr. Sarah Naiem

SOFTWARE QUALITY STANDARDS AND MODELS

Software quality assurance (SQA) is critical to software development, ensuring that products meet predefined quality benchmarks.

Software quality standards and models provide frameworks and guidelines to ensure software products meet specified quality criteria.

These standards help organizations implement best practices, achieve consistency, and improve customer satisfaction.

This chapter explores key software quality standards and models, including ISO 9001, ISO/IEC 25010, IEEE standards, CMMI, Six Sigma, Lean methodologies, ITIL, and COBIT.

ISO STANDARDS FOR SOFTWARE QUALITY

ISO 9001: Quality Management System

ISO 9001 is a globally recognized standard for quality management systems (QMS). While it is not specific to software, it provides a framework for process improvement and customer satisfaction.

Organizations seeking ISO 9001 certification must stick to its principles, including:

- **Customer focus** – Ensuring that customer needs and expectations drive quality efforts.
- **Leadership commitment** – Management must actively support and promote quality practices.
- **Process approach** – Emphasizing process optimization for consistent outcomes.
- **Continuous improvement** – Regularly reviewing and improving quality processes.
- **Evidence-based decision-making** – Using data and analytics to drive quality decisions.

ISO MODEL FOR SOFTWARE QUALITY

ISO/IEC 25010: Software Product Quality Model

- ISO/IEC 25010 is a comprehensive model for evaluating software product quality. It categorizes software quality into eight key characteristics:
  - **Functional Suitability** – Ensuring software meets specified functional requirements and business needs.
  - **Performance Efficiency** – Measuring response time, resource utilization, and scalability.
  - **Compatibility** – Ensuring interoperability and co-existence with other systems.
  - **Usability** – Assessing ease of use, accessibility, and user satisfaction.
  - **Reliability** – Evaluating fault tolerance, recoverability, and availability.
  - **Security** – Covering data protection, authentication, and access control.
  - **Maintainability** – Assessing modularity, reusability, and code maintainability.
  - **Portability** – Ensuring adaptability across different environments.

IEEE STANDARDS FOR SOFTWARE QUALITY

IEEE Standards for Software Quality

- The IEEE (Institute of Electrical and Electronics Engineers) has established multiple standards addressing software quality. Key IEEE standards include:

- **IEEE 730 (Software Quality Assurance Plans)** – Provides guidelines for developing structured SQA plans, ensuring quality activities are well-documented.
- **IEEE 829 (Software Test Documentation)** – Defines standard documentation for software testing processes, ensuring traceability and completeness.
- **IEEE 1012 (Software Verification and Validation)** – Establishes best practices for software verification (ensuring correct implementation) and validation (ensuring customer requirements are met).
- **IEEE 12207 (Software Life Cycle Processes)** – Standardizes software development and maintenance processes, emphasizing consistency and best practices in software engineering.

CAPABILITY MATURITY MODEL INTEGRATION (CMMI)

CMMI is a globally recognized process improvement framework developed by the Software Engineering Institute (SEI).

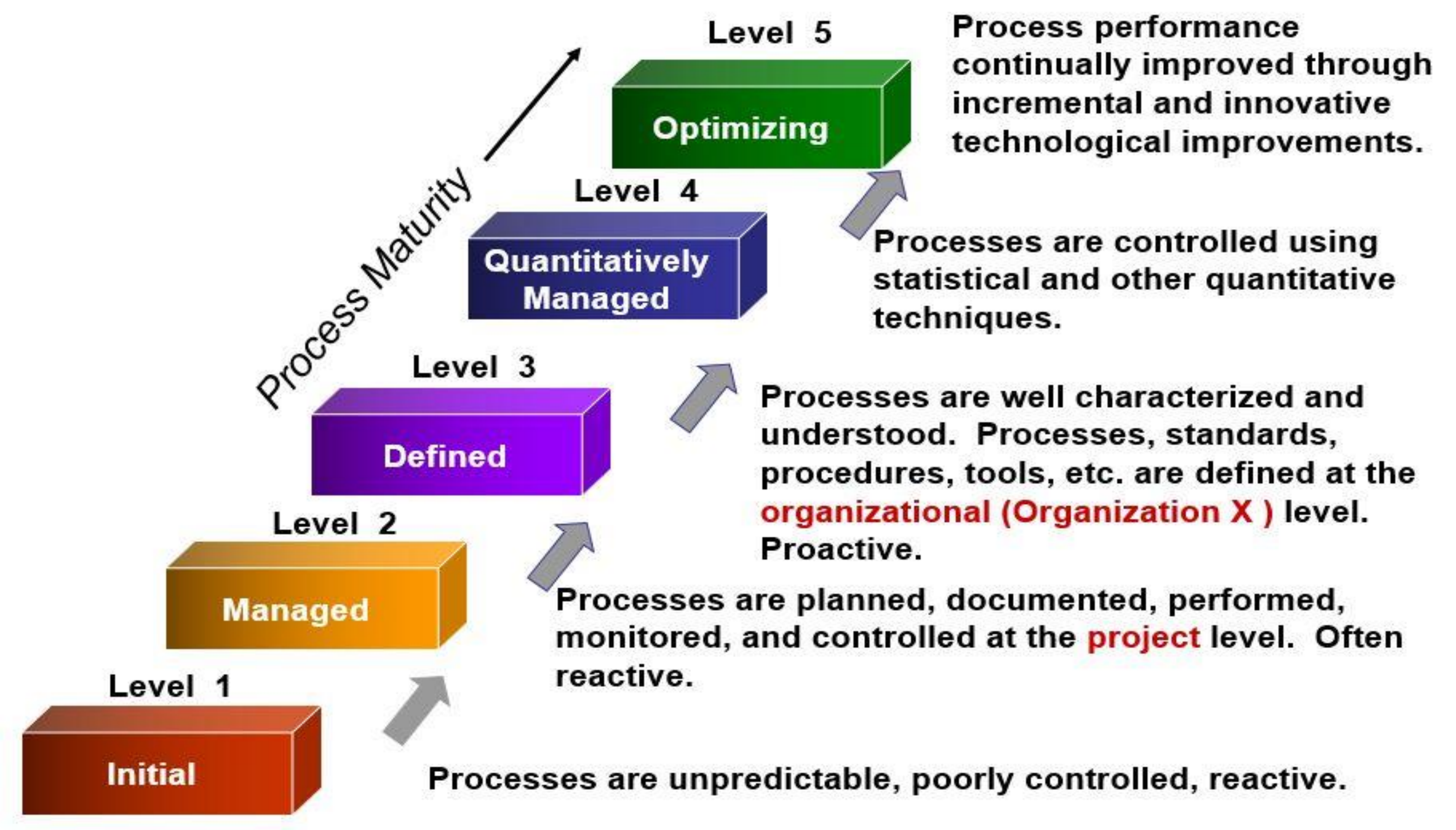
CMMI is designed to:

- Improve **process standardization and consistency**
- Reduce **software defects and project risks**
- Enhance **predictability of project outcomes**
- Increase **customer satisfaction** by delivering high-quality software
- Foster **continuous improvement** within an organization.

It provides a **structured roadmap** to move from ad-hoc, unpredictable processes to well-defined, optimized workflows.

CAPABILITY MATURITY MODEL INTEGRATION (CMMI)

- The Capability Maturity Model describes a continuum of five stages based on how well a company or organization follows common and repeatable processes to get work done.
- The low end of the scale describes companies without repeatable processes, where much of the work is chaotic and ad-hoc.
- The highest end describes companies that use defined and repeatable processes, collect metrics to help them continuously improve their processes, and look for creative ways to do things better on an ongoing basis.
- In the past, there were a number of different CMM models. These models were combined in 2002 into one integrated model – hence the newer acronym of CMMI



Process performance continually improved through incremental and innovative technological improvements.

Processes are controlled using statistical and other quantitative techniques.

Processes are well characterized and understood. Processes, standards, procedures, tools, etc. are defined at the organizational (Organization X) level. Proactive.

Processes are planned, documented, performed, monitored, and controlled at the project level. Often reactive.

Processes are unpredictable, poorly controlled, reactive.

CMMI MATURITY LEVELS

- There are different interpretations of the CMMI framework, and some companies have even created their customized versions. However, in general, CMMI consists of **five maturity levels**:
  - **Ad-hoc / Crisis Mode** – The organization lacks standardized processes, and project success depends entirely on individual skills. There is little organizational support to ensure consistency, making project outcomes unpredictable.
    - Many companies fall into this category, with some jokingly considering themselves at level 0 or even -1.
  - **Basic Project Management** – The organization has introduced **standard project management processes**, which are applied across all projects. This serves as a foundational step toward further improvements. Companies that begin adopting CMMI typically aim to reach this level.
  - **Standardized Software Development** – The company now focuses on **consistent software development practices**, just as it did for project management at level 2. This includes standardized and repeatable processes, common tools, and defined deliverables.
  - **Managed & Data-Driven** – The organization actively **collects and analyzes project metrics** to track performance and improve decision-making. Past project data is stored in a repository and used as a reference for new projects.
  - **Optimized & Continuously Improving** – The company has established a **cycle of process execution, measurement, and continuous improvement**. Feedback loops, data insights, and innovation drive ongoing process optimization.

SIX SIGMA FOR SOFTWARE QUALITY

- Six Sigma is a data-driven methodology aimed at reducing defects and improving quality. It follows the **DMAIC (Define, Measure, Analyze, Improve, Control)** approach:
  - **Define** – Identify quality goals, customer requirements, and problem areas.
  - **Measure** – Collect and analyze performance data to understand current quality levels.
  - **Analyze** – Identify root causes of defects and inefficiencies.
  - **Improve** – Implement solutions and process changes to enhance quality.
  - **Control** – Maintain improvements through monitoring and continuous feedback.
- Six Sigma methodologies help software teams minimize defects, improve performance, and enhance customer satisfaction.

LEAN SOFTWARE DEVELOPMENT FOR SOFTWARE QUALITY

Lean principles focus on eliminating waste and improving efficiency. The key Lean principles include:

- **Eliminate waste** – Remove unnecessary code, processes, and documentation that do not add value.
- **Build quality in** – Ensure quality at every stage of development rather than relying on post-release testing.
- **Deliver fast** – Prioritize quick releases and iterative development to respond to customer needs efficiently.
- **Empower teams** – Encourage collaboration, ownership, and continuous improvement among software teams.
- **Optimize the whole** – Consider the entire software development lifecycle rather than isolated tasks to improve efficiency.

Framework	Purpose	Key Features	Focus Area
ISO 9001	Quality Management System	Process-based approach, customer focus, continuous improvement	Organization-wide quality management
ISO/IEC 25010	Software Product Quality Model	Defines quality attributes (e.g., functionality, performance, security, maintainability)	Software product evaluation
IEEE Standards	Software Engineering Standards	Guidelines for testing, verification, validation, and lifecycle management	Software development best practices
CMMI (Capability Maturity Model Integration)	Process Maturity Framework	Five maturity levels (Initial, Managed, Defined, Quantitatively Managed, Optimizing)	Process improvement and maturity
Six Sigma	Quality Improvement Methodology	DMAIC (Define, Measure, Analyze, Improve, Control), defect reduction	Data-driven quality enhancement
Lean Software Development	Efficiency & Waste Reduction	Eliminating waste, empowering teams, delivering fast, optimizing the whole	Agile and DevOps software practices