軟體品質保證大師課

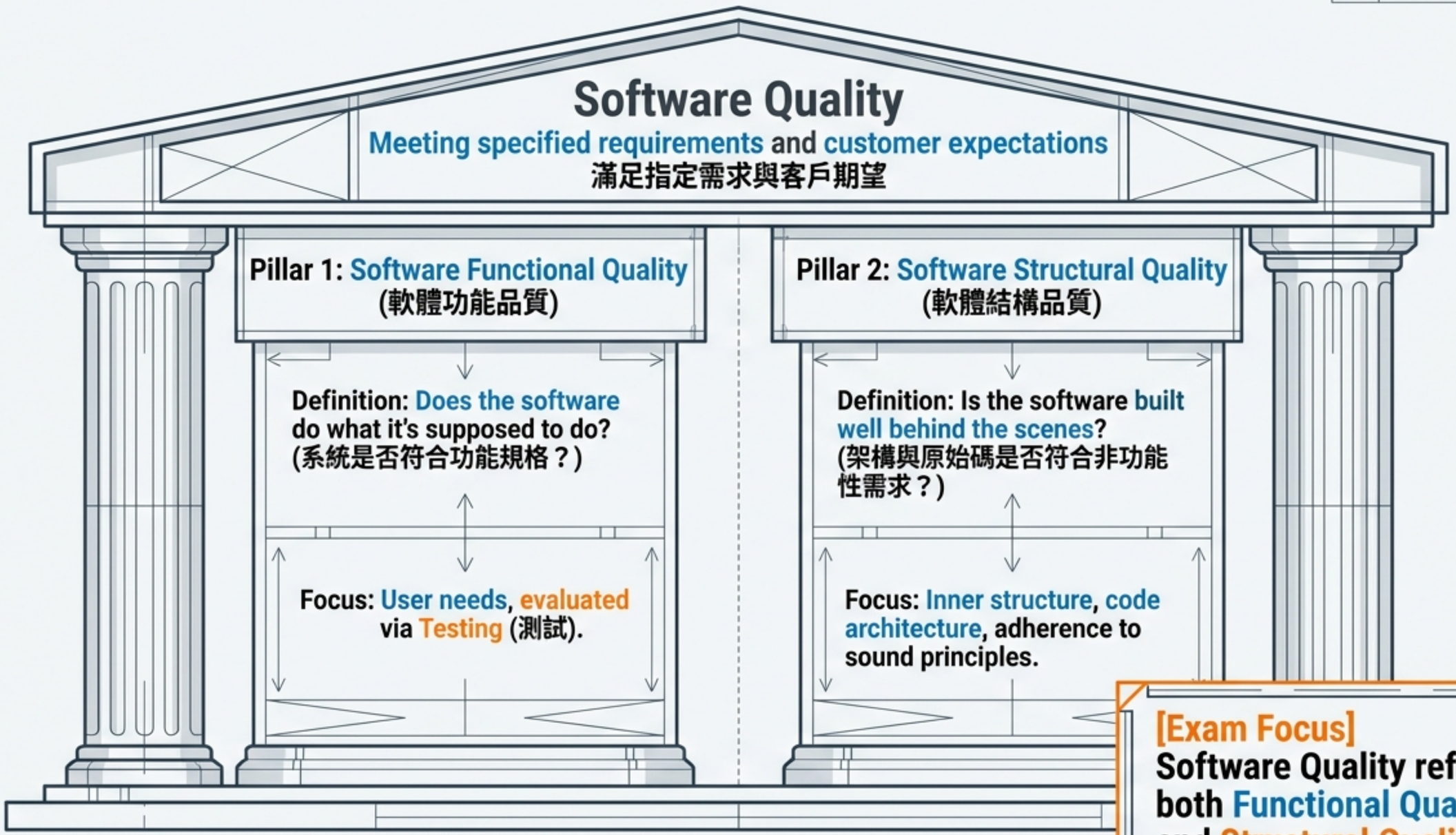
Masterclass in Software Quality Assurance

From Zero to Mastery: The Feynman Approach to SQA

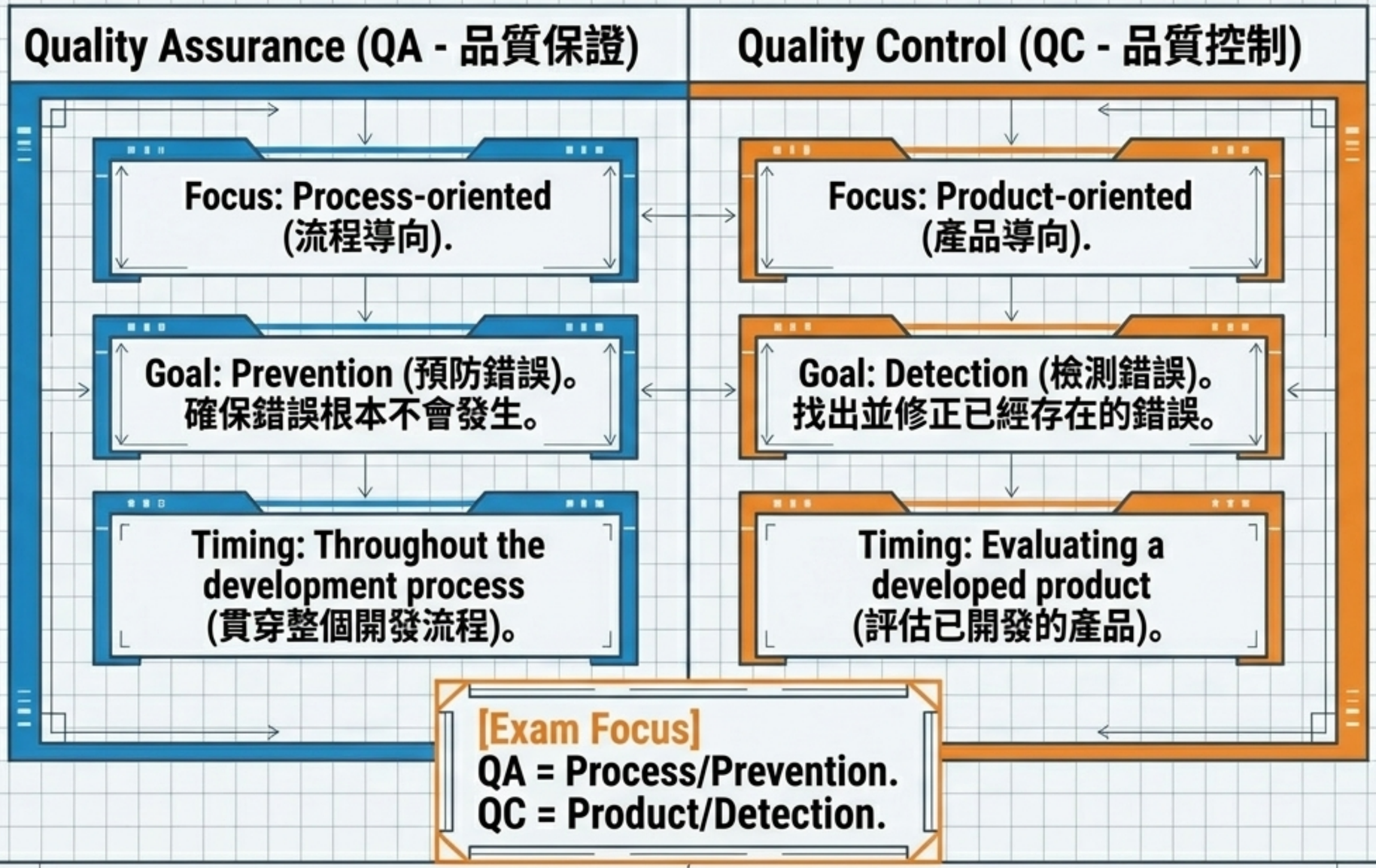
Professor's Note

「品質是設計出來的，不是測出來的。」(Quality is built in, not tested in.)

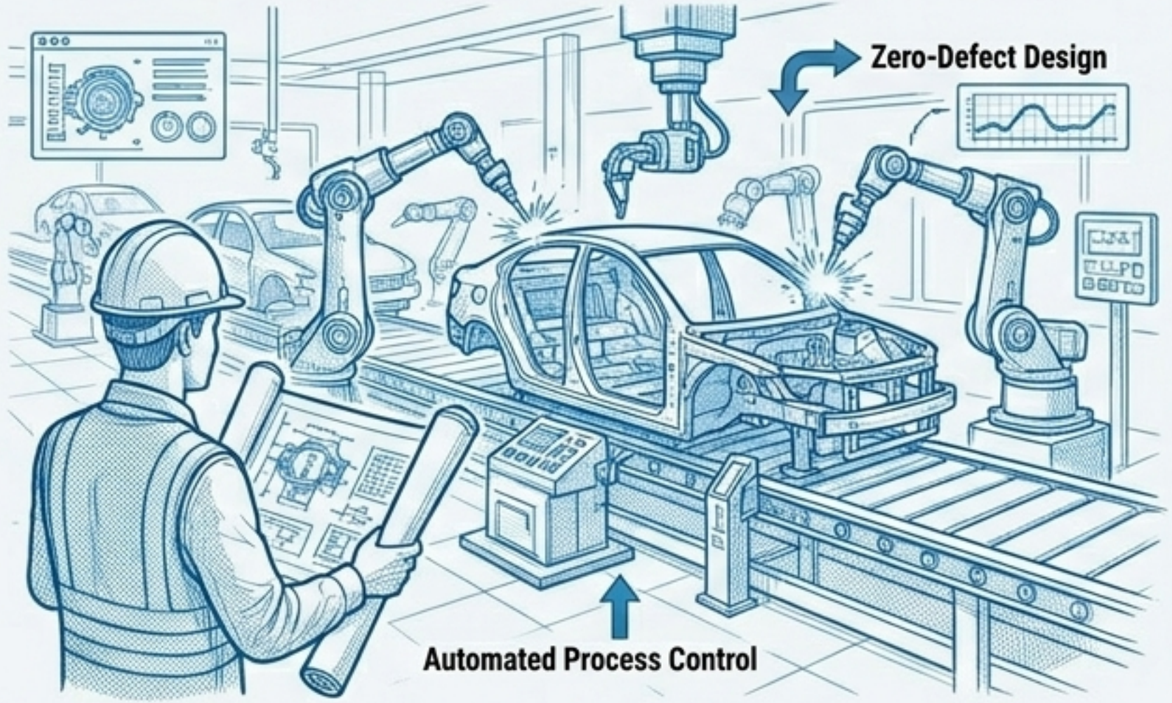
歡迎來到本課程。我們的首要目標是帶領你從零基礎，徹底掌握 Software Quality Assurance (SQA) 的核心邏輯與實務標準。


Software Quality
Meeting specified requirements and customer expectations
滿足指定需求與客戶期望
Pillar 1: Software Functional Quality
(軟體功能品質)
Definition: Does the software do what it's supposed to do?
(系統是否符合功能規格？)
Focus: User needs, evaluated via Testing (測試).
Pillar 2: Software Structural Quality
(軟體結構品質)
Definition: Is the software built well behind the scenes?
(架構與原始碼是否符合非功能性需求？)
Focus: Inner structure, code architecture, adherence to sound principles.
[Exam Focus]
Software Quality refers to both Functional Quality and Structural Quality.

Quality Assurance (QA - 品質保證) vs. Quality Control (QC - 品質控制)



Feynman Conceptual Analogy: QA (Quality Assurance - 品質保證)



Feynman Explanation: QA 就像是在「設計完美的汽車組裝線」。如果組裝線設計得無懈可擊，工廠就「不可能」製造出有缺陷的汽車。這就是 Prevention (預防).

Feynman Conceptual Analogy: QC (Quality Control - 品質控制)

Feynman Explanation: QC 就像是「在生產線末端檢查成品的檢驗員」。如果大燈不亮，就把車退回。這就是
這就是 Detection (檢測).

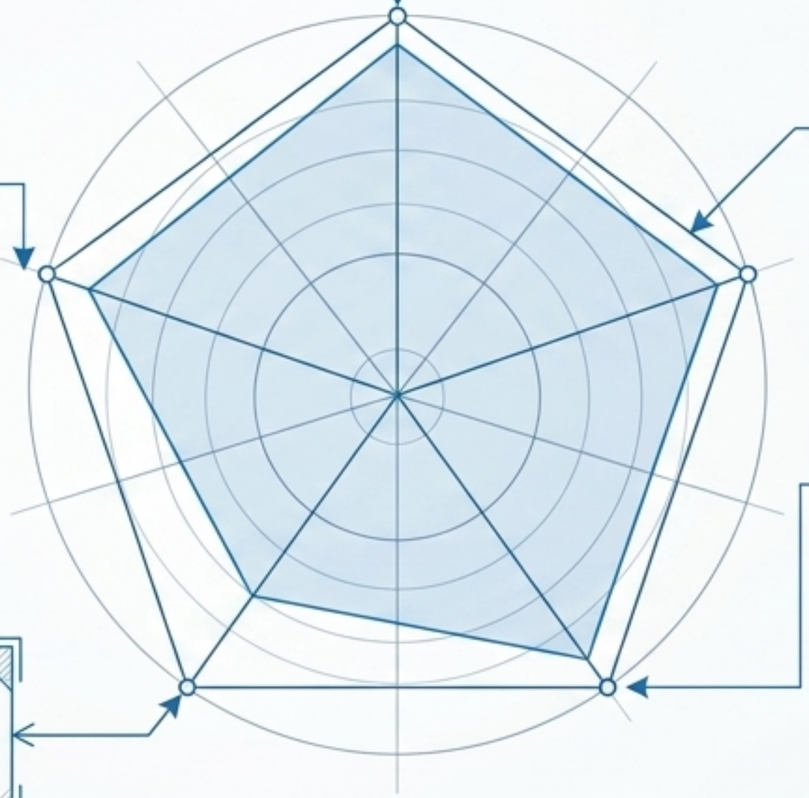
[Professor's Note]
如果只有 QC，你會丟掉很多壞掉的車（浪費成本）。 QA 確保你一開始就不會做出壞車。

The 5 Measurable Attributes of Software Quality (軟體品質的五大可測量屬性)

Reliability (可靠性):
Mean time between failures.
系統穩定運作不當機的能力。

Efficiency (效率):
Resource usage and performance.
系統執行速度與資源消耗。

Size (規模):
Lines of code or function points.
專案的龐大程度。

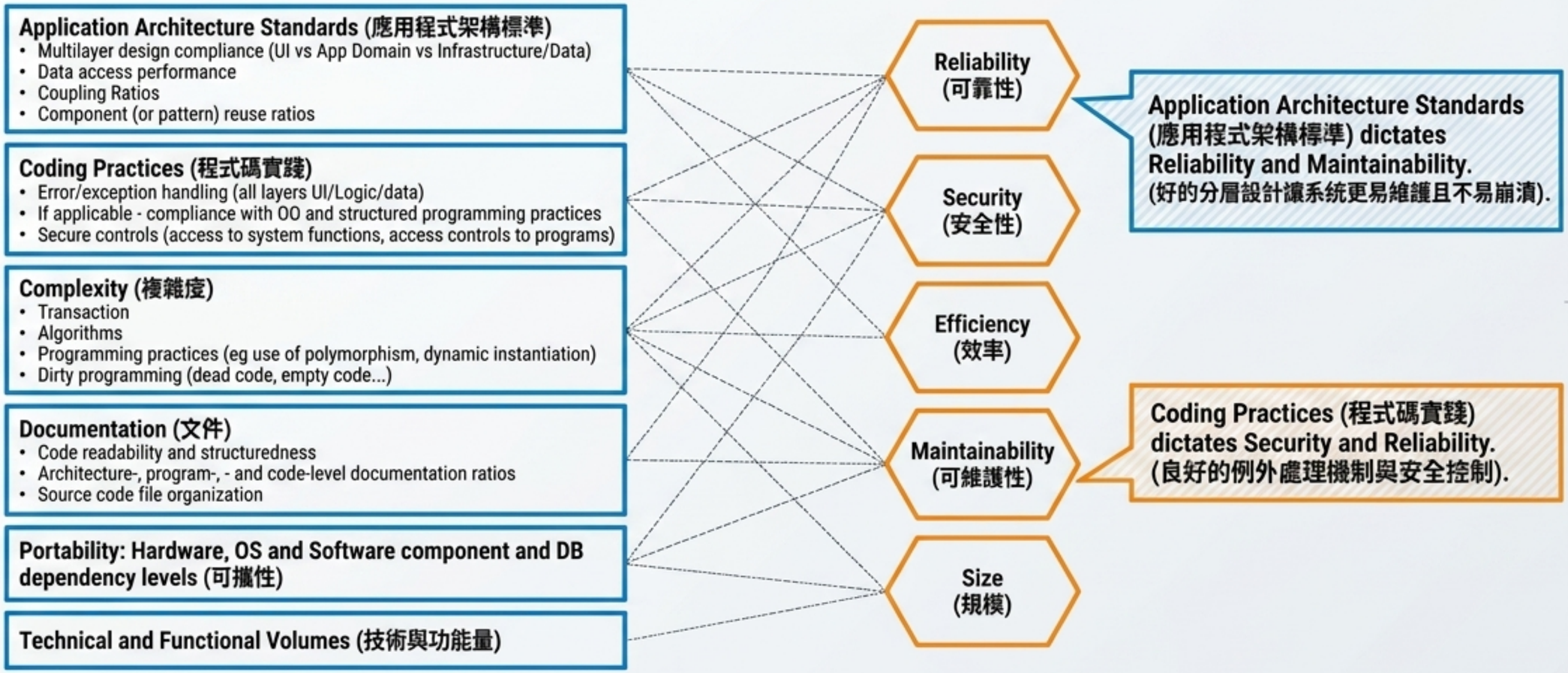


Security (安全性):
Protection of data/system.
防禦攻擊與資料保護。

Maintainability (可維護性):
Ease of fixing/updating.
程式碼修改與擴充的難易度。

[Exam Focus]
Memorize these 5 attributes: **Reliability, Efficiency, Security, Maintainability, Size.**

Mapping Characteristics to Attributes (特徵與屬性的關聯)



Professor's Note | 結構特徵是「因 (Cause)」，可測量屬性是「果 (Effect)」。

Deep Dive: Coding Practices & Complexity (深入探討：程式碼實踐與複雜度)

Concept Expansion (概念增潤)

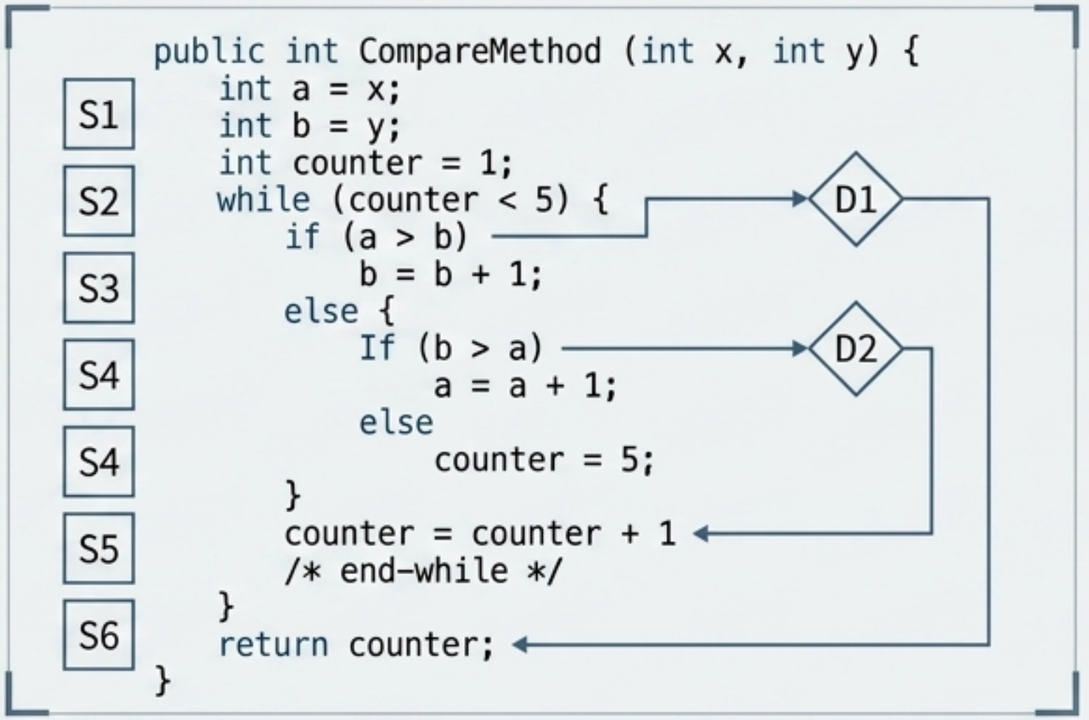
為了徹底掌握程式複雜度 (Complexity)，我們必須了解核心單位。

Class (類別)：

is a blueprint for objects. (類別是物件的藍圖)。

Method (方法)：

is an action the object performs. (方法是執行的動作)。



Complexity (複雜度)：圖中的 D1, D2, D3 代表「決策節點 (Decision nodes)」。演算法越複雜、包含越多 'Dirty programming' (如無效程式碼)，**Efficiency** (效率) 與 **Maintainability** (可維護性) 就越低。

Quality Assurance Standards (品質保證標準)

ISO 9000 Family (品質管理系統)

- **ISO 9001:2008:** Sets requirements (制定要求).
- **ISO 9004:2009:** Focuses on efficiency/effectiveness (專注於效率與效能).

ISO/IEC 25000:2005 (SQuaRE)

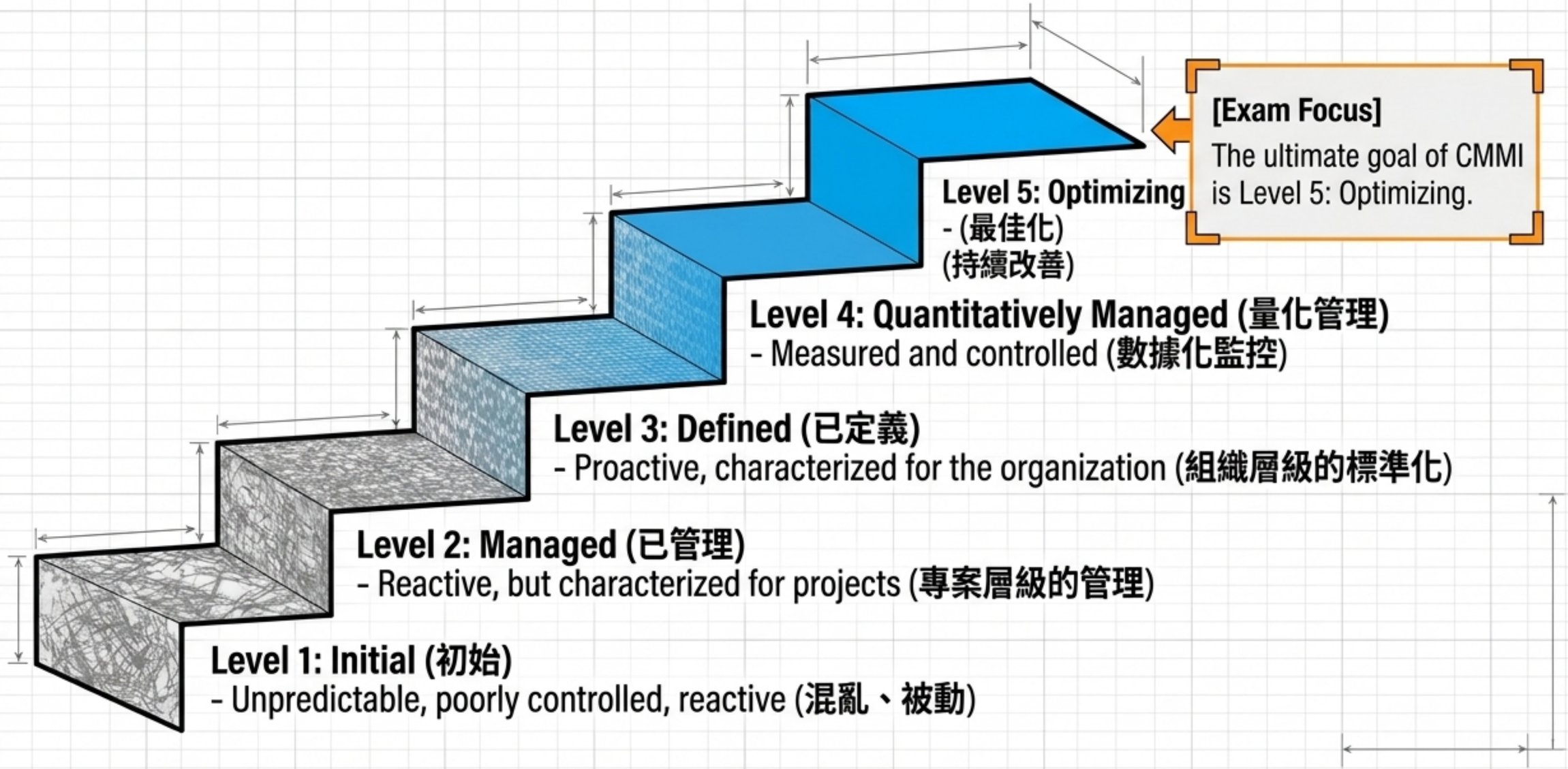
Software product Quality Requirements and Evaluation.
(專注於軟體產品的品質要求與評估).

IEEE SA 1061

Methodology for Software Quality Metrics spanning the entire lifecycle.
(涵蓋整個生命週期的軟體品質度量方法)

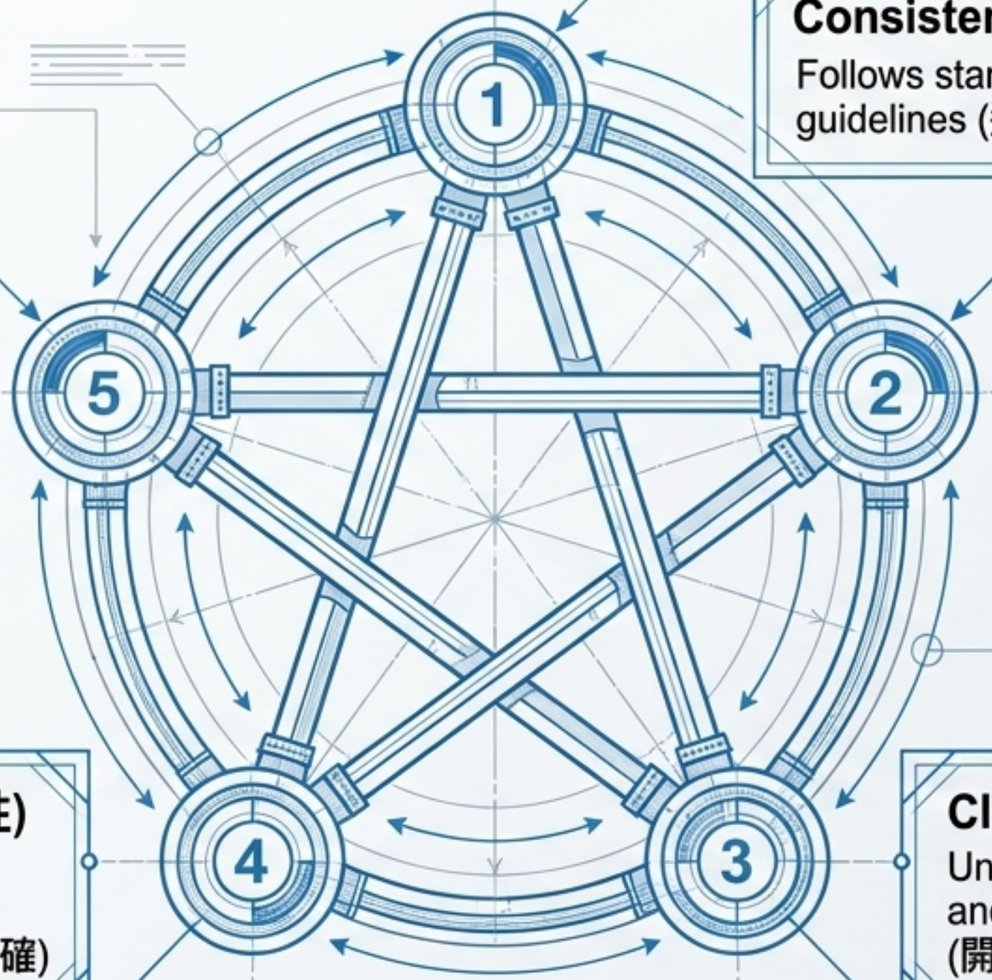
[Exam Focus]
Match ISO 9001 with “requirements”, and SQuaRE with ISO/IEC 25000.

Process Maturity: CMMI Levels (能力成熟度模式整合)



The 5 C’s of Quality Planning (品質計畫的五大核心原則)

在執行 QA/QC 前，必須先制定計畫：



Consistent (一致性)
Follows standards and guidelines (遵循業界標準與規範)

Complete (完整性)
Includes overall representation of requirements
(完整涵蓋所有專案需求)

Clear (清晰性)
Understandable to developers and stakeholders
(開發者與利害關係人皆能輕易理解)

Correct (正確性)
Accurate details regarding delivery
(交付細節與時程絕對正確)

Constructible (可建構性)
Adaptable if design errors occur
(若發生設計錯誤，計畫具有可調整與重構的彈性)

Quality Control Activities (品質控制具體行動)

QC 是一套結構化的審查與檢驗流程：

Step 1: Quality Control Staff
(指派專業品管人員)。

↓

Step 2: Quality Control Reviews
(品管審查，包含 Constructibility 可建構性審查)。

↓

Step 3: Verification & Validation (V&V)
(驗證與確認 - 做對的事，把事做對)。

↓

Step 4: Software Control Methods & Testing
(執行軟體控制方法與最終測試)。

[Exam Focus]

Verification & Validation (V&V) and Testing are core QC activities.

Software Testing Stages (軟體測試階段)

Acceptance Tests (驗收測試): Goal is to confirm user needs are met; key tester is the User/Client.

System Tests (系統測試): Evaluating the complete, integrated system (e.g., Performance testing).

Integration Tests (整合測試): Testing combined parts of an application to determine if they function together correctly.

Unit Tests (單元測試): Testing individual components/methods.

```
public class TestStudent extenda TestCase (

    private Student student1 = new Student();
    private Student student2 = new Stodent();
    private Student student3 = new Student();
    private Student student4 = new Student();
    private Student student5 = new Student();

    public void testCase1() {
        // B5(a) to be completed
    }
    public void testCase2() {
        // B5(b) to be completed
    }
    public veid testCase3() {
        // B5(d) to be complsted
    }
    public void testCase5() {
        // B5(e) to be complated
    }
```

[Exam Focus]

Testing techniques include **Black-box** (focus on inputs/outputs) and **White-box** (focus on internal code structure).

Modern SQA Perspectives (現代 SQA 觀點 - AST Report)

1. Skill is Paramount (技術至上)

測試人員的技能至關重要，需持續更新與升級。

2. Context-Driven (情境導向)

測試方法必須根據專案情境量身打造 (fit for the purpose)。

3. Start at Requirements (從需求開始)

品質改善應從 Requirement Level (需求層級) 就開始落實 (這正是 QA 的精髓)。

4. Shared Responsibility (共同責任)

程式設計師同樣對品質負責；測試員應採用 Exploratory Testing (探索性測試) 來找出隱藏的 Bug。

Exam Spotlight: Connecting Theory to Practice (考試重點解析)

Prompt Box

Distinguish QA vs. QC.

Answer Box

QA = Process-oriented, Prevention-focused (預防).
QC = Product-oriented, Detection-focused (檢測).

Prompt Box

List 4 measurable attributes of Software Quality.

Answer Box

Reliability (可靠性)
Efficiency (效率)
Security (安全性)
Maintainability (可維護性)

Prompt Box

Who is the key tester in Acceptance Testing?

Answer Box

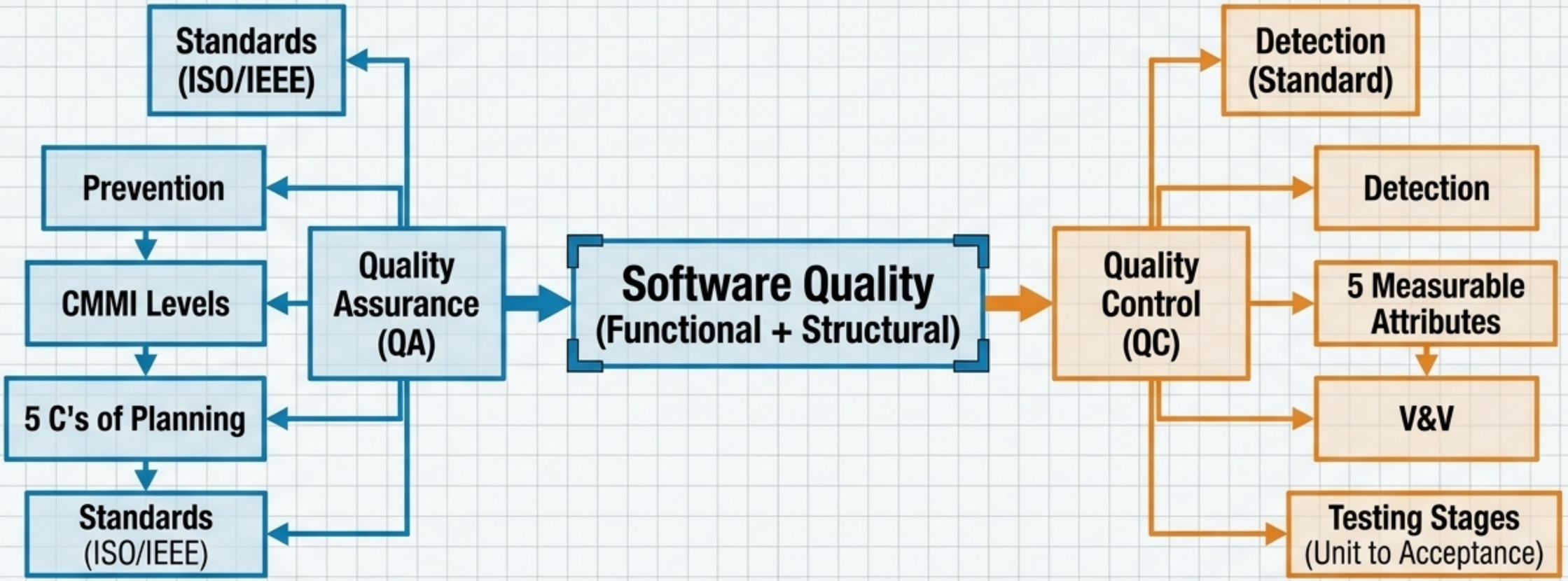
The **End-User / Client** (終端使用者 / 客戶)

[Exam Focus]

Consolidate key concepts and definitions for high-yield exam questions.

Understand the core distinctions between Quality Assurance (process) and Quality Control (product) activities, and recall fundamental software quality characteristics.

The SQA Master Blueprint (總結：軟體品質保證藍圖)



Professor's Final Word

「真正的品質保證，是在寫下第一行程式碼之前就開始的。掌握流程 (QA)，嚴格把關 (QC)，你就是卓越的軟體工程師。」

(True QA starts before the first line of code is written. Master the process, rigorously control the product, and you become an elite software engineer.)