SOFTWARE QUALITY ENGINEERING
BSE : 7th SEMESTER

DAY:

9/9/25

⊛ **Quality:**

Measurable thing/characteristic related to software.

→ **Types:**

Both Quality matters { ⊙ Product, ⊙ Process

→ Main Process Models will be discussed

→ **Terms:**

Quality Assurance (QA)

Testing (Python)
- Unit Testing
- Python Testing

⊛ To ensure quality products, software quality engineering is used.

⊛ SQA (Test Developed Product)

DATE: DAY:

* SQE Vs. SQA
  - SQE → CI/CP → Proactive
  - SQA → QC → Reactive

11/9/25

Case Study

* Maintenance problem
* Lack of Monitoring and testing of dead code.
* No alert message or triggering of error.

** PLO's of Degree (study it) & Course Outline

16/09/25

* Quality intangible trait but measurable.

→ Process → Project → Product

* Quality Parameters
  - Technical Quality Parameters
    - Reliability
    - Performance
    - correctness
    - Capability
    - Maintainability
  - User Quality Parameters
    - Easy to use (Usability)
    - Documentation
    - Portability
    - Availability

DATE: DAY:

⊛ Class Activity.

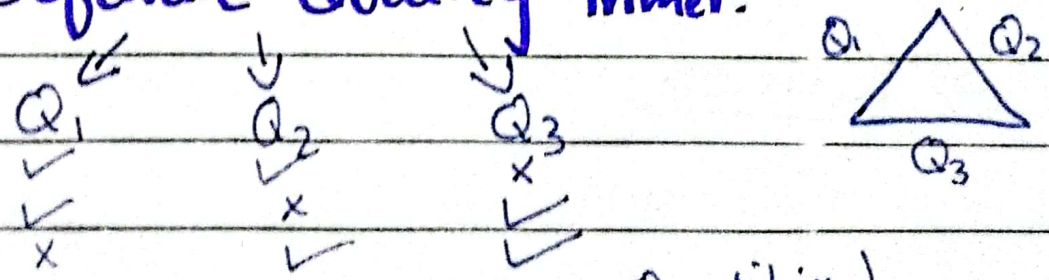
(i) Reliability
(ii) Documentation/Maintainability
(iii) Usability
(iv) Correctness
(v) Availability

⊛ Prioritizing the Parameters is need for generalization of projects. (Budget and cost constraints)

⊛ Highest Priority before launch:

(i) Correctness
(ii) Reliability
(iii) Performance

⊛ Software Quality Trimer:



(Trade-off between Qualities)

→ Operational (Well work in use) → Transition (Can be modified) (Major change)

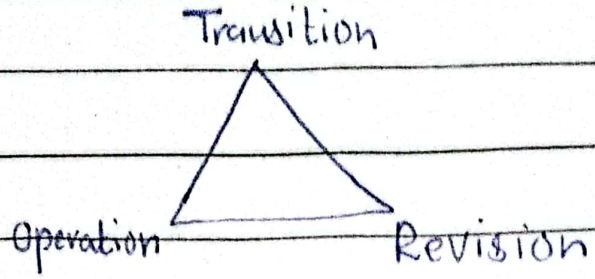
→ Revision (Change focused) (Minor change)

DATE: DAY:

18/09/25

€ A software can't be 100%. reliable, super fast, cheap to maintain

⊙ Software Quality Triangle:



→ Operation:

- ⊙ Used focus, how well it is defined.
- ⊙ Integrity - security wise.

→ Revision:

- ⊙ Change focused
- ⊙ Maintainability, Flexibility, Testability.

→ Transition:

- ⊙ Adaptation focused
- ⊙ Portability, Reusability, Interoperability (Components / Different platforms compatible)

DATE: DAY:

23/09/25

→ QC vs QA vs QE

→ △

→ Audit (formal) vs walkthrough

⊙ TQM

Total Quality (Continuous Improvement) Management

:Change is a bit difficult

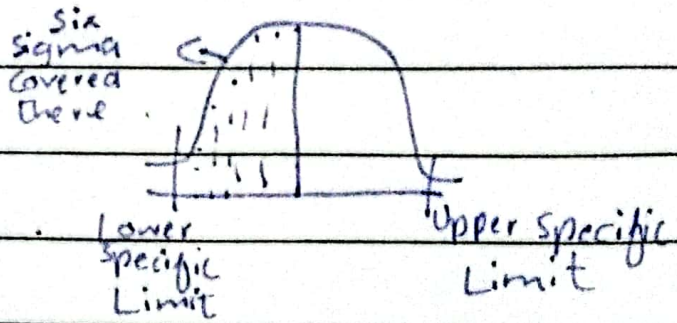
⊛ Costs:

→ Training

25/09/25

⊛ Six Sigma:

→ Sigma – "Standard Deviation"
(Derivation from mean)



→ Six Sigma Level:

1δ, 2δ, 3δ, 4δ, 5δ, 6δ

(Values will lies within range or limit)

→ SD↓ Performance↑ and vice versa.

DATE: DAY:

→ Defects per million opportunity.

: Defects / Total opportunity : for million

$\frac{D}{To} \times$ million

2/10/25

SIX SIGMA

- DPMO
- Two categories

DMAIC (Repairing is better)

DMADV (Discard old solution and make new to reduce efforts)

⊙ DMA ≈ Analyze (Analysis techniques)

Define (Goals)

(Measure) (Criteria + Quality attributes)

→ DMAIC → Improve → control (surveys)

→ DMADV → Design → Verify & Validate

⊙ D Phase:

→ Wait Time ↑

→ Call duplicate ↑ (drop rate)

DATE: DAY:

⊙ M Phase :

→ Timings drop
→ Experience.

⊙ A Phase :

→ Not experience and call drops
→ Check calls drop rate.

⊙ I & C : (Improve) (control)

Phase

→ Hiring
→ Training
→ Continuous cycle analysis and improvement
→ Load controlled and Self Support.

⊙ CMMI : → Integration

Capability Maturity Model

→ 5 levels exist to achieve highest quality

↓ ① Adhoc (without any plan or problem)
② Managed (management practices)
③ Defined (Processes not working properly)
④ Quantity Measure (Process management) (Qualitative aspects)
↑ ⑤ Prevention Before

: Key Process/ Performance Areas (KPAs)

DATE: DAY:

⊙ **Six Sigma Belts:**

Yellow Green Black Master Belt

→ Experience and skills set according.

7/10/25

⊛ **Measure:** Number + Descriptor

⊛ **Metrics:** Two related measures/ Combination of two measures.

→ **Category:**

- Direct Measure (which thing is measured directly)
- Indirect Measures (Perform calculations on Direct to achieve some meaningful thing)

→ **Common Mistakes:**

⊙ What or how to measure

→ **Kind of metrics:**

- Product
- Project

⊛ Code Quality not depends on endpoint or functionality but also on flexibility and maintainability. (LOC Measures in Software Development).

: Python (Numbers compare, addition, subtraction)

DATE: DAY:

→ Best/Better Size Measures:

Complexity measures
→ Software Science
→ Cyclomatic complexity.

ⓔ Software Science Metrics:

→ Halsteads see program as operator (symbol) and operand (data types & variables).

⊙ n1 (Number of unique operators)
⊙ n2 (Number of unique operands)
⊙ N1 (Total number of operator uses)
⊙ N2 (Total number of operand uses)

e.g: `int x = n + 1`
operand operand operand operand

: Length of program: $N = N1 + N2$

Vocabulary: $n1 + n2$

→ More Tools:

⊙ Radon (Python tool)
→ McCabe's complexity
→ Raw metrics
→ Halstead metrics
→ Maintainability

DATE: DAY:

9/10/25

(*) Software Science Metrics:

→ Not only physical code is focused but also deep things should be considered.

Languages dependant → (1) Operand
(2) Operators

→ N1, N2, n1, n2

→ e.g:

```
int x = x+1;
```

(operands) (operators)

N1 / 4 , n1 / 3 , N2 / 3 , n2 / 3

** (Matrix multiplication) → Testwell CTC++ (→ Framework), Python, Radon (Tools for Metrics measure)

→ In General Manual and tools, both values can vary from each other (In Python Language)

: +=, ++=

: for i in range(10)

: tokenize

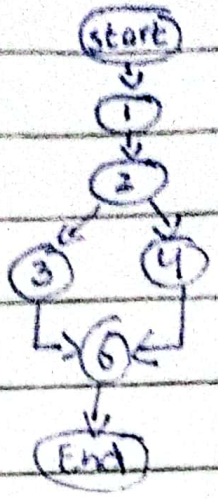
→ Waiting & Development Cycle are also the factors remaining in the code till now.

DATE: DAY:

⊙ Cyclomatic Complexity:

→ CFG (control Flow Graph)

→ e.g:




```
int x, y
if (x > 1)
    print(.... )
else
    print(.... )
```

: $C.C = E - N + 2(P)$

(Connected components)

E → function

P → (Number of functions)

→ class A:

f1()

class B:

f2()

f3()

: $CC = (D) + 1$

D → Decision Points

: 1-10 (Normal Level testing)

: 10-20 (Moderate level testing)

: 20-50 (Recommended level testing)

: >50 (Must testing)

⊙ Function Point Analysis:

→ Analysis is done before and programming level dependency is waived off.

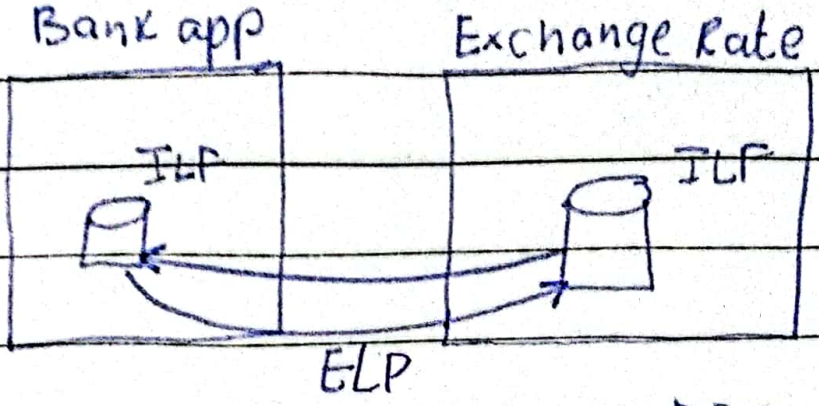
DATE: _____ DAY: _____

- Fonctional Point
  - EI (External Inputs)
  - EO (Derived Data)
  - EQ (External Queries) (No Change in Data)

Retrieval level

→ File Type:

→ ILF (Internal Logic File)
→ ELF (External Logic File)



→ RET (Record Element Type) and DET (Data Element Type)

: Instructor:

→ Name, experience, salary, cnic

: COURSES:

Course ID
Course Name
Credit-Hours

: RET: 2
DET: 7

DATE: DAY:

=> Low (L), Medium (M), High (H)

DETS

RET	1-10		
1	◎	H	L
2		M	

	Count				
E_1	C	0			= 3 x 5
E_0					
⋮					

: Medium: 4
High: 5

Σ □

FC (Function Counts)

: But we need function point.

$$FP = 0.65\,FC + \sum_{i=0}^{N} W_i$$ (weights)

: Pros and Cons of FP

		W 0.5	
1	:	3	(Maintability)
2	1	0	(Flexibility)
3	:	0	
4			
5	⋮		
n			

(ΣW)

: Then we can find FP (function point) which is indirect measure.