Software Quality

Course 1

Introduction

Course info

- Course + lab (interactive sessions)
- TO DO: (Teams)
  - 2 labs
  - Project:
    - Intermediate delivery
    - Final delivery & presentation
  - Final examination: oral exam
  - Teams (communication, other resources) n5ysuuy

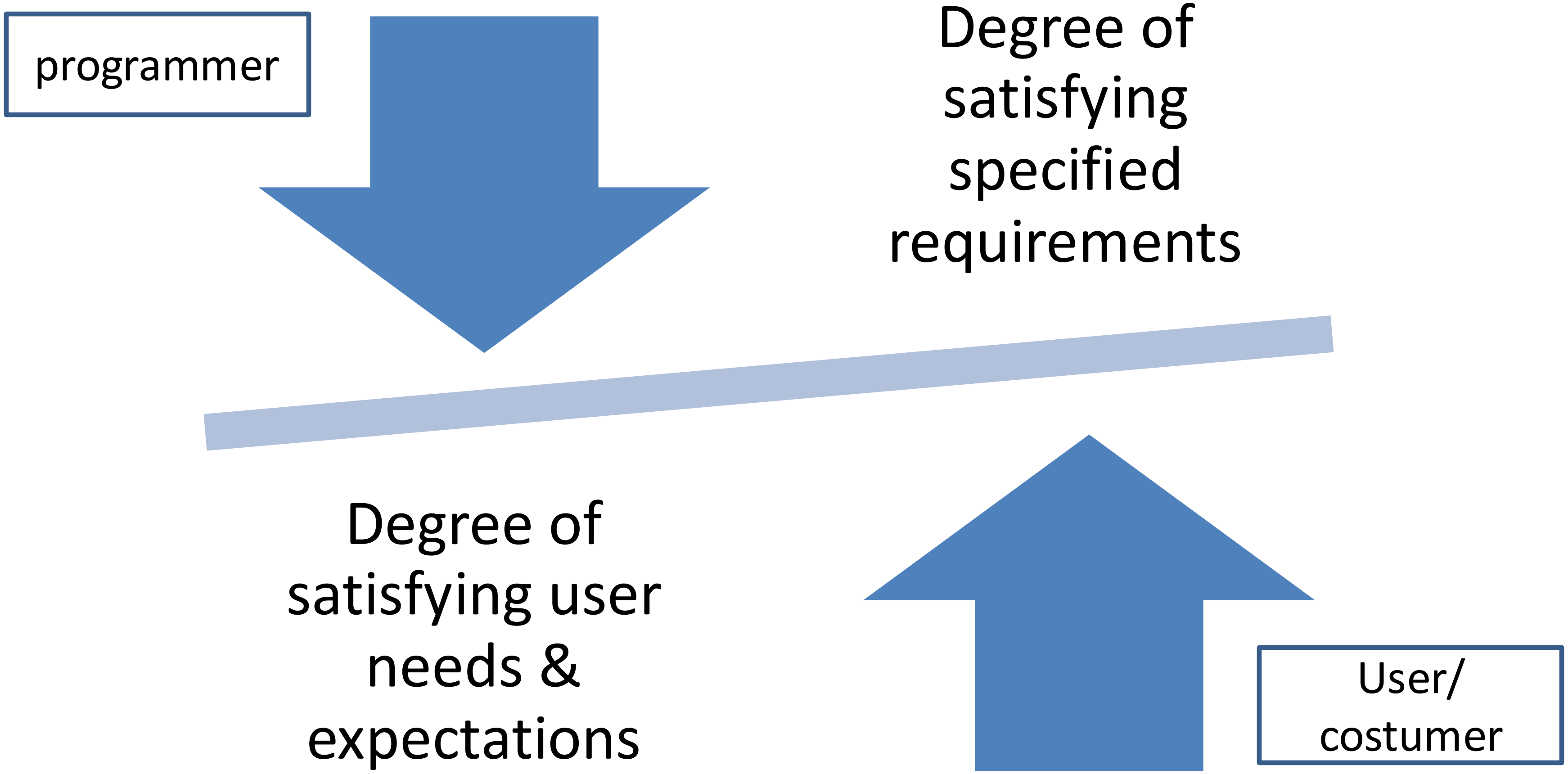
Evaluation:

- 30% exam +
- 10% participate at workshop/webinar:
  - SWOT analysis
  - Risk analysis
- 60% lab activity:
  - 10% lab 1 + 10% lab 2 + 40% project
- Bonus: lab presentation: GenAI in Code Quality – to be discussed

References

- D. Galin – Software quality assurance – From theory to implementation, Addison Wesley, 2003
- S.H. Kan –Metrics and models in Software Quality Engineering. Addison Wesley, 2nd ed., 2003
- R.A. Khan, K. Mustafe, S.I. Ahson – Software Quality: Concepts and Practice, Alpha Science, 2006
- G. Schulmeyer - Handbook of Software Quality Assurance , Artech House, 2007
- D. Spinellis. *Code Quality: The Open Source Perspective*. Addison Wesley, 2006
- S. McConnell – Code Complete, 2nd Edition, Microsoft Press, 2004
- C. Laporte, A. April – Software Quality Assurance, Wiley - IEEE Press, 2017
- A. Gillies – Software Quality: Theory and Management (3rd ed.), Bearswood Press, 2011

What is Software Quality?



programmer

Degree of satisfying specified requirements

Degree of satisfying user needs & expectations

User/ costumer

Characteristics/Factors of SQ

External

user

Internal

developer

Characteristics of SQ

user

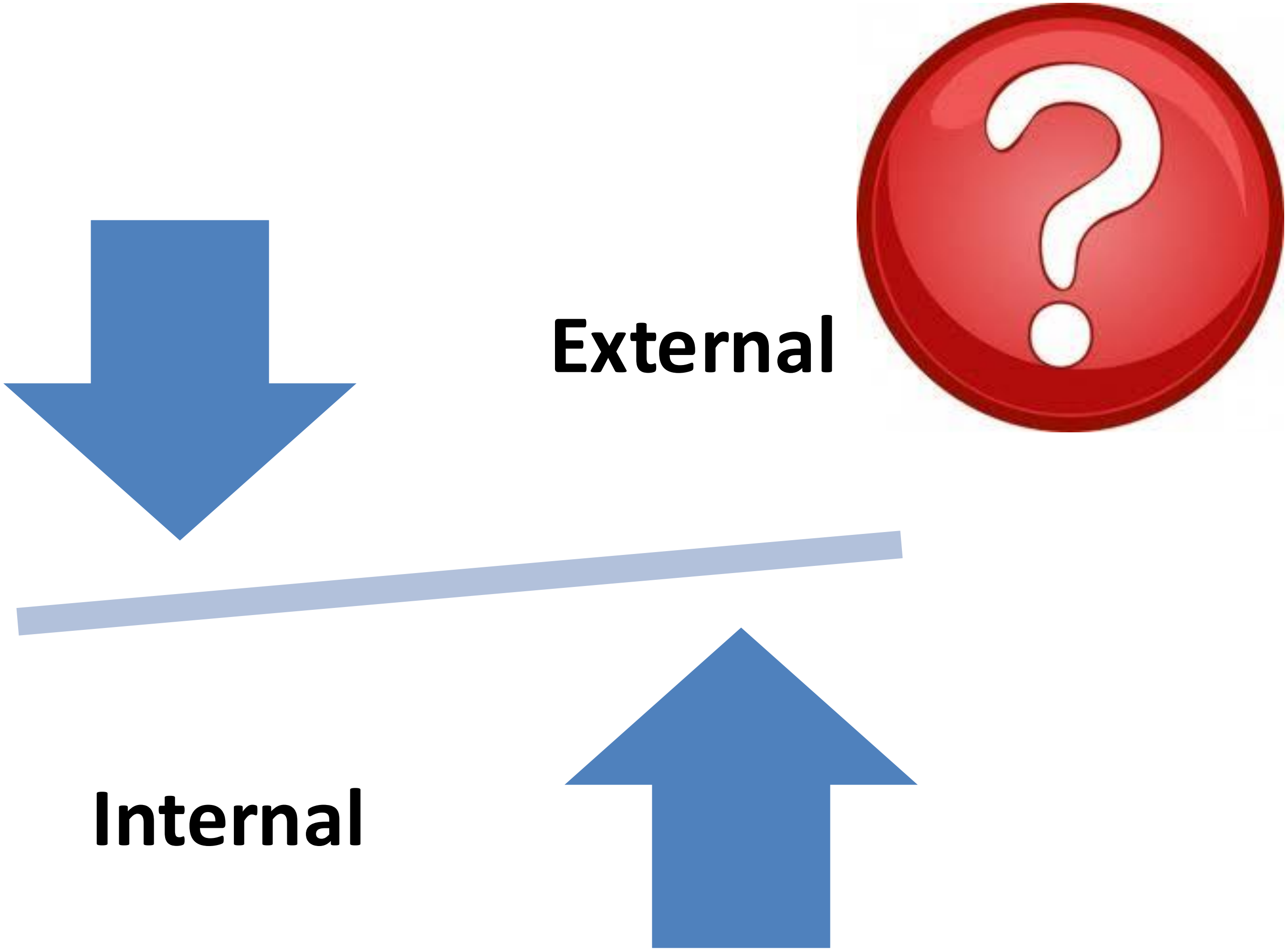
External

- Correctness – system is free of faults in specif, design, implementation
- Usability – easy to use
- Efficiency – minimal use of resources
- Reliability – perform required tasks in stated conditions
- Integrity – prevent unauthorized or improper access
- Adaptability – easy to be used in other environments
- Accuracy – how well does the job
- Robustness – continue to function in invalid conditions

developer

Internal

- Maintainability – easy to modify / extend
- Flexibility – easy to modify for other purposes
- Portability – easy to use on different platforms, environments
- Reusability – easy to be reused in other systems
- Readability – easy to read and understand
- Testability – easy to verify if it meets the requirements
- Understandability – coherence of a system



External

Internal

Dependency between external characteristics

(S. McConnell – Code Complete)

	Correct.	Usab.	Effic.	Relia.	Integ.	Adapt.	Acc.	Robust.
Correct.	↑		↑	↑			↑	↓
Usability		↑				↑	↑	
Efficiency	↓		↑	↓	↓	↓	↓	
Reliability	↑			↑	↑		↑	↓
Integrity			↓	↑	↑			
Adaptability					↓	↑		↑
Accuracy	↑		↓	↑		↓	↑	↓
Robustness	↓	↑	↓	↓	↓	↑	↓	↑

- Why study software quality?
- Why apply a software quality assurance plan?

✓ Best answer – manager

? Have you perform SQA?

Facts - statistics

- Fact: *improving quality reduces development costs*

- Stats:
  - Biggest (all resources) activity in software development is debugging and correcting code

Stats

- NASA
  - 50 projects/ 400 work-years/ 3 mil lines of code
  - *“ increased quality assurance decreased error rate, didn’t increase overall development costs”*
- IBM
  - *“less defects – shortest development projects”*
  - *“removing errors – most expensive and time-consuming activity”*

Stats

- Experiment (1985)
  - 166 programmers – same specification => programs ≈ 220 lines + ≈5 hours
  - *“less errors – average time”*

Test

- Given the 3 integers, representing the lengths of a triangle sides determine if the triangle is:
  - Any
  - Isosceles
  - Equilateral
  - Rectangular (dreptunghic)

  OR "Is NOT a triangle"

- Write test cases

Evaluate your test

G. Myers – The Art of Software Testing

1. Do you have a test case that represents a *valid* scalene triangle? *(No guarantee for 1, 2, 3 and 2, 5, 10)*
2. Do you have a test case that represents a valid equilateral triangle?
3. Do you have a test case that represents a valid isosceles triangle? (*(No guarantee for* 2, 2, 4)
4. Do you have at least three test cases that represent valid isosceles triangles such that you have tried all three permutations of two equal sides? (*ex. 3, 3, 4; 3, 4, 3; and 4, 3, 3*)
5. Do you have a test case in which one side has a zero value?
6. Do you have a test case in which one side has a negative value?
7. Do you have a test case with three integers greater than zero such that the sum of two of the numbers is equal to the third?

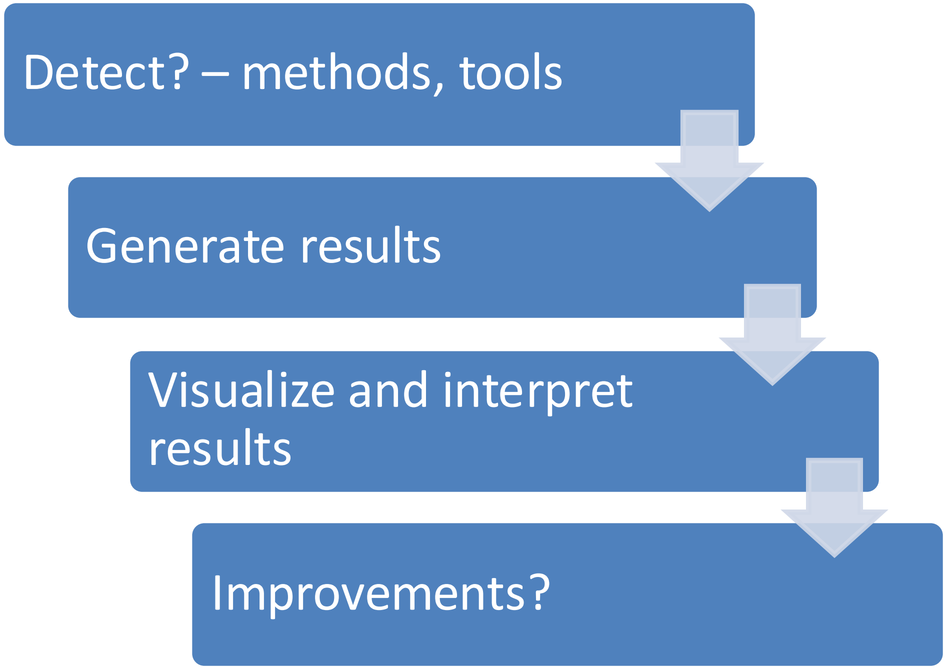
8. Do you have at least three test cases in category 7 such that you have tried all three permutations where the length of one side is equal to the sum of the lengths of the other two sides? (*ex. 1, 2, 3; 1, 3, 2; and 3, 1, 2*)
9. Do you have a test case with three integers greater than zero such that the sum of two of the numbers is less than the third? (*ex. 1, 2, 4 or 12,15,30*)
10. Do you have at least three test cases in category 9 such that you have tried all three permutations? (*ex. 1, 2, 4; 1, 4, 2; and 4, 1, 2*)
11. Do you have a test case in which all sides are zero (0, 0, 0)?
12. Do you have at least one test case specifying noninteger values? (*ex. 2.5, 3.5, 5.5*)
13. Do you have at least one test case specifying the wrong number of values (two rather than three integers, for example)?
14. For each test case did you specify the expected output from the program in addition to the input values?

Highly professional ≈ 7.8 / 14

Compute your score

What we will talk about?

- SQ models
- SQ factors
- QA
- Risks, SWOT



Detect? – methods, tools

Generate results

Visualize and interpret results

Improvements?