

Other Topics of Software Engineering

Lecture #19
Software Engineering and
Project Management

Instructed by Steven Choy on Mar 19, 2007



1

Software Development Professional

Certification
Ethics and Practice



2

Software Development Professional – Certification



- Different from other mature fields of engineering, such as civil engineering and electrical engineering, the field of software engineering lacks established ways to help developing professional's credentials.
- What does it mean to be a professionally qualified software engineer?
- How do you get there?
- In recent years, a number of software engineering certification programs come to address these issues.

3

Software Development Professional – Certification...



- IEEE Computer Society's Certified Software Development Professional (CSDP) Program comes provides a solid path to software engineering certification.



<http://www.computer.org/certification>

4

Software Development Professional – Certification...



- One requirement for CSDP certification is that applicants have to demonstrate mastery of a body of knowledge by passing the **CSDP examination**.
- The 3.5-hours examination consists of 180 multiple-choice questions drawn from a number of knowledge areas in software engineering.

<http://www.computer.org/certification>

5

Software Development Professional – Certification...



- Topics
 - I. Business Practices and Engineering Economics
 - II. Software Requirements
 - III. Software Design
 - IV. Software Construction
 - V. Software Testing
 - VI. Software Maintenance
 - VII. Software Configuration Management
 - VIII. Software Engineering Management
 - IX. Software Engineering Process
 - X. Software Engineering Tools and Methods
 - XI. Software Quality

6

Software Development Professional – Certification...



- CSDP Sample Questions
 2. **Which of the following is not required of a software component?**
 - [a] A unit of independent deployment
 - [b] Exploits commonalities in large software systems
 - [c] A unit of third-party composition
 - [d] Exposes source code for modification

<http://www.computer.org/certification>

7

Software Development Professional – Certification...



- CSDP Sample Questions
 7. **During a software development project two similar requirements defects were detected. One was detected in the requirements phase, and the other during the implementation phase. Which of the following statements is mostly likely to be true?**
 - [a] The most expensive defect to correct is the one detected during the requirements phase.
 - [b] The most expensive defect to correct is the one detected during the implementation phase.
 - [c] The cost of fixing either defect will usually be similar.
 - [d] There is no relationship between the phase in which a defect is discovered and its repair cost.

<http://www.computer.org/certification>

8

Software Development Professional – Certification...



- CSDP Sample Questions
 - 15. Paul has drafted a software project management plan. Which of the following items should be discussed in this plan?
 - I. Schedule
 - II. Budget
 - III. Requirements
 - IV. Staffing
- [a] I, III, IV only
[b] I, II, III only
[c] I, II, IV only
[d] I, II, III, IV

<http://www.computer.org/certification>

9

Software Development Professional – Ethics



- ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices
- <http://www.acm.org/serving/se/code.htm>

10

Software Engineering Ethics and Professional Practices



1. **PUBLIC** - Software engineers shall act consistently with the public interest.
2. **CLIENT AND EMPLOYER** - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.
3. **PRODUCT** - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. **JUDGMENT** - Software engineers shall maintain integrity and independence in their professional judgment.
5. **MANAGEMENT** - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. **PROFESSION** - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. **COLLEAGUES** - Software engineers shall be fair to and supportive of their colleagues.
8. **SELF** - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

11

Code Complexity Measurement



12

Code Complexity Measurement



- How do you measure the code complexity of a software?
 - LOC (line of code)
 - Halstead's software science
 - Cyclomatic number
 - Function points

13

Function Points



- Developed in 1979
- Function Points measure a system from its functional perspective independent of technology (tool/programming language)
- A better alternative to measure code size or complexity than LOC

14

Function Points Analysis



- For measurement, function points analysis identifies five user function types:
 - **External input (EI)** – data enter the system to access/update an internal logical file (e.g. data from input screen or other applications)
 - **External output (EO)** – data leave the system for the outside world
 - **External inquiry (EQ)** – data enter and leave the system without modifying an internal logical file
 - **Internal logical file (ILF)** – a group of record element types (RET) that are often accessed at the same time
 - **External logical file (ELF)** – a group of record element types (RET) that are often accessed together by this application

15

Complexity Multiplier



- Each component is assigned with a ranking (low, average or high)

	Complexity Multiplier		
	Low	Average	High
EI	3	4	6
EO	4	5	7
EQ	3	4	6
ILF	7	10	15
ELF	5	7	10

16

Calculating Function Points



- Suppose the system has 21 EIs in total:
 - 7 at low complexity
 - 9 at average complexity
 - 5 at high complexity
- What's the function points contributed by the above EI?

17

Calculating Function Points



- By looking up the complexity multiplier table, the multiplier of EIs are:
 - low complexity → 3
 - average complexity → 4
 - High complexity → 6
- So, function points for EI is:
 - $(7 \times 3) + (9 \times 4) + (5 \times 6) = 87$
- Function points for other user function types can be calculated using the above method

18

User Function Complexity



- How do you determine the complexity of a user function types?
- An International Function Point Users' Group (IFPUG) proposed some rules to determine the complexity
- The following table shows how EI's complexity is determined:

Number of file element types	Number of data element types		
	1-4	5-15	16 or more
0-1	Low	Low	Average
2	Low	Average	high
3 or more	Average	High	High

- Definition:
 - A "Data element type" is a field
 - A "record element type" is a group of related fields
 - A "file element type" is a group of record element types that is often accessed together

19

Adjusting Function Points



- What we have calculated before is just known as an "**unadjusted function point**"
- The unadjusted function points do not capture all aspects of complexity such as performance, data communication, reusability, etc.
- To calculate the "**adjusted**" FP, 14 additional factors are taken into account
- Each factor is assigned with a value of 0 to 5 based on their influence

20

Calculating the “Adjusted FP”



- To calculate the “Adjusted FP”, we first sum up all the influenced factors to get the **degree of influence (DI)** (i.e. 31)
- We then determine the technical complexity adjustment (TCA):
 - $TCA = 0.65 + (0.01 \times DI)$
 - i.e. $TCA = 0.65 + (0.01 \times 31) = 0.96$
- Finally, the adjusted FP:
 - Adjusted FP = unadjusted FP x TCA

Factor	Rating
Data Communication	0
Distributed Functions	0
Performance	2
Heavy Hardware Usage	2
Transaction Rate	1
Online data entry	3
End-user Efficiency	4
Online Update	4
Complex Processing	2
Reusability	3
Ease of installation	3
Ease of operation	1
Multiple Sites	3
Ease of modification	3

21

Factor	Rating
Data communication	0
Distributed functions	0
Performance	2
Heavy hardware usage	2
Transaction rate	1
Online data entry	3
End-user efficiency	4
Online update	4
Complex processing	2
Reusability	3
Ease of installation	3
Ease of operation	1
Multiple sites	3
Ease of modifications	3

Eg. Adding up the values, we have a DI value of 31.

$$TCA = 0.65 + (0.01 \times 31) = 0.96$$

If the unadjusted FP is 700, the adjusted FP will be $700 \times 0.96 = 672$

22

Object-Oriented Measurement



- Rule of thumb: Low coupling, High cohesion
- How do you measure the cohesion of methods?
 - Lack of cohesion of methods (LCOM)

23

How LCOM measures cohesion?



- The LCOM of a class is defined as:

$$LCOM = \text{no. of dissimilar pairs} - \text{no. of similar pairs}$$
- A low value of LCOM → a highly cohesive class

24

Similar & Dissimilar Pairs



- If two methods access the same data member, they are said to be **similar**
- If two methods do not access the same data member, they are said to be **dissimilar**

```
public class DemoClass {
    public int data_1;
    public int data_2;

    public void method1() {
        data_1 = 6;
        ...
    }

    public void method2() {
        if (data_1 == 10) {
            ...
        }
        ...
    }

    public void method3() {
        data_2 = 10;
        ...
    }
}
```

similar

dissimilar

Example: Calculating LCOM



- Given the following access information of a class, where m1,m2,m3 are methods and d1,d2,d3,d4,d5 are data members, calculate the LCOM value.

	d1	d2	d3	d4	d5
m1	X		X		
m2				X	
m3		X		X	X

Example: Calculating LCOM



- Similar pairs: (m2,m3)
- Dissimilar pairs: (m1,m3), (m1,m2)
- Applying the LCOM formula:

$$\text{LCOM} = \text{no. of dissimilar pairs} - \text{no. of similar pairs}$$

$$= 2 - 1$$

$$= 1$$
- NOTE: If you have a negative LCOM value, assign it to zero

Example: Calculating LCOM



	v1	v2	v3	v4	v5	v6
m1	X		X			
m2	X			X		
m3		X		X	X	
m4			X		X	X

4 similar pairs: (m1,m2), (m2,m3), (m3, m4), (m1,m4)
 2 dissimilar pairs: (m1,m3), (m2,m4)
 LCOM = 2 - 4 = -2, which is 0

Cyclomatic number



- Measure the complexity of a program module by the number of branches
- Simplest program: no branch at all. Lowest possible cyclomatic number of 1.
- Cyclomatic number of a program is defined as the number of branches plus one.
- Consider the following:
 - A program with no conditional statement
 - A program with "if-then-else"
 - A program with nested conditional statements

29

Halstead's software science



- Based on 4 fundamental variables:
- n_1 : no. of distinct operators used in a program. It includes all the arithmetic and logical operators and method names
- n_2 : no. of distinct operands in a program. It includes variables and constants
- N_1 : no. of times operators are used
- N_2 : no. of times operands are used
- A program's *vocabulary* (n) is defined as $n = n_1 + n_2$
- A program's *length* (N) is defined as $N = N_1 + N_2$
- A program's *volume* (V) is defined as $V = N \log_2 n$

Many readers were not convinced of the accuracy and validity of software science

30

More...



- More topics can be found on the course website.

31