Prof. Dr. A. Podelski, Dr. B. Westphal
S. Feo Arenis

Sommersemester 2015

Softwaretechnik/Software Engineering

http://swt.informatik.uni-freiburg.de/teaching/SS2015/swtvl

Exercise Sheet 2

Early submission: Wednesday, 2015-05-06, 12:00 Regular submission: Thursday, 2015-05-07, 12:00

Exercise 1 – (Processes) (11/20 Points)

In this exercise, you are requested to model the development plan of the *Softwarepraktikum*. If you have not participated and are not currently participating in it, please consider the *Softwarepraktikum* homepage[1] or interview your classmates that are currently or have been enrolled.

Hint: The plan on the Softwarepraktikum *website is a* customer plan*, i.e., the time planning from the viewpoint of the receiver of the developed software. For this task, please refer to* ***an internal development plan****, i.e. the viewpoint of the software developer.*

1. Process model (6/11 Points)

(i) Give a list of the **artifacts** of the project. Artifacts are both the intermediate and the final results of the project. (1)

(ii) Describe the **activities** to be performed. (1)

(iii) Describe the **roles** necessary for the completion of the project. (1)

(iv) Describe the **relations** between artifacts, activities and roles. What activities are necessary to produce or process what artifacts? What activities are performed by which roles? (2)

(v) Assume the following situation: In the first Scrum developer meeting, there is an agreement, that coding conventions should be used but there are conflicting views. An agreement on which coding conventions to use cannot be reached and the project cannot continue. More precisely, out of 7 group members, 3 insist that identifiers in code should use "CamelCase"[2], other 3 members insist that identifiers should use all lowercase letters and prefix the identifier with the type and underscore (like in "`m_count`"). The remaining group member abstains from giving an opinion, she doesn't care *which* coding conventions are established as long as *some* are.

How do you plan to resolve conflicts such as the one arising in this situation? How would the decision about which convention to adopt be made? (1)

2. Cost Estimation (5/11 Points)

(i) Estimate the effort in person-months (PM) of the *Softwarepraktikum* project. Give an estimate range. Discuss how you achieve that result. (1)

Hint: a PM is usually the working time of a full-time position (which may differ from company to company). Please fix a concrete number to use in your estimation. Note that the question is about the PMs actually *spent (including night-shifts, etc.), not only the hour-equivalent of ETCS credit points.*

[1] See `https://sopra.informatik.uni-freiburg.de/`
[2] See `http://en.wikipedia.org/wiki/CamelCase`

Characteristics of the Type				a	b	c	d	Software Project Type
Size	**Innovation**	**Deadlines/ Constraints**	**Dev. Environment**					
Small (<50 KLOC)	Little	Not tight	Stable	2.4	1.05	2.5	0.38	Organic
Medium (<300 KLOC)	Medium	Medium	Medium	3.0	1.12	2.5	0.35	Semi-detached
Large	Greater	Tight	Complex HW/ Interfaces	3.6	1.20	2.5	0.32	Embedded

Table 1: Classification of software projects and corresponding coefficients for the COCOMO estimation model.

(ii) Classify each of the activities described in 1 (ii) into management or development. What is the proportion of management and development effort of your development plan? (1)

(iii) Use the *basic COCOMO*[3] (constructive cost model) algorithm to compute the software development effort of the *Softwarepraktikum* project. Use Table 1 for the task.

- First, determine the size, the innovation, the deadline criticality and the development environment to determine the type of the project. Justify your choices.
- Determine the expected size of the project in thousands of lines of code (kLOC). How do you arrive at your result?
- Calculate the effort using the basic COCOMO equations:

$$E \text{ (effort required)} = a(kLOC)^b \quad \text{[person-months]}$$
$$TDEV \text{ (time to develop)} = cE^d \quad \text{[months]}$$

Hint: if you want to, you may also use intermediate COCOMO for comparison; yet there are no extra points in it for you. (2)

(iv) How big is the difference between your estimate and the result of the COCOMO calculation? What are possible reasons for the deviation? Which estimate is more plausible to you? (1)

[3] See `http://en.wikipedia.org/wiki/COCOMO`

Exercise 2 – (Metrics) (9/20 Points)

(i) From your programming course (or any other where submitting source code is required), take the largest / most complex piece of software that you have coded yourself and calculate:

- The following lines of code metrics:
  - LOC_{tot} = Total number of lines of code
  - LOC_{ne} = Number of non-empty lines of code
  - LOC_{pars} = Number of lines of code that not only contain comments or non-printable characters.
- For the biggest method/function in your program,
  - the **cyclomatic complexity** (cf. Figure 1),
  - Halstead complexity measures.[4]

Make sure to submit the source code together with your solution. By submitting code, you grant us a license which allows the use of your code for the purposes of the exercise (reading by tutors, showing during the exercise session, etc). If you don't wish to grant this license to us, use some appropriate public domain code. (3)

(ii) The metrics from (i) are often used as derived measure for the complexity or effort required to develop the code being measured.

In particular the family of LOC metrics is notorious for being subvertible, that is, two semantically equivalent programs (that perform the same computation, and thus should have needed roughly similar effort) have substantially different metric values.

a) Convince yourself of this claim for the case of LOC_{pars}: give two semantically equivalent programs with substantially different metric values. (1)

b) Choose either cyclomatic complexity or a Halstead measure and discuss their subvertibility, i.e. are there two semantically equivalent programs with substantially different values under the chosen metric? (2)

(iii) Assume you meet a colleague from the programming course after some years, you are now working as a developer for a company. You have signed a non-disclosure agreement and cannot reveal details on the software you are developing.

Discuss which of the metrics would you consider to be more appropriate to communicate to your colleague the complexity and the size of the project you are working on without revealing specific details. Justify your choices. (2)

(iv) Recall the software metrics for object-oriented programs from Chidamber and Kemerer. In particular LCOM enjoys great popularity, there are people who recommend to use the LCOM value to guide the review and testing effort.

- Explain in your own words what aspects of the measured class are distinguished by LCOM values, i.e. give examples of classes with *good* (low) and *bad* (high) LCOM value.
- Discuss whether LCOM is plausible to guide review and testing effort.
- Discuss whether LCOM can be subverted by developers (i.e., can they introduce artifacts in the code that artificially modify the resulting value of LCOM without changing the code's observable behaviour). (1)

References

Ludewig, J. and Lichter, H. (2013). *Software Engineering.* dpunkt.verlag, 3. edition.

[4] See `http://en.wikipedia.org/wiki/Halstead_complexity_measures`

For program:

```
1  void insertionSort(int[] array) {
2    for (int i = 2; i < array.length; i++) {
3      tmp = array[i];
4      array[0] = tmp;
5      int j = i;
6      while (j > 0 && tmp < array[j-1]) {
7        array[j] = array[j-1];
8        j--;
9      }
10     array[j] = tmp;
11   }
12 }
```

Cyclomatic complexity is defined for graph G corresponding to the program as

$$v(G) = e - n + p$$

Number of edges: $e = 11$
Number of nodes: $n = 6 + 2 + 2 = 10$
External connections = $p = 2$

$v(G) = 11 - 10 + 2 = 3$

Corresponding graph G

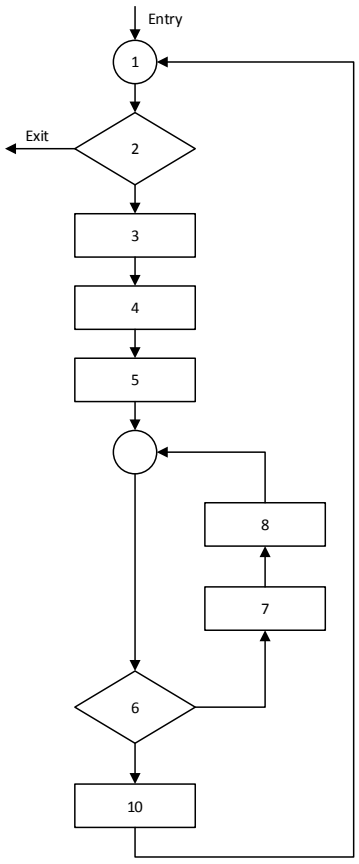

Figure 1: Example of the calculation of cyclomatic complexity