Assignment 1 - Solution

Handout: **Monday, 30 September 2024**
Due: **14:00, Thursday, 10 October 2024**

1. ACM/IEEE SE Code of Ethics and Professional Practice (10 marks)

Read the full version of the Code at
https://ethics.acm.org/code-of-ethics/software-engineering-code/,
pick two clauses, one from Principle 2 CLIENT AND EMPLOYER and the other from Principle 7 COLLEAGUES, and then for each clause 1) describe a situation where software engineers may violate the clause and 2) discuss what could be done to avoid the situation and in consequence also the violation. (< 200 words)

2. Software Processes (8 marks)

Please 1) briefly describe one software project you worked with before, 2) decide whether a plan-driven approach based on the waterfall model or an agile method like eXtreme Programming would be more appropriate for the project, and 3) justify your decision. (< 100 words)

A plan-driven approach based on the waterfall model is more suitable for the development of the following systems:

- *Embedded systems where the software has to interface with hardware systems;*
- *Critical systems where there is a need for extensive safety and security analysis of the software specification and design;*
- *Large software systems that are part of broader engineering systems developed by several partner companies;*
- *Software systems with clear and stable requirements.*

An agile method like XP is more suitable for developing other systems with changing requirements.

A justification is acceptable if it is in line with the above guidelines.

3. Manifesto for Agile Software Development (12 marks)

Agile software development values individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. Can you think of four situations, one for each pair of these items, where focusing too much on the first item while too little on the second item could lead a software development team to trouble? (< 150 words)

Individuals and interactions: E.g., when developers are less competent

Working software: E.g., when a system has a long life and therefore high maintainability is key

Customer collaboration: E.g., when the development team and the customers have major disagreement

Responding to change: E.g., when a tight schedule is the most important constraint on the project

4. eXtreme Programming (12 marks)

Schneider et al. discussed several issues they noticed when teaching eXtreme Programming (XP) at universities [1]. Pick two issues from the paper that intrigue you, and then 1) briefly explain what the issues were about, 2) indicate where in the paper the issues were mentioned, and 3) describe what lesson(s) you learned from the discussion on the issues and how the lesson(s) could be helpful if you are to apply XP to a course project in the future. (< 300 words)

[1] Jean-Guy Schneider and Lorraine Johnston. 2003. EXtreme Programming at universities: an educational perspective. In Proceedings of the 25th International Conference on Software Engineering (ICSE '03). IEEE Computer Society, USA, 594–599.

5. Requirements Specification (8 marks)

Suppose you and your friends plan to implement an open-source communication software system like MS Teams and ZOOM, and you are responsible for developing the requirements document for the new system. Based on your experience with MS Teams and ZOOM, please

1. define one functional system requirement in natural language for the new system's “schedule a meeting” function. (< 50 words)

2. define one non-functional system requirement in natural language for the new system regarding its usability. (< 50 words)

Note: 1) Use “shall” for compulsory requirements and “should” for desirable requirements; 2) Be careful with the differences between functional and non-functional requirements; 3) Make sure requirements are verifiable; 4) Provide an explanation in parentheses if necessary.

Functional system requirement: A user shall be able to specify the meeting's starting and ending time and date.

Non-functional system requirement: A user shall be able to use all the system functions after 5 hours of training. After the training, the average number of errors experienced users make shall not exceed 1 per hour of system use.

How to hand in:

Submit your typed, instead of handwritten, answers in a PDF file on Blackboard.