MTAT.03.094 – Software Engineering

Written Exam – 06 January 2017

Start: 14:15 – End: 16:45

Important Notes:

- The exam is open-book and open-laptop. Web browsing is allowed, but you are neither allowed to use e-mail clients or Instant Messaging clients nor to share any information "live" with anybody inside or outside the exam room.
- This document (question sheet) contains 6 pages (including the cover page). Please check that you have received 6 pages.
- At the end of the exam you must submit both the question sheets and your answer sheets. To avoid that any of your solutions get lost, make sure to write your name (and student ID) on each sheet of paper that you submit.
- Write clearly. Answers that are illegible cannot be counted as correct answers. Only answers written in English will be marked.
- To answer Part 1, use the separately distributed answer sheet. Answers given on the question sheets will not be marked!
- To answer Parts 2 and 3, use the separately distributed blank paper. Answers given on the question sheets will not be marked! Also, please number the pages on your answer sheets.
- At the end of the exam you must return the problem sheet. If you take the question sheets with you (out of the exam room), this will be considered academic fraud (cheating) and treated accordingly.
- Total exam marks: 30 (equivalent to 30% of total course grade). You must get at least 10 marks in this exam to not fail the course. There are 2 bonus marks available.

==========

PART 1: Multiple-Choice Questionnaire (10 marks + 2 bonus marks)

PART 2: Open Questions (10 marks)

PART 3: Constructive Tasks (10 marks)

Total: 30 marks (=100%) plus 2 bonus marks

PART 1: Multiple-Choice Questionnaire (10 marks + 2 bonus marks)

Important: For Part 1, please check boxes on the separate answer sheet. Read carefully before you answer and observe instructions carefully!

The following questions (up to question Q-10) have exactly one correct answer, thus, you must check exactly one answer box on the separate answer sheet. If you think that more than one answer is correct, choose the one answer that seems to be most correct/suitable/relevant.

Question Q-11 at the end of Part 1 is a bonus questions. Please read related instructions carefully.

Q-01 (1 mark): Which of the following statements does not describe a good work practice of a software engineer?
Answer choice:
A: To adhere to the existing body of knowledge of software development techniques/methods
B: To observe constraints, e.g., time and budget constraints, and standards
C: To skip an essential quality assurance activity (e.g., system testing), because a deadline is approaching
D: To develop and use design and quality models

Q-02 (1 mark): Which of the following indicate a waterfall approach to software development is not appropriate?
Answer choice:
A: Business is volatile
B: There are few requirements
C: Business requirements are stable
D: Legal regulations require completing requirements analysis before starting software design

Q-03 (1 mark): What are User Story Points typically not used for?
Answer choice:
A: To estimate project effort
B: To estimate project quality
C: To estimate project duration
D: To estimate team velocity

Q-04 (1 mark): Which of the following is a non-functional requirement of a web-based application?
Answer choice:
A: When the user clicks a "read me" link, the color of the link should change from blue to pink
B: When the user clicks a "read me" link, the next page should be opened within 5 seconds
C: When the user clicks a "read me" link, the mouse over should show the target page title
D: When the user clicks a "read me" link, the read me page should load correctly

Q-05 (1 mark): Which of the following are the key elements of use case diagrams?
Answer choice:
A: People, computer
B: Actors, use cases
C: People, classes, objects
D: Uses, use cases

Q-06 (1 mark): Which of the following statements about code refactoring is correct?
Answer choice:
A: Refactoring does always change the program design
B: Refactoring is done in order to add new functionality
C: Refactoring is a test activity
D: Before refactoring a component, a test suite for this component must be in place

Q-07 (1 mark): Which of the following statements is correct?
Answer choice:
A: Regression testing can efficiently be done manually
B: Equivalence class partitioning is a white-box testing technique
C: Black-box testing techniques exploit knowledge about the code that is tested
D: White-box testing techniques exploit knowledge about the code that is tested

Q-08 (1 mark): You execute a calculation program and see the following message on the screen: 'Program aborted due to division by 0'. Which of the following describes your experience best?
Answer choice:
A: I made an error (while using the program)
B: I debugged an error
C: I triggered a failure
D: I localized a fault

Q-09 (1 mark): Which of the following is not a characteristic of SCRUM?
Answer choice:
A: Incremental development
B: Agility
C: Stable requirements
D: Time-boxing

Q-10 (1 mark): If three developers work 5 days each, how much effort have they spent?
Answer choice:
A: 15 days
B: 15 person-days
C: 5 person-days
D: 3 developer-days

The following question (question Q-11) is a bonus question and can have more than one correct answer. You must check all correct answer choices to get full marks. You get partial marks, if you check some of the correct answer choices. You will get a penalty, if you check an incorrect answer choice. You don't get a penalty, if you miss a correct answer choice. Overall, the lowest possible mark you can get is 0 (i.e., even if everything you check is wrong, you won't get a negative mark).

Q-11 (2 marks - bonus): Which of the following types of meetings are defined in the SCRUM method?
Answer choice:
A: Daily Scrum review
B: Sprint retrospective
C: Scrum master meeting
D: Sprint review

PART 2: Open Questions (10 marks)

Note: Please give your answers on separate answer sheet(s) and state clearly to which question number each answer refers. Answers that have no question number stated will not be marked. Don't forget to write your name on each separate answer sheet.

Q-12 (4 marks): Assume that a list of requirements contains the following two user stories:

(US1) As a hotel receptionist, I want to see the list of rooms not yet booked.

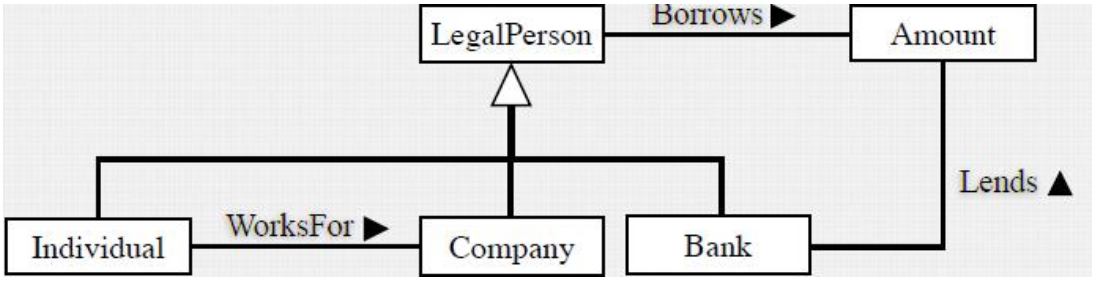
(US2) As a travel planner, I want to enter city names in up to three different languages (e.g., 'Mailand' and 'Milano'), so I can find more information (i.e., information provided in different languages) about a city I am planning to visit.

Answer the following questions Q1 to Q3:

- Q1: Which of the two user stories is of better quality? Justify your answer.
- Q2: What are the three elements of a complete user story? Explain each part briefly
- Q3: What is the purpose of the 'why'-part in a user story? For whom and why is it useful?

Q-13 (3 marks): The figure below shows an initial (incomplete) sketch of a domain model for the following situation:

- An individual can work for one or several companies
- A car is owned by an individual, a bank, or a company
- Banks give loans for buying cars
- A loan can be secured against a car

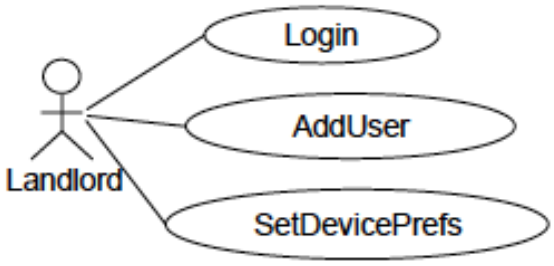


To Do:

- Say which domain class is missing (only using the information given).
- Place the missing domain class in the domain model, i.e., show the association(s) required to connect the missing domain class properly with the existing domain class(es) in the model. Provide name and direction of the added association(s). Explain/justify what you are doing.

Q-14 (3 marks): Do the following:

1. Briefly describe the key elements contained in a use case diagram
2. Describe the steps to follow for identifying use cases
3. Explain what is wrong in the use case diagram shown below. Then say how to fix it.



PART 3: Constructive Tasks (10 marks)

Note: Please provide solution on separate answer sheet(s) and state clearly to which task number each solution relates. Solutions that have no task number stated will not be marked. Don't forget to write your name on each separate answer sheet.

T-01 (2 marks): Assume, you have 20 user stories (US1-US20) in your project backlog. You have estimated the user stories to have a difficulty/complexity expressed in story points as follows:
- Each of US1 to US5 equals 2 story points
- Each of US6 to US10 equals 3 story points
- Each of US11 to US15 equals 7 story points
- Each of US16 to US20 equals 13 story points

To do:
Assume that all user stories have equal value for the end user. If you have a team of 4 developers and weekly sprints (1 week = 5 days = 40 hours), which user stories would you be able to implement in the sprint and achieve the highest possible value without violating your capacity (effort) constraint?
For your calculations, assume that in the past, on average, one developer could implement 1% of the sum of user story points of your current project in one day. Assume that we don't allow overtime. Show all your calculations and explain all your choices. (If you just present the list if User Stories that can be accommodated you will get 0 marks, even if the list is correct.)

T-02 (3 marks): Point out the code smells in the code shown below. Then briefly describe what refactoring steps you would take to refactor the code. Note: You don't need to write down the refactored code, it's sufficient, if you say how you refactor the code to remove the code smells.

```
class BookRental {
   String bookTitle;
   String author;
   Date rentDate;
   Date dueDate;
   double rentalFee;
   boolean isOverdue() {
      Date now=new Date();
      return dueDate.before(now);
   }
   double getTotalFee() {
      return isOverdue() ? 1.2*rentalFee : rentalFee;
   }
}
class MovieRental {
   String movieTitle;
   int classification;
   Date rentDate;
   Date dueDate;
   double rentalFee;
   boolean isOverdue() {
      Date now=new Date();
      return dueDate.before(now);
   }
   double getTotalFee() {
      return isOverdue() ? Math.max(1.3*rentalFee, rentalFee+20) : rentalFee;
   }
}
```

T-03 (5 marks): The code snippet below shows a function that calculates fines for speeding when driving a car. Fines are calculated based on age of the driver (age), the excess speed (overspeed), and the number of penalty points (penaltypts) already accumulated in the driver's police record.

```
0  public static int speedingfine (int age, int overspeed, int penaltypts) {
1    int fine = 0;
2    if ((age >= 25) && (overspeed < 30) && (penaltypts < 3)) {
3       fine = fine + 100 * overspeed;
4       return fine; }
5    if ((age < 25) || (penaltypts >= 3))
6       fine = fine + (200 * overspeed);
7    if (overspeed >= 30)
8       fine = fine + 5000;
9    return fine; }
```

To do:

a) Provide a minimum set of test cases that achieves 100% statement coverage. Remember that complete test cases include both input values and expected output values. You must also say which lines are covered by each test case. (Note: You will get a penalty, if the number of test cases is not the minimum)

b) Provide a minimum set of test cases that achieves 100% decision coverage. Remember that complete test cases include both input values and expected output values. You must also say which decisions are covered by each test case in which direction (i.e., 'true' or 'false'). (Note: You will get a penalty, if the number of test cases is not the minimum)

You can assume that all inputs have been checked for syntactic/semantic correctness, i.e., all parameter values are of the correct type and in a reasonable range (e.g., age is in [18, 99]).

Hint:

Use the following table format to present your test cases of part a):

Test case number	Age (in)	Overspeed (in)	Penaltypts (in)	Fine (out)	Line covered
1					
...					
...					

Use the following table format to present your test cases of part b):

Test case number	Age (in)	Overspeed (in)	Penaltypts (in)	Fine (out)	Decision covered (direction)
1					
...					
...					