MTAT.03.094 – Software Engineering (Tarkvaratehnika)

Final Exam – 6 January 2009

EXAM STRUCTURE

– 20 multiple-choice questions in **Section A** worth one point each.

– 5 short-answer questions in **Section B** worth 20 points in total.

– 3 problem-oriented questions in **Section C** worth 20 points in total.

SECTION A

Please answer these questions in the "Multiple Choice Answer Sheet" to be found at the end of the exam.

QUESTION 1
Which of the following is not a software development method?

(a) Waterfall

(b) RUP

(c) *CMM

(d) XP

(e) Crystal

QUESTION 2
In the software development V-model, the output of the requirements analysis phase is used to design which of the following types of test?

(a) Unit tests

(b) Integration tests

(c) System tests

(d) *User acceptance tests

(e) Maintenance tests

QUESTION 3
Consider a Skype "add-on" that allows a Skype user to sign up for notifications about events related to his/her Skype friends ("buddies"). The add-on notifies users of *events*, like for example: a buddy has just signed in or signed out, or a buddy's mood

message has changed. Which of the following is an example of a non-functional requirement for this add-on?

(a) It must be possible to add new types of events and to remove existing ones

(b) Users must be able to see and signup to any of the available types of events

(c) Every event is visible for 1 month after the user has seen this particular event

(d) Users must be able to filter events by time period (last day, week, month), type and buddy

(e) *The add-on must be free of charge and available for download by any Skype user

QUESTION 4

Consider an application that manages an address book containing contact details of companies and individuals? This address book can be exported in several formats, including: (i) HTML so that it can be viewed in a browser; (ii) Windows Contact Schema – an XML format for representing contact details – so that it can be uploaded into other applications; and (iii) as a Comma-Separated Values files that can be uploaded into a spreadsheet application. Which of the following design patterns would you employ to design the export function of this address book application?

(a) Façade

(b) Singleton

(c) Command

(d) *Strategy

(e) Composite

QUESTION 5

Consider an application that manages an address book containing contact details of companies and individuals. This address book can be exported in Windows Contact Schema – an XML format for representing contact details. In order to export an object of class AddressBook into XML, you will need to generate XML code for each of the objects composing the AddressBook (which are objects of a class called "Contact"). And in order to generate XML code for an object of class Contact, you need to generate XML code for each of the objects that compose an object of class Contact. You foresee that in the future, the class Contact will be extended to include additional details. Which of the following patterns would you consider using to implement this XML export function?

(a) Proxy

(b) Singleton

(c) *Visitor

(d) Strategy

(e) Composite

QUESTION 6
Which of the following notations would be most appropriate to describe the process of handling an insurance claim in an insurance company?

(a) *UML activity diagram

(b) UML object diagram

(c) UML class diagram

(d) UML sequence diagram

(e) UML use-case diagram

QUESTION 7
Which of the following notations would be most appropriate to describe the messages exchanged between an Automatic Teller Machine (ATM) and a bank's internal IT systems when a customer uses the ATM to withdraw cash?

(a) UML activity diagram

(b) UML object diagram

(c) UML class diagram

(d) *UML sequence diagram

(e) UML use-case diagram

QUESTION 8
Which of the following notations would be most appropriate to describe the actions that a user can perform with an ATM?

(a) UML activity diagram

(b) UML object diagram

(c) UML class diagram

(d) UML sequence diagram

(e) *UML use-case diagram

QUESTION 9
Which of the following testing methods can be applied only if you have access to the source code?

(a) Boundary value analysis

(b) Equivalence partitioning

(c) Random input

(d) *Path coverage

(e) Acceptance testing

QUESTION 10

Which of the following types of test is most appropriate for validation?

(a) Unit testing

(b) Boundary value analysis

(c) System testing

(d) White-box testing

(e) *Beta testing

QUESTION 11

Which of the following testing methods guarantees that every possible bug is found?

(a) Equivalence partitioning

(b) Boundary value analysis

(c) Path coverage

(d) Decision coverage

(e) *None of the above

QUESTION 12

A traceability matrix allows one to find out which test cases are associated to a given:

(a) Class in a class diagram

(b) Method in a class

(c) Activity in an activity diagram

(d) Test protocol

(e) *None of the above

QUESTION 13

Which of the following best describes the difference between RPC and messaging?

(a) RPC requires more lines of code than messaging

(b) *RPC is a synchronous communication technology while messaging is asynchronous

(c) System failures are more likely in messaging than in RPC

(d) RPC is always faster than messaging

(e) Messaging systems only allows point-to-point communication while RPC systems supports point-to-point and broadcast communication

QUESTION 14

Which of the following is an example of an XP practice?

(a) Iterations of around 2 months

(b) Documenting user requirements in terms of functions

(c) *Open workspace for developers, in close proximity to the customer

(d) 2-hours meetings at the end of each working day

(e) Each programmer owns an identifiable part of the code

QUESTION 15

In XP, a burn-down chart is used to track:

(a) The velocity of a programmer, measured for example in terms of new lines of code written by that programmer each day

(b) The number of story points that have been added to the project each day

(c) The number of story points that have been tested each day

(d) *The number of story points (or the amount of work) left to do at different points in time until the deadline

(e) The number of man-hours used each day

QUESTION 16

Consider the following Java method. What is its McCabe's cyclomatic complexity?

```
static int func(int x) {
  int y;
  if (x==0)
     y = 3;
  else if (x==1)
    y = 4;
  else if (x==2)
    y = 5;
  else
    y = 0;
  return y;
}
```

(a) 1

(b) 2

(c) 3

(d) 6

(e) *None of the above.

QUESTION 17

Imagine that you have a Java method written by one of your colleagues, with a cyclomatic complexity of 7. The method calculates a shortest-path between two points in a graph. What would you do?

(a) Try to split the method into two methods by moving the first 50% of the lines of code to one method and the other 50% of lines to another method.

(b) Try to factor away some of the conditional (if/then/else) statements and nested loops into separate methods.

(c) Perform a boundary value analysis of the method in order to test it.

(d) Try to refactor the entire class and the classes it is connected with.

(e) *Nothing in particular.

QUESTION 18
Imagine that you have a Java method written by one of your colleagues, with a cyclomatic complexity of 30. The method performs a calculation of the price of a product, taking into account different factors, such as optional components included in the product, chosen delivery method and applicable discounts. What would you do?

(a) Try to split the method into two methods by moving the first 50% of the lines of code to one method and the other 50% of lines to another method.

(b) *Try to factor away some of the conditional (if/then/else) statements or nested loops into separate methods.

(c) Try to refactor the entire class and the classes it is connected with.

(d) Perform a boundary value analysis of the method in order to test it.

(e) Nothing in particular.

QUESTION 19
Consider an application that manages an address book containing contact details of companies and individuals. When calculating the number of unadjusted function points for this application, which of the following is an example of a user inquiry?

(a) Add a new contact (person or company) into the address book

(b) *Retrieve all contacts sorted and grouped by city

(c) Save all contacts into an XML format

(d) Delete an existing contact

(e) Display the entire address book

QUESTION 20
Consider an application that manages an address book containing contact details of companies and individuals. When calculating the number of unadjusted function points for this application, which of the following is an example of a user input?

(a) Display the entire address book sorted by name

(b) Retrieve all contacts located in a given city

(c) Save all contacts into an XML format

(d) *Edit an existing contact

(e) None of the above

SECTION B

Provide a concise answer to each of the following questions. Please answer these questions in a separate piece of paper. Answers can be given in Estonian or English.

QUESTION 21
What is test-driven development? What are the main steps in a test-driven development cycle? [**4 points**]

TDD is an iterative software development method based on short development iterations. Each iteration is driven by a pre-written set of test cases, which define desired improvements or new functions . The purpose of an iteration is to produce the code necessary to pass the iteration's initial tests.
The main steps in a TDD iteration are: (i) Add a test;(ii) Run all tests and see if new one fails; (iii) Write code that will cause the test to pass;(iv) Run the tests cases again and check if they all pass; (v) Refactor code. Then move to next iteration.

QUESTION 22
What is equivalence partitioning? What are the advantages of using equivalence partitioning over random input? How does equivalence partitioning relate to boundary value analysis? [**4 points**]

Equivalence partitioning is a black-box testing method in which each tests case is designed to test a given partition of the space of input or output values. In general, only one test case is designed per partition.
The advantage of equivalence partitioning over random input is that the test cases are chosen in such a way as to cover different cases, so one can use less test cases and still have some degree of confidence that all discernable cases (i.e. all the identified partitions) have been tested.
To set up boundary value analysis test cases, the tester first determines which boundaries are at the interface of a software component. This is done by applying the equivalence partitioning technique. So equivalence partitioning can be seen as a first step needed before applying boundary value analysis.
QUESTION 23
What is a wireframe? What is it used for? [**4 points**]

A wireframe is a line drawing that shows the main elements in graphical user interface. Wireframes are often completed before any artwork is developed. They are used to show the information architecture and layout of the user interface. They also provide a visual reference upon which design decisions about the user interfaces can be identified and discussed.
QUESTION 24
What are the main differences between the Cocomo I and the Cocomo II models for software cost estimation? [**4 points**]

Cocomo I model is primarily intended to estimate software development effort for green-field projects (i.e. development "from scratch") that follow a waterfall model and use imperative programming languages.

Cocomo II is a more complex model consisting of four sub-models that produce estimates at different levels of detail. Cocomo II also takes into account non-waterfall development models (e.g. spiral, iterative), and other modern software development practices. Cocomo II therefore contains many more parameters than Cocomo I.

QUESTION 25

In which circumstances would you use function points for software cost estimation? In which circumstances would you use lines of code (LOC)? [**4 points**]

Function points are suitable for use in the early phases of development, where the requirements have been documented, but there is no detailed software design available.

LOC are suitable when a software design is available, so that an analysis of LOCs can be performed by analyzing the complexity of each component.

SECTION C

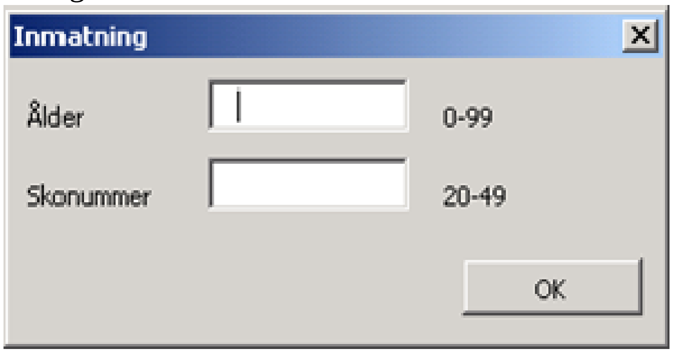
Please answer these questions in a separate piece of paper. Answers can be given in Estonian or English.

QUESTION 26 [**5 points**]

Consider the following application screen with 2 input fields

- Which black-box testing would you choose and why?
- Which test cases would you write?

Note: Ålder means "Age" and Skonummer means "Shoe number".



Ålder	|	0-99
Skonummer		20-49

Boundary value analysis is highly suitable in this case because there are clearly defined input ranges, so the partitions and the boundaries of each partition can be easily identified.
In this example, tests cases should be written with Ålder = -1, 0, 99, 100 and Skonummer = 19, 20, 49, 50, for example:
Negative test cases: (-1, 20), (100, 20), (0, 19), (0, 50)
Clean test cases: (0, 20), (99, 20), (0, 20), (0, 49)
In addition, empty inputs and non-integer inputs should be considered in the test cases.
[Note for grading purposes: Remove one point if equivalence partitioning is used instead of boundary value analysis, and remove two points if random input is used. Remove 1-2 points if the test cases do not include the values -1, 100 for Ålder and 19, 50 for Skonummer. Remove one point if non-integer inputs are not mentioned.]

QUESTION 27 [**6 points**]

Here are hypothetical coefficients for the "basic" COCOMO model:

Project	**a**	**b**	**c**	**d**
Organic	**0.01**	**1.0**	**1.0**	**1.0**
Semi-Detached	**0.02**	**1.0**	**1.0**	**2.0**
Embedded	**0.03**	**2.0**	**1.0**	**2.0**

Recall that the COCOMO equations are:

$E = a\ KLOC^{b}$

$D = c\, E^{d}$

Using the above hypothetical coefficients, what would the COCOMO basic model predict to be the development effort needed for a new release of an HTML editor? How long is the development time? Assume the original development team is available to staff the project, and that the new release is anticipated to involve 50,000 lines of code. Justify any assumptions you make and show the calculations that led to your results.

In this scenario, a semi-detached mode is applicable. An organic mode is not applicable because the project can not be classified as "small". On the other hand, the tight constraints associated with "embedded" projects are not present.

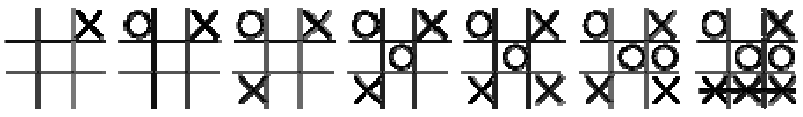
Under this assumption:
$E = 0.02(50)^{1} = 1$ person-months
$D = 1(1)^{2} = 1$ months

Note: This might be a bit of a borderline case, and an "organic" assumption can perhaps be justified if more information was available. In that case:
$E = 0.01(50)^{1} = 0.5$ person-months
$D = 1(0.5)^{1} = 0.5$ months

[Note for grading purposes: One point will be removed if the organic assumption is adopted. Two points will be removed if multiple solutions are given (i.e. one solution for each of the three modes). Three points will be removed if the embedded assumption is adopted. One or two points will be removed for calculation errors.]

<u>QUESTION 28</u> [**9 points**]
The goal of this exercise is to design an object-oriented model for the Tic-tac-toe game (in Estonian: Trips-Traps-Trull)

Description (taken from Wikipedia):
Tic-tac-toe, alternatively called noughts-and-crosses, is a pencil-and-paper game for two players, O and X, who take turns marking the spaces in a 3×3 grid, usually X going first. The player who succeeds in placing three respective marks in a horizontal, vertical or diagonal row wins the game. For example, the following game is won by the first player, X:



1. *User Story*. Consider the usecase of winning a tic-tac-toe game. Write one example of a user-story for this usecase.
2. *Object Diagram*. Draw the corresponding object diagram of the pre- condition of the winning state.
3. *Class Diagram*. Draw a class diagram which captures the internal states of a tic-tac-toe game for two human players (ignore the user interface or any other methods like saving or loading). Specifying methods in the classes is optional.

1. User Story

Title: User story for winning a tic-tac-toe game [If student forgets title this should be considered a minor flaw]

Precondition:

Alice and Bob are playing tic-tac-toe. Alice plays the X's, Bob the O's. The game field looks like the following:

```
X|X|
O|O|
 | |
```

It is Alice's turn.

[Another description would be to say the game field consists of a 3x3 table, in (1,1) and (2,1) is an X in (2,1) and (2,2) is an O]

[If user is used instead of Alice (User1 and User2 is not right but better than only User), the assignment of O's or X's is missing, whose turn it is, or the specific X's and O's on the game field are missing this should be considered a mistake]

Action [or turn, description, story]:

Alice draws an X on the upper right corner of the game field [(1,3)]

Post condition:

The game field looks like this:

```
X|X|X
O|O|
 | |
```

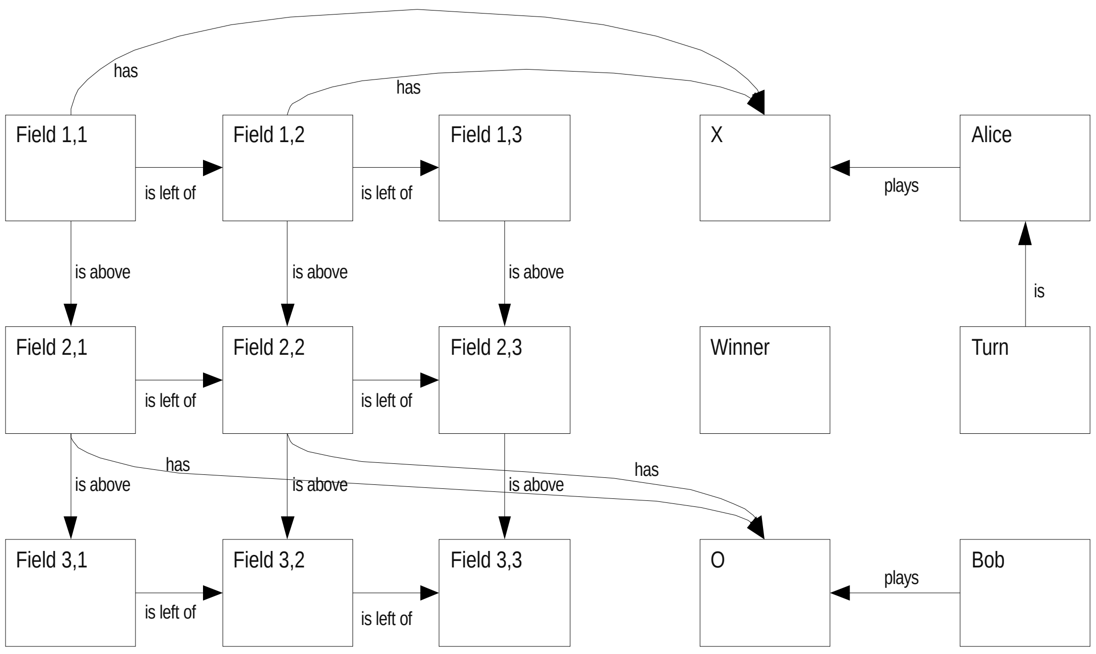
Alice has won the game [can be also mentioned in the action]

The game is over.

[Writing more like for example the actual names, age or relations of the players is not to be considered an error, can be considered a plus. On the other hand, it is important to specify the pre-condition, the action and the post-condition, otherwise remove one or two points.]

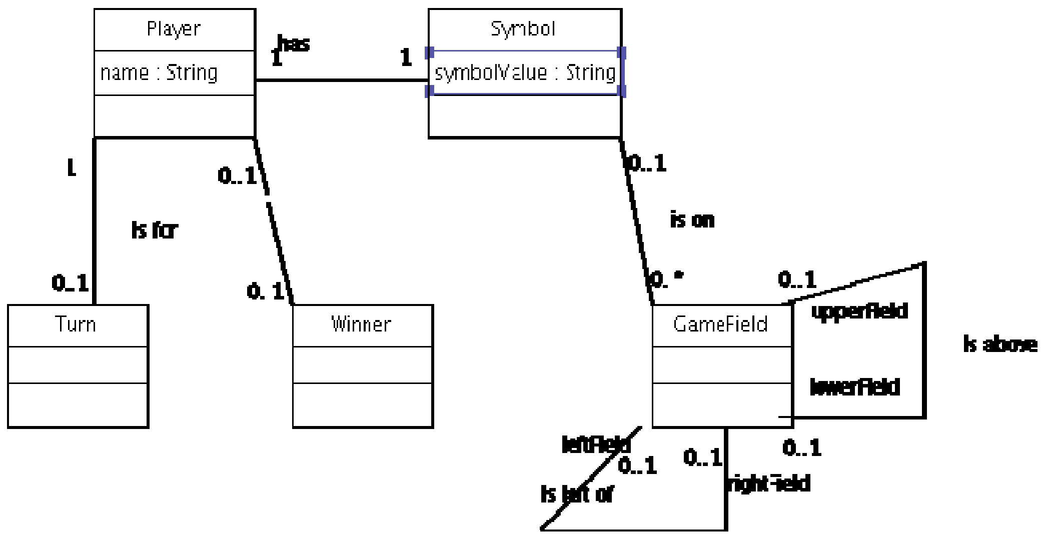
2. Object Diagram

[Game field elements should show up (nine pieces) they should be interconnected or connected to one game object, Alice and Bob (or other names) should be visible, Turn, X, O, must be visible. The connection between gamefield and X and O and between the Users and the Turn should be visible, specifying the Winner-Object is a plus]



3. Class Diagram

[Should contain Player, Turn, X and O or direct connections to the Player [optional Winner], Gamefield/Gameelement and reasonable associations]



[An alternative is to model the “Winner” not as a class, but as an association between Player and GameField (in fact this alternative is cleaner).]

[A completely different alternative is to model everything two classes, namely GameField and Player. The GameField contains a two-dimension array (presumably a 3x3 array) of booleans (with the assumption for example that ‘X’ is true and ‘O’ is false). The ‘turn’ can be modelled as an association between GameField and Player or as a method in the class GameField since the winner can be determined from the two-dimensional array. The class player only needs to have a “name” attribute.]