Internal working document, contains grammar and spelling mistakes.

The intention of the document is to give the criteria for grading and an idea of what solutions might look like. There might be many more solutions giving credits. Reservation for changes

Linköpings Universitet 2021-10-27
IDA
Kristian Sandahl

Written exam for Software Engineering Theory

Course codes TDDC88, TDDC93, 725G64

Note: When we visit the exam, I and/ or Lena will pass by all students, so you don't need to sit with your hand raised. Just call for our attention when we pass your desk.

Instructions to students, please read carefully

- **Explicitly forbidden aids:** Textbooks, machine-written pages, photocopied pages, pages of different format than A4, other electronic equipment than the computer for running Wiseflow. A silent keyboard, mouse and a webcam might be brought.
- Try to solve as many problems as possible.
- Motivate all solutions.
- Please, draw clearly.
- You may write solutions in either Swedish or English, but don't mix in the solution of the same problem.
- Please, note that the problems are not necessarily written in order of difficulty.
- TIP! Read all exercises in the beginning of the exam. This will give you the possibility to ask questions about all parts of the exam since the examiner will visit you in the beginning of the exam time.

Grading

The exam consists of two parts: Fundamental and Advanced.

The Fundamental part has problems worth 10 credits per area. Areas are Requirements, Planning & Processes, Design & Architecture, Testing & SCM, and Software Quality. Thus, the Fundamental part can give maximally 50 credits.

The Advanced part has problems worth 50 credits in total. Each problem typically requires a solution of several pages.

The maximum number of credits assigned to each problem is given within parentheses at the end of the last paragraph of the problem.

Pass condition: At least 4 credits per area in the Fundamental part **and** at least 50 credits in total. The total amount of credits also includes the bonus credits you might have got in lecture exercises autumn 2021. This gives you the mark 3. If you have at least 4 credits for 4 of the areas in the Fundamental part, then you can still pass if you have more than 60 credits in total.

Higher marks are given based on fulfilled *pass condition* **and** higher amounts of credits according to the following table:

Total credits	Mark
0-49	U (no pass)
50-66	3
67-83	4
84-	5

Multiple-choice questions

In multiple-choice questions, we will ask you to write down the letters A, B, C, or D for the one or two statements that you think are true. Note that you should not write down the statements that you think are false. There are exactly two true statements per question, so answering with three or four alternatives gives 0 credits.

For each statement that you select that is correct (i.e., that the statement is, in fact, true) you get one credit. For each statement that you select that is incorrect (i.e., that the statement is, in fact, false, but you believed it was true) you get minus one credit. Each multiple-choice question can give a maximum of 2 credits and minimum 0 credits, i.e., you cannot get negative credits for one multiple choice question.

Example 1: Assume that you have written down statements A and C. If now statements A and B were true, and statements C and D were false, you would get +1 credit for writing down A, but -1 credit for writing down C. Hence, the total credits for the multiple-choice question is 0.

Example 2: Assume that you have written down statement B. If now statement A and B were true, and statements C and D were false, you would get +1 credit for the multiple-choice question.

Example 3: Assume you correctly wrote both statements A and B. If now statements A and B were true, and statements C and D were false, you would get +1 credit for writing down A, and +1 for writing down B. Hence, the total credits for the multiple-choice question is 2.

Good Luck!

Kristian

Problems

Part 1: Fundamental

Area 1: Requirements

1 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. A requirement is said to be *ambiguous* if it cannot be interpreted in more than one way.
B. A requirement is said to be *traceable* if it is easy to find the software components that realize the requirements.
C. Requirements analysis involves *conceptual modelling*, for instance writing *use-cases*.
D. Requirements *validation* is achieved when all requirements are correctly spelled and made up by complete sentences.

B, C

1 b) *Scenario:* In a social media system users can upload pictures, short movies, and sound files. There are features of basic editing of the media files and sharing and commenting functions. There are personal user pages where you can show examples of your work and describe ambition level and dreams when it comes to media production. It's possible to submit files to panels of well-renowned members that can give you constructive criticism on how to improve.

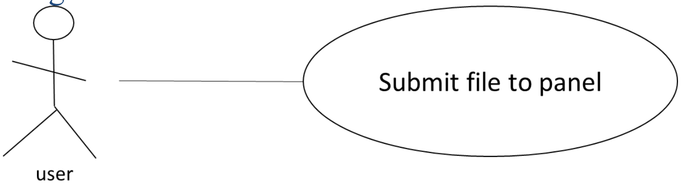
Task: Write a *use-case* of the system described above and draw a *UML use-case diagram*. Write two *user stories* of the above system where at least one is related to the use-case. (4)

Take a photo of the diagram and add that as an appendix to the answer. Let the name of the appendix start with the characters "1b".

Use-case: Submit file to panel.
Actor: User
Text:
The user logs in to the system.
The user navigates to the album of the media files for submission.
The user selects the media files to submit.
The user selects the panel from a dropdown list.
The user clicks on the SEND button and gets a confirmation question.
The user answers yes.
The media files are submitted.

The user gets a receipt of submission.

Diagram:



User stories:

As a user I want to submit photos for judgement so that I can get feed-back to improve.
Priority: x, Estimate y

As a user I want to upload photos so I can inspire my connections.
Priority: z, Estimate w

Use-case text of 3-5 sentences, 1credit.
Use-case diagram with correct actor and a verb phrase as a title, 1 credit.
We accept if the student draws a system boundary around the use-case.
Two correct user stories, I credit each.
Students must know the difference between the terms use-case and user stories.

1 c) For each of the requirements R1 and R2 below, describe one good property that the requirement has and one good property that it is lacking.

R1: The processing time of an average customer registration shall be optimal.
R2: The customer shall be able to login with a Facebook-ID and then be able to listen to music from the personal playlist. (4)

Sample solutions:
Good properties:

R1 complete sentence, use of verb "shall", numbered
R2 complete sentence, use of verb "shall" and "be able", numbered

Lack of good properties:

R1 "optimal" is a strong claim and lacks lot of information to formulate a goal function and constraints.
R2 is two requirements in one which makes it hard for effective management, e.g. prioritization.

1 credit per sensible statement.

Area 2: Design and Architecture

2 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. An *implementation (code) view* of a system can be illustrated with a *UML package diagram*.
B. An *execution view* of a system shows which source code *artifacts* that shall be compiled together.
C. A *deployment view* of a system shows on which hardware different components shall execute.
D. An *implementation (code) view* of a system is used to analyze the need of bandwidth and processing power for a *client-server architecture.*

A, C

2 b) Describe two benefits of using a *layered architectural style*. A benefit is documented with:

- one sentence description of the benefit
- a quality factor that is improved by the benefit
- a motivation of why the quality factor will be improved as a result of using the *layered architecture*. (4)

Sample solution:

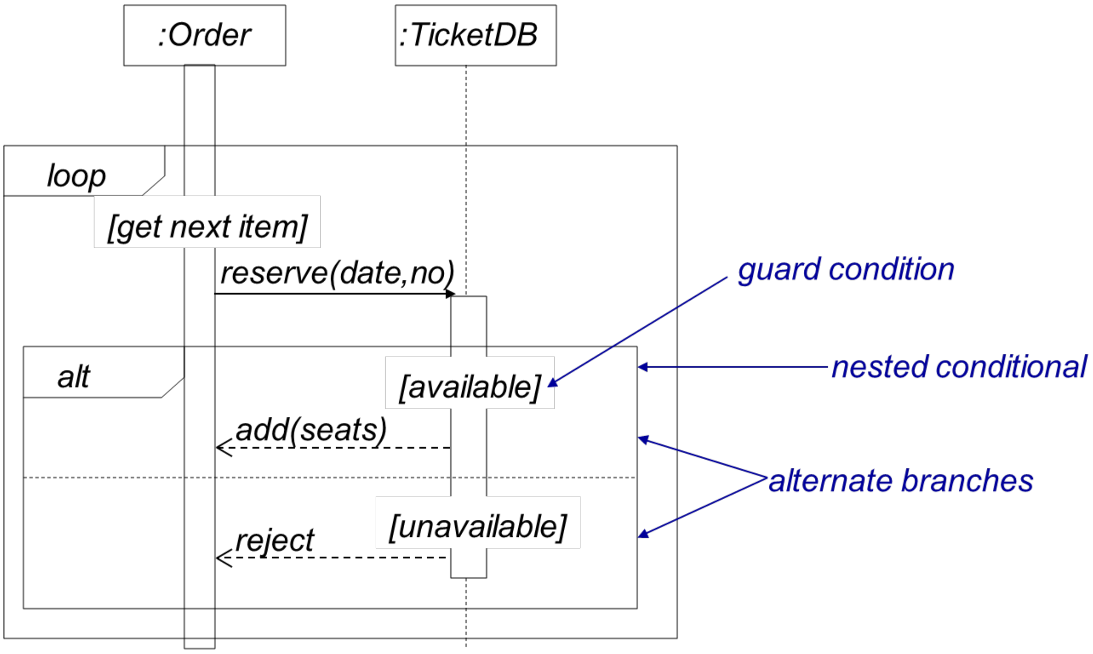
Dependencies are kept local to the layer – you don't need to analyze other layers if you change the layer in focus. This improves maintainability since changes will be faster and easier to test.

1 credit for a description of the benefit
1 credit for a sensible quality factor
2 credits for a good motivation

2 c) Draw a *UML sequence diagram* that shows an *ALT fragment*. It does not need to be a large, complicated diagram. The important thing is that you demonstrate that you understand what an ALT fragment is. The syntax must be correct and the semantics reasonable. (4)

Take a photo of the diagram and add that as an appendix to the answer. Let the name of the appendix start with the characters "2c".

Sample solution, the theatre booking system from the slides



4 credits for a good solution, the loop-fragment and the annotations are not needed just to explain the ALT-fragment. It is OK if the return messages are not dashed.

-1 p per semantic ULM fault
-1 p per syntactic UML faults, but only one instance. If for example, both guards can lack the brackets. It's still only -1 p

Area 3: Testing and SCM

3 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. A good thing about the *top-down* integration testing strategy is that you most likely will test the user interface early.
B. A good thing about the *big-bang* integration testing strategy is that you don't have to spend time writing *drivers* and *stubs*.
C. A potential problem with the *bottom-up* integration testing strategy is that *stubs* can have very many conditions.
D. A potential problem with the *top-down* integration testing strategy is that lower-level functions are often trivial, off-the shelf software.

A, B

3 b) The HR-department of the Royal Opera asked you to do a boundary value analysis testing of their system for retirement offerings. The software sends retirement information to dancers in the year when they will become 41, to singers when they will be 52, and to the rest of the personnel when they become 65. It is assumed that you can read the position and birthdate from the personnel database, and that you can access the current year from the computer clock. Answer with your test table for boundary value testing. (4)

Sample solution

Input varables: Position, birthyear, current year
Output varable: Send retiremet information

The boundaries interesting for the system is the age = current year – birthyear.
Assumption: If the person is over the age, we assume that the information was sent last year.

Test table

Id	In: Position	In: Birthyear	In: Current year	Out: send retirement information
1	Dancer	1979	2021	No
2	Dancer	1980	2021	Yes
3	Dancer	1982	2021	No
4	Singer	1968	2021	No
5	Singer	1969	2021	Yes
6	Singer	1970	2021	No
7	Other	1955	2021	No
8	Other	1956	2021	Yes
9	Other	1957	2021	No

4 credits for a correct test cases

-1 credit per missing test case
-1 credit per missing Id or variable
-2 credits if age is direct input

3 c) Describe three activities, automated or manual, that are performed in *Continuous Delivery*. What is the difference between *Continuous Delivery* and *Continuous Deployment*? (4)

Sample solution:

Activities in Continuous Delivery:

- Automatically building the system from sources
- Automated testing
- Manual testing
- Feeding back test results
- Set up test environments

The major difference is that you deploy the system at the customer environment. You might also collect data from the running system.

1 credit per activity, max 3
1 credit for the difference

Area 4: Planning and Processes

4 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. The *buffer time* is the difference between *internal* and *external deadlines*.
B. A *tollgate* is described by activities that has to be finished before a certain date.
C. The *critical path* in a Gantt-chart contains activities performed by the most expensive resources.
D. A consequence of *time-boxing* can be that you need to postpone the implementation of low-priority functions.

A, D

4 b) Shortly describe each step in the *risk management process*. 1-2 sentences per step is enough. (4)

Sample solution:

1. Risk identification means that you find out what can possibly go wrong, for instance with brain storming

2. Risk prioritization means that you analyze the probability and impact to calculate the risk magnitude indicator.

3. Risk planning means that you formulate actions to take if the risk occurs.

4. Risk monitoring means that you evaluate the risks regularly to check if the probability has changed.

1 credit per correct step

4 c) Describe two advantages and two disadvantages of moving from the *classical waterfall* model to an *iterative* model. (4)

Advantages:

- Misunderstandings and inconsistency are made clear early (e.g. between requirement, design, and implementation)
- Encourage to use feedback -> elicit the *real* requirements
- Forced to focus on the most critical issues
- Continuous testing offers project assessment
- Workload is spread out over time (especially test)
- The team can get "lesson learned" and continuously improve the process
- Stakeholders get concrete evidence of progress

Disadvantages:

- Problem with current business contracts, especially fixed-price contracts.
- With short iterations it can be hard to map customer requirements to iterations.
- Overhead added
- Requirements selection problem
- Stressful learning period

2 credits per sensible advantage, max 2 credits
2 credits per sensible disadvantage, max 2 credits

Area 5: Software Quality

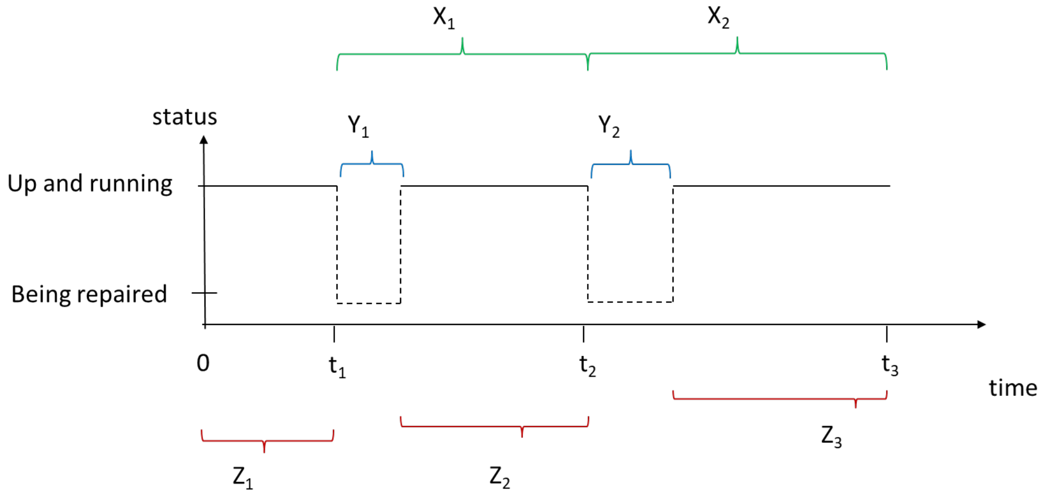
5 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

Consider the failure model below where failures occur at time t_1, t_2,.... Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. The mean of all X's can be used (together with other metrics) to estimate *availability*.
B. A high value of the mean of all X's indicates high *reliability*.
C. The mean of all Y's is called *Mean Time To Failure*.
D. The mean of all Z's is called *Mean Time Between Failures*.

A, B

Simplified model with repair time



5 b) Describe four types of Software reviews in terms of the goal and who is performing the review. (4)

Sample solution

Review	goal	actors
Inspection	Find defects	Inspectors (inspection leader, recorder, reader, author)
Management review	Find deviations from plans, risks	Management staff
Audit	Evaluate standard conformance	External personnel
Walk-throughs	Find anomalies	Author, peers, experienced persons

1 credit per correct description, there are also Technical reviews and Peer-reviews mentioned in the course.

5 c) Scenario: You have a fairly large development company and you have a good reputation when it comes understand the customers' true needs and you deliver fast. However, you need to spend too much time in repairing defects of the products, since your reliability of the products vary a lot.

Task: Describe a *CMMI process area* that you think would help you to deliver less faulty software. Summarize the process area's *purpose*, *specific goals*, and *introductory notes* in 5-6 sentences and motivate how this could help you. (4)

Sample solution:

VER – verification
Verification means that you check your work products against their specified requirements. Work products can be code, architecture description, design, test cases etc. This is typically done iteratively throughout the development phase. Software reviews are generic techniques for verifying products, in the best case with many stakeholders. To be effective, reviews have to be planned, performed, and the results are analyzed and corrective actions are identified.

If applied to the project many problems with there will be an opportunity to find potential defects early and repair them when the memory is fresh. The analyses give an opportunity to learn, which will reduce the variance.

Alternative solution:

PPQA – Process and Product Quality Assurance:
The purpose is to provide the staff and management an objective insight into products and processes. This involves evaluating product and processes either by external evaluators or against a list of specified criteria. These activities can be coordinated with the verification processes, such as testing and peer-review. The evaluations shall be performed in all phases of the project. PPQA also involves communicating the results of the evaluations and resolve anomalies.

If applied to each project, the organization will get a clear picture of the state of each project. This will help staff and management to prioritize activities and identify the most pressing needs for problem solving. If evaluations are done often, they will be smother, faster and more exact. Using this process area requires an investment in education of performing evaluation, maybe working together with customers.

2 p for a good PA description, showing that more than the purpose is described
1 p if only the purpose of the PA is described
2 p for a good motivation for how the described PA helps the company

Part 2: Advanced

6. *Scenario:* Your company of 10 skilled software engineers has been assigned to develop a safety critical software in a medical equipment. The calendar time for the development is not too short, but there is no room for unnecessary delays. After the start in January 2022 you are supposed to deliver in June 2022. Hence, you have started making preparations by:

- Sending one of your developers to a course in formal methods for requirements engineering.
- Placing an order for a logic inference tool to help you proving the consistency and correctness of your requirements.
- Outsourcing the coding to a company in India that is evaluated on CMMI level 5
- Contracting a medical expert of 100 hours to be used in different types of review and testing.

Task: Identify two *project-specific* risks with high *risk magnitude*. Create four different *plans* per risk: *avoidance*, *transfer*, *mitigation*, and a *contingency plan*. The risks shall use information from the scenario. You can add your own assumptions about the project if you write them down.
(10)

Examples:

Risk: The use of formal methods might prolong the development time.
Plan:

- Avoidance: Stick to known methods e.g. inspections and simulation.
- Transfer: Hire an expert consultant to work with formalization and verification.
- Mitigation: Start a study circle in using the tool with known requirements.
- Contingency plan: Send more people to the course so they can help each other.

Risk: There are communication problems between your company and the vendor in India.
Plan:

- Avoidance: Let your own company also do the coding.
- Transfer: Outsource to a company in your own cultural neighborhood.
- Mitigation: Arrange visits on both sites for people involved.
- Contingency plan: Let your own company also do the coding and pay overtime. Buy a tool with code generation ability.

1 credit per sensible, project specific risk
1 credit per sensible item in the plans

We accept several risks but they only get 2 credits, so just phrasing 10 risks will only give 2 credits.

7. *SCRUM* and *Kanban* are popular frameworks for organizing development teams in software engineering. SCRUM is more prescriptive than Kanban since there are more practices, roles, and artifacts mandated by SCRUM. Select five concepts that are prescribed by SCRUM, but not in Kanban. For each of the concepts, write down

a) A description of the concept
b) A motivation of why the concept is *agile*
c) A motivation of whether the concept could be used by a team using Kanban
d) A motivation of whether the concept could be used by a team using the *classical waterfall* model.

A description typically has one sentence, motivations 2-4 sentences.
(20)

Sample solutions:

Sprint:

a) A sprint is a time-boxed iteration where the team is working on product backlog items. Normally 1-4 weeks.
b) It's agile in the sense that the work is divided into several iterations where the work is evaluated after each iteration.
c) Theoretically no. It might be used in Kanban but with the reservation that the team might be disturbed during a sprint.
d) No, the waterfall model assumes that all requirements are known from the start of the project and that testing is done after implementation. Of course, a development team might use sprints when they tick of the requirements, but only unit testing will be done.

SCRUM-meeting:

a) A SCRUM meeting is a short daily, stand-up meeting where the team-members answer three questions: What have I done since last meeting? What will I do until next meeting? Are there any problems?
b) It's agile since it promotes direct communication over written reports.
c) Works well with Kanban, it can actually be good to meet in front of the board every day. This assumes that the team is small enough to make the meeting efficient.
d) Works well, nothing in the ideas behind the classical Waterfall model is against a joint coordination meeting every day.

Cross-functional teams:

a) The team is itself a role in SCRUM. The team has the cross-functional knowledge needed and is self-organized.
b) It's agile in the sense that collaboration between individuals is prioritized.
c) Cross-functional teams in Kanban works well. Teams are freer in Kanban, it is possible to have a specialist team.

d) No, there is limited use of requirements people in other phases on a regular basis. They can be invited occasionally, but not as regular members.

Prioritized product backlog:

a) The product backlog is a list of items that need to be performed in the project. The items are sorted in order of priority.
b) It's agile if the backlog contains user stories that are short fairly independent and can be re-poetized often. It can also replace detailed requirements and detailed release plans. It can, however, be misused if the items are large and complex.
c) Works well, the difference is marginal. In Kanban the backlog does not need to be prioritized, but often is.
d) Yes. If the requirements in the specification are prioritized. It is a static prioritized backlog.

Burndown chart:

a) A burndown chart shows how much work is left as a function of time(days)
b) It is agile as it counts work in agile estimation points and have product backlog items as the granularity for ticking things off.
c) Generally, no. Items might be added to the backlog any time and iterations are optional.
d) It's meaningless, since most requirements are done in the end of the project.

1 credit per sensible item a)-d) if the concept is prescribed in SCRUM but not in Kanban. Max 20 credits.

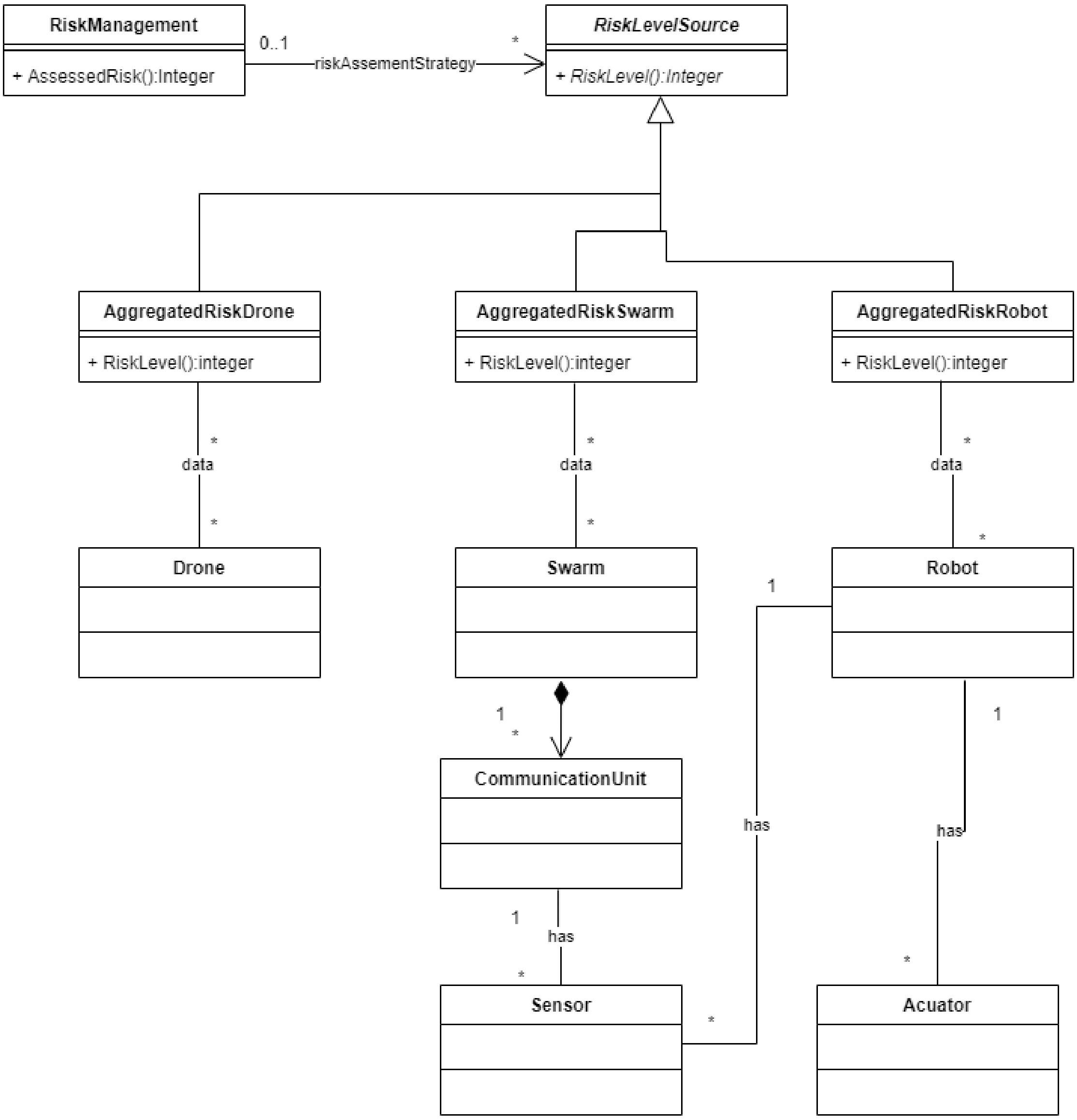
8. *Scenario*: You are developing a risk management system for missions in dangerous zones in catastrophic areas and your goal is to assess the aggregated risk level (0-100) in a zone with three different means:
1. You send up a drone to measure different properties from the air and calculate a risk level from the measurements.
2. You drop a swarm of several simple and expendable measurement and communication units and calculate the risk level from these measurements.
3. You drop an expensive robot with advanced sensors and capability to lower the risk. Likewise, a risk level is calculated.

It is important to be able to shift the source of the risk level. For instance, you might want to send in a swarm first before sending the expensive robot.

Task: Draw a *UML class diagram* of the concepts involved. You only need to have the most important attributes and methods of the classes. Use the *Strategy design* patterns for full credits. If you don't know the Strategy design pattern, make a sensible diagram, you will get credits for such model too. (10)

Take a photo of the diagram and add that as an appendix to the answer. Let the name of the appendix start with the characters "8".

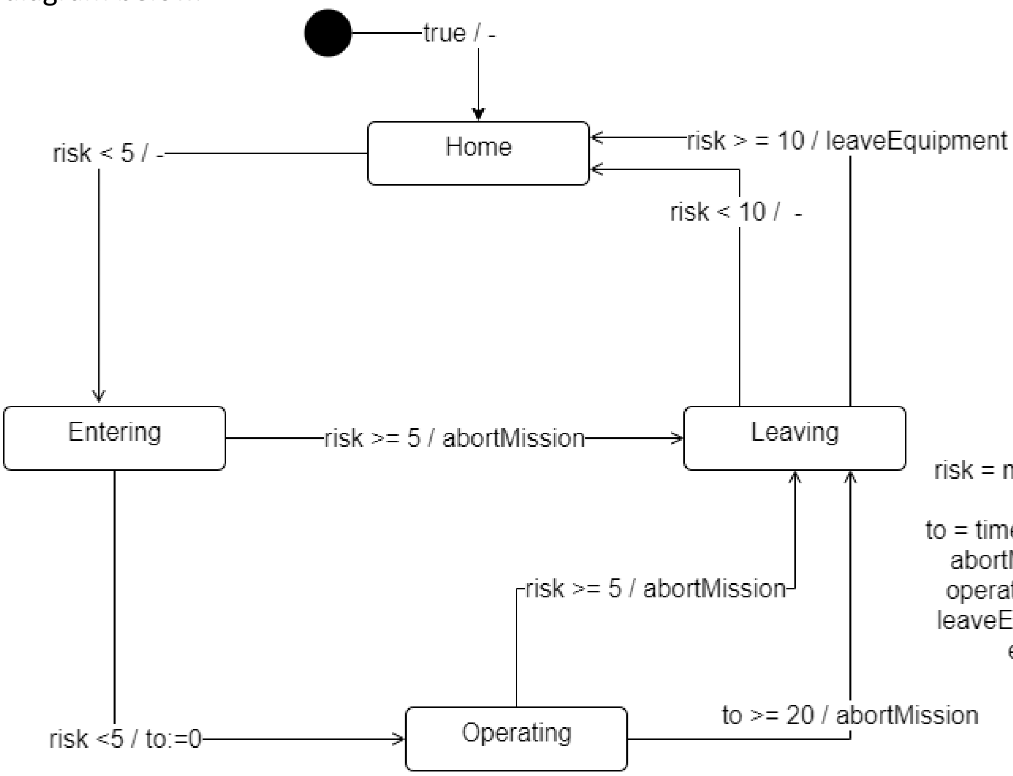
Sample solution



It is a quite straight-forward solution. You can probably find even more elegant ones. Full credits if both the data-view and the physical is there, the Strategy pattern is used, there are names and multiplicity on non-specialized associations.

-2 credits if the Strategy Pattern is not used
-2 credits if the physical equipment is forgotten and only the data view
-1 p if names are missing on association apart from generalization and composition
-1 p if there are associations missing multiplicity
-1 p per type of syntactic of semantic mistake, but only one minus per misunderstanding. For instance, assume that they have turned two generalization associations upside down, then we withdraw 1 credit, for the wrong understanding. But we don't withdraw 2 credits even though the problem was manifested twice.

9. *Scenario*: Same as in problem 8, but with the extension that you can send in a human operator into the zone. The states of the class human are shown in the state diagram below.



risk = most recently assessed risk (0-100)
to = time in operation (minutes)
abortMission = discontinue operation; prepare to leave
leaveEquipment = don't bring equipment back

Task: Create a *test table* where the union of the *test cases* exercises all *transitions* shown in the diagram at least once. Besides the usual properties of a test case, the test case shall specify which state(s) that are covered. Also include current state (before the execution of the test case) and next state (the state that is reached after the execution of the test case) for each test case in the table. (10)

Id	Current state	In: risk	In: to	Out: action	Next state	Passed states
1	Start	-	-	-	Home	Start, Home
2	Home	4	-	to:=0	Operating	Home, Entering, Operating
3	Entering	6	-	abortMission	Home	Entering, Leaving, Home
4	Operating	4	20	abortMission	Home	Operating, Leaving, Home
5	Leaving	12	-	leaveEquipment	Home	Leaving, Home

The table above is just an example but gives 10 credits

2 credits per correct test case, max 10 credits

-1 per transition not taken (it is OK to skip the first one since the start is a pseudo state
-1 per column forgotten (but we are kind when it comes to to, it is a bit tricky since it is both set and used as input)
-2 if major parts of a test case is missing, such as input variables.