

Grading Instructions to Exam 2023-10-25

There are many different solutions possible. This is working material. Misspellings and grammar errors do appear.

In almost reverse order

9. Scenario: Suppose that you've got the job to develop a software system for a computerized examination room for Linköping University. The examinations are taken by students in a specially equipped room as soon as they feel ready. Biological sensors determine the identity of the students, and electronic sensors aim to ensure that no communication equipment is brought into the room. Most exam exercises are graded automatically. There is grading support for essay questions using topic and phrase extraction from answers. The requirements on security and reliability are high.

You and your four team members start the development project on 8 January 2024 and start alpha testing on 26 August 2024. You need to use formal procurement procedures to buy the hardware. This includes publishing specifications, waiting three weeks for bids, evaluating bids formally, publishing the evaluation result, providing an appeal period, and writing a contract with the chosen vendor.

Since this project is supported by EU funding, you get paid in EUR and need to outsource some of the development to another EU country.

Task: Make a list of *five relevant risks* that need to be monitored. Use your risks to demonstrate how you can calculate the *risk magnitude indicator*. Moreover, make a *plan for each identified risk*. Select your risks so that you can *give examples of four different types of risk planning*: Risk avoidance, risk transfer, risk mitigation, and contingency plan definition.

Hint: You don't have to give examples of all types of risk planning for each risk. It is sufficient if all types of risk planning occur in your entire solution. So Risk #1 demonstrates risk avoidance, Risk #2 highlights risk transfer, etc. (10)

9.

Sample solution:

1. You might lose money when paying staff in local currency. Plan, Avoidance: Pay all people in EUR. Probability: 2 moderate. Impact: tolerable: 2. Risk Magnitude Indicator=4.
2. You might make a mistake in ordering the hardware. Plan, Transfer: Let a procurement consultant handle the process. Probability: 2 moderate, Impact: 3 serious. RMI=6.
3. There will be communication problems with the team outside Sweden. Plan, mitigation: Make study visits at both sites. Probability: 4 very high, Impact: 2 tolerable. RMI=8
4. The hardware might be delivered too late to be used in testing. Plan, contingency plan: Use simulators during testing. Probability: 3 high. Impact: 2 tolerable. RMI=6
5. You might find that you have a too small data set to use topic recognition. Plan, contingency plan: cooperate with more universities with similar courses. Probability: 1 low, Impact: 2 tolerable: RMI=2

9. Grading:

2 p per sensible risk and plan

Probability and Impact are just examples and can vary a lot

-2 p if not all four types of plans are covered.

8. In the list below you find the twelve principles of agile software development from the Agile manifesto. Select five of them and write down how they can be realized in SCRUM or eXtreme Programming (XP). For SCRUM, you define which artifacts, roles, or meetings that are involved. For XP, you mention the practice (rule) that you are referring to. Your motivations shall be thorough, 5-10 sentences per agile principle. (10)

The twelve principles behind agile software development from the Agile Manifesto

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

8. Sample solution

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

In XP you use *user stories*, small releases and *spike solutions*. User stories are obtained directly from the stakeholders. Spike solutions demonstrate the implementation of the user story. These rules focus on things highly valued by the customer and postpone generalizing the implemented functions. With small and frequent releases, the customer doesn't have to wait for long to start seeing valuable software and can confirm if you are on the right track.

2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

In SCRUM we divide the work into *sprints* and each sprint starts with consulting the continuously prioritized product *backlog* list. A sprint is typically 1-4 weeks long. This ensures that the current focus is the top prioritized requirements. Selected items are refined into a sprint backlog list. Unless there aren't very large dependencies between backlog items of many sprints, each sprint begins with a clean table, which means that both new and old requirements can be handled throughout the process.

(cont'd)

8. Sample solution (cont'd)

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

In XP we have the rule to *make frequent small releases* and each *release is divided into iterations*. As the names of the rules indicate, it fulfills the principle as regards the size and frequency. Other rules, such as *integrate often* and run *acceptance tests* assure that we have working software in a continuous flow of the development project. XP does not mandate any length of releases and iteration but mentions a few weeks as a recommended pace.

4. Business people and developers must work together daily throughout the project

This principle is probably not selected. Maybe you can make something from the implication of SCRUM that uses cross-functional teams, and business people might be included. The Product owner is supposed to stay in contact with all stakeholders, including business people. Maybe smart students can give us more material for this.

(cont'd)

8. Sample solution (cont'd)

5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The idea of having the *team* as a role in SCRUM realizes this. The assumption in SCRUM is that the same team members continue working together for many sprints, thus fostering good cooperation and high group motivation. The team consists of people who have all the knowledge needed and can thus become autonomous. The role of the SCRUM master is to help the team to get resources, remove bottlenecks, and to work undisturbed during the sprint. There is no guarantee that the individuals will become highly motivated by working together; that is more a general experience.

6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

This is the core idea behind the term SCRUM and the daily SCRUM meeting. Making sure that everyone gets at least a condensed version of first-hand information increases productivity by removing misunderstandings in communication. A physical meeting captures unspoken things as body language. It is also valuable that each team member recapitulates what was done and makes a quick plan for the day. The SCRUM meeting puts special emphasis on perceived problems so that people with the right competence can gather to solve them fast. Without the daily SCRUM meeting, time will be spent in formulating problem descriptions that need to be interpreted and can contain ambiguities.

And so on...

8. Grading:

For each principle we give 2p for sensible motivations of how they are realized.

The number of sentences is not a strict requirement as long as descriptions are complete

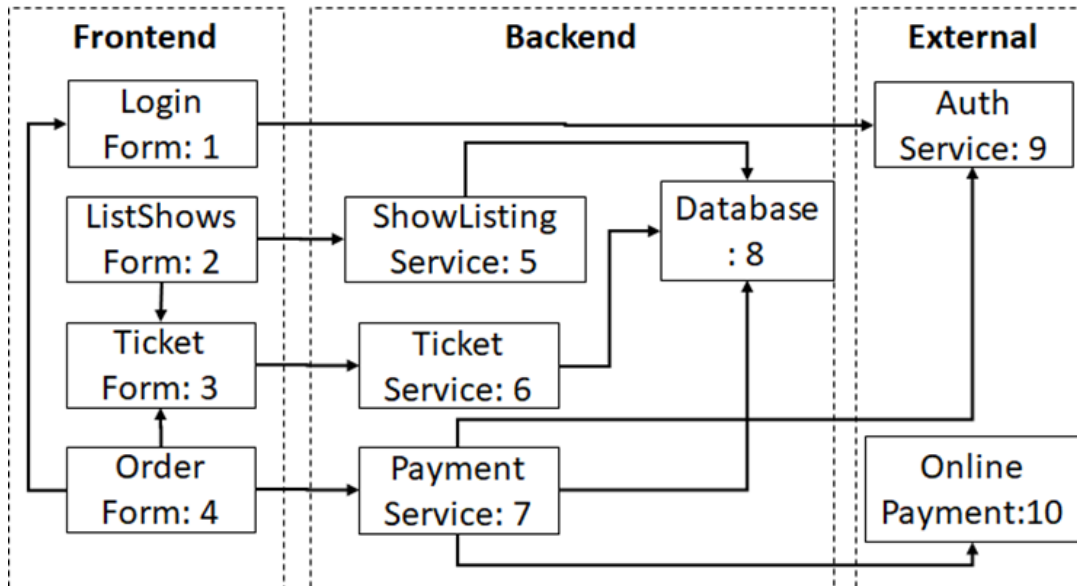
We withdraw 1p if no references to rules of XP or SCRUM concepts are referred to.

We withdraw 1p if the student mixes concepts both from XP and SCRUM

We ignore superfluous information unless it's a directly false statement, such as, there is no limit of team size in SCRUM

Section 7

Consider the following block diagram representing the architecture and dependencies of a multi-tier software system (where an arrow from component A to component B means that A calls / uses B).



Task: Propose and briefly explain an **integration strategy** which (a) minimizes the number of stubs and (b) postpones integration of external services to a later stage integration.

For each integration stage, specify clearly *what components are tested together at the given stage, what stubs are needed*. Moreover, define what is the *integration order* between the stages! (Each component is uniquely identified by a number next to it, feel free to use those numbers as abbreviations to help describe your strategy in a more compact way)

Key considerations and principles:

- The number of stubs is minimized by a bottom-up integration strategy,
- But as the integration of external services needs to be postponed, we need to design a **sandwich integration strategy**.
- At each step, we need to integrate **one** new component
- We should not integrate a component until all its dependent components (other than the external ones 9 and 10) have already been integrated.
- The dependency graph defines a partial order between components, the correct solutions should satisfy this partial order.
- At most two stubs are needed, a solution should not introduce more than 2 stubs

Step #1: {**6**, 8}

Step #2: {**5**, 6, 8}

Step #3: {**3**, 5, 6, 8}

Step #4: {**2**, 3, 5, 6, 8}

Step #5: {**1**, 2, 3, 5, 6, 8} - Stub: 9

Step #6: {1, 2, 3, 5, 6, **7**, 8} - Stub: 9, 10

Step #7: {1, 2, 3, **4**, 5, 6, 7, 8} - Stub: 9, 10

Step #8: {1, 2, 3, 4, 5, 6, 7, 8, **9**} - Stub: 10

Step #9: {1, 2, 3, 4, 5, 6, 7, 8, 9, **10**}

A sandwich strategy is designed and named (1p: sandwich strategy, 0.5p: bottom-up strategy) [1/1]

The proposed integration strategy is explained [1/1]

The integration plan is precisely defined and meaningful steps [1/1]

Integration plan integrates all components [1/1]

Stubs are consistently specified in integration strategy (even if not minimal number) [1/1]

The integration strategy uses the minimal number of stubs (2 stubs, not more, not less) [1/1]

One new component is integrated at each step (1p: always, 0.5p: sometimes, 0p: rarely) [1/1]

Components are only integrated after all their dependent components (except for externals) [2/2]

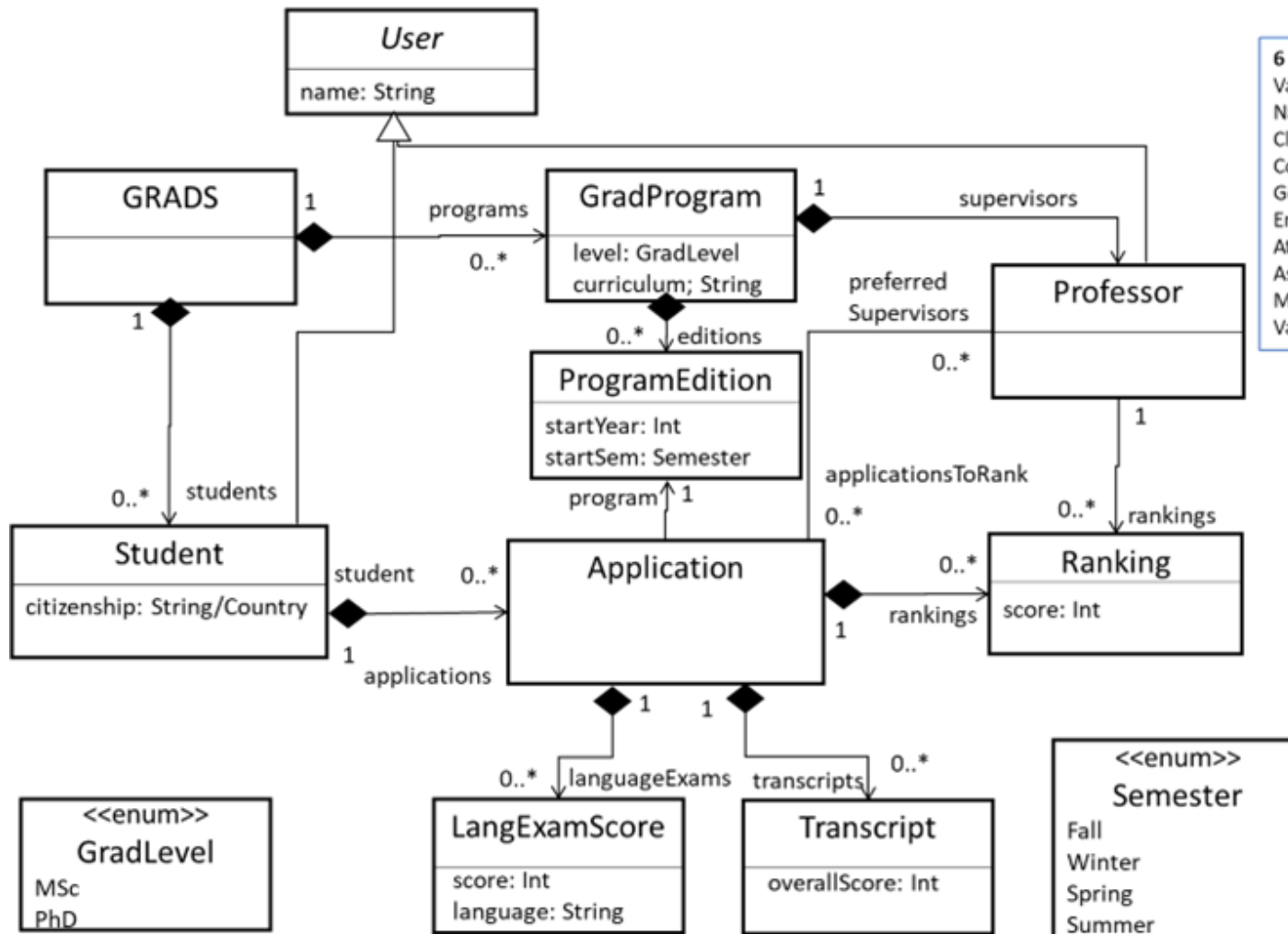
Integration of external components 9 and 10 is sufficiently delayed [1/1]

Section 6 (a)

Scenario: A graduate student application management system (GRADS) helps collect and review applications from prospective graduate students from all around the world to graduate programs offered on different levels (MSc vs. PhD), in different curricula (e.g., Computer Science, Software Engineering) and at a given starting time (e.g. Fall 2023). Prospective graduate students (MSc or PhD students) create a personal profile (with their name and citizenship) and then upload their application, which must contain (a) their language exam score and (b) their official transcripts including their aggregated score. When submitting their application, students specify their preferred supervisors. GRADS checks if all minimum criteria of graduate admission are fulfilled and sends automated emails to prospective students to complete missing information. Professors then rank applications of those students in GRADS who meet the minimum criteria and who selected them as a preferred supervisor by assigning a numeric score. A student may be admitted to the graduate program if he or she is selected by at least one professor, otherwise his/her application is rejected.

Task: Draw a domain model in the form of a UML Class Diagram for the GRADS system showing the domain concepts and their relationships as well as potential generalizations. Specify multiplicities for your associations and compositions, and give them meaningful names. Provide a total of 5 key attributes mentioned in the description above together with their type. (10)

Solution (Section 6.a)



6 a) Feedback:

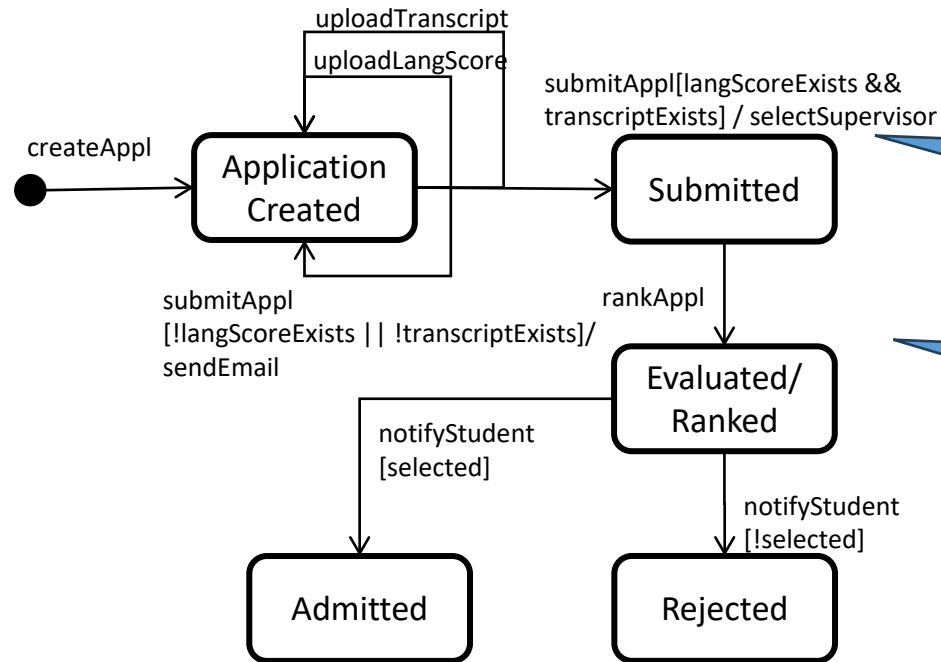
Valid syntax for UML CD (1/1)
 Naming is appropriate (1/1)
 Classes are consistent and complete (1/1)
 Containment is consistent and complete (1/1)
 Generalizations are consistent and complete (1/1)
 Enumerations are consistent and complete (1/1)
 Attributes are consistent and complete (1/1)
 Associations are consistent and complete (1/1)
 Multiplicities are consistent and complete (1/1)
 Valid justification provided (1/1)

Section 6 (b)

Scenario: A graduate student application management system (GRADS) helps collect and review applications from prospective graduate students from all around the world to graduate programs offered on different levels (MSc vs. PhD), in different curricula (e.g., Computer Science, Software Engineering) and at a given starting time (e.g. Fall 2023). Prospective graduate students (MSc or PhD students) create a personal profile (with their name and citizenship) and then upload their application, which must contain (a) their language exam score and (b) their official transcripts including their aggregated score. When submitting their application, students specify their preferred supervisors. GRADS checks if all minimum criteria of graduate admission are fulfilled and sends automated emails to prospective students to complete missing information. Professors then rank applications of those students in GRADS who meet the minimum criteria and who selected them as a preferred supervisor by assigning a numeric score. A student may be admitted to the graduate program if he or she is selected by at least one professor, otherwise his/her application is rejected.

Task: Describe the state-based behavior of the *GraduateApplication* class by a UML Statechart diagram. (The *GraduateApplication* class of the GRADS system represents the application created or submitted by a graduate student.) (10)

Solution (Section 6.b)



Supervisor selection can be a condition (but then a related trigger is also needed)

A state for a complete application can be introduced

Triggers:

createAppl
uploadAppl / uploadTranscript /
uploadLangScore
submitAppl
sendEmail
rankAppl
notifyStudent

6 b) Feedback:

UML statemachine syntax is valid (2/2)
States are appropriately named (1/1)
Triggers and actions have proper names (1/1)
Triggers are consistent and complete (1/1)
States are consistent and complete (1/1)
Guards are consistent and complete (1/1)
Transitions are consistent and complete (1/1)
Behavior is compliant with spec (2/2)

Section 1.a

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

- ✓ a. A key difference between a *functional* and *non-functional requirement* is that the latter cannot be attributed to a single component, but to the system as a whole.
- b. As *user requirements* do not contain technical details, but describe what services the system is expected to provide, user requirements only include functional requirements but exclude non-functional requirements.
- ✓ c. *Workflow models* used in requirements engineering describe how the system operates in a business context, e.g. within the business processes at a company.
- d. *Interviews* during the requirement elicitation process help to formally capture the true needs of the customer hence eliminating all potential fuzziness in requirements.

Section 1.b

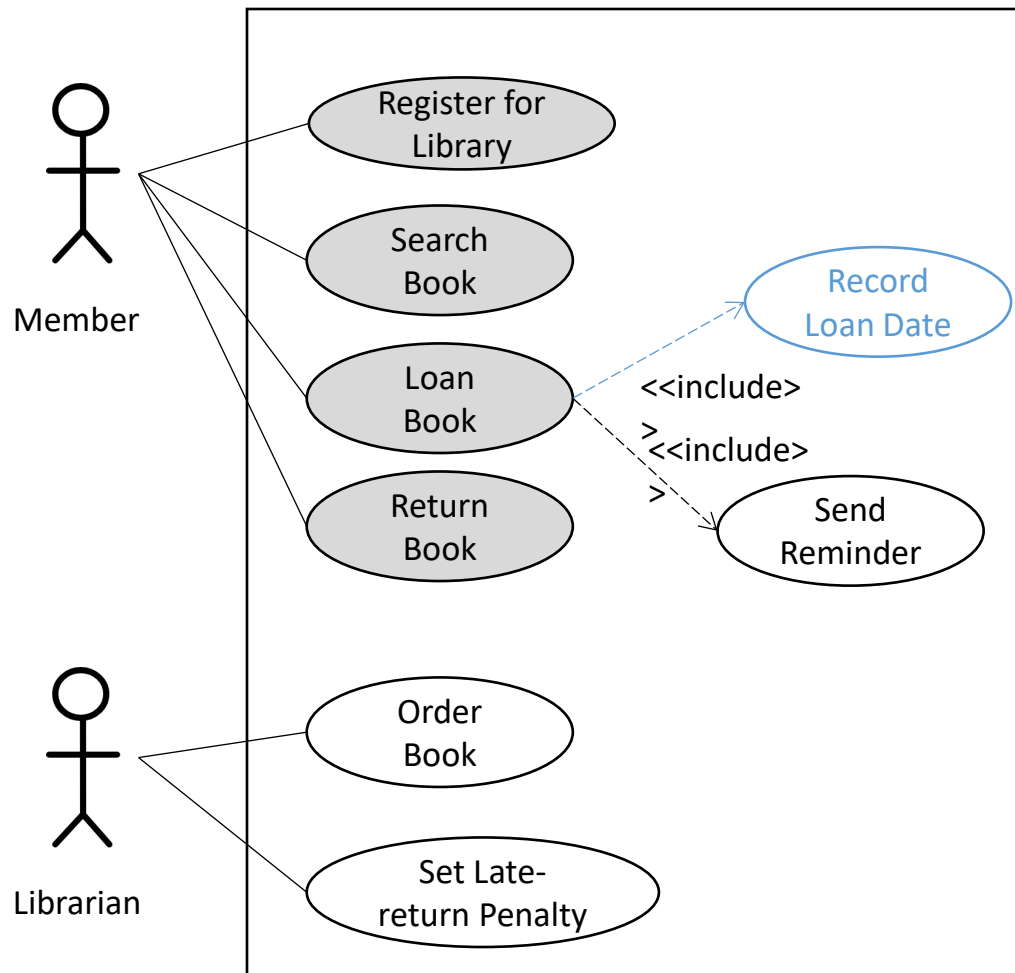
Scenario:

A web-based library management system (LMS) allows library members to take books on loan. A member can register for the library by providing his/her address and personal number. Then (s)he can search for a book with a given title or from designated authors. The library keeps a limited number of copies for each book. The librarians may order new books by the end of each month. If a copy of the designated book is available for a target period, then it can be taken for loan by a member, and the system records the loan date and return date. One week before the return date, the system sends an automated reminder to the member. If the book is returned by the member after the return date, then the librarian sets a late-return penalty.

Task: Create a UML Use Case diagram for the LMS with key actors and use cases. Give appropriate names for these actors and use cases, but no detailed descriptions are needed. (Logging in and logging out are basic functions, not to be considered as individual use-cases.)

Append a diagram of the use-cases and write in the text which appendix that contains the answer to this question. (4)

Solution (Section 1.b)



Grading instructions:

- Correct UML syntax (0.5),
 - Naming conventions are kept (0.5)
 - 2 Actors (0.5)
 - System boundary but no System actor (0.5)
 - 4 (grey) UCs for Member (0.25 each)
 - 2 UCs for Librarian (0.5)
 - 1-2 UCs for System - at least 1 include (0.5)
- Partial score can be assigned to each item

Section 1.c)

Scenario: Same as in Problem 1b) above

Task: Write *two user stories* of the system described above. Write *one non-functional requirement* for the system together with a *short recommendation* (1-2 sentences) on which level of testing can it be verified. (4)

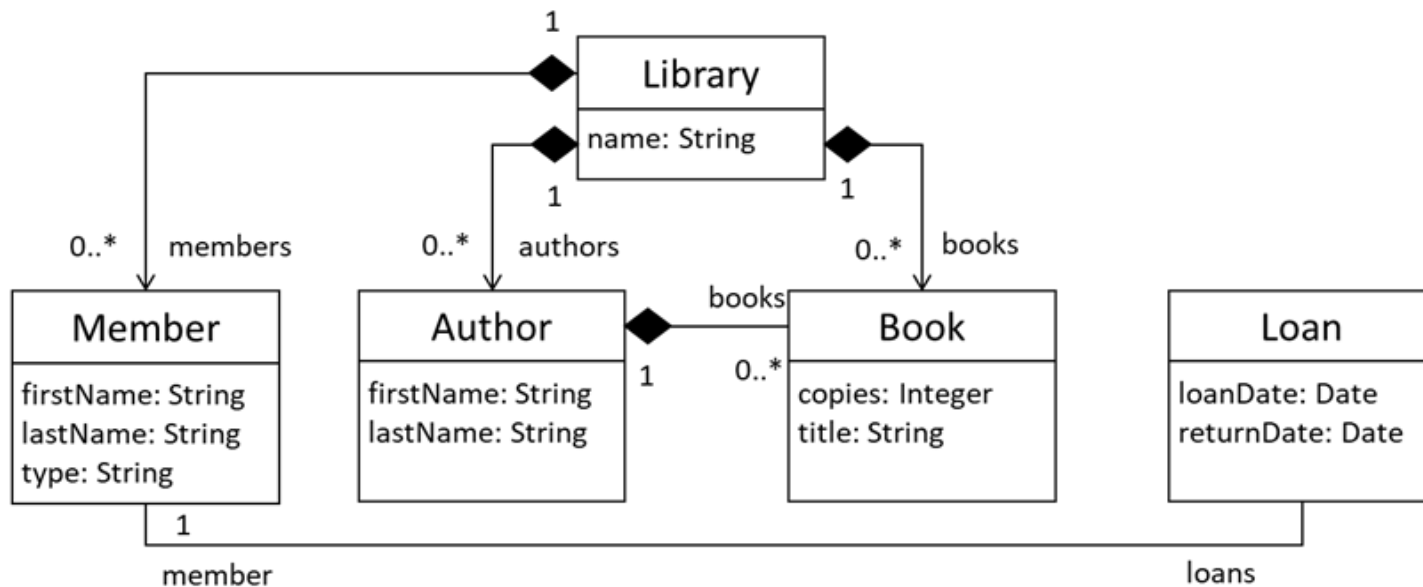
- I) *As a member, I want to loan a book from the library so that I can read more literature.*
II) *As a librarian, I want to order books for the library so that our library has a wide selection of books.*
- **1p each if the general template is followed:** “As a (role) I want (something) so that (benefit)”
- The Library Management System shall allow the members to obtain the list of all books they loaned in less than 1 second for 99.9% of attempts. (**1p** if it is a NF requirement, and the recommended contents are there – defines the system, use modal verbs, defines positive outcome, measurable)
- Non-functional requirements are typically *tested on system-level*. As *NF requirements are properties of the entire system, hence the entire system needs to be integrated* (cannot be integration test). Moreover, NF requirements are typically system requirements and not user requirements, so they are not typically part of an acceptance test suite. (**1p**)

Section 2.a)

- ✓ a. The system architecture already contains all decisions on which functional component needs to be deployed on what platform element.
- ✓ b. For modern web-based systems, the model-view-controller architecture is often combined with a layered architecture.
- c. The performance of a system can be scaled up by creating a large number of subsystems with redundant functionality.
- d. By introducing a unified interface, the Façade design pattern promotes high coupling between subsystems.

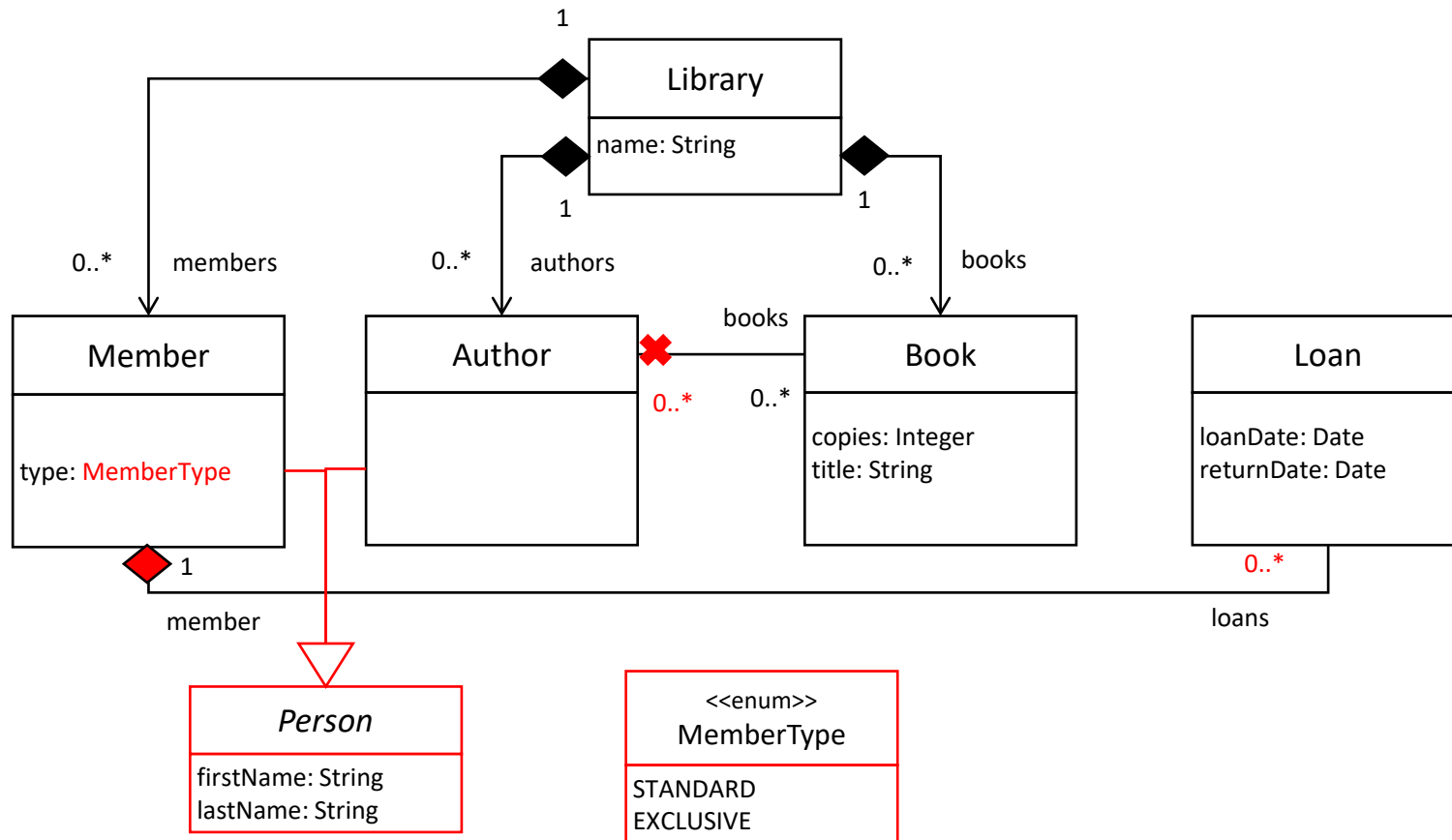
Section 2.b

Carry out a thorough design review of the following domain model captured by a UML Class Diagram (5)



Write down your recommendations how the design should be improved in constructive way as you would propose as a reviewer as part of a design review.

Solution (Section 2.b)



Section 2.C)

Which *non-functional system characteristics* can be enhanced by redundancy and diversity? Give *two examples* of a software system where the respective non-functional characteristic is important. (3)

- Typically, safety is enhanced by *redundancy* and *diversity*. The same functionality is implemented and deployed by multiple components (redundancy) by different teams (diversity), the output may be calculated by voting.
- (1p for naming safety and explaining at least redundancy or diversity)
- *Examples* of safety-critical systems: Civil avionics, modern cars, nuclear plants (1p for each example)

Section 3.a)

Which of the following statements are true? There are exactly two correct answers, wrong answers give negative score.(2)

- ✓ a. *White box testing* can only reveal extra functionality but not missing functionality.
- b. Tests for non-functional requirements are carried out as part of *unit testing*.
- c. The role of a *stub* is to call the function under test and pass test data to it.
- ✓ d. A test suite that ensures *full branch coverage* also ensures *full statement coverage*.

Section 3.b)

Scenario: A company is paying bonuses to their employees if they work more than a year and the company made a profit in the given year. Moreover, an employee (with at least one year of employment) receives individual bonus if he or she worked in a project that was successfully completed.

Task: Create a test table to test a business functionality that controls the paying of (company and individual) bonuses. (5)

Solution 3.B)

Inputs			Outputs	
Worked for >1 year	Company profit?	Project completed?	Company Bonus	Indiv Bonus
F	DC	DC	F	F
DC	F	DC	F	F
T	T	F	T	F
T	F	T	F	T
T	T	T	T	T

It is totally acceptable to have a table with 8 rows, one for each combination of truth values of inputs. The exercise can be understood in two different ways concerning when individual bonus is paid (one possible interpretation is highlighted above)

1p for correctly identifying the three input conditions, and the two outputs

1p for each of the four semantically different cases (which may be covered by multiple test cases).

Lines 4 and 5 are treated as a single case.

DC: don't care, any value

T: true

F: false

Underspecified

Section 3.C)


What is the difference between *Continuous Integration* and *Continuous Delivery*? Explain it in 1-2 sentences.


Name *two expected benefits* and *two main challenges* for each of them. (3)

- **Continuous integration** is a *developer practice* where developers integrate their work frequently.
- **Continuous delivery** is a *development practice* where every change is treated as a potential release candidate to deploy and/or to release (**1p** for defining this difference)
- **CI benefits:** improved quality, improved predictability, efficiency, speed, easier troubleshooting, transparency and feedback, enabling continuous deployment, it feels good (**0.5p** for naming at least two)
- **CI challenges:** scalability (with increasing team size), some constant pain, team-internal integration, manufacturability (**0.5p** for naming at least two)
- **CD benefits:** improved quality, improved predictability, efficiency, speed, easier troubleshooting, transparency and feedback, enabling continuous deployment, it feels good (**0.5p** for naming at least two)
- **CD challenges:** last 1% is 90% of effort, limited time and capacity to test, test flakiness, establishing trust (**0.5p** for naming at least two)

4 a) Which of the following statements are true? There are exactly two correct answers, wrong answers give negative score.(2)

☐ The definition of an iterative life-cycle model is that components are developed and integrated with a fully working sub-system.

 ☒ A potential drawback of an iterative life-cycle model is that extra administrative overhead for planning and coordination is added.

 ☒ The classical Waterfall life-cycle model can be a good alternative for fixed-priced contracts.

☐ An agile method constantly adapts the requirements to fit the system being developed.

4 b) Describe the following properties of Kanban in 1-2 sentences: *Kanban board*, *lead time*, and *work in progress*. Give one expected advantage of using Kanban. (4)

4 b) Sample solution:

A Kanban board consists of columns describing the state of work items. Each work item is represented with a yellow sticker card. When an item has fulfilled all criteria for a state, the card is moved to the next column. Typical columns can be backlog, selected, under development, deployed, delivered.

Lead time is the time it takes on average to process an item from customer order to delivery.

Work in progress is the number of items that are simultaneously in a state other than backlog or delivered. The work in progress is limited to a predefined number for each state to avoid multitasking.

(cont'd)

4 b) (cont'd)

Advantages with Kanban are:

- Eliminate over-production, the #1 waste
- Produce only what is ordered, when ordered, & quantity ordered
- Increase flexibility to meet customer demand
- Competitive advantage by sequencing shipments to customers (what they want, when they want it, in the order they want it!)
- The Kanban board gives a good overview of the status of the project

Grading:

1p per correct and understandable description of concepts

1p for an advantage

4 c) Suppose you are developing software in time-boxed iterations. Name each of the four dependent project parameters and describe if it is locked or free to change. Give two examples of what you can do if one of the (free) parameters changes. (4)

4 c)

Sample solution:

For time-boxed iterations the four dependent project parameters are:

- Calendar time – locked
- Resources – free
- Features – free
- Quality – free

Example 1: Change: Resources – a person leaves the team. Solution: Features – cut some features for the delivery.

Example 2: Change: Quality – the customer wants the software to run on more platforms. Solution: Resources – add people with knowledge about the new platforms.

Grading:

2p for all parameters correctly named and classifies


1p per sensible example


Adding more people late to a project will lead to further delays (Brooks law). We still allow this solution, since we only mentioned this orally at the lectures.

5 a) Which of the following statements are true? There are exactly two correct answers, wrong answers give negative score. (2)

☐ In an inspection, it is good if the inspectors have very similar competence to give a statistical ground for process improvement.

☐ An audit is a software review with no written documents; the participants discuss about the ideas of the software

 ☒ You normally record data about the inspection itself in order to improve the expensive inspection process.

 ☒ Both the author and the inspection leader take part in the exit-and-follow-up phase of the inspection process

5 b) Suggest *two different software metrics* for measuring the *maintainability* of software. Don't forget to briefly motivate (in 2-3 sentences) why the metrics can give information of the maintainability. (4)

5 b)

Sample solution:

Description: Average Cyclomatic complexity

How to obtain data: This metric is often included in development tools and metrics plugins

How to calculate the metric: Read off the value, if the average is not an option, calculate the average from the values per function or method.

Relevant quality factor: Analysability. Several studies and recommendations try to keep the value low since in general it is hard for a human to understand the code with high cyclomatic complexity. Understanding the code is a necessary step to change it. Changing the code is inevitable in most maintenance efforts.

Description: Average lead time

How to obtain data: Time stamp change requests, and record when the change request is done (according to your definition of done)

How to calculate the metric: For each change request, calculate time done – time received. Take the average for a given number of change requests.

Relevant quality factor: Modifiability. A high speed indicates good maintainability since most maintenance implies change. If change is relatively fast, this indicates that both the product and the process are fit for maintainability work.

Grading:

1p for the first three items per metric

1p for the description of relevant quality factor per metric

5 c) Scenario: Life isn't easy: Your end-users complain that even though your software is failure-free and has a good-looking GUI, you miss certain features (functions) that "everyone" in their company know are sometimes needed. Your employees are really appreciating your personal feedback, but it comes sporadically and is based on your feelings only. As a complement, they would like to have more regular and objective feedback.

Task: Describe a *CMMI process area* that you think will help you the most. A description is typically 5-6 sentences that covers *Introductory notes* and/or *Specific goals*. Also, describe how this will help your company.
(4)

5 c) Sample solution:

PPQA – Process and Product Quality Assurance

The purpose of PPQA is to provide stakeholders with objective insights of process and products. Assessing the product and process quality can be done in different ways, for instance, by measuring characteristics or by reviewing artifacts and documents. An important part is that the methods used are based on predefined criteria. This ensures objectivity and alignment with the goals of the organizations. Objectivity can be further strengthened if the evaluation is performed by an independent organization, but this is not possible in a small organization. The result is fed back to the developing organization and deviations from targets are handled in the project, sometimes with help from company leadership or experts.

The company will be helped by the necessity to clearly formulate its goals. These goals are then broken down into criteria for different work-products and processes. In the process of setting goals, there will be reviews that make sure that nothing is forgotten. It will take a while of experimentation until well accepted criteria are in place, but once this is done the employees will accept the feedback as objective. The more the personnel are involved, the faster the experiments will be finished.

Grading:

2p for a good PA description. If only the purpose statement is elaborated 1p.

2p for a good motivation.

There are different PAs that can be applicable to the problem. For example, Verification or Validation can provide a statement of the product to ship.

Marks

Total credits	Mark
0-49	U
50-66	3
67-83	4
84-	5



Allowed aids

- Two sheets of handwritten A4 papers (you can write on both sides)
- One volume of dictionary to or from English or an English wordbook.

Explicitly forbidden aids

- Textbook
- Machine-written pages
- Printout from drawing software
- Photocopied pages
- Pages of another format than A4
- Electronic equipment



Hints

- Register for the exam
- If necessary, reserve a guest computer
- Never guess on the first multiple-choice questions of each KA
- Bring a good pen for diagrams and sketches
- Have the nerve to read through the exam first
- Use time-boxing and buffer time
- Do as the exam invigilators say
- If you have questions, wait until we pass your table.

Thanks for listening!



GOOD LUCK!