Example answers of exam 2013-11-01 in reverse order

8. *Scenario:* You are developing a global peace stabilization system (GPSS). The idea with the system is to collect lot of information about potential conflicts in the world. The more different evidence you have, the stronger is the degree of alert:

1. **Green**: the region is observed due to some incidents.
2. **Yellow**: there are conflicts in the region.
3. **Red**: immediate attention by the United Nations is required.

You have several sources classified as:

- **Indicators**, for instance, changes in the argumentation in media.
- **Reliable**, for instance, video clips taken by habitants.
- **Highly reliable**, for instance, satellite photos.

Each source has its own data collection method, data base, and completely automated data analysis. The output from each source is the percentage of conflict likelihood and a threshold value, indicating that conflicts are detected.

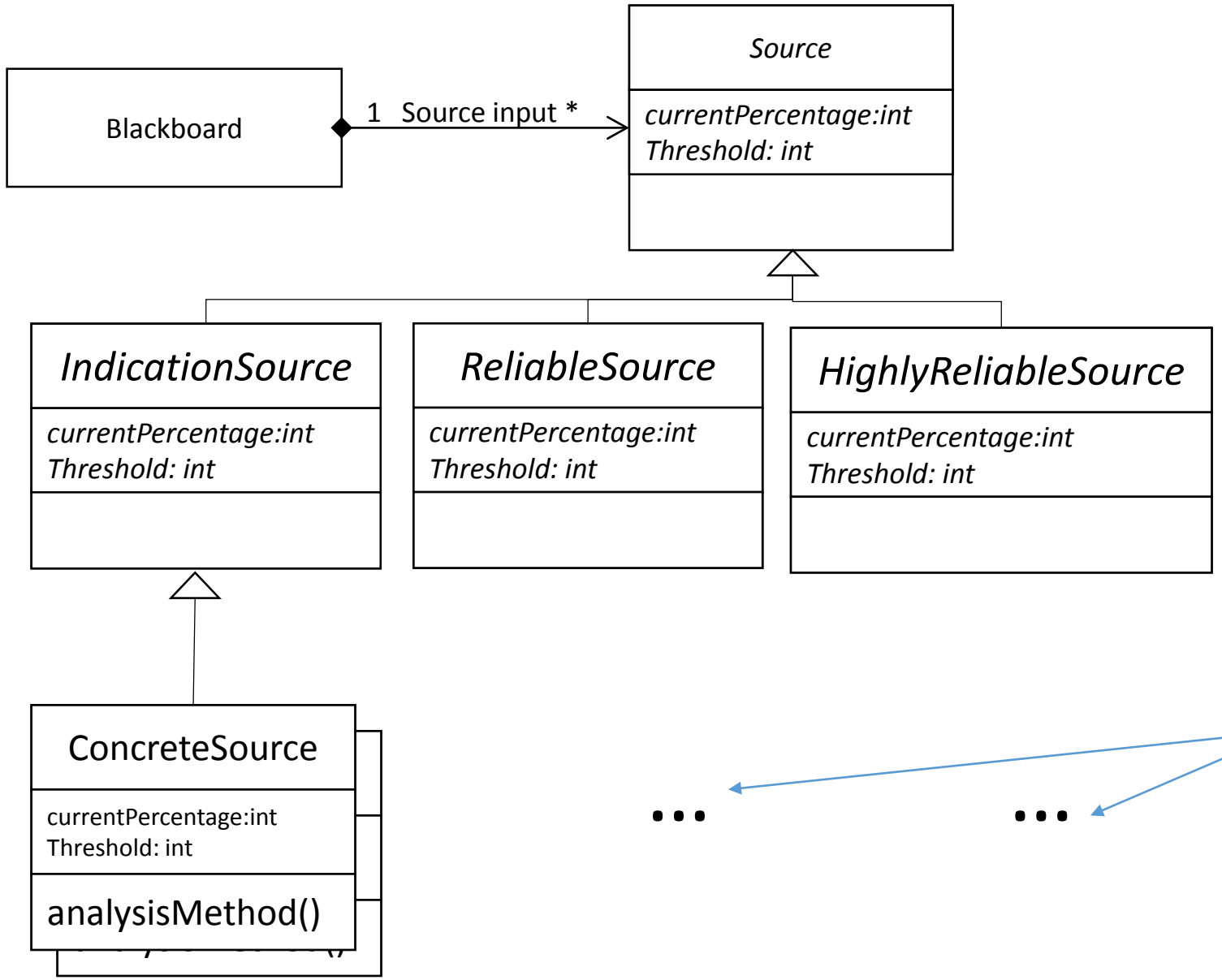
The rules for the alerts are:

- Green: two indicators pass threshold value, or a reliable source is 10 per cent units below its threshold.
- Yellow: two reliable sources pass the threshold, or one highly reliable source is 10 per cent units below threshold.
- Red: one highly reliable source passes the threshold and two reliable sources pass the threshold.
- If the conditions for the current alert level are changed, the level is lowered to the level where the conditions are fulfilled.

Task: Your task is now:

a) Show with an example *UML class diagram* how the *strategy design pattern* can be used to model the decision-making software, called Black-board, and its relation with the sources. Don’t forget that the diagram probably needs a small explaining text. If you have forgotten the design pattern, you can get some credits for a general UML class diagram. (10)

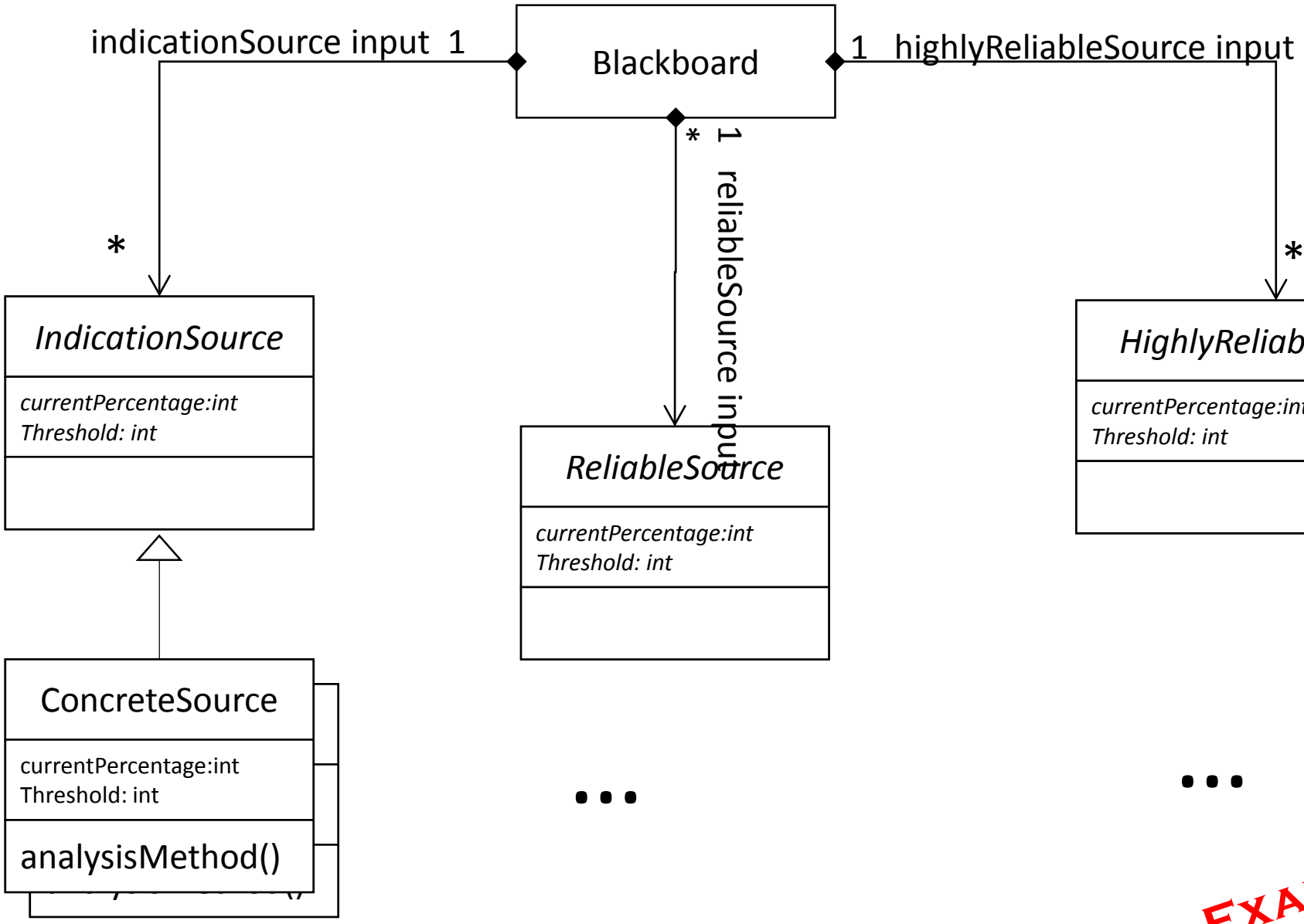
Standard solution



Text: The idea is to use the source as the strategy. To distinguish between different kinds of sources we have a layer of abstract classes under the Source. At the bottom there are concrete sources of varying kinds for each kind of source. As a result the Blackboard can use the same decision routines regardless of which concrete source that gives the inpu.

EXAMPLE ONLY

Practical solution



Comment: Here we use separate strategies for each kind of source. Otherwise, same as previous solution.

EXAMPLE ONLY

Grading criteria:

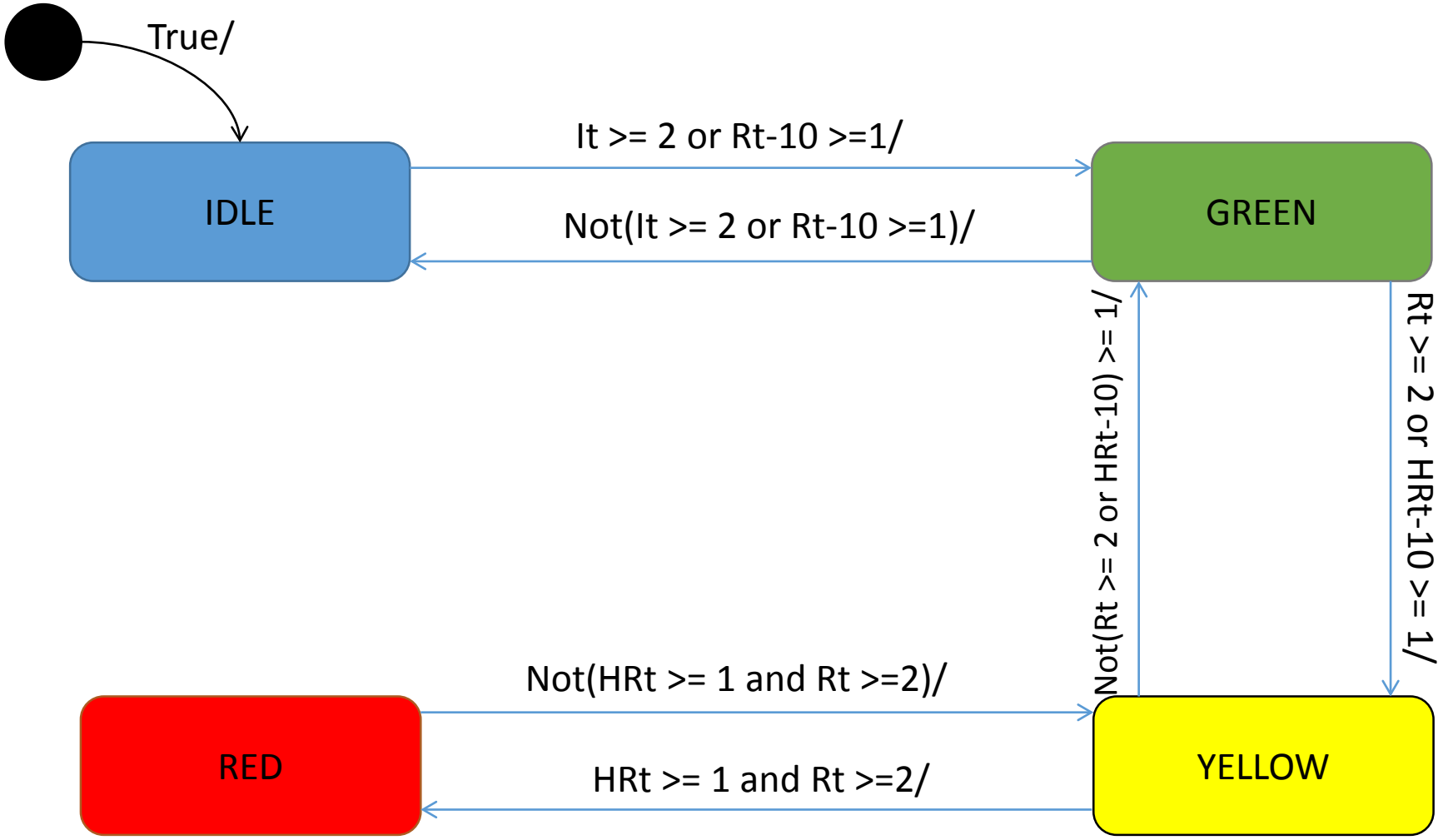
Strategy pattern: max 4 p
-Common things in interface, high level, or abstract class.
-Special things in concrete classes
-Polymorphism possible, things in subclass with same name
-Some kind of client

Representing sources: max 4 p
-Source classification
-Current percentage
-Threshold value
-It is possible to reach the properties of the sources from the client (introspection is OK)

If the UML–style is OK 2p extra.

Semantic problems with UML -1 credit each (for instance, using generalization up-side down)
1st syntactic problem -1 credit, then
2 syntactic problems -1 credit (for instance, using rounded corners)

b) Make a *UML State chart* for a class named `AlertLevel` that holds the current level of alert. The rules specified above shall be obeyed. Don’t forget to document assumptions made. (10)
Hint: you need a no-alert state.



Note: you are not allowed to use red color on the exam.

Variables:

It	= number of indicators with here risk >= threshold
Rt-10	= number of reliable sources with risk >= t – 10
Rt	= number of reliable sources with risk >= t
HRt-10	= number of highly rel. sources with risk >= t-10
HRt	= number of highly rel. sources with risk >= t

EXAMPLE ONLY

Grading criteria:

Correct behaviour and notation: 10p

The answer suggestion follows the rules, but the rules themselves are not complete.
We can accept that any change of the sources lowers the alert.

Underspecified, but correct solution 5 p

Forgotten return transitions max 7p

Semantic problems -1 each
1st syntactic problem -1 credit, then
2 syntactic problems -1

7. Make a comparative analysis of the three agile frameworks. SCRUM, Kanban, and eXtreme Programming (XP). For each of the frameworks write:

- A short description of what developers and other stakeholders can expect from the framework. 2-3 sentences
- A list of important concepts of the framework, for instance, roles, artefacts, meetings, and technical practices. For each concept, make a 1-2 sentence description and give a short motivation if you think this concept is unique to the framework or if there is a similar concept in another framework.

For full credits you describe at least two concepts per framework and 7 concepts in total. (20)

	SCRUM	Kanban	XP
Description of what to expect.	Product backlog items are broken down to sprint backlog items.	User stories are used as input.	User stories are used as input.
	There is a demo after each sprint.	The process is not necessarily iterative.	Demonstrations to customers very often (1 per week)
	Emphasis on time-boxed sprints with a self-organizing teams.	Emphasis on flow of stories, from arrival to shipping.	Emphasis on using practices.
Description of some important concepts.	The product owner is the voice of the stakeholders (1).	The Kanban board is used to visualize the work-flow. (3)	Pair programming, which means that two programmers work at the same computer, one using the keyboard, the other watching and commenting.(5)
	There is a daily SCRUM, a 15-minute stand-up meeting (2).	Each project as a limited amount of work items in progress (WIP).(4)	Test-driven development. Before any code is written, test-cases are written first.(5)
	The burn-down chart shows the progress of the team (2).		

(1) Can be used in Kanban, XP only mention customer
(2) Can be used in all frameworks
(3) Large similarity to SCRUM boards
(4) Unique
(5) Practices are not in other frameworks but can be combined with them if needed.

EXAMPLE ONLY

Grading criteria:

Two credits per good description of a framework.

Two credits per well-described concept with motivation.
We give maximally 14 credits for concepts.

Only two frameworks –2p.
Only one framework –4p

6. Make a template of headlines of a *project plan.* For each headline, add a small instruction within brackets of what information that goes under that headline.

Example:

1. Project description
1.1 Background to the project

< Major functions. Identify orderer and main users. Expected benefits from the perspective of orderer. Earlier solutions used. >

1.2 Constraints

<…>

(end example)

You shall assume that requirements specification and quality management are documented in separate documents. At least 10 headlines are needed for full credits. 1.1 in the example does not count; 1.2 in the example does if you write instructions. (10)

1 Project description
1.1 Background
<see problem description>
1.2 Constraints
<Limitations of resources, budget, calender time, or technical solution>
1.3 Project goal
<one or a few, well-formulated SMART goals>
1.4 Start and expected end date
<dates on format 20xx-xx-xx>

Comment: 10 new headlines

2 Project organization
2.1 Roles
<a list of roles needed. For each role, describe major responsibilities, authorities and preferred competence>
2.2 Knowledge/skill
<an inventory of the technical, managerial and social knowledge and skill needed. A gap-analysis between present and needed state.
2.3 Training
<a schedule and content of the planned training, to reach the needed competence>
2.4 Communication and reports
<description of meetings, minutes of meeting, status reports, time recording, and social media>

3 Time and resource plan
3.1 Milestones
<deadline and deliverables of the internal and external mile-stones>
3.2 Toll-gates
<date, purpose, participants, and decisions to be taken at the toll-gate meetings>
3.3 Deliverables
<list and short description of the code and documentation delivered, perhaps divided into different iterations>

EXAMPLE ONLY

Grading criteria: 1p per relevant suggestion of content

Other possible headlines

Suggested headlines

Project Description

- Background to the project
- Relevant constraints (budget etc.)
- Project Goal
- Start and expected end date.

Time and Resource Plan

- Milestones
- Tollgates
- Deliverables
- Activities
- Resources

Project Organization

- Roles
- Knowledge / skill
- Training
- Communication and reports

Risk Management

- Risks, Probability, and Impact
- Mitigation and Contingency plan

Area 5: Software Quality

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

A. When you have a requirement on *reliability* you often express the requirement in terms of *failure intensity* since this is more straight-forward to measure instead of estimating a probability.
B. The number of hours that your team puts on testing can be said to be a measurement of the structure of your code.
C. McCabe suggested in his article about *cyclomatic complexity* that all code modules shall have a *cyclomatic* complexity of 10 or higher. Otherwise there is a risk that people make errors of omission by forgetting conditions.
D. *Efficiency* of interactive software can be measured by logging the time it takes for representative users to complete a specified task.

Answer:
A
D

Comment:
A, see slides of requirements
D, see list of usability metrics

5 b) Two questions on Software Reviews: i) What is the difference between an *inspection* and an *audit*? ii) Apart from finding faults, the *author* of the inspected document has two responsibilities in the inspection process. Describe these two shortly. (4)

Answer: i)
The goal of an inspection is to find defects, and audit ensures conformance to standards or accounting.

Grading criteria: 2p for this or a similar comparison.
1p if either the inspection or audit is described well, but no comparison

ii) The author is responsible for meeting the entry criteria of the inspection process. The author is responsible for carrying out the changes mandated by the inspection record.

Grading criteria: 1p per good description of the author role, maximally 2 credits

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 functions that "everyone" in their company know are sometimes needed. Your employees are really appreciating your personal feed-back, but it comes sporadically and is based on your feelings only. As a complement they would like to have more regular and objective feed-back.

Task: Your task is now to find the two most relevant *CMMI process areas* that can help you to take care of the criticism. Write down their names and a short motivation. (4)

Answer:
Closest areas:

Requirements Development, RD. Some of the goals are to elicit customer needs, and transform them into requirements. Should take care of missing requirements.

Process and Product Quality Assurance, PPQA. Some of the goals are to Objectively evaluate Process and Product performance. That is what employees are asking for.

EXAMPLE ONLY

Grading criteria:
2 p for each of these two areas with a good motivation. We do accept other areas, if the motivation is clear about why the person in the scenario is helped.

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

A. If you want to start integration testing with high-level user interface functions you will need several *drivers* to emulate lower level functions.
B. *Bottom-up integration testing* can help you to discover performance problems of lower level components early in the testing process.
C. With *big-bang integration testing* there is a high risk that localizing the fault of a failure will take time compared to other strategies.
D. *Sandwich strategies* start integrating and testing modules at a *target level*, continue with a *bottom-up method* to the highest level, and then with a *top-down* method to integrate the lowest level modules.

Answer:
B
C

Comment: B, see slides on testing
C, see slides on testing
Or table in the book.

4 b) *Scenario:* With a SAS economy class ticket you can pay EUR 80 to check in second bag. The maximum weight is 23 kg and the sum of height, width and depth can be 158 cm or lower. There is an app developed where users can enter weight, height, width, and depth of their bag. If the baggage is allowed the app calls a page for making a reservation. If the baggage is not allowed the app displays a text in red "baggage is oversized".

Task: Your task is to identify *equivalence classes* of the input parameters to the app. You shall also make a *table of test-cases* that test all equivalence classes. (4)
Hint: You may assume that the input pad for the app only allows entry of digits.

Assumption: I assume that it is not possible to enter negative numbers, since only digits were allowed.

Equivalence classes:
EC1: Weight <= 23 kg
EC2: Weight > 23 kg

EC3: 0 <= Height+width+depth <= 158 cm
EC4: Height+width+depth > 158 cm

You can work on height, weight and depth independently without being wrong but then you get many classes and loose time.

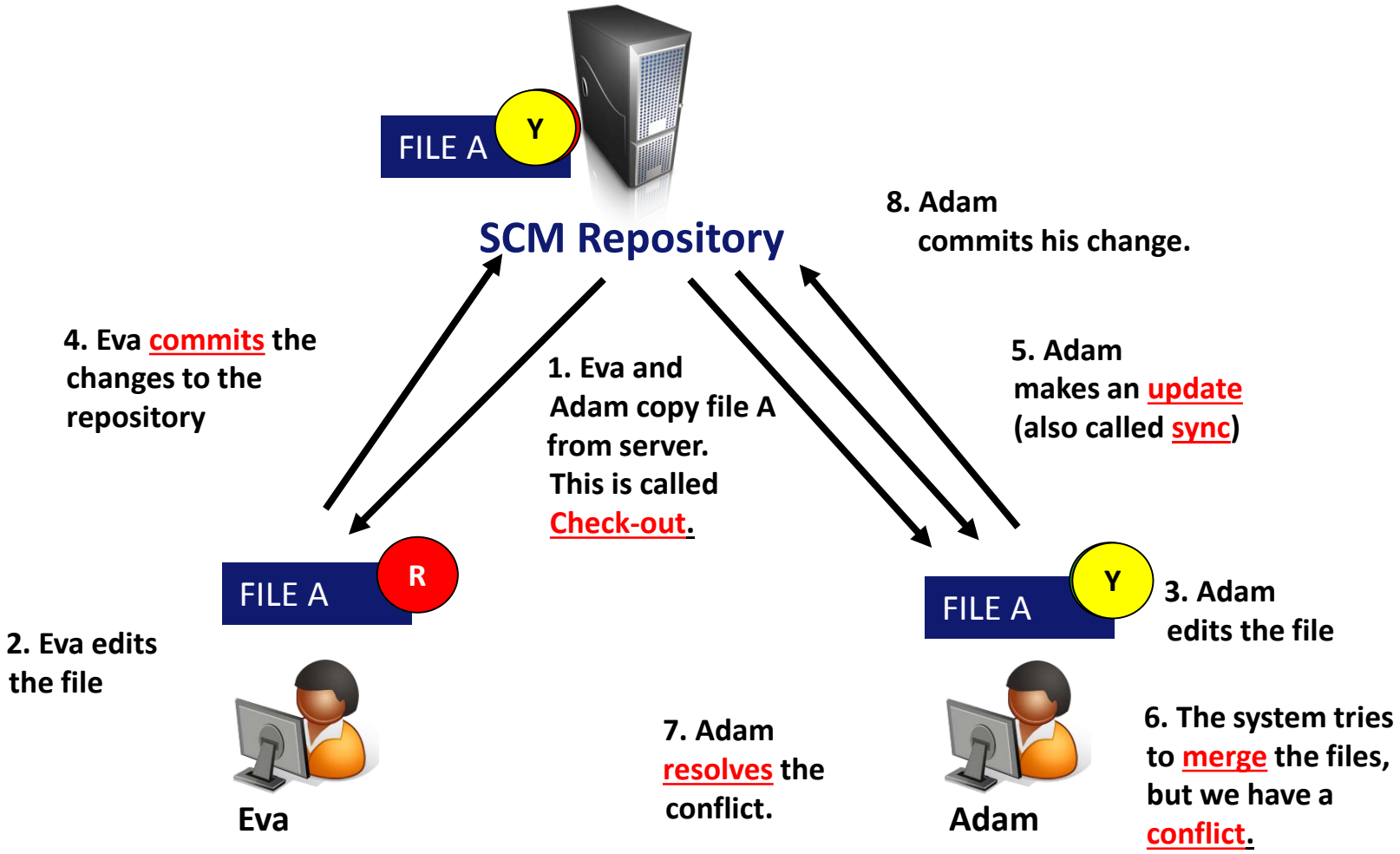
Test table

Id	Weight	Height	Width	Depth	Action	EC
1	20	50	40	30	reservation	1
2	25	50	50	40	“oversized”	2
3	21	50	50	50	reservation	3
4	22	60	60	60	“oversized”	4

EXAMPLE ONLY

4c) *Scenario:* Adam is working on the file `TestCase.java`. He gets his copy from the *code repository* and edits the method `getInput`. While he is working, Eva also gets her copy of `TestCase.java`, edits the same method, and writes back the new code to the repository before Adam is finished. Poor Adam now has two revisions of `TestCase.java`: His own and the one in the repository. Adam and Eva are using a *Centralized Modify-Merge* version handling tool, for instance, Subversion.

Task: Your task is to describe a workflow for Adam that starts with him getting the copy of the file and ends with both his and Eva's changes of the testCase.java stored in the repository. Use the correct commands instead of the everyday language in the scenario. (4)



Grading criteria:

It is OK to use other SCM tools than SVN provided that the tool is named and the result is correct.

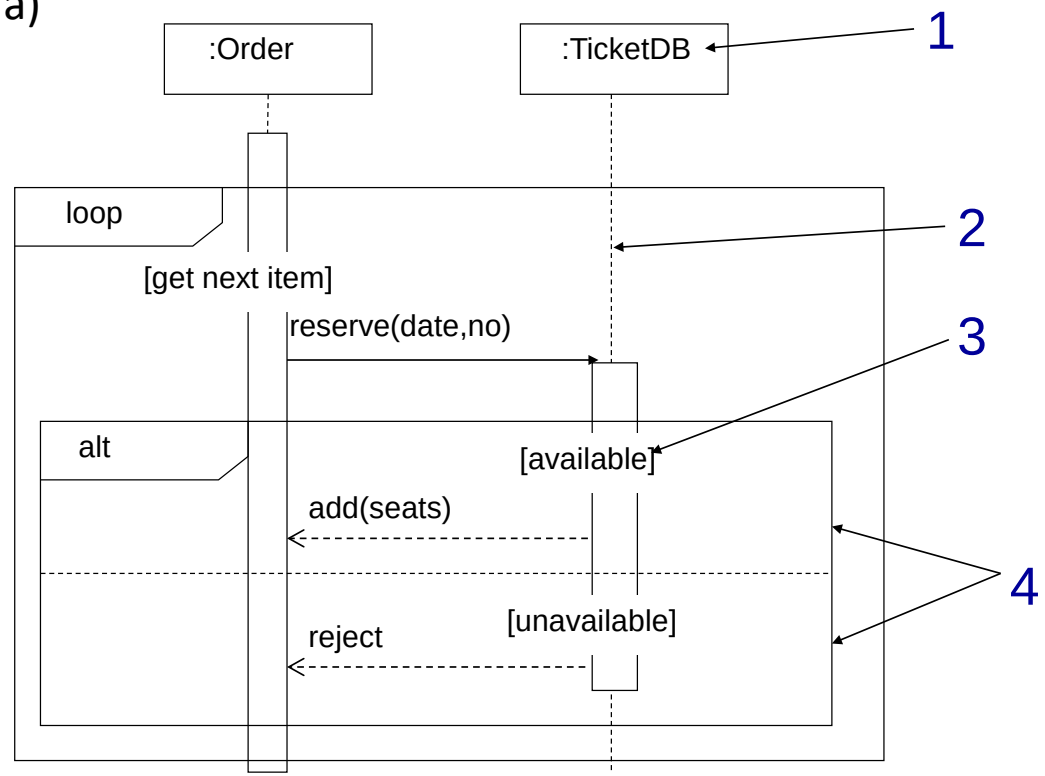
Wrong command, but intention clear -1

Wrong end result -2

Missed name tool, if other than SVN -1

EXAMPLE ONLY

3 a)



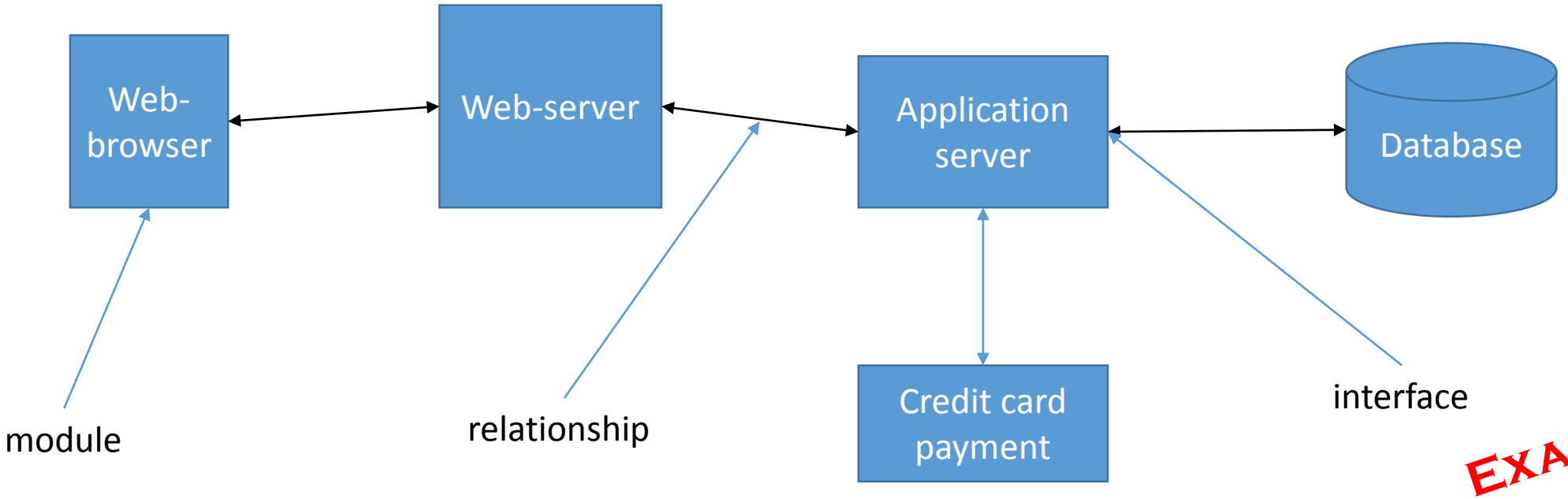
Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. Arrow number 1 is pointing to a *class*.
B. Arrow number 2 is pointing to a *dependency association*.
C. Arrow number 3 is pointing to a *guard condition*, which means that this region is entered only if the guard condition is true.
D. Arrows labelled with number 4 is pointing to two *alternate branches*.

Answer
C
D

Comment See slides of lecture 7

3 b) Draw an architecture of a web-shop with a line-and-box diagram. Point out the following elements of the diagram: *module*, *relationship*, and *interface*. Which architectural view is your diagram showing? Don't forget to motivate your answer to the last question. (4)
Hint: Don't spend too much time in making the architecture perfect, 5-10 elements will do.



This is an execution view since it shows the flow of data and control between modules when the software is used.

Grading criteria:

1 credit per correct identification of module, relationship, and interface
1 credit per motivation of the correct architectural view
If UML class diagrams have been used it is OK, but they still have to point out where you find the module, relationship, and interface
If they only describe a part of the webshop, they need to explain that in a sentence, otherwise -1p

3 c) Define the two concepts *pipe* and *filter*. Give one advantage and one disadvantage of the *pipe-and-filter architectural style*. (4)

Answer:
Pipe – channel of data
Filter – the processing unit
Advantage: Modular building blocks gives high flexibility
Disadvantage: Redundancy, for instance, input validation

Grading criteria:

Pipe, 1p
Filter, 1p
Advantage: 1p
Disadvantage 1p

EXAMPLE ONLY

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

A. Linköping University is a good example of a *project*, since we have a well elaborated *strategy*.
B. According to Frederick Brooks it is advisable to have a risk list of about $\frac{n(n-1)}{2}$ *risks* in a team with *n* members.
C. The dependent project parameters of a project are: *Resources*, *Features*, *Calendar time*, and *Quality*.
D. In a *Gantt chart* a *task* and its *predecessor* do not need to belong to the same *phase*.

Answer:
C
D

Comment: slides on lecture 4

2b) Describe two advantages of working with a *Waterfall life-cycle model* and two advantages of working with *Iterative development methods*. (4)

Answer:
The waterfall model is easy to manage.
The waterfall model is suitable for fixed-price contracts.
An iterative model allows for more frequent feed-back from the customer.
An iterative model evens out the workload during the project.

Grading criteria:
1p per sensible advantage

EXAMPLE ONLY

2c) Describe the four tasks performed in the *risk management process*. (4)
Hint: don't mix this with the *strategies*: avoidance, transfer, and acceptance.

Answer:

Risk identification: find potential risks, for instance with brainstorming
Risk analysis: calculate probability, consequences, and risk magnitude indicator
Risk planning: determine the strategies to be used for handling the risks, see text in question.
Risk monitoring: Periodically check if any information of risks has been changed?

Grading criteria:
1p per sensible description

EXAMPLE ONLY

1 a) Consider the following requirements of an interactive project-planning tool:

R1: The user shall be able to add a new task to the project.
R2: The critical path of the Gantt chart shall be displayed in red.
R3: The colorizing according to **R2** shall be done maximally 2.0 seconds counted from the last keystroke creating or modifying the critical path.
R4: When a change in the critical path is made, the tasks that no longer are on the critical path shall be colorized in their default colour after the new critical path has become red.
R5: When a change in the critical path is made, the tasks that no longer are on the critical path shall be colorized in their default colour before the new critical path has become red.

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

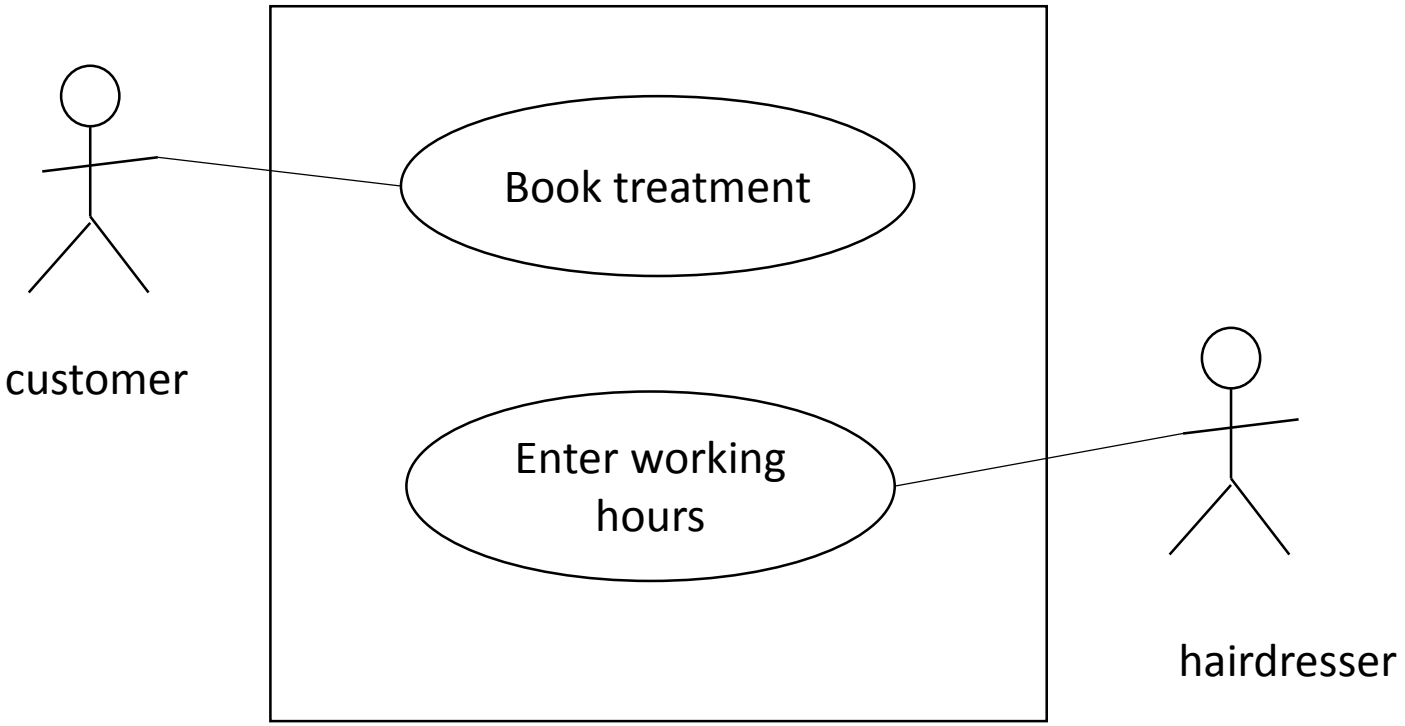
A. R1 is a *functional requirement*
B. R2 is a *non-functional requirement*
C. R3 is a *non-functional requirement*
D. R4 and R5 are *consistent*

Answer:
A
C

Comment: refer to definitions of RE-lecture

1 b) *Scenario:* Your hairdresser salon has started a web-site for frequent customers. If you have signed up for the free membership you can login and book treatments. You can search for different employees, date, or time. The hairdressers use the system for planning, for instance, entering working hours and serving drop-in customers. The customer can erase bookings if more than 24 hours remain before the start of the treatment. The hairdressers can edit anything, and can also edit advertisements from their vendors.

Task: Now, create a *use-case diagram* of the web-site consisting of two different actors and two different use-cases. Don't forget the use case texts. Only logging in and logging out are basic functions, not to be considered as use-cases. (4



Comment:
1 diagram
2 use-cases
2 actors

EXAMPLE ONLY

Use-case descriptions:

Book treatment:

The customer logs in to the system and selects type of treatment. The customer is shown a list of available time-slots and personnel. The customer selects one of the available items and presses OK. The customer logs out from the system.

Enter working hours:

The hairdresser logs in to the system and selects the change working hour – tab. The hairdresser selects the week number and is shown a graphical schedule. The user marks the available working hours in the schedule and presses SAVE. The hairdresser logs out from the system.

Grading criteria:
2p per good use-case, with at least 3 sentences per use-case description
Actors are roles, not user1 and user2, if so -1p
An actor can be a sub-system
Two single use-case diagrams -2p.
Only a correct diagram, max 1p.
Diagram missing -1.
Use-case name: a verb phrase, if not -1p per use-case.

EXAMPLE ONLY

1 c) Describe two techniques for performing *requirements elicitation*. Describe two different goals for *requirements analysis*. (4)
Hint: by "describe" we mean that a student like you but who has not taken the course shall be able to understand what you wrote.

Answer:
Requirements elicitation techniques:
Interviews: Stakeholder representatives are interviewed about their needs and expectations of the system.
Prototyping: Small prototypes of the system or parts of the system are developed, stakeholder representatives are asked to try the prototypes and are asked for missing functions or superfluous functions.

Requirements analysis goals:
Find conflicting requirements with inspections or making models of the system.
Make a conceptual model, for instance a UML class diagram, of all concrete objects and properties handled by the system.

Grading criteria:
1 p per sensible description of an item. Max 2 for RE and mas 2 for RA.

EXAMPLE ONLY

kristian.sandahl
@liu.se

Literature (Recommended Reading)

References and information that are fundamental for the exam (1/3)

SE General

- Pfleeger and Atlee: Chapter 1
- **Geppert, L. (2004). Lost Radio Contact Leaves Pilots on Their Own, IEEE Spectrum 41(11), pp 16-17**
- **Charette, R.N. (2005). Why software fails, IEEE Spectrum 42(9), pp 42-49**

Area 1: Requirements

- Pfleeger and Atlee: Chapter 4.
- **IEEE-Std-830-1998 (high-level)** (Navigate from within the LiU-domain: LiU-home page, Library, Databases. Search for IEEE Xplore, enter and Browse standards, search for "830")

Area 2: Planning and Processes

- Pfleeger and Atlee: Chapter 2.
- Pfleeger and Atlee: Chapter 3.
- **Manifesto for Agile Software Development**
- **Scrum introduction YouTube video**
- **Official Scrum guide**
- **Kanban vs Scrum**
- **OpenUP resource web (high-level)**
- **Extreme programming website (high-level)**

Area 3: Design and Architecture

- Pfleeger and Atlee: Chapter 5.
- Pfleeger and Atlee: Chapter 6.
- **Design pattern (high-level)**
- **SOA and Amazon (high-level)**

kristian.sandahl
@liu.se

Literature (Recommended Reading)

References and information that are fundamental for the exam (2/3)

Area 4: Testing and SCM

- Pfleeger and Atlee: Chapter 8.
- Pfleeger and Atlee: Chapter 9.
- **Selenium webpage (high-level)**
- **Git- SVN Crash Course**
- **Free online book on subversion (high-level)**
- **The Git Community Book (high-level)**
- **Continuous Integration according to Martin Fowler**

Area 5: Software Quality

- Pfleeger and Atlee: Chapter 4.9, 8.3, 13.2
- **IEEE Standard for Software Reviews and Audits 1028-2008 (high-level)**, esp Section 6 Inspections.
- Pfleeger and Atlee: 6.7, 8.1, 8.8, 9.3, 9.9, 11.4
- Pfleeger and Atlee: 12 (Only very general questions about reuse can appear in a written exam)
- 13 (13.1 is outside the scope of this course. If it looks interesting, go for TDDD30)
- **Short intro to TQM**
- **CMMI-DEV 1.3** Ch 1-3: Only the staged representation.
  Read purpose and introductory notes for the areas CM, OPD, PMC, PP, PPQA, RD, REQM, RSKM, TS, VAL, and VER.

Complementary Literature (Further Reading)

kristian.sandahl
@liu.se

These references are not needed for passing the exam.

Area 1: Requirements

- Philip A. Laplante. **Requirements Engineering for Software and Systems**, CRC Press, 2009, ISBN-13 978-1-4200-6467-4

Area 2: Planning and Processes

- Philippe B. Kruchten. **The Rational Unified Process: An Introduction** 3rd Edition, ISBN 0321197704, Addison-Wesley Professional, 2003
- Kent Beck and Cynthia Andres. **Extreme Programming Explained: Embrace Change**. ISBN 0321278658, Addison-Wesley Professional, 2004
- Ken Schwaber and Mike Beedle. **Agile Software Development with Scrum**. ISBN 0130676349, Prentice Hall, 2001

Area 3: Design and Architecture

- Martin Fowler, **UML Distilled: A Brief Guide to the Standard Object Modeling Language**, Third Edition, Addison-Wesley Professional, 2003, ISBN: 0321193687.
- Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. **Design Patterns: Elements of Reusable Object-Oriented Software**, Addison-Wesley Professional, 1994, ISBN: 0201633612
- Len Bass, Paul Clements, and Rick Kazman, **Software Architecture in Practice, Second Edition**, ISBN 0321154959, Pearson Eduction, Inc, Boston, USA, 2003
- Frank Buschmann et. al. **Pattern-Oriented Software Architecture Volume 1: A system of patterns**, ISBN 9780471958697, John Wiley & Sons Ltd, England, 1996

Complementary Literature (Further Reading)

These references are not needed for passing the exam.

Area 4: Testing and SCM

- Lee Copeland. **A Practitioner's Guide to Software Test Design**, ISBN 978-1580537919, Artech House, 2004

Area 5: Software Quality

- Norman E. Fenton and Shari Lawrence Pfleeger. **Software Metrics: A Rigorous and Practical Approach**, 2nd edition, ISBN-13: 978-0534954253, PWS Pub. Co., Boston, MA, USA, 1998
- Rini van Solingen and Egon Berghout. **The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software Development**. ISBN 007-709553-7, McGraw Hill. 1999[**link**]
- Daniel Galin. **Software Quality Assurance: From Theory to Implementation** ISBN-13: 978-0201709452, Addison Wesley, 2003

Questions and Answers Session

kristian.sandahl
@liu.se

During your study many questions might arise. Collect your questions and come to this occasion.

Friday, October 24, 13.15-15.00
(Alan Turing, E-house)

Thursday, October 30, 12.30-14.00
(John von Neumann, B-house

Examination - content? to pass?

kristian.sandahl
@liu.se

To pass the exam (alternatives)

1. a) at least 4 credits in all areas in fundamentals **and**
   b) at least 50 credits in total
2. a) at least 4 credits in at least 4 areas **and**
   b) at least 60 credits in total

Part I: Fundamentals

- Requirements
- Planning and Processes
- Design and Architecture
- Testing and SCM
- Software Quality

10 credits per area. Max 50 credits.

Part II: Advanced

50 credits, distributed over 2-5 questions.

- argue, compare, and analyze different concepts and techniques.
- construct and/or design solutions to larger problem.
- explain more advanced and specific topics.

Examination - grades?

kristian.sandahl
@liu.se

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

Exam Aids

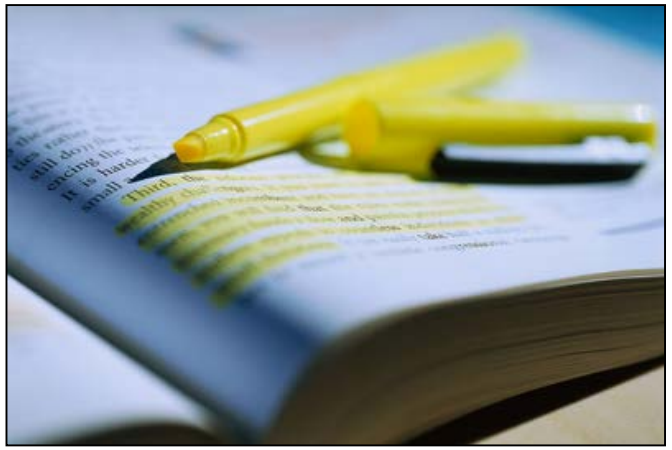
kristian.sandahl
@liu.se

Allowed aids

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

Explicitly forbidden aids

- Textbook
- Machine-written pages
- Photocopied pages
- Pages of other format than A4
- Electronic equipment



The End...

kristian.sandahl
@liu.se

Thanks for listening!

me, when all students pass