Lecture "Software Engineering SoSe 2023

Free University of Berlin, Institute of Computer Science, Software Engineering Group Lutz Prechelt, Linus Ververs, Oskar Besler, Tom-Hendrik Lübke, Nina Matthias

Exercise sheet 3 **UML Models and Diagrams** for 2022-05-08

Work on all tasks in such a way that you are **well prepared for** the exercises, i.e. that you can **show**, **explain** and **discuss** your solutions in the exercise group in an appropriate way. Always indicate your **sources** used.

Task 3-1: UML elements

Learning objective: Know, read and be able to use important UML model elements.

Consider the following UML model elements:

Classes and *objects*, *attributes* and *operations*, *associations, multiplicity*, ge- *neralization*, and *composition* and *aggregation.*

For *each* element, answer the following questions:

a) What is the purpose or semantics of the element? What are the relationships to the other elements mentioned?

b) How is the element notated graphically or textually? Which are the necessary notation components and which are the optional ones? Note: Most model elements have a lot of optional notation components!

You may refer to the UML specification itself to complete this task (see Task **3-3**). However, you may also use other sources that are easier to read. As always, remember to cite them as well.

Task 3-2: UML class diagrams

Learning objective: To be able to completely and precisely capture static facts expressed in natural language as a class model and represent them as a UML class diagram.

Model the following situation as completely as possible and represent the result as a UML class diagram. *Complete* means that all statements from the description text have been modeled. Also make sure that the class diagram does not say *more than* the text.

In the dangerous Troll Forest there are many caves. Each cave has a certain number of entrances and a certain depth. Each of these caves serves as a dwelling for one or more trolls. There are two different types of trolls in the Troll Forest: Stone trolls and Mountain trolls. All trolls live in caves and are distinguishable by their nose length and size. Stone trolls are made of a specific type of rock. Mountain trolls are not made of stone, but they have scales in the most iridescent colors. Every troll has a club, of course, with which he can strike and which is his property. Great trolls even have several such clubs. The clubs are either made of oak or spruce.

Task 3-3: UML metamodel

Learning Goal: Be able to navigate structured technical documents (don't be discouraged; the task itself is not difficult); understand difference between diagram and model, and multiple levels of abstraction.

Background: Concrete UML *diagrams* (e.g., your result from Task **3-2**) consist of graphical elements, each of which represents elements of the underlying *UML* model. The model elements available in principle (see e.g. Task **3-1**) are specified in the so-called UML *metamodel* as metaclasses (= classes in the metamodel).

Each of the chapters 7 to 20 of the UML specification highlights a UML topic area. *Con- crete metaclasses* each describe a variety of UML model elements that occur in actual UML models and diagrams (such as *classes*, *associations*, or *gene- ralizations*). *Abstract metaclasses,* on the other hand, are only technical aids in the UML metamodel and are not used directly in actual UML models and diagrams.

The automatically generated "Classifier Descriptions" sections of each of Chapters 7 through 20 list all metaclasses: Concrete metaclasses have the suffix "[Class]", abstract metaclasses are marked with "[Abstract Class]".

Task: Download the specification of the UML meta model in the current version 2.5.1 to work on the subtasks: http://www.omg.org/spec/UML/2.5.1/ PDF/.

a) The following UML model elements are mainly used in *class diagrams.* Find the corresponding (concrete) metaclasses in the specification and specify the name of the metaclass and the relevant section numbers in each case.

Note: Chapters 9-11 should be particularly helpful here.

> *Example:* Normal classes from a UML model are graphically represented in diagrams as boxes (as in Task **3-2**). Technically, they are copies of the `Class` type from the UML metamodel described in Section 11.4 and summarized in 11.8.3.

UML model element	Metaclass name	Relevant sections
Class Object (copy of a class) Generalization Attribute Operation Composition & Aggregation Interface Implementation (an interface) Association	`Class`	11.4 and 11.8.3

b) The following UML model elements can be found in other diagram types. Find out the associated metaclasses here as well.

Note: Chapters 13 and 14 should be particularly helpful here.

Model element	Diagram type	Metaclass	Relevant sections
Method call	Sequence diagram		
Internal transition	State diagram		
External transition	State diagram		

c) The nodes in activity diagrams can be either *atomic actions* or *composite activities*. In UML, a number of *actions* are predefined.

Note: Chapter 16 should be particularly helpful here.

For each of three of these actions, find out the metaclass and describe in your own words what each of these actions does.

Task 3-4Æ: UML sequence diagrams

a) What is a sequence diagram and what is its purpose in the context of software development?

b) What model elements are used in both class diagrams and sequence diagrams?

c) What is shown in the vertical?

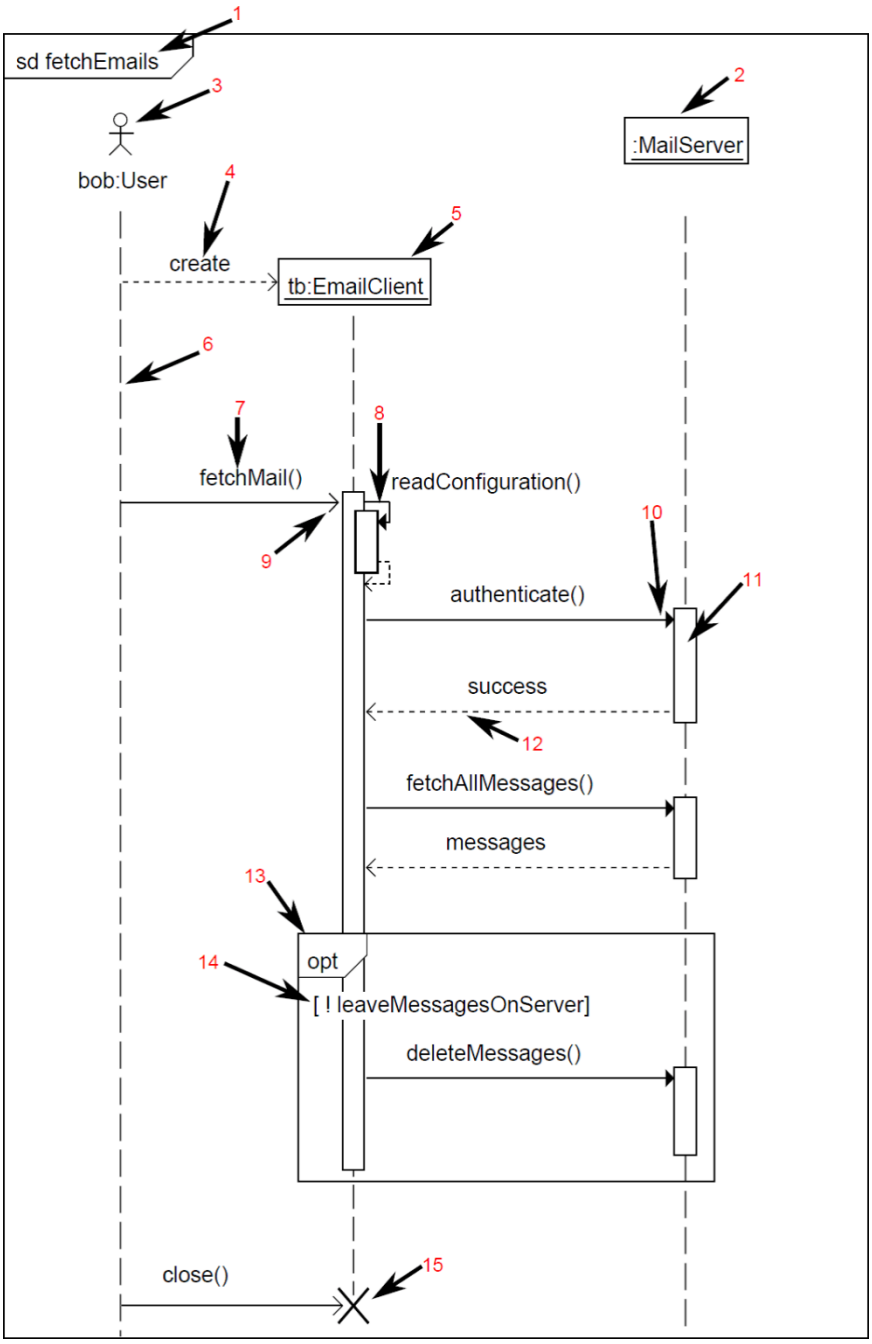
d) How is a call to a method of an object represented?

e) Is a message the same as a method call?

f) How do the objects "approach each other", i.e. what is the connection between the sender object and the receiver object through which a message is sent?

g) How can you clearly see that sequence diagrams always represent only one of many possible sequences?

h) For each of the 15 diagram elements marked with arrows, enter the name or what they stand for. Example: *1 ← Diagram name*


sd fetchEmails
1
2
:MailServer
3
bob:User
4
create
5
tb:EmailClient
6
7
fetchMail()
8
readConfiguration()
9
10
11
authenticate()
success
12
fetchAllMessages()
messages
13
opt
14
[! leaveMessagesOnServer]
deleteMessages()
close()
15