Assignment 6 - Communication Diagram and Implementation Class model

Version: April 12, 2024

Objectives

- Derive Communication diagram from System Class model and Operational Pattern
- Derive Implementation Class model
- Implement what has been designed

Prerequisites

1. Lecture on Communication diagram and Implementation diagram
2. Sample solution for Operational Pattern from assignment 5, but you will only need the createAppointment one which is already available with assignment 4 and also linked with this current assignment.

General Info

In this assignment, you will create a Communication diagram for your system operation *createAppointment* (the same one you created an OperationalPattern for). In Task 2, you will derive a partial Implementation Class Model from this Communication Diagram, and in Task 3, you will implement what you have.

If you are struggling with the Communication Diagram, you might want to think about your implementation first. Of course, this goes a little against the process since your implementation should be based on the design, but I think it might help you understand what is happening and what objects need to communicate with each other.

The diagrams should be completed in Astah (use one Astah file).

Task 1: Communication diagram (25 points)

Create a communication diagram for the system operation *createAppointment* for the tutor. Make sure you use the correct syntax and that the diagram is based on the sample solution of the System Class model (please use the version provided with this assignment), Operational Pattern, and Sequence Diagram you were given (will also be provided as a document with this assignment when everyone has submitted assignment 5 – you can start earlier based on your solution but be aware that if your solution is not

good, it might make things more complicated or wrong).

Task 2: Implementation Class model (15 points)

Based on the Communication diagram you created in Task 1, create one Implementation class model that includes all methods, attributes, and classes used in the Communication diagram. Do not include methods or classes that are not used in your Communication diagram (this is just a partial diagram). You can include attributes, though, so you do not have to delete them from the classes. Also, make sure to include types and access specifiers for your attributes and methods.

Task 3: Implementation (10 points)

Using your implementation from Assignment 3, change your code implementation based on your Communication Diagram. All of your attributes should now be "as private as possible." Thus, you need methods to access them. You might also want to include other methods that make your life easier. Your implementation should exactly reflect what your Communication diagram does (Use the exact same names for your methods, attributes, etc.). If you already had your attributes private and used access methods in Assignment 5, you might not have to change your implementation at all and will only create the Communication diagram for it.

If you are doing this first, then you should build your Communication diagram on the exact flow of events happening here for your *createAppointment* method (we will only grade it if it fits together with your Communication Diagram). It hopefully gives you insight into how your Diagram should look like.

Make sure your *createAppointment* method is implemented in the appropriate control class.

Again, make sure there are no compilation errors in your implementation so we can run it.

Submission:

Submit the following on Canvas

- one PDF named comm_asurite.pdf containing your diagrams from task 1 and 2 (Communication and Implementation class model).
- one zip file impl_asurite.zip containing your code implementation.