Assignment 3 - Activity Diagram, UI Design, System Class model

Version: April 3, 2025

Objectives

- Create an Activity Diagram
- Apply the User-centered design process from ISO 13407 'in the small' to elicit the value of iteration
- Apply techniques for persona development
- Identify Users, Goals, and Tasks
- Develop proficiency in low-fidelity prototyping

The solution for this assignment should be based on the sample solution of assignment 2 which will be provided when everyone has submitted.

Task 1: Activity Diagram (15 points)

Create an Activity diagram for the Use Case *ManageAppointment* from the Student point of view. Remember that an Activity diagram sets the different Scenarios from that Use Case and Actor in relation. It includes all scenarios this actor can actively use; check the given Use Case diagram to see what scenarios are included in this Use Case.

Check the provided sample solution for the Sequence diagrams given to you.

If you do not have a Sequence diagram for a scenario, then you should use an abstract action that you should not specify in detail. (See lecture slides example *Create Purchase Order action*).

Make sure the diagram is consistent with your Sequence Diagram.

Task 2: UI Design (15 points)

For this task, you will 1) develop personas representing the categories of stakeholders and 2) generate "low fidelity" sketches of the UIs. Finally, you will do so in the context of the UCD process (just to emphasize the process again!).

You should not spend too much time on this; I want to see a rough prototype. I am giving you estimates on how long I think you should spend on these tasks so you have an understanding of what is asked.

Your process for doing this task should follow the UCD process "in the small" (**only doing some steps, see below**). Remember the ISO 9241-210 process from the video. We will start with Step 2 of that process.

Namely:

Step 2: Understand the context of use

Create simple "personas" for **each** type of actor that can use the system (create one persona per actor – see Use Case diagram for the actors). These can be simple in that you do not have to write multiple paragraphs, just a few sentences. Use the "Rhonda Wilson" example from your reading as a guide as to the length and detail. Be sure to identify goals and represent each category of user you may have.

I added an example on the assignment page and the tool used to create this example. This was created in the Summer of 2020 by students, and the personas turned out really well.

approx: 20min

Step 3: Produce Design Prototype as "sketch(es)" for the Student

Start your prototype on paper (or in a tool right away) for the **Student** actor. The prototype is a limited prototype that only needs to include the scenarios that you have in your Activity Diagram. If a scenario does not have an SD, then you only show where the user can reach this scenario, but you do not include details about the scenario.

Make sure you have a good workflow and appropriate UI design. The workflow should be the same as in your Activity Diagram from Task 1. It should include all input fields, buttons, errors, and success messages. You can create a web app, phone app, or whatever you feel fits best.

When you are happy with your design, create it in Pencil or any other tool for UI design.

Pencil is an open-source UX prototyping tool available from:

http://pencil.evolus.vn/en-US/Home.aspx.

approx: 25min

Step 5: Test Design Prototype

Have individuals review your sketch(es) and provide feedback as to whether they are Â“directionally correct".

The best would be if you can find another student from this course to review your design, e.g., one of your team members, someone you know from the Course Slack, a random student, or the 2-3 students that are the next in the sequence of names after your name on the course Slack. Get at least one feedback (the feedback should not take the reviewers longer than 5-10 min - it is just a quick feedback). Please do NOT just post it publicly, but DM other students.

As it is an initial draft, your evaluations should be high-level. â€¢ Are you on the right track? Take the feedback notes and put them in your submission document.

In case you do not find anyone from the course to do the review: Write a brief explanation of why you could not have your design reviewed by a fellow student from the course. Let an external person look over your design (spouse, friend, parents, etc.) and take notes.

Iterate

Then iterate! Based on the feedback from the outside individuals, go back to step 2 and revisit your personas and requirements, and then revise your prototype in step 4.

approx: 15min

Under task 2, include your draft, feedback, and final version in your PDF document. (Do not submit the Pencil files; instead, include them as images in your document.)

Task 3: System Class model (15 points)

Base this diagram on the sample solution of the Domain Class model provided.

Create another Class diagram. I call this System class model (it is, however, just a regular Class diagram with a specific level of abstraction).

The easiest is to clone the Domain class model and start to change this diagram into the System Class model (watch the Astah video I provided in weeks 1 and 2).

The System Class Model should include everything that was explained in the lecture slides (and video) about the System Class Models (use the step-by-step guide). Make sure the level of abstraction is correct.

Task 4: Implementation (10 points)

We want to do a basic implementation again.

We want to create the exact same functionality as in assignment 2.

The only thing that should change is your structure. Meaning: in the previous version your Main included all the objects and your one method. Now, the objects should be moved to the control classes and your Main class should have instances of your control classes. The control classes should now have the "lists" of tutors, appointments, etc., in the same way you set it up in your System Class model. The method that you had in your Main in Assign 2 should now be moved into the appropriate control class. This is the control that is responsible for this scenario (check the UC model).

You do not have to implement Boundary classes; the Boundaries would be UIs. The UIs would be connected to controls and call the scenario based on decisions your actor makes. We skip this UI part, and your Main method calls the scenario methods from the controls directly. For example, in your Main, you call appControl.cancelAppt().

In theory, you would implement all scenarios in your control classes now (based on where the scenario is located in your Use Case). Here, I only want you to implement the method "Cancel Appointment" again, which you can take from the sample solution implementation. This method should be a public method in one of your control classes (the one responsible for this scenario). Have good outputs for your method and especially test the "Cancel Appointment" method with all its alternatives. When we run your Main we want to see the output of your method. The method does the exact same thing as in assignment 2. Check the sample solution; you can use the sample implementation as a base for your implementation here.

Create an object view for your implementation (which describes the system state), which includes your Control Objects now.

Important: Make sure the implementation is based on your diagrams. If it is not consistent with the diagrams, we will not grade it. Make sure there are no compile errors so we can run your implementation. Also, make sure that you have a Main that shows your functionality.

Submission Part

You will need to submit

- One PDF document containing
  - Your Activity diagram
  - Everything from Task 2 (first draft, feedback comments, end draft)
  - Your System class model and object view
- one zip file containing your implementation from task 4.