Mission 3: Architecture and Use-Case Model

Deadline

See WebCT

Overview

1. This is your third task in the project.
2. The objective of this task is to produce a high-level design of your project (music player or book net).
3. The task is partitioned into four sub-tasks which are:
    a. Use-case models
    b. Architecture
    c. Static analysis
    d. Traceability matrix
4. Each sub-task is described in further detail below.
5. All deliverables are to be handed in electronically WebCT, as agreed upon with the TA supervising your team.
6. Your submission should follow a similar format to the other documents you have submitted. The cover page should list the project team number and names of all team members. This should be followed by a table of contents, and then the rest of the document. Make sure the level of details is "right" – important classes should receive more attention than others.
7. In addition to the deliverables detailed below, you must submit a page identifying the responsibilities of each member in your team. Specifically, list what deliverables will she or he be in charge of (documents, sub-systems, classes, etc.). The member in charge of a certain part can get the help of other team members, but ultimately the responsibility will be hers or his. You will be committed to this partition at later stages (implementation & testing) of the project as well.
8. If you suggested extra features in your requirement document beyond the original specifications and you now plan to address them in the design phase, please mention this explicitly.
9. Also, in addition to the high-level design document and use-case model, you should submit a revised version of your requirements analysis document, correcting any issues raised about your original submission.

Sub-task A: Use-Case Model

The use case model should include:

1. Use case diagrams
2. List of all actors, along with a brief description of the role of each actor and the use-cases they participate in
3. List of all use-cases with a one- or two-sentence description
4. Detailed descriptions of the use cases you feel are the most important ones. Identify at most 15 use cases you feel are the *most important* ones, and include these in the main document.

Additional use cases may be included in an appendix, but will not necessarily be graded. As studied in class and in the tutorials, each use-case description should have

a. A meaningful name
b. Participating actors
c. Entry conditions
d. Flow of events
e. Exit conditions
f. Quality requirements (if any)
g. Exceptions

Sub-task B: Architecture

Explain the general structure of your system and the architecture(s) you decided to use. What is the rationale for your choices? What are the main subsystems and what service do they provide?

Add a diagram that describes the main subsystems and the relationship between them.

Sub-task C: Static Model

In this part you should describe the preliminary (system-level) static structure of the system. Focus on finding the main classes in the system, their attributes, and their associations with other classes. **Note:** There is no need to identify and specify the methods of the classes at this stage.

Please perform domain analysis and identify classes that will be used in the system. Use the requirements and use-case documents as an aid.

When discussing GUI classes, keep the level of details manageable. (There is no need to describe scroll bars or label text locations.)

For this sub-task you should submit the following:

1. List of main classes. For each class you should include:
   a. A meaningful name
   b. Role – describe what it does
   c. Important attributes and methods
   d. Backward traceability – reference to the use case or requirements that were used to come up with the class
2. Class diagrams. Describe the classes used in the system and the relationship between them. Remember, at this point, this is not a detailed design. No need for design patterns or other forms of reuse at this point.

Prepare tidy and clear diagrams. Messy or unreadable diagrams will be penalized. Pay attention to associations between classes, and multiplicity. Try not to have too many classes; the focus at this point is on major classes so you do not need to include smaller classes (e.g., Date). Follow the instructions that were taught in class and in the tutorials.

Sub-task D: Requirement Traceability Matrix

For every requirement listed in your requirement specification document, show how it will be implemented in use cases and in classes. That is, for every requirement (or group of requirements, if convenient), you must ensure that it is implemented by some class, and that it appears in a use-case (for functional requirements). Include a table where each column is a requirement, each row is a use case or class. For each column, indicate with marks in the appropriate cells which use-case(s) address that requirement, which classes are involved in its implementation. This will later be a useful tool for testing use cases, and verifying that all of the requirements have been properly implemented by your system.