Carleton University
Department of Systems and Computer Engineering

SYSC-4805 PROJECT MILESTONES

MILESTONE 1

Milestone 1 Goals: Requirement Specification and Design System Structure

Design the Use Case Model for the system (use case diagram and textual description based on a template of the flows of events corresponding to each use-case). Design the structural model of the system and scenarios representing the realization of the use cases.

Milestone 1 Steps

1.1. Establish the list of requirements and create the Use Case Model that captures the functionality of the system. Show <<include>> and <<extend>> relationships between use cases. Note that it is important to clearly define the boundary of the system being built (i.e., the software system responsible for the control of the elevator bank, which includes device interfaces but not the actual devices). The textual description of the use-cases indicates briefly the role and purpose of each use-case, and describes its flows of events:

- basic flow, a. k. a. the "happy path"
- alternative flows, which take into account regular variants, odd cases and exception flows for handling errors.

1.2. Model the system structure (top-down approach):

- Analysis phase: Identify problem domain objects (classes) and their relationships. Categories of classes include external classes from the environment (i.e., users, external I/O devices, sensors and actuators, external systems) and application classes contained in the system (i.e., entity classes, controllers, boundary classes). Note that attributes may be identified in this step, but no operations.
- Design phase: UML-RT class and capsule structure diagram
    - Map the classes identified in the analysis phase to capsules, passive classes or protocols/ports.
    - Identify the internal structure of each capsule (may contain nested capsules).

1.3. Use case realization

- For each use case: develop use case realizations (expressed as interaction diagrams) that show how the capsules/ objects interact in order to realize the functionality expressed by the use cases.
- This step will help to:
    - identify the signals/information exchanged between interaction participants and define the required protocols
    - discover new capsules/classes that were omitted before.

Milestone 1 Demo

a) Requirements and Use Case Model from step 1.1: use case diagram (*with UML-RT tool*) and use case textual description (*text editor*)

b) Structure model step 1.2: problem domain class diagram(s) (*with drawing tool*); design class diagram showing capsules, protocols and classes (*with UML-RT tool*) and capsule structure diagrams (*with UML-RT tool*)

c) Use Case Realizations from step 1.4: sequence diagrams for all use cases (*with drawing tool*); add signals to protocols (*with UML-RT tool*).

A marking scheme for the demo will be posted. Every group should submit on cuLearn a zipped folder containing the models, documents and diagrams prepared for the demo. We do NOT expect an integrated cohesive report for each milestone. Collecting these documents will help you gather material for the final report and will help the TA to better evaluate your work.

MILESTONE 2

Milestone 2 Goals: Design of capsule behaviour, Implementation and Unit testing

- Design phase: system behaviour represented as capsule state machines
- Implementation of the model using the UML-RT tool
- Unit testing.

Milestone 2 Steps:

2.1. Design the system behaviour

- For each use case: refine the use case realizations from the previous milestone and update them, if necessary.
- For each capsule: design its state diagram, by taking into account the behaviour of the capsule on behalf of every use case. Use nested state charts where appropriate.
- For each passive class: design its operations.

2.2. Implementation: build the UML-RT state machines for three system capsules: elevator capsule, controller capsule and coordinator capsule.

2.3. Perform unit testing for the three capsules (elevator, controller and coordinator), by sending appropriate signals to the capsule posts and tracing its behaviour. Unit testing is meant to cover the behaviour of each capsule at a time as completely as possible. You need to identify the list of unit test cases for each capsule, to indicate the input signal(s) and the expected outcome for each test. There are two ways of performing unit testing:

a) injecting the signals for each test case "by hand" with the help of port probes and inject windows

b) build a test harness capsule (which will be thrown away after the test) whose role is to send the input signals for each test case to the capsule under test and receive its outputs. The alternative (b) requires more effort to build the test harness but simplifies the task of repeating the unit testing every time the capsule under test is changed.

2.4. Plan the end-to-end test cases based on the use cases (cover the alternatives)

Milestone 2 Demo

- Model implementation
- Description and discussion of your test plan for unit testing.
- Unit testing of three capsules (elevator, controller, and coordinator).
- Present the list of end-to-end test cases based on the use cases.

Every group should submit on cuLearn a zipped folder containing the documents and diagrams prepared for the demo. We do NOT expect an integrated cohesive report for each milestone. Collecting these documents will help you gather material for the final report.

(Final) MILESTONE 3

Milestone 3 Goals:

- Complete the implementation of the entire system.
- Perform integration testing of the model.
- Prepare the draft of the final report.

Milestone 3 Steps:

3.1. Complete the model implementation.
3.2. Perform integration testing. The set of test cases used for integration testing should contain end-to-end tests corresponding to each event flow of each use-case from step 1.1, and concurrency tests (overlapping of use cases).
3.3. Obtain trace-based sequence diagrams by executing the model for different test cases, and compare them with the original requirements.

Final Demo

- Demonstrate end-to-end behaviour of the system
- Demonstrate overlapping of use cases
- Answer questions on integration testing and how the system satisfies the original requirements.

Every group should submit on cuLearn a zipped folder containing the documents and diagrams prepared for the demo. We do NOT expect an integrated cohesive report for each milestone. Collecting these documents will help you gather material for the final report.

FINAL REPORT

WHEN: Thursday, April 7, 2016, 23:55 PM.
WHAT: The following items are all required to be submitted in a zipped folder on cuLearn:

1. Electronic copy of the project report with the contents listed below.
2. Electronic copy of the UML-RT model
   - Save with your model traces/sequence diagrams that you obtained by running the model.
   - Test the electronic copy of your model before submitting it, to make sure it's your latest version and the tool can read it without any problem! The model will be tested by the TA.
3. REAME.txt file with running instructions and the signals to be injected for every test case.

REPORT CONTENT

Title Page and Table of Contents

1. Introduction: explain the contribution of each team member.

2. System Requirements and Use Case Model

- System requirements
- Use case diagrams
- Use-case textual description as prepared for milestone 1.

3. Analysis and Design documentation

- Describe the model as prepared for milestone 1 and 2, showing the system structure and behaviour. Explain the role and responsibilities of different components.
- Discuss the choice of system architecture, concurrency issues, other design choices you have considered, role and responsibilities of different capsules/classes. Explain the use of UML-RT features such as encapsulation, multiplicity, different types of ports, capsules versus passive classes, inheritance, hierarchical state machines, etc.
- Illustrate the text with model printouts obtained from the tool (class diagrams, capsule structure and state diagrams).

4. Testing strategy

- Explain your testing strategy
- List of test cases for unit testing and test harnesses, if you have them.
- List of test cases for integration testing (derived from use cases)

- Illustrate both unit and integration test cases with printouts of traces and/or sequence diagrams obtained by running the model. Discuss how different sequence diagrams satisfy the original requirements.
