Metropolitan State University
ICS 372 Object-Oriented Design and Implementation

Group Project 2

Points: 100

Goals:

1. Perform finite-state modeling techniques to come up with the state transition table and diagram.
2. Identify the classes in an FSM-based system with minimal conditionals.
3. Use the Unified Modeling Language to document work.
4. Work in small groups.
5. Implement a design utilizing structures such as classes and interfaces.
6. Employ Java coding standards.

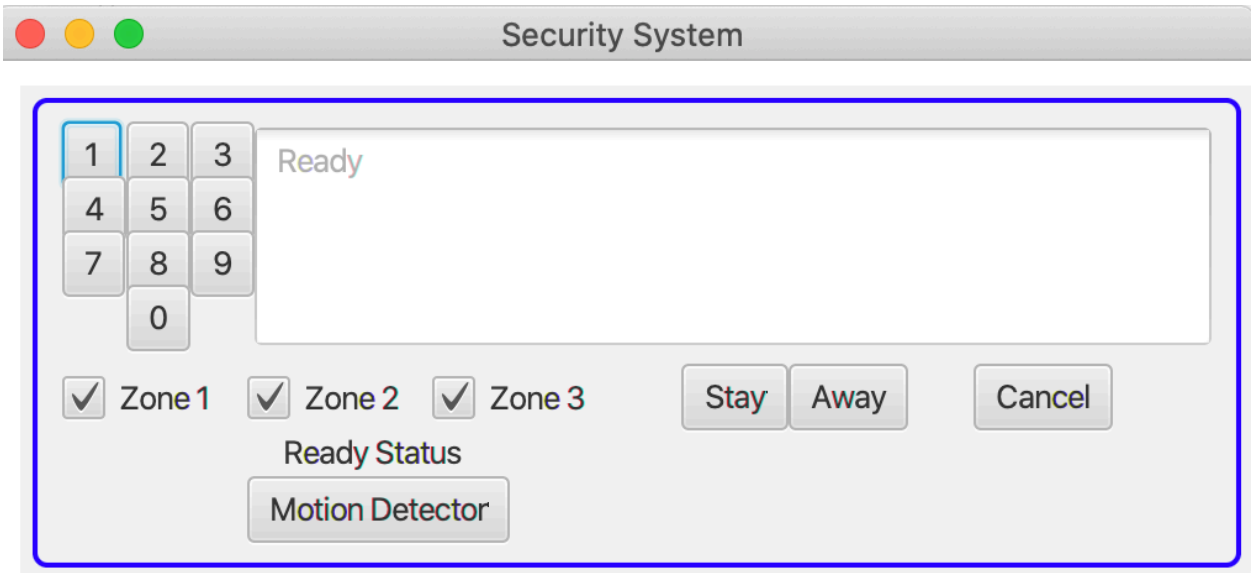
The Problem

You need to use the FSM approach to model, design, and implement a controller for a home security system. The system works as follows.

When the home is unoccupied, the system may be **armed**, which means many intrusions can be detected and alarms raised, so law enforcement can be notified. The system can also be armed even when the home is occupied, perhaps at night.

To arm the system, all exterior doors should be closed. In this project, we call these doors **zones**. A zone is said to be **ready**, when the corresponding door is closed. When the door is open, that zone is **not ready**.

Users manage security via a panel, which looks like the following.



When the home is going to be unoccupied, all doors are closed (that is, all zones are ready), and the user presses the Away button. The user then has 60 seconds to leave the house. This will mean making one or more zones not ready, but the system ignores door openings and closings as well as movements during this 60-second period.

When exactly 60 seconds elapse after the user presses Away, the system checks that all zones are ready. If not, the system unarms and the system would be not ready. If all zones are ready, the system is fully armed. If there is any movement inside the house or any zone becomes not ready, the system issues a warning to enter a secret password via a keypad (see the picture). If no valid password is entered in 60 seconds after movement or no-ready-zone is detected, the system decides that security has been breached. In an actual system, a massive audible alarm is raised (like from that of a fire truck or ambulance) and police may be called. The breached situation persists until the correct password is entered.

At night, the owner may press the Stay button The system behaves similarly as in the Away case, except for two important differences:

1) Movements within the house do not result in a warning.
2) If a zone becomes unready, the system immediately declares that security has been breached. There is no warning period.

Since we can't have real doors and motion detectors, we simulate these using checkboxes for the zones and a button for motion. A checkbox when checked means that zone is ready. A movement is simulated by clicking the Motion Detector button.

The system can be disarmed by pressing the Cancel button and then entering the password. We assume that in our system, it is not possible to disarm the system during the one-minute waiting period after pressing Away or Stay. This makes things a bit simpler, although is may not be reflective of real-world systems.

You must implement the system such that the waiting period after Stay or Away is pressed is 10 seconds and the warning period after a zone is made unready or motion is detected is 15 seconds. This will be a factor in grading.

Watch the video posted on D2L. The GUI should be similar to the one given here, but does not have to match 100%; but all components should be there and properly organized. The displays must be exactly as in the demo.

You may use FXML to create the interface. However, all event handling and processing must be in pure Java. Design and implement using the state and observer patterns. Your work should demonstrate mastery over the concepts discussed in class. You will have a fair number of conditionals in your code, but they shouldn't be excessive.

Things to be Submitted

You need to submit two files: one for analysis and design, and a second one for implementation.

Analysis and Design

Analyze and design the system using the FSM approach. Submit a single PDF document that contains all of the following.

1. The state transition table. This must be in the standard format, with states appearing in the rows and events shown in the columns. The states and events must be properly named in the table, reflecting their role in the application. Difficult-to-read tables will not get much credit and hand-written tables will be ignored.
2. The state transition diagram. Draw the diagram showing all the states. But restrict the transitions to show only what happens when the Stay, Away, Zone, and Motion Detector buttons are pressed. The diagram must use the standard notations. The states and events must be properly shown in the diagram. Difficult-to-understand diagrams will not get much credit and hand-drawn diagrams will be ignored.

You can talk to me to get any clarifications you need, especially with respect to the requirements.

I will also ignore any document that is in any other format (Word, GIF, JPEG, etc.) If you submit multiple PDF documents, I will choose one of the PDF files and ignore everything else. Be sure to ensure that the information is all in the portrait mode if at all possible. But it is more important to not have any part of the information cut off or unviewable in any way.

Implementation

Implement the design following sound object-oriented principles. Submit the entire application as an Eclipse Java project using JavaFX. *Do not create a module*. All code

(classes, interfaces, constructors, and methods) must be properly formatted and documented. Use packages to clearly group your classes. The documentation, naming conventions, and code formatting must follow the coding conventions we have discussed before. Document and lay out your code properly as specified under the CodingStandards.pdf document.

I will provide no debugging support. You must resolve such issues within the group.

Grading

Your assignment will be graded as written in this section.

Design:

The states and events must correctly reflect the behavior of the application and be such that they promote clean implementation that adheres to sound object-oriented principles.

Correctness of states and events and state transitions will count as below. Note that if you don't have the correct states and events, you will also lose points for correctness of state transitions.

Correctness of States	10
Correctness of Events	10
Correctness of State Transitions	20

State Transition Table (10 points)

State Transition Diagram (10 points)

Implementation (40 points)

Grading Issues Related to Group Work

Usually, all members of a group get the same grade. But I have had multiple groups disputes, which have often made grading a somewhat difficult process. I am forced to reserve the right to give students within a group different grades. This will not be done capriciously; I will be meeting with every group every week, so will have at least weekly updates on how things are going on within groups, which might obviate the need to assign different grades. I will inform the student involved before I give him/her a different grade.