SWEN30006 Software Modelling and Design
Workshop 11: Getting a Better GRASP on Design

School of Computing and Information Systems
University of Melbourne
Semester 2, 2020

Introduction

We have already covered domain modelling and design modelling in lectures and workshops. This workshop brings together many of the elements covered in the subject, and provides you with further modelling and pattern application practise. You should expect some questions on these topics in the exam. This workshop also provides you with practise at drawing models individually, a task you will need to perform quickly and correctly in the exam.

General Instruction

In answering these questions, you may need to draw UML diagrams or fragments of these diagrams. You should draw those diagrams with a software for the purpose of conducting peer review and getting feedback more easily. However, for the exam, the diagrams will be hand-drawn. Hence, for the exam, we recommend you use a pencil and have an eraser handy. We also remind you that you can draw a UML model using more than one diagram, in particular using frames.

In keeping with instructions you have already received in regard to UML diagramming, your diagrams must use correct notation. For example, we will not be guessing or inferring whether a particular arrowed line is an association showing navigation or an inheritance relationship; this must be clear from the choice of notation.

You may need to make some assumptions about the problem being solved in order to complete these tasks. Feel free to make consistent and reasonable assumptions; you should explicitly list them.

Case Study: Security System

StopEm Enterprise is developing a new security product line CatchAll, covering both the home market and small enterprise market. CatchAll is modular, and provides for cameras, various detectors, and flashers/-sounders to be placed around a home or premises. It can be monitored over the internet, with triggered alarms being reported, zones in the secured building being turned off or on, and cameras being monitored.

The modularity is supported by Zone Units, which support up to 6 sensors, up to 2 cameras, and up to two flashers/sounders being connected in a zone, i.e. an area of the building physically close to where the Zone Unit is placed, typically a large room or several adjacent rooms. These Zone Units must in turn be connected to a Central Control unit which can support up to 16 Zone Units.

In CatchAll version one, there will only be three sensor types, an infrared detector, a movement detector, and a configurable movement detector that can be adjusted to ignore (most) pets. If a sensor in an active zone is set-off, all the flashers and sounders in all zones will be activated, and an alert will be sent to all registered Monitors of that system via the CatchAll smart phone app.

A Monitor can use the app to select a camera and see the camera image (incl. panning and zooming– only one Monitor can control a camera at a time of course), to reset the alarm system, and to turn zones on or off.

The user operates a CatchAll security system by entering a preset (6-8 digit) code through the CatchAll app or CatchAll console keypad. This code will cause the system to enter a countdown phase; after a preset period of time the alarm is armed. During the countdown period, active zone units beep at an increasing rate to provide an audible cue for people on the premises to leave to avoid triggering the alarm. At any time, the code can be re-entered to disarm the system. This will also shut-off flashers/sounders if they have been triggered.

The console display and app will show the current status of the overall system, and the console or app can be used to activate or deactivate zones at any time (this will not shutdown triggered flashers/sounders). When the system is armed, the app can be used to see which detectors, if any, are being triggered, whether or not they are in an active zone. In this way the app can be used to monitor movement or heat detection in areas which are zoned off.

> **suggestion:** While most of the following exercises are to be completed **individually** to simulate the exam environment, we recommend you to conduct peer review after each exercise.

Part A

Requisite Knowledge and Tools

It is expected by this point that you are familiar with the following terms and concepts:

- UML Class Diagram Notation (Ch. 16)
- Domain (Conceptual) Modelling (Ch. 9, 31)
- Design Class Modelling and GRASP Patterns (Ch. 17, 18, 19, 25)
- UML State Machine Modelling (Ch. 29)

It is suggested that you have attended the lectures on the above topics and read, at a minimum, the corresponding chapters, as listed above, of Applying UML and Patterns by Larman (the prescribed text).

Exercise One - Core

Draw a domain class diagram for CatchAll.

Exercise Two - Core

Draw a design class diagram for CatchAll.

Where your design model differs from your domain model, justify this difference.

Exercise Three - Core

Draw a state-machine diagram for the CatchAll security system, including triggers, guards, and any known actions on the transitions/states.

Part B

Requisite Knowledge and Tools

It is expected by this point that you are familiar with the following terms and concepts:

- UML Sequence Diagrams (Ch. 10, 15)
- GRASP Patterns (Ch. 17, 18, 25)
- GoF Patterns (Ch. 26)

It is suggested that you have attended the lectures on the above topics and read, at a minimum, the corresponding chapters, as listed above, of Applying UML and Patterns by Larman (the prescribed text).

General Instruction

To complete the exercises below, you may want to modify your domain and design class models.

You are free to discuss the requirements and the class models with other students, but suggest that you complete the exercises below individually.

Your design solution to the CatchAll system should involve at least the four GRASP patterns are listed below:

- Polymorphism
- Pure Fabrication
- Indirection
- Protected Variations

In your solutions to each of the exercises below, identify where patterns have influenced the design.

Exercise Four

You are to draw the following design sequence diagram.

`Forgot William is working late`. Movement detector sets off alarm and Jessica is alerted via the app. Jessica responds to maintain security while allowing William to continue working in the office zone.

Exercise Five

You are to draw the following design sequence diagram.

`System start-up`. Show a system startup sequence that demonstrates creation of at least one instance of each of the classes in your design class diagram (treat sensors as one entity and flashers/sounders as one entity for the purposes of this exercise).