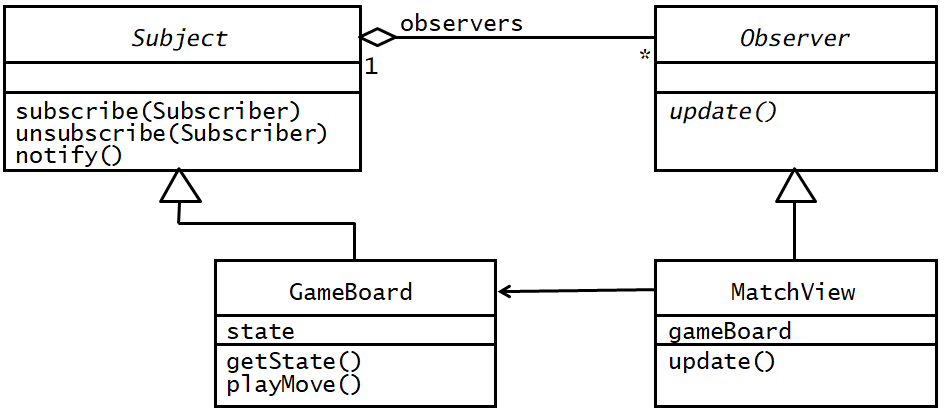
**Individual Assignment 2: Object Modeling and Dynamic Modeling**

1. Please perform reverse engineering to compose a UML class diagram for the provided java file, i.e., Question1.java.

Diagram

Description automatically generated

1. Please implement the following design using Java, C#, or C++.



The following code is created and generated from starUML, and I also edited the code a little. These files can be found inside the zip file which is inside the submission folder.

Subject.java

import java.util.\*;

/\*\*

\*

\*/

public class Subject {

/\*\*

\* Default constructor

\*/

public Subject() {

}

/\*\*

\* @param Subscriber

\*/

public void subscribe(void Subscriber) {

// TODO implement here

}

/\*\*

\* @param Subscriber

\*/

public void unsubscribe(void Subscriber) {

// TODO implement here

}

/\*\*

\*

\*/

public void notify() {

// TODO implement here

}

}

MatchView.java

import java.util.\*;

/\*\*

\*

\*/

public class MatchView extends Observer {

/\*\*

\* Default constructor

\*/

public MatchView() {

}

/\*\*

\*

\*/

int gameBoard;

/\*\*

\*

\*/

public void update() {

// TODO implement here

}

}

GameBoard.java

import java.util.\*;

/\*\*

\*

\*/

public class GameBoard implements Subject {

int state;

/\*\*

\* Default constructor

\*/

public GameBoard() {

}

/\*\*

\*

\*/

public void getState() {

// TODO implement here

}

/\*\*

\*

\*/

public void playMove() {

// TODO implement here

}

}

Observer.java

import java.util.\*;

/\*\*

\*

\*/

public class Observer {

/\*\*

\* Default constructor

\*/

public Observer() {

}

/\*\*

\*

\*/

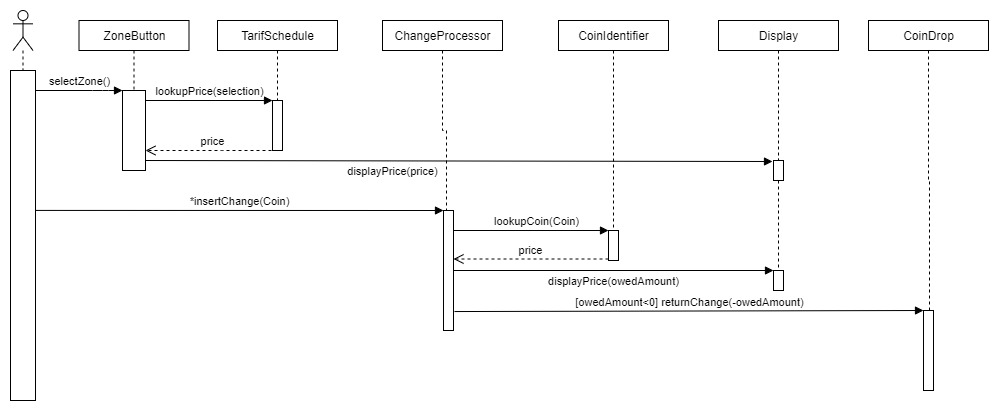
public void Update() {

// TODO implement here

}

}

1. In our lecture 11 (dynamic modeling), please combine the sequence diagrams on pages 14 and 15 into a single sequence diagram. I used app.diagrams to create this because starUML wouldn’t let me use texts that contain “()” under sequence diagrams.



1. Some products, such as GPS, iPhone 7, etc., are currently out of stock in Bestbuy®. Some customers want to be notified if these products become available again. Please create ***a sequence diagram and a class diagram*** for the analysis and design.

Sequence Diagram:

Diagram

Description automatically generated

Class Diagram:Diagram

Description automatically generated

1. Imagine that you are developing a lighting system. A switch has “on” and “off” states. When the switch is turned to “on”, the light should be turned on. Otherwise, the light should be turned off. Suppose that we have different types of switches, such as a regular switch attaching to the wall or a remote control switch that can remotely turn on/off the light. ***Please draw a statechart diagram and a class diagram for your analysis and design***.  
   State chart:

Diagram

Description automatically generated

Class Diagram:

Diagram

Description automatically generated

1. Imagine that you will develop a web system for a rural area, which only has very slow Internet connection. As downloading images is pretty time consuming, you want to make a place holder for the image when initializing a web page. We want to show the text portion as soon as possible. The real images are displayed after the text portion has been downloaded. This strategy is good because users have something to see when they try to open a web page. Otherwise, they will lose patience if they have to wait for a long time to see the web page. Please use the **class diagram** to make the design.

Diagram

Description automatically generated