CS586 PROJECT

CS 586; Spring 2018

Deadlines:
MDA-EFSM (5 points): **March 28, 2018**
After **April 1** the MDA-EFSM will not be accepted.

Final Project (30 points): **April 25, 2018**
Late submissions: 50% off
After **April 30** the final project will not be accepted.

This is an **individual** project not a team project.
The **hardcopy** of the project must be submitted. Electronic submissions are not acceptable. Notice that the Blackboard project submissions are only considered as a proof of submission on time (before the deadline).

Goal:
The goal of this project is to design two different *GasPump* components using the Model-Driven Architecture (MDA) and then implement these *GasPump* components based on this design.

Description of the Project:
There are two *GasPump* components: *GasPump-1* and *GasPump-2*.

The **GasPump-1** component supports the following operations:

Activate (float a, float b)	// the gas pump is activated where *a* is the price of the Regular gas // and *b* is the price of Diesel gas per gallon
Start()	//start the transaction
PayCredit()	// pay for gas by a credit card
Reject()	// credit card is rejected
PayDebit(string p)	// pay for gas by a debit card, where *p* is a pin #
Pin(string x)	// pin # is provided, where *x* represents the pin #
Cancel()	// cancel the transaction
Approved()	// credit card is approved
Diesel()	// Diesel gas is selected
Regular()	// Regular gas is selected
StartPump()	// start pumping gas
PumpGallon()	// one gallon of gas is disposed
StopPump()	// stop pumping gas
FullTank()	// Tank is full and pump is stopped

The **GasPump-2** component supports the following operations:

```
Activate (float a, float b, float c)     // the gas pump is activated where a is the price of Super gas, b is
                                         //the price of Regular gas and c is the price of Premium gas per liter
PayCash(int c)                           // pay for gas by cash, where c represents prepaid cash
PayCredit()                              // pay for gas by a credit card
Approved()                               // credit card is approved
Reject()                                 // credit card is rejected
Cancel()                                 // cancel the transaction
Premium()                                // Premium gas is selected
Regular()                                // Regular gas is selected
Super()                                  // Super gas is selected
StartPump()                              // start pumping gas
PumpLiter()                              // one liter of gas is disposed
Stop()                                   // stop pumping gas
Receipt()                                // Receipt is requested
NoReceipt()                              // No receipt
```

Both *GasPump* components are state-based components and are used to control simple gas pumps. Users can pay by cash, a credit card or a debit card. The gas pump may dispose different types of the gasoline. The price of the gasoline is provided when the gas pump is activated. The detailed behavior of *GasPump* components is specified using EFSM. The EFSM of Figure 1 shows the detail behavior of *GasPump-1* and the EFSM of Figure 2 shows the detailed behavior of *GasPump-2*. Notice that there are several differences between *GasPump* components.

Aspects that vary between two *GasPump* components:

a. Types of gasoline disposed
b. Types of payment
c. Display menu(s)
d. Messages
e. Receipts
f. Operation names and signatures
g. Data types
h. etc.

The goal of this project is to design two *GasPump* components using the Model-Driven Architecture (MDA) covered in the course. In the first part of the project, you should design an executable meta-model, referred to as MDA-EFSM, for *GasPump* components. This MDA-EFSM should capture the "generic behavior" of both *GasPump* components and should be de-coupled from data and implementation details. Notice that in your design there should be **ONLY** one MDA-EFSM for both *GasPump* components. In addition, in the Model-Driven Architecture coupling between components should be minimized and cohesion of components should be maximized (components with high cohesion and low coupling between components). The meta-model (MDA-EFSM) used in the Model-Driven architecture should be expressed as an EFSM (Extended Finite State Machine) model. Notice that the EFSMs shown in Figure 1 and Figure 2 are **not acceptable** as a meta-model (MDA-EFSM) for this model driven architecture.

SUBMISSIONS & DEADLINES

I. MDA-EFSM submission: Wednesday, March 28, 2018
After **April 1, 2018** the MDA-EFSM will not be accepted.

MDA-EFSM model report for the *GasPump* components should contain:

- A list of meta events for the MDA-EFSM
- A list of meta actions for the MDA-EFSM.
  - The responsibility of each action must be described.
- A state diagram of the MDA-EFSM
- Pseudo-code of all operations of Input Processors of *GasPump-1* and *GasPump-2*

The **hardcopy** of the assignment must be submitted. Electronic submissions are not acceptable. Notice that the Blackboard assignment submissions are only considered as a proof of submission on time (before the deadline). If the hardcopy is different than the electronic version submitted on the Blackboard, then **50% penalty** will be applied. If the assignment is submitted on the Blackboard on time, we must receive the hardcopy of the assignment by **4:00pm on Thursday, March 29**. If the hardcopy is received after this deadline, **20% penalty** will be applied.

II. Final Project submission: April 25, 2018
After **April 30** the final project will not be accepted.

The detailed description of the final project report and deliverables will be posted later.