PROJECT

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 Super gas per gallon
Start()	//start the transaction
PayCredit()	// pay for gas by a credit card
Reject()	// credit card is rejected
Cancel()	// cancel the transaction
Approved()	// credit card is approved
Super()	// Super gas is selected
Regular()	// Regular gas is selected
StartPump()	// start pumping gas
PumpGallon()	// one gallon of gas is disposed
StopPump()	// stop pumping gas

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

Activate (int a, int b, int c)	// the gas pump is activated where *a* is the price of Regular gas, *b* is //the price of Premium gas and *c* is the price of Super gas per liter
Start()	//start the transaction
PayCash(int c)	// pay for gas by cash, where *c* represents prepaid cash
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 or a credit 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.