MDA-EFSM Events:

Activate()	
Start()	
PayType(int t)	//credit: t=1; cash: t=2; debit: t=3
Reject()	
Cancel()	
Approved()	
StartPump()	
Pump()	
StopPump()	
SelectGas(int g)	// Regular: g=1; Super: g=2; Premium: g=3; Diesel: g=4
Receipt()	
NoReceipt()	
CorrectPin()	
IncorrectPin()	
Continue()	

MDA-EFSM Actions:

StorePrices	// stores price(s) for the gas from the temporary data store
PayMsg	// displays a type of payment method
StoreCash	// stores cash from the temporary data store
DisplayMenu	// display a menu with a list of selections
RejectMsg	// displays credit card not approved message
SetPrice(int g, int M)	// set the price for the gas identified by *g* identifier as in SelectGas(int g); if M=1, the price may be increased
ReadyMsg	// displays the ready for pumping message
SetInitialValues	// set *G* (or *L*) and *total* to 0;
PumpGasUnit	// disposes unit of gas and counts # of units disposed
GasPumpedMsg	// displays the amount of disposed gas
StopMsg	// stop pump message and receipt? msg (optionally)
PrintReceipt	// print a receipt
CancelMsg	// displays a cancellation message
ReturnCash	// returns the remaining cash
WrongPinMsg	// displays incorrect pin message
StorePin	// stores the pin from the temporary data store
EnterPinMsg	// displays a message to enter pin
InitializeData	// set the value of price and cash to 0

Operations of the Input Processor
(GasPump-1)

```
Activate(float a, float b) {
        if ((a>0)&&(b>0)) {
             d->temp_a=a;
             d->temp_b=b;
             m->Activate()
        }
}

Start() {
        m->Start();
}

PayCredit() {
        m->PayType(1);
}

Reject() {
        m->Reject();
}

PayDebit(string p) {
        d->temp_p=p;
        m->PayType(3);
}

Pin(string x) {
        if (d->pin==x) m->CorrectPin()
        else m->InCorrectPin();
}

Cancel() {
        m->Cancel();
}

Approved() {
        m->Approved();
}

Diesel() {
        m->SelectGas(4)
}

Regular() {
        m->SelectGas(1)
}

StartPump() {
        if (d->price>0) {
             m->Continue();
             m->StartPump();
        }
}

PumpGallon() {
        m->Pump();

StopPump() {
        m->StopPump();
        m->Receipt();
}

FullTank() {
        m->StopPump();
        m->Receipt();
}
```

Notice:
m: is a pointer to the MDA-EFSM object
d: is a pointer to the Data Store object

Operations of the Input Processor (GasPump-2)

```
Activate(int a, int b, int c) {
        if ((a>0)&&(b>0)&&(c>0)) {
              d->temp_a=a;
              d->temp_b=b;
              d->temp_c=c
              m->Activate()
        }
}

PayCash(float c) {
        if (c>0) {
              d->temp_cash=c;
              m->start();
              m->PayType(2)
        }
}

PayCredit() {
        m->start();
        m->PayType(1);
}

Reject() {
        m->Reject();
}

Approved() {
        m-> Approved();
}
Cancel() {
        m->Cancel();
}

Super() {
        m->SelectGas(2);
        m->Continue();
}

Premium() {
        m->SelectGas(3);
        m->Continue();
}

Regular() {
        m->SelectGas(1);
        m->Continue();
}

StartPump() {
        m->StartPump();
}

PumpLiter() {
if (d->cash>0)&&(d->cash < d->price*(d->L+1))
                m->StopPump();
else m->Pump()
}

Stop() {
        m->StopPump();
}

Receipt() {
        m->Receipt();
}

NoReceipt() {
        m->NoReceipt();
}
```

Notice:
cash: contains the value of cash deposited
price: contains the price of the selected gas
L: contains the number of liters already pumped

cash , *L*, *price* are in the data store
m: is a pointer to the MDA-EFSM object
d: is a pointer to the Data Store object