GP-1

MDA-EFSM *m
DataStore *d

Activate( )
Start( )
PayCredit( )
Reject( )
Cancel( )
Approved( )
StartPump( )
PayCash( )
StopPump( )
Pump( )

MDA-EFSM

OP *p

Activate( )
Start( )
PayType( )
Reject( )
Cancel( )
Approved( )
StartPump( )
Pump( )
StopPump( )
SelectGas( )
Receipt( )
NoReceipt( )
Continue( )

OP

DataStore *d

StorePrices( )
PayMsg( )
StoreCash( )
DisplayMenu( )
RejectMsg( )
SetPrice( )
SetInitialValues( )
PumpGasUnit( )
GasPumpedMsg( )
PrintReceipt( )
CancelMsg( )
ReturnCash( )
SetPayType( )
EjectCard( )

GP-2

MDA-EFSM *m
DataStore *d

Activate( )
PayCash( )
Cancel( )
Premium( )
Regular( )
StartPump( )
Stop( )
Receipt( )
NoReceipt( )
Diesel( )
PumpGallon( )
Start( )

DataStore

DS-1

int temp_a
int price
int L
int total
int temp_c
int w
int cash

DS-2

float temp_a
float temp_b
float temp_c
int temp_cash
float Dprice
float Rprice
float Pprice
int cash
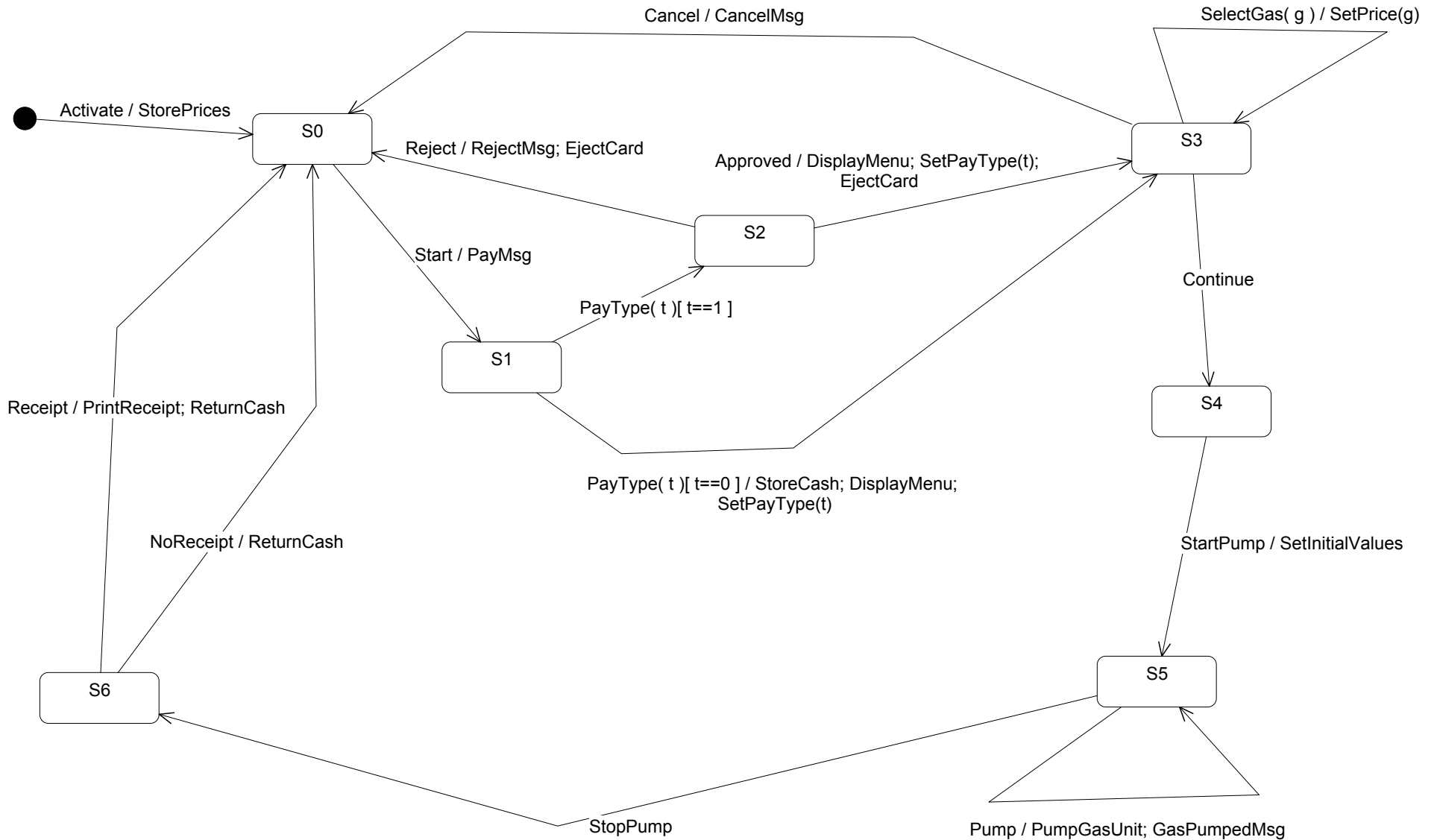float total
int G
float price

MDA-EFSM Events:
Activate()
Start()
PayType(int t)   //credit: t=1; cash: t=0;
Reject()
Cancel()
Approved()
StartPump()
Pump()
StopPump()
SelectGas(int g)   // Regular: g=1; Diesel: g=2; Premium: g=3
Receipt()
NoReceipt()
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)   // set the price for the gas identified by g identifier as in SelectGas(int g);
SetInitialValues()   // set G (or L) and total to 0;
PumpGasUnit()   // disposes unit of gas and counts # of units disposed and computes Total
GasPumpedMsg()   // displays the amount of disposed gas
PrintReceipt()   // print a receipt
CancelMsg()   // displays a cancellation message
ReturnCash()   // returns the remaining cash
SetPayType(t)   // Stores pay type t to variable w in the data store
EjectCard()   // Card is ejected

Cancel / CancelMsg

SelectGas(g) / SetPrice(g)

Activate / StorePrices

S0

Reject / RejectMsg; EjectCard

S3

Approved / DisplayMenu; SetPayType(t);
EjectCard

S2

Start / PayMsg

Continue

PayType(t)[t==1]

S1

S4

Receipt / PrintReceipt; ReturnCash

PayType(t)[t==0] / StoreCash; DisplayMenu;
SetPayType(t)

StartPump / SetInitialValues

NoReceipt / ReturnCash

S6

S5

StopPump

Pump / PumpGasUnit; GasPumpedMsg

MDA-EFSM for Gas Pumps

Operations of the Input Processor
(GasPump-1)

```
Activate(int a) {
        if (a>0) {
                d->temp_a=a;
                m->Activate()
        }
}

Start() {
        m->Start();
}
PayCash(int c) {
        if (c>0) {
                d->temp_c=c;
                m->PayType(0)
        }
}

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

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

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

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

Pump() {
if (d->w==1) m->Pump()
else if (d->cash < d->price*(d->L+1)) {
                m->StopPump();
                m->Receipt(); }
        else m->Pump()
}

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

Notice:
cash: contains the value of cash deposited
price: contains the price of the gas
L: contains the number of liters already pumped
w: pay type flag (cash: w=0; credit: w=1)
cash, *L*, *price, w:* are in the data store
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(float a, float b, float c) {
        if ((a>0)&&(b>0)&&(c>0)) {
                d->temp_a=a;
                d->temp_b=b;
                d->temp_c=c
                m->Activate()
        }
}


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


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

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


Diesel() {
        m->SelectGas(2);
        m->Continue();
}
```

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


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


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


PumpGallon() {
if (d->cash < d->price*(d->G+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
G: contains the number of Gallons already
pumped

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