Exam #2

Wednesday, October 29

Closed books and notes exam

The coverage is posted

COVERAGE FOR EXAM #2

CS 586; Fall 2025

Exam #2 will be held on **Wednesday, October 29, 2025,** at **5:00 p.m.**
Location: 111 Stuart Building
The exam is a **CLOSED books and notes** exam.

Coverage for the exam:

- OO design patterns: proxy, adapter, strategy, and abstract factory patterns. [Textbook: Section 3.4 (pp. 263-275); Handout #1, class notes]
- Interactive systems. Model-View-Controller architectural pattern [Textbook: Section 2.4, pp. 123-143]
- Client-Server Architecture
    - Client-Dispatcher-Server [Textbook: Section 3.6: pp. 323-337]
    - Client-Broker-Server Architecture [Textbook: Section 2.3; pp. 99-122]

Sources:

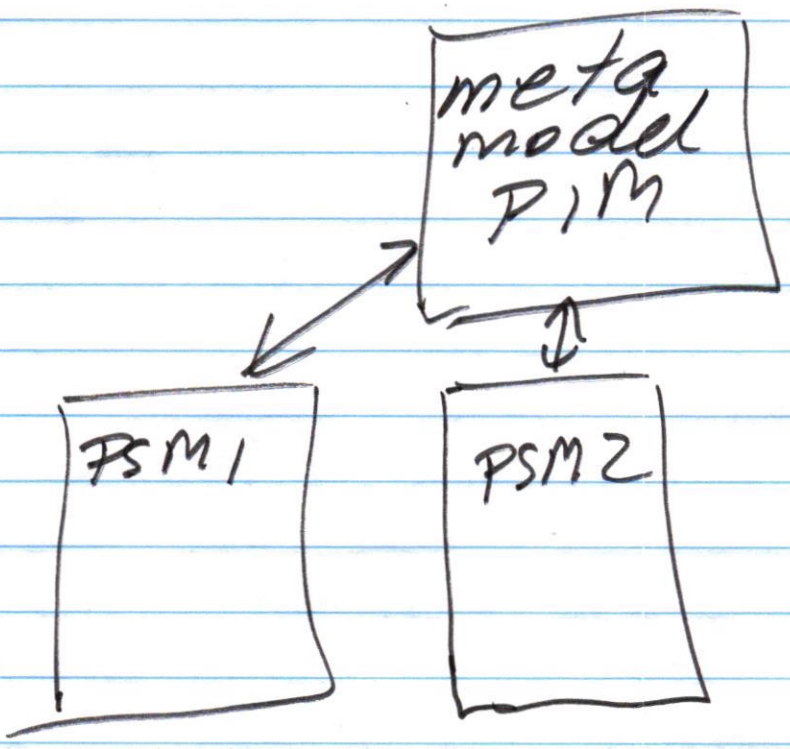
- Textbook: F. Buschmann, et. al., Pattern-oriented software architecture, vol. I, John Wiley & Sons.
- Class notes
- Handout #1

Model-Driven-Architecture

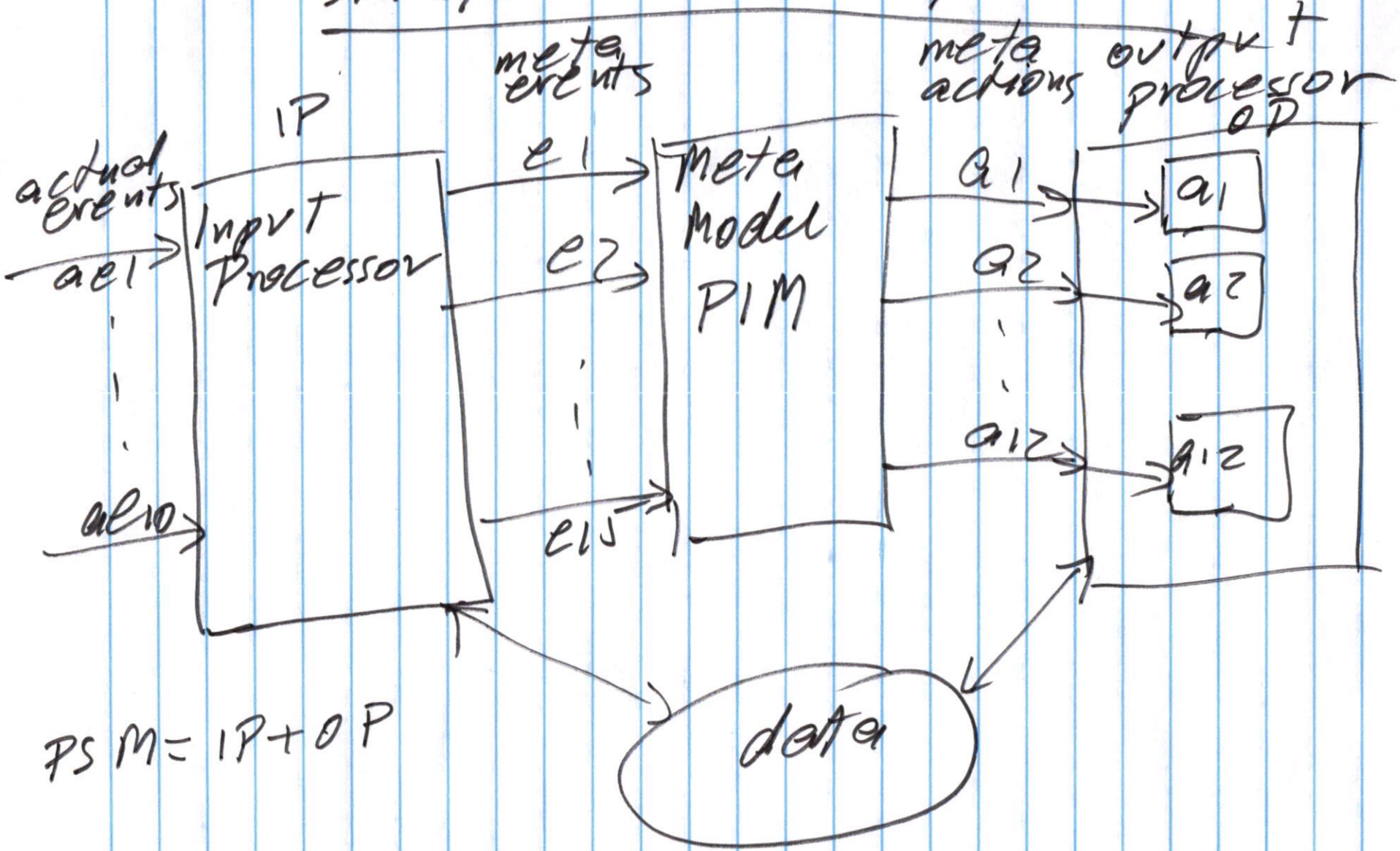
Idea: seperate the Platform independent behavior (PIM)

FROM

platform specific behavia (PSM)



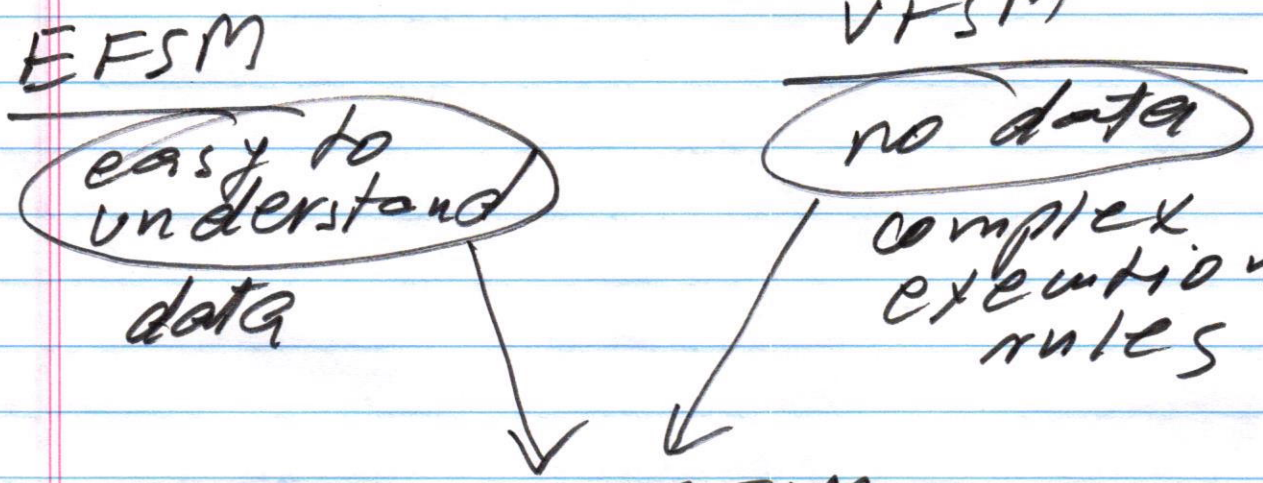
state/event based systems



PSM = IP + OP

Modeling Languages

EFSM
VFSM

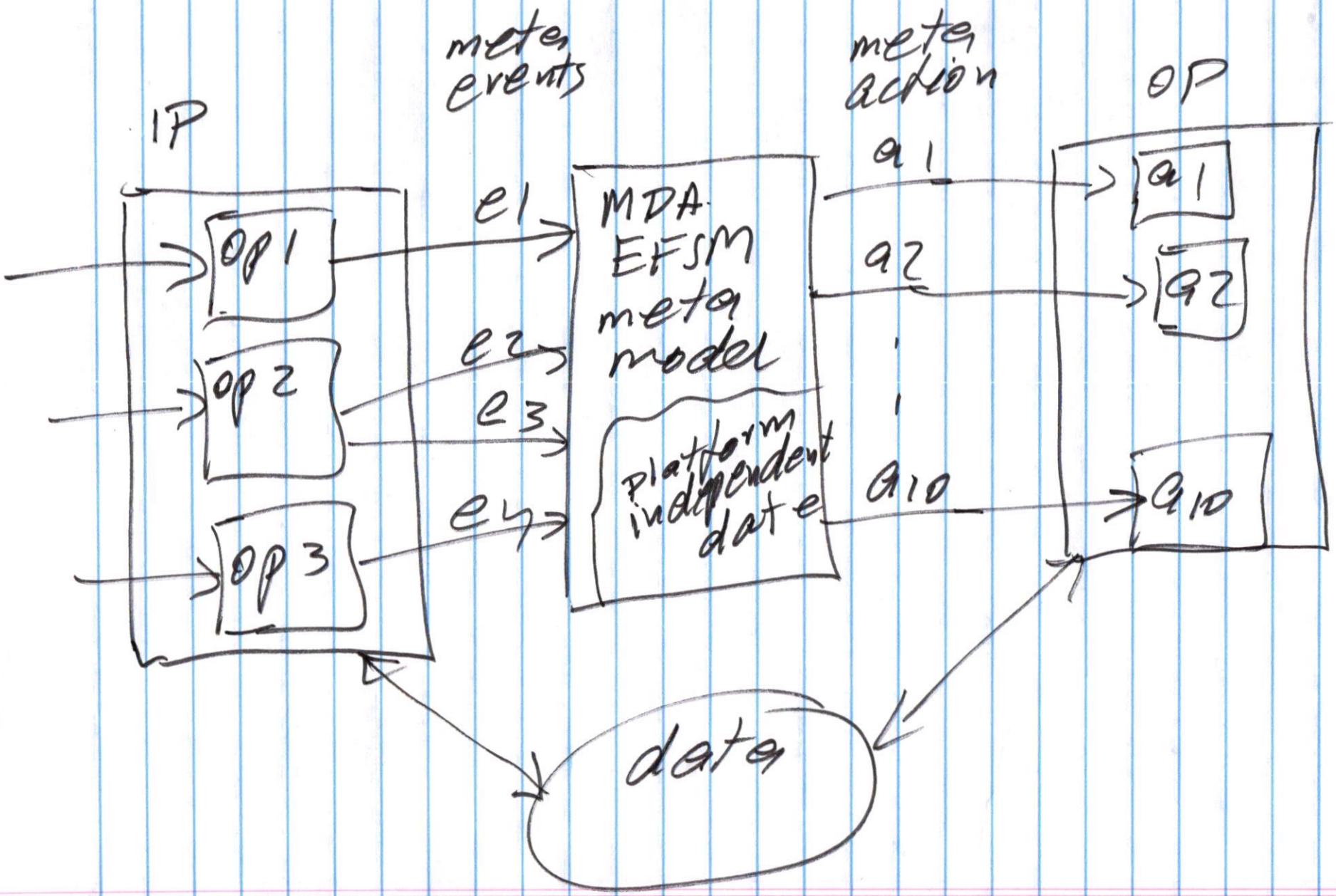


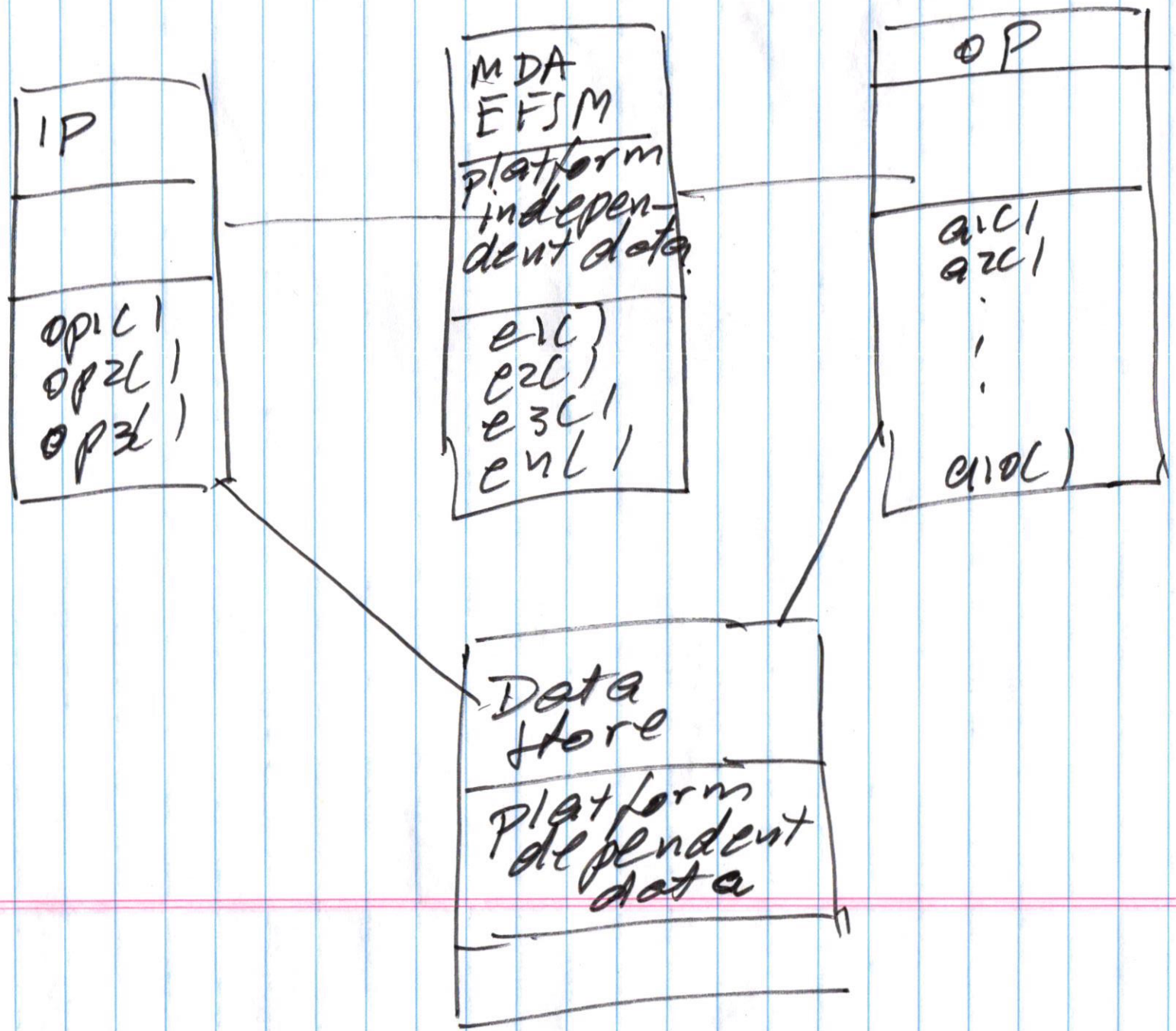
MDA-EFSM

EFSM with restrictions

1. no data
   or
   platform independent data

2. events/operations/actions must be platform independent


IP
meta events
meta action
OP
op1
op2
op3
e1
e2
e3
e4
MDA EFSM meta model
Platform independent data
q1
q2
q10
q1
q2
q10
data


IP
op1C1
op2C1
op3C1
MDA
EFSM
Platform independent data
e1C1
e2C1
e3C1
enC1
OP
q1C1
q2C1
qmOC1
Data Store
Platform dependent data

MDA-EFSM

We need to identify:

1. meta events: $e_1, e_2, e_3, \ldots$
2. meta actions: $a_1, a_2, \ldots$
3. state diagram

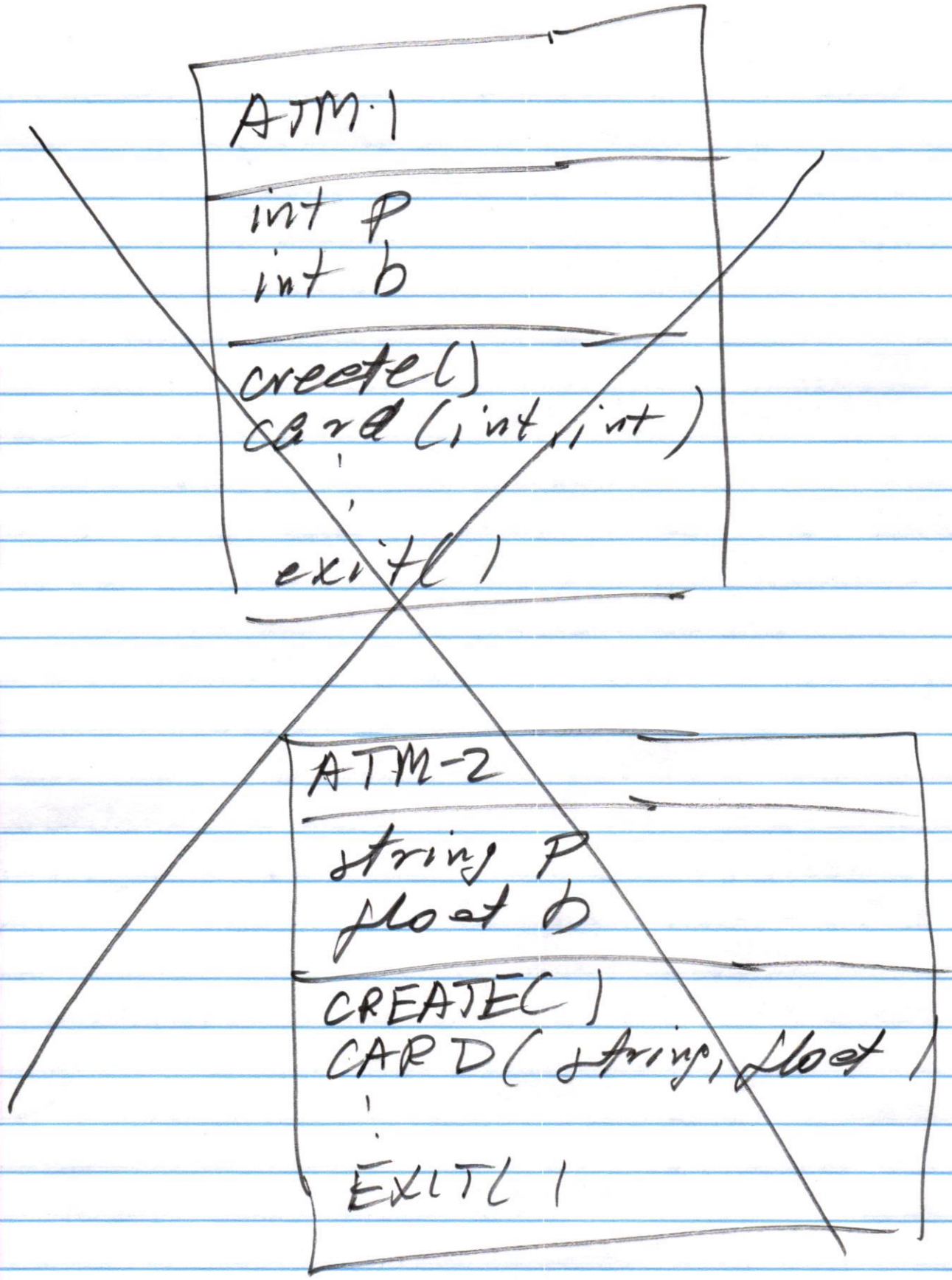
MDA-EFSM

ATM-1 component

create()
card(int x, int y)
pin(int a)
deposit(int d)
withdraw(int w)
balance()
exit()

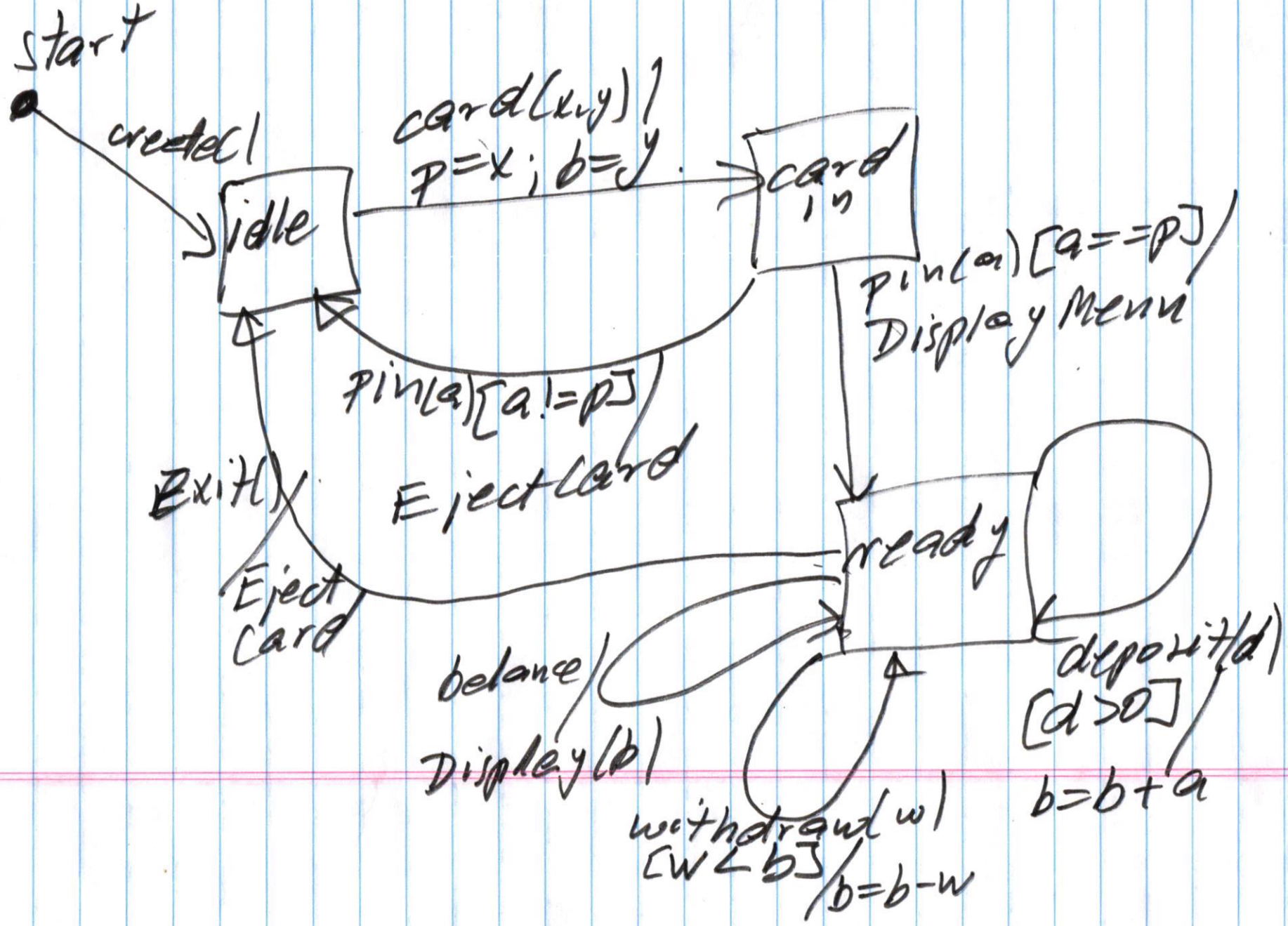
ATM-2 component

CREATE()
CARD(string x, float y)
PIN(string a)
DEPOSIT(float d)
WITHDRAW(float w)
BALANCE()
EXIT()


ATM-1
int P
int b
create()
card(int, int)
exit()
ATM-2
string P
float b
CREATEC()
CARD(string, float)
EXITC()

Platform dependent EFSM

ATM-1



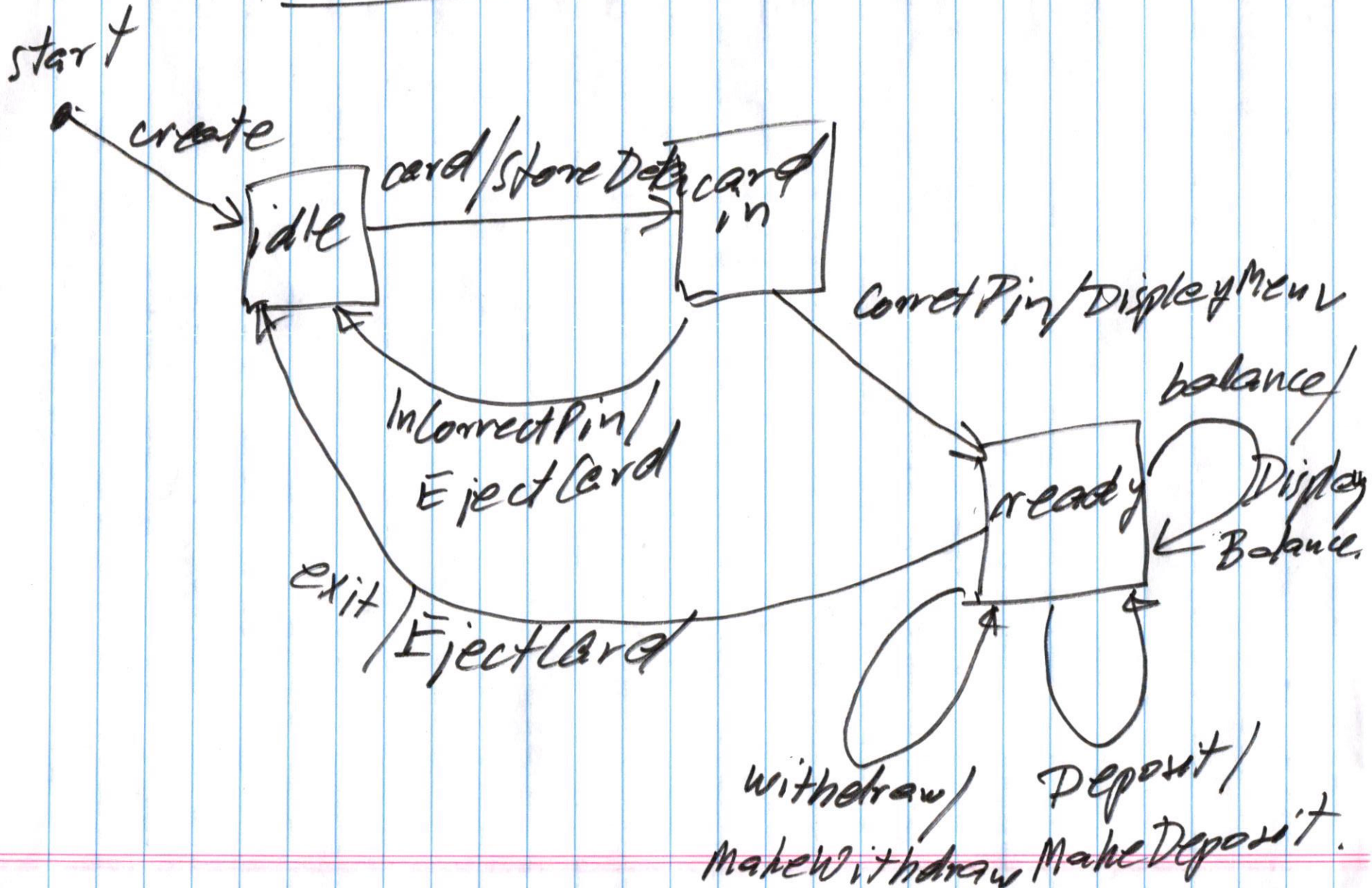
meta events

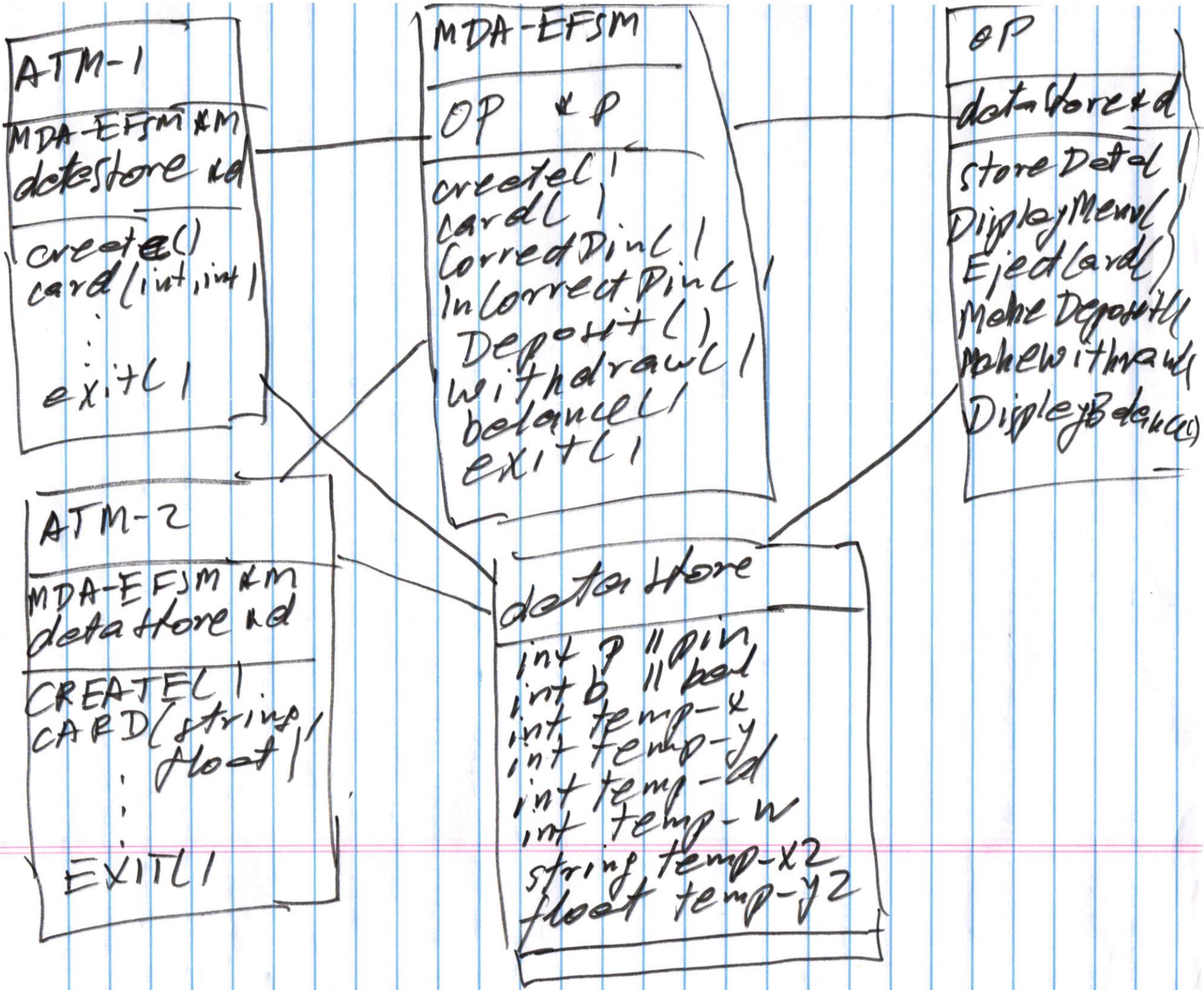
createC)
cardC)
CorrectPinC)
InCorrectPinC)
DepositC)
WithdrawC)
balanceC)
exitC)

meta action

StoreDataC)
DisplayMenu
EjectCardC)
MakeDepositC)
MakeWithdrawC)
DisplayBalanceC)

MDA - EFSM


start
create
idle
card/StoreData
card in
CorrectPin/DisplayMenu
InCorrectPin/
EjectCard
balance/
Display
Balance
ready
exit/EjectCard
withdraw/
MakeWithdraw
Deposit/
MakeDeposit.


ATM-1
MDA-EFSM *M
datastore *d
create()
card(int,int)
exit()
ATM-2
MDA-EFSM *M
datastore *d
CREATE()
CARD(string, float)
EXIT()
MDA-EFSM
OP *p
create()
card()
correctPin()
incorrectPin()
Deposit()
withdraw()
balance()
exit()
datastore
int P //pin
int b //bal
int temp-x
int temp-y
int temp-d
int temp-w
string temp-x2
float temp-y2
OP
datastore *d
storeDet()
DisplayMenu()
EjectCard()
MakeDeposit()
MakeWithdraw()
DisplayBalance()

ATM-1 class

```
create()
{ m -> create()
}
```

```
card (int x, int y)
{     d -> temp-x = x
      d -> temp-y = y.
      m -> card()
}
```

```
pin( int a )
{. if (a == d -> p) m -> CorrectPin()
   else  m -> InCorrectPin()
}
```

```
deposit(int d)
    if (d > 0)
      { d -> temp-d = d
        m -> Deposit()
      }
```

ATM-1 class

```
withdraw(int w)
{. if (w < d->b)
   {. d->temp-w = w
      m->withdraw()
   }
}
```

```
balance()
{ m->balance()
}
```

```
exit()
{ m->exit()
}.
```

OP class

```
StoreData()
{   d->p = d->temp-x
    d->b = d->temp-y;
}
```

```
DisplayMenu()
{ Implementation of
      Display Menu
}
```

```
EjectCard()
{ Implementation
    to eject card
}
```

```
MakeDeposit()
{   d->b = d->b + d->temp-d
}
```

```
MakeWithdraw( )
{
  d->b = d->b - d->temp-w
}
```

```
Display Balance( )
{ implementation of
  displaying
        d->b
}
```

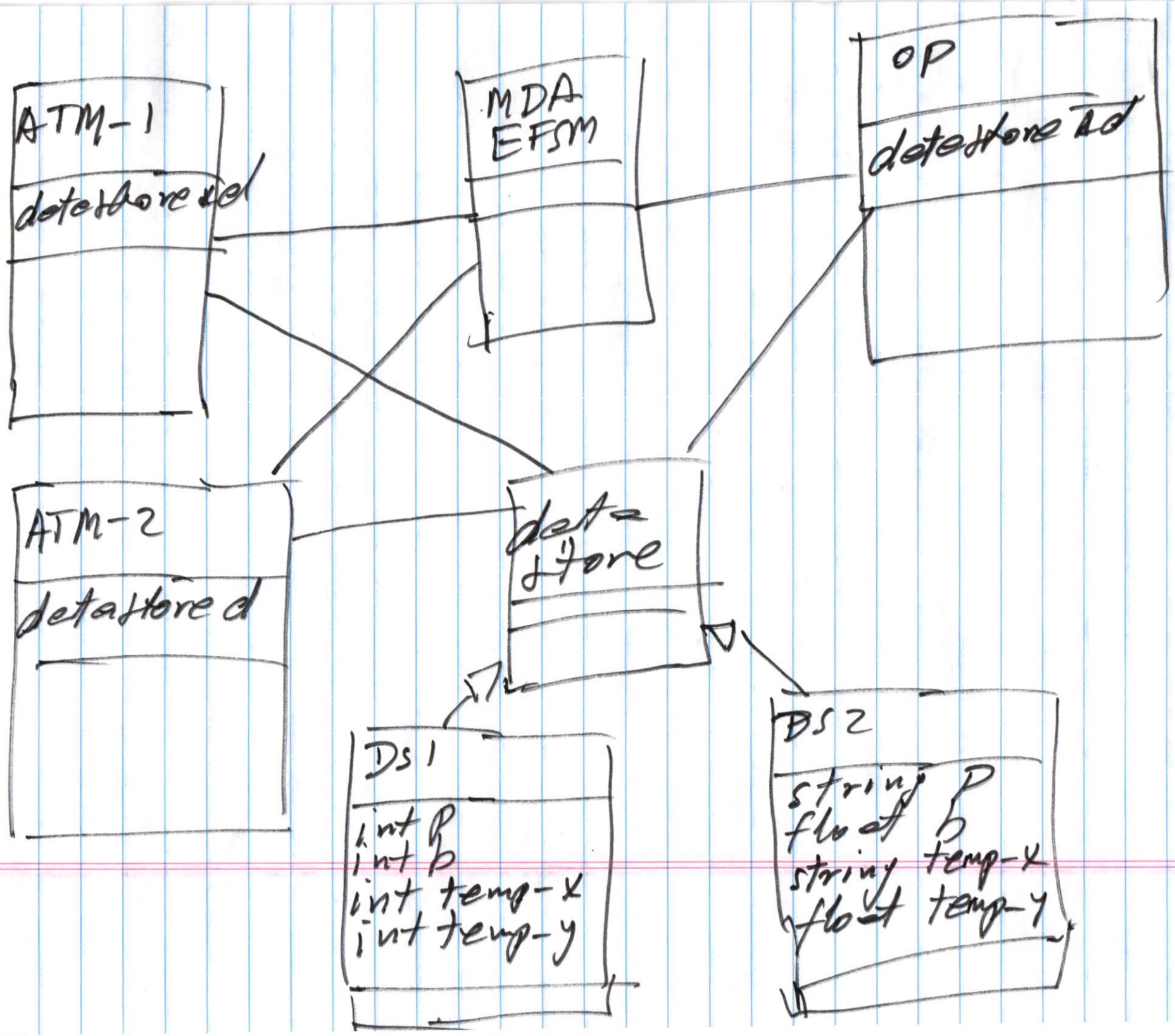
ATM-2 class

CREATE()
{
m → create();
}

CARD(string x, float b)
{
d → temp_x = x
d → temp_y = y
m → card();
}

CARD(string x, float y)
{
d → temp_x2 = x
d → temp_y2 = y.
m → card();
}

```
CARD(string x, float y)
{  d->temp_x = x
   d->temp_y = y.
      m -> card()
}
```


ATM-1
datastored
MDA
EFSM
OP
datastored
ATM-2
datastored
data
store
DS1
int P
int b
int temp-x
int temp-y
DS2
string P
float b
string temp-x
float temp-y

ATM-1:
objects: ATM-1, MDA-EFSM,
OP, DS1

ATM-2
objects: ATM-2, MDA-EFSM, OP
DS2