

MACHINE ATM_R4

REFINES ATM_R3

SEES ATM_R4_implicitContext

VARIABLES

`account` refined class instances
`atm` refined class instances
`atmB` class instances
`bal` inherited attribute of `account`
`atm_card` inherited attribute of `atm`
`atm_cashA` inherited attribute of `atm`
`atm_wdam` inherited attribute of `atm`
`atm_acbalA` inherited attribute of `atm`
`idle` state from refined statemachine, `ATM.SM`
`active_atm` state from refined statemachine, `ATM.SM`
`validating` state from refined statemachine, `active_atm.SM`
`transOption` state from refined statemachine, `active_atm.SM`
`invalidCard` state from refined statemachine, `active_atm.SM`
`performTrans` state from refined statemachine, `active_atm.SM`
`trans` state from refined statemachine, `transOption.SM`
`reqWD` state from refined statemachine, `transOption.SM`
`reqCB` state from refined statemachine, `transOption.SM`
`sentReqWD` state from statemachine, `reqWD.SM`
`recvdReqWD` state from statemachine, `reqWD.SM`
`sentReqCB` state from statemachine, `reqCB.SM`
`recvdReqCB` state from statemachine, `reqCB.SM`
`processedWDFail` state from refined statemachine, `performTrans.SM`
`processedWDOK` state from refined statemachine, `performTrans.SM`
`processedCB` state from refined statemachine, `performTrans.SM`
`endTrans` state from refined statemachine, `performTrans.SM`
`rspWDOK` state from refined statemachine, `performTrans.SM`
`rspWDFail` state from refined statemachine, `performTrans.SM`
`rspCB` state from refined statemachine, `performTrans.SM`
`processWDFail` state from statemachine, `processedWDFail.SM`
`sentRspWDFail` state from statemachine, `processedWDFail.SM`
`processWDOK` state from statemachine, `processedWDOK.SM`
`sentRspWDOK` state from statemachine, `processedWDOK.SM`
`processCB` state from statemachine, `processedCB.SM`
`sentRspCB` state from statemachine, `processedCB.SM`
`atm_acbalB` attribute of `atmB`
`atm_cardB` attribute of `atmB`

INVARIANTS

$\text{atmB.type} : \text{atmB} \in \mathbb{P}(\text{atm})$
 $\text{sentReqWD.type} : \text{sentReqWD} \in \mathbb{P}(\text{reqWD})$
 $\text{recvdReqWD.type} : \text{recvdReqWD} \in \mathbb{P}(\text{reqWD})$
 $\text{sentReqCB.type} : \text{sentReqCB} \in \mathbb{P}(\text{reqCB})$
 $\text{recvdReqCB.type} : \text{recvdReqCB} \in \mathbb{P}(\text{reqCB})$
 $\text{processWDFail.type} : \text{processWDFail} \in \mathbb{P}(\text{processedWDFail})$
 $\text{sentRspWDFail.type} : \text{sentRspWDFail} \in \mathbb{P}(\text{processedWDFail})$
 $\text{processWDOK.type} : \text{processWDOK} \in \mathbb{P}(\text{processedWDOK})$
 $\text{sentRspWDOK.type} : \text{sentRspWDOK} \in \mathbb{P}(\text{processedWDOK})$
 $\text{processCB.type} : \text{processCB} \in \mathbb{P}(\text{processedCB})$
 $\text{sentRspCB.type} : \text{sentRspCB} \in \mathbb{P}(\text{processedCB})$
 $\text{atm_acbalB.type} : \text{atm_acbalB} \in \text{atmB} \leftrightarrow \mathbb{N}$
 $\text{atm_cardB.type} : \text{atm_cardB} \in \text{atmB} \leftrightarrow \text{ValidCard}$
 $\text{reqWD_SM_partitions_reqWD} : \text{partition}(\text{reqWD}, \text{sentReqWD}, \text{recvdReqWD})$
 $\text{reqCB_SM_partitions_reqCB} : \text{partition}(\text{reqCB}, \text{sentReqCB}, \text{recvdReqCB})$
 $\text{processedWDFail_SM_partitions_processedWDFail} : \text{partition}(\text{processedWDFail}, \text{processWDFail}, \text{sentRspWDFail})$
 $\text{processedWDOK_SM_partitions_processedWDOK} : \text{partition}(\text{processedWDOK}, \text{processWDOK}, \text{sentRspWDOK})$
 $\text{processedCB_SM_partitions_processedCB} : \text{partition}(\text{processedCB}, \text{processCB}, \text{sentRspCB})$
 $\text{Invariant1} : \forall a \cdot a \in (\text{recvdReqWD} \cup \text{recvdReqCB}) \Rightarrow a \in \text{dom}(\text{atm_card})$
 $\text{Invariant2} : \forall a \cdot a \in (\text{recvdReqWD} \cup \text{recvdReqCB}) \wedge a \in \text{dom}(\text{atm_card}) \wedge a \in \text{dom}(\text{atm_cardB}) \Rightarrow \text{atm_card}(a) = \text{atm_cardB}(a)$

EVENTS

Initialisation

begin

$\text{account.init} : \text{account} := \emptyset$
 $\text{atm.init} : \text{atm} := \emptyset$
 $\text{atmB.init} : \text{atmB} := \emptyset$
 $\text{bal.init} : \text{bal} := \emptyset$
 $\text{atm_card.init} : \text{atm_card} := \emptyset$
 $\text{atm_cashA.init} : \text{atm_cashA} := \emptyset$
 $\text{atm_wdam.init} : \text{atm_wdam} := \emptyset$
 $\text{atm_acbalA.init} : \text{atm_acbalA} := \emptyset$
 $\text{idle.init} : \text{idle} := \emptyset$
 $\text{active_atm.init} : \text{active_atm} := \emptyset$
 $\text{validating.init} : \text{validating} := \emptyset$
 $\text{transOption.init} : \text{transOption} := \emptyset$
 $\text{invalidCard.init} : \text{invalidCard} := \emptyset$
 $\text{performTrans.init} : \text{performTrans} := \emptyset$
 $\text{trans.init} : \text{trans} := \emptyset$
 $\text{reqWD.init} : \text{reqWD} := \emptyset$
 $\text{reqCB.init} : \text{reqCB} := \emptyset$
 $\text{sentReqWD.init} : \text{sentReqWD} := \emptyset$
 $\text{recvdReqWD.init} : \text{recvdReqWD} := \emptyset$
 $\text{sentReqCB.init} : \text{sentReqCB} := \emptyset$

```

recvdReqCB.init : recvdReqCB := ∅
processedWDFail.init : processedWDFail := ∅
processedWDOK.init : processedWDOK := ∅
processedCB.init : processedCB := ∅
endTrans.init : endTrans := ∅
rspWDOK.init : rspWDOK := ∅
rspWDFail.init : rspWDFail := ∅
rspCB.init : rspCB := ∅
processWDFail.init : processWDFail := ∅
sentRspWDFail.init : sentRspWDFail := ∅
processWDOK.init : processWDOK := ∅
sentRspWDOK.init : sentRspWDOK := ∅
processCB.init : processCB := ∅
sentRspCB.init : sentRspCB := ∅
atm_acbalB.init : atm_acbalB := ∅
atm_cardB.init : atm_cardB := ∅

```

end

Event *createAccount* $\hat{=}$

refines *createAccount*

any

self contextual instance of refined class account

where

self.type : *self* \in *account*

then

skip

end

Event *deposit* $\hat{=}$

refines *deposit*

any

self contextual instance of refined class account

where

self.type : *self* \in *account*

then

skip

end

Event *insertCard* $\hat{=}$

refines *insertCard*

any

selfATM contextual instance of class atm

c

where

selfATM.type : *selfATM* \in *atm*

c.type : *c* \in *ValidCard*

ATM_SM_isin_idle : *selfATM* \in *idle*

```

    insertCard.Guard1 :  $selfATM \notin dom(atm\_card)$ 
  then
    ATM_SM_enterSuperState_active_atm :  $active\_atm := active\_atm \cup \{selfATM\}$ 
    ATM_SM_leaveState_idle :  $idle := idle \setminus \{selfATM\}$ 
    active_atm_SM_enterState_validating :  $validating := validating \cup \{selfATM\}$ 
    insertCard.Action1 :  $atm\_card := atm\_card \cup \{selfATM \mapsto c\}$ 
  end

Event reloadCash  $\hat{=}$ 
refines reloadCash

  any
    selfATM      contextual instance of class atm
  where
    selfATM.type :  $selfATM \in atm$ 
    ATM_SM_isin_idle :  $selfATM \in idle$ 
    reloadCash.Guard1 :  $atm\_cashA(selfATM) < MIN\_CASH$ 
  then
    reloadCash.Action1 :  $atm\_cashA(selfATM) := MAX\_CASH - atm\_cashA(selfATM)$ 
  end

Event ejectCard1  $\hat{=}$ 
refines ejectCard1

  any
    selfATM      contextual instance of class atm
    c
  where
    c.type :  $c \in ValidCard$ 
    selfATM.type :  $selfATM \in atm$ 
    active_atm_SM_isin_invalidCard :  $selfATM \in invalidCard$ 
    ejectCard1.Guard1 :  $selfATM \in dom(atm\_card)$ 
    ejectCard1.Guard2 :  $atm\_card(selfATM) = c$ 
  then
    ATM_SM_leaveSuperState_active_atm :  $active\_atm := active\_atm \setminus \{selfATM\}$ 
    active_atm_SM_leaveState_invalidCard :  $invalidCard := invalidCard \setminus \{selfATM\}$ 
    ATM_SM_enterState_idle :  $idle := idle \cup \{selfATM\}$ 
    ejectCard1.Action1 :  $atm\_card := atm\_card \setminus \{selfATM \mapsto c\}$ 
  end

Event withdrawOK  $\hat{=}$ 
refines withdrawOK

  any
    selfATM      contextual instance of class atm
    c
    am
    ac
  where

```

```

    am.type :  $am \in \mathbb{N}$ 
    ac.type :  $ac \in account$ 
    selfATM.type :  $selfATM \in atm$ 
    c.type :  $c \in ValidCard$ 
    reqWD.SM.isin_recvdReqWD :  $selfATM \in recvdReqWD$ 
    withdrawOK.Guard1 :  $selfATM \in dom(atm\_cardB)$ 
    withdrawOK.Guard7 :  $selfATM \in atmB$ 
    withdrawOK.Guard6 :  $selfATM \in dom(atm\_wdam)$ 
    withdrawOK.Guard2 :  $atm\_cardB(selfATM) = c$ 
    withdrawOK.Guard3 :  $bal(ac) \geq am$ 
    withdrawOK.Guard4 :  $card\_account(c) = ac$ 
    withdrawOK.Guard5 :  $am = atm\_wdam(selfATM)$ 

  then
    performTrans.SM_enterSuperState_processedWDOK :  $processedWDOK := processedWDOK \cup \{selfATM\}$ 
    active_atm.SM_enterSuperState_performTrans :  $performTrans := performTrans \cup \{selfATM\}$ 
    transOption.SM_leaveSuperState_reqWD :  $reqWD := reqWD \setminus \{selfATM\}$ 
    active_atm.SM_leaveSuperState_transOption :  $transOption := transOption \setminus \{selfATM\}$ 
    reqWD.SM_leaveState_recvdReqWD :  $recvdReqWD := recvdReqWD \setminus \{selfATM\}$ 
    processedWDOK.SM_enterState_processWDOK :  $processWDOK := processWDOK \cup \{selfATM\}$ 
    withdrawOK.Action1 :  $bal(ac) := bal(ac) - am$ 
    withdrawOK.Action2 :  $atm\_acbalB(selfATM) := bal(ac)$ 

  end

Event withdrawFail  $\triangleq$ 

refines withdrawFail

any
  selfATM      contextual instance of class atm
  c
  am
  ac

where
  am.type :  $am \in \mathbb{N}$ 
  ac.type :  $ac \in account$ 
  selfATM.type :  $selfATM \in atm$ 
  c.type :  $c \in ValidCard$ 
  reqWD.SM.isin_recvdReqWD :  $selfATM \in recvdReqWD$ 
  withdrawFail.Guard1 :  $selfATM \in dom(atm\_cardB)$ 
  withdrawFail.Guard7 :  $selfATM \in atmB$ 
  withdrawFail.Guard6 :  $selfATM \in dom(atm\_wdam)$ 
  withdrawFail.Guard2 :  $atm\_cardB(selfATM) = c$ 
  withdrawFail.Guard3 :  $card\_account(c) = ac$ 
  withdrawFail.Guard4 :  $bal(ac) < am$ 
  withdrawFail.Guard5 :  $am = atm\_wdam(selfATM)$ 

  then
    performTrans.SM_enterSuperState_processedWDFail :  $processedWDFail := processedWDFail \cup \{selfATM\}$ 
    active_atm.SM_enterSuperState_performTrans :  $performTrans := performTrans \cup \{selfATM\}$ 
    transOption.SM_leaveSuperState_reqWD :  $reqWD := reqWD \setminus \{selfATM\}$ 

```

```

    active_atm_SM_leaveSuperState_transOption :  $transOption := transOption \setminus \{selfATM\}$ 
    reqWD_SM_leaveState_recvdReqWD :  $recvdReqWD := recvdReqWD \setminus \{selfATM\}$ 
    processedWDFail_SM_enterState_processWDFail :  $processWDFail := processWDFail \cup \{selfATM\}$ 
    withdrawFail.Action1 :  $atm\_acbalB(selfATM) := bal(ac)$ 
end

Event checkBalance  $\hat{=}$ 
refines checkBalance

any
    selfATM      contextual instance of class atm
    c
    ac
where
    selfATM.type :  $selfATM \in atm$ 
    c.type :  $c \in ValidCard$ 
    ac.type :  $ac \in account$ 
    reqCB_SM_isin_recvdReqCB :  $selfATM \in recvdReqCB$ 
    checkBalance.Guard1 :  $selfATM \in dom(atm\_cardB)$ 
    checkBalance.Guard4 :  $selfATM \in atmB$ 
    checkBalance.Guard2 :  $atm\_cardB(selfATM) = c$ 
    checkBalance.Guard3 :  $card\_account(c) = ac$ 
then
    performTrans_SM_enterSuperState_processedCB :  $processedCB := processedCB \cup \{selfATM\}$ 
    active_atm_SM_enterSuperState_performTrans :  $performTrans := performTrans \cup \{selfATM\}$ 
    transOption_SM_leaveSuperState_reqCB :  $reqCB := reqCB \setminus \{selfATM\}$ 
    active_atm_SM_leaveSuperState_transOption :  $transOption := transOption \setminus \{selfATM\}$ 
    reqCB_SM_leaveState_recvdReqCB :  $recvdReqCB := recvdReqCB \setminus \{selfATM\}$ 
    processedCB_SM_enterState_processCB :  $processCB := processCB \cup \{selfATM\}$ 
    checkBalance.Action1 :  $atm\_acbalB(selfATM) := bal(ac)$ 
end

Event ejectCard2  $\hat{=}$ 
refines ejectCard2

any
    selfATM      contextual instance of class atm
    c
where
    selfATM.type :  $selfATM \in atm$ 
    c.type :  $c \in ValidCard$ 
    transOption_SM_isin_trans :  $selfATM \in trans$ 
    ejectCard2.Guard1 :  $selfATM \in dom(atm\_card)$ 
    ejectCard2.Guard2 :  $atm\_card(selfATM) = c$ 
then
    active_atm_SM_leaveSuperState_transOption :  $transOption := transOption \setminus \{selfATM\}$ 
    ATM_SM_leaveSuperState_active_atm :  $active\_atm := active\_atm \setminus \{selfATM\}$ 
    transOption_SM_leaveState_trans :  $trans := trans \setminus \{selfATM\}$ 
    ATM_SM_enterState_idle :  $idle := idle \cup \{selfATM\}$ 

```

```

    ejectCard2.Action1 : atm_card := atm_card \ {selfATM ↦ c}
end

Event ejectCard3 ≐
refines ejectCard3

any
    selfATM    contextual instance of class atm
    c
where
    selfATM.type : selfATM ∈ atm
    c.type : c ∈ ValidCard
    performTrans.SM_isin_endTrans : selfATM ∈ endTrans
    ejectCard3.Guard1 : selfATM ∈ dom(atm_card)
    ejectCard3.Guard2 : atm_card(selfATM) = c
then
    active_atm.SM_leaveSuperState_performTrans : performTrans := performTrans \ {selfATM}
    ATM.SM_leaveSuperState_active_atm : active_atm := active_atm \ {selfATM}
    performTrans.SM_leaveState_endTrans : endTrans := endTrans \ {selfATM}
    ATM.SM_enterState_idle : idle := idle ∪ {selfATM}
    ejectCard3.Action1 : atm_card := atm_card \ {selfATM ↦ c}
end

Event start ≐
refines start

any
    selfATM    constructed instance of class atm
where
    selfATM.type : selfATM ∈ ATM \ atm
then
    atm.constructor : atm := atm ∪ {selfATM}
    atm.atm_cashA_initialise : atm_cashA(selfATM) := MAX_CASH
    ATM.SM_enterState_idle : idle := idle ∪ {selfATM}
end

Event validateCardFail ≐
refines validateCardFail

any
    selfATM    contextual instance of class atm
    c
    p
where
    selfATM.type : selfATM ∈ atm
    c.type : c ∈ ValidCard
    p.type : p ∈ Pin
    active_atm.SM_isin_validating : selfATM ∈ validating
    validateCardFail.Guard1 : selfATM ∈ dom(atm_card)
    validateCardFail.Guard2 : atm_card(selfATM) = c

```

```

    validateCardFail.Guard3 :  $card\_pin(c) \neq p$ 
  then
    active_atm_SM_leaveState_validating :  $validating := validating \setminus \{selfATM\}$ 
    active_atm_SM_enterState_invalidCard :  $invalidCard := invalidCard \cup \{selfATM\}$ 
  end
Event validateCardOK  $\hat{=}$ 
refines validateCardOK

  any
    selfATM    contextual instance of class atm
    c
    p
  where
    p.type :  $p \in Pin$ 
    selfATM.type :  $selfATM \in atm$ 
    c.type :  $c \in ValidCard$ 
    active_atm_SM_isin_validating :  $selfATM \in validating$ 
    validateCardOK.Guard1 :  $selfATM \in dom(atm\_card)$ 
    validateCardOK.Guard2 :  $atm\_card(selfATM) = c$ 
    validateCardOK.Guard3 :  $card\_pin(c) = p$ 
  then
    active_atm_SM_enterSuperState_transOption :  $transOption := transOption \cup \{selfATM\}$ 
    active_atm_SM_leaveState_validating :  $validating := validating \setminus \{selfATM\}$ 
    transOption_SM_enterState_trans :  $trans := trans \cup \{selfATM\}$ 
  end
Event retry  $\hat{=}$ 
refines retry

  any
    selfATM    contextual instance of class atm
  where
    selfATM.type :  $selfATM \in atm$ 
    active_atm_SM_isin_invalidCard :  $selfATM \in invalidCard$ 
    retry.Guard1 :  $selfATM \in dom(atm\_card)$ 
  then
    active_atm_SM_leaveState_invalidCard :  $invalidCard := invalidCard \setminus \{selfATM\}$ 
    active_atm_SM_enterState_validating :  $validating := validating \cup \{selfATM\}$ 
  end
Event doAnother  $\hat{=}$ 
refines doAnother

  any
    selfATM    contextual instance of class atm
  where
    selfATM.type :  $selfATM \in atm$ 
    performTrans_SM_isin_endTrans :  $selfATM \in endTrans$ 

```



```

    doAnother.Guard1 :  $selfATM \in dom(atm\_card)$ 
  then
    active_atm.SM_enterSuperState_transOption :  $transOption := transOption \cup \{selfATM\}$ 
    active_atm.SM_leaveSuperState_performTrans :  $performTrans := performTrans \setminus \{selfATM\}$ 
    performTrans.SM_leaveState_endTrans :  $endTrans := endTrans \setminus \{selfATM\}$ 
    transOption.SM_enterState_trans :  $trans := trans \cup \{selfATM\}$ 
  end

Event  $sendReqWD \triangleq$ 

refines  $requestWD$ 

  any
     $selfATM$       contextual instance of class atm
     $am$ 
  where
     $selfATM.type : selfATM \in atm$ 
     $am.type : am \in \mathbb{N}$ 
     $transOption.SM\_isin\_trans : selfATM \in trans$ 
     $sendReqWD.Guard3 : selfATM \in dom(atm\_card)$ 
     $sendReqWD.Guard1 : atm\_cashA(selfATM) > MIN\_CASH$ 
     $sendReqWD.Guard5 : am \leq MIN\_CASH$ 
  then
     $transOption.SM\_enterSuperState\_reqWD : reqWD := reqWD \cup \{selfATM\}$ 
     $transOption.SM\_leaveState\_trans : trans := trans \setminus \{selfATM\}$ 
     $reqWD.SM\_enterState\_sentReqWD : sentReqWD := sentReqWD \cup \{selfATM\}$ 
     $sendReqWD.Action1 : atm\_wdam(selfATM) := am$ 
  end

Event  $sendReqCB \triangleq$ 

refines  $requestCB$ 

  any
     $selfATM$       contextual instance of class atm
  where
     $selfATM.type : selfATM \in atm$ 
     $transOption.SM\_isin\_trans : selfATM \in trans$ 
     $sendReqCB.Guard1 : selfATM \in dom(atm\_card)$ 
  then
     $transOption.SM\_enterSuperState\_reqCB : reqCB := reqCB \cup \{selfATM\}$ 
     $transOption.SM\_leaveState\_trans : trans := trans \setminus \{selfATM\}$ 
     $reqCB.SM\_enterState\_sentReqCB : sentReqCB := sentReqCB \cup \{selfATM\}$ 
  end

Event  $recvReqWD \triangleq$ 

  any
     $selfATM$       contextual instance of class atm
  where
     $selfATM.type : selfATM \in atm$ 
     $reqWD.SM\_isin\_sentReqWD : selfATM \in sentReqWD$ 

```

```

    recvReqWD.Guard1 :  $selfATM \in dom(atm\_wdam)$ 
    recvReqWD.Guard2 :  $selfATM \in dom(atm\_card)$ 
  then
    reqWD.SM_leaveState_sentReqWD :  $sentReqWD := sentReqWD \setminus \{selfATM\}$ 
    reqWD.SM_enterState_recvdReqWD :  $recvdReqWD := recvdReqWD \cup \{selfATM\}$ 
    recvReqWD.Action1 :  $atmB := atmB \cup \{selfATM\}$ 
    recvReqWD.Action2 :  $atm\_cardB(selfATM) := atm\_card(selfATM)$ 
  end
Event  $recvReqCB \hat{=}$ 
  any
     $selfATM$       contextual instance of class atm
  where
    selfATM.type :  $selfATM \in atm$ 
    reqCB.SM_isin_sentReqCB :  $selfATM \in sentReqCB$ 
    recvReqCB.Guard1 :  $selfATM \in dom(atm\_card)$ 
  then
    reqCB.SM_leaveState_sentReqCB :  $sentReqCB := sentReqCB \setminus \{selfATM\}$ 
    reqCB.SM_enterState_recvdReqCB :  $recvdReqCB := recvdReqCB \cup \{selfATM\}$ 
    recvReqCB.Action2 :  $atm\_cardB(selfATM) := atm\_card(selfATM)$ 
    recvReqCB.Action1 :  $atmB := atmB \cup \{selfATM\}$ 
  end
Event  $recvRspWDFail \hat{=}$ 
refines  $responseWDFail$ 
  any
     $selfATM$       contextual instance of class atm
     $b$ 
  where
    selfATM.type :  $selfATM \in atm$ 
     $b.type : b \in \mathbb{N}$ 
    processedWDFail.SM_isin_sentRspWDFail :  $selfATM \in sentRspWDFail$ 
    recvRspWDFail.Guard1 :  $selfATM \in dom(atm\_card)$ 
  then
    performTrans.SM_leaveSuperState_processedWDFail :  $processedWDFail := processedWDFail \setminus \{selfATM\}$ 
    processedWDFail.SM_leaveState_sentRspWDFail :  $sentRspWDFail := sentRspWDFail \setminus \{selfATM\}$ 
    performTrans.SM_enterState_rspWDFail :  $rspWDFail := rspWDFail \cup \{selfATM\}$ 
    recvRspWDFail.Action1 :  $atm\_acbaA(selfATM) := b$ 
  end
Event  $recvRspWDOK \hat{=}$ 
refines  $responseWDOK$ 
  any
     $selfATM$       contextual instance of class atm
     $b$ 

```

where

$\text{selfATM.type} : \text{selfATM} \in \text{atm}$
 $\text{b.type} : b \in \mathbb{N}$
 $\text{processedWDOK.SM.isin_sentRspWDOK} : \text{selfATM} \in \text{sentRspWDOK}$
 $\text{recvRspWDOK.Guard1} : \text{selfATM} \in \text{dom}(\text{atm_card})$

then

$\text{performTrans.SM.leaveSuperState_processedWDOK} : \text{processedWDOK} := \text{processedWDOK} \setminus \{\text{selfATM}\}$
 $\text{processedWDOK.SM.leaveState_sentRspWDOK} : \text{sentRspWDOK} := \text{sentRspWDOK} \setminus \{\text{selfATM}\}$
 $\text{performTrans.SM.enterState_rspWDOK} : \text{rspWDOK} := \text{rspWDOK} \cup \{\text{selfATM}\}$
 $\text{recvRspWDOK.Action1} : \text{atm_acbalA}(\text{selfATM}) := b$

end

Event $\text{recvRspCB} \triangleq$

refines responseCB

any

selfATM contextual instance of class atm
 b

where

$\text{selfATM.type} : \text{selfATM} \in \text{atm}$
 $\text{b.type} : b \in \mathbb{N}$
 $\text{processedCB.SM.isin_sentRspCB} : \text{selfATM} \in \text{sentRspCB}$
 $\text{recvRspCB.Guard1} : \text{selfATM} \in \text{dom}(\text{atm_card})$

then

$\text{performTrans.SM.leaveSuperState_processedCB} : \text{processedCB} := \text{processedCB} \setminus \{\text{selfATM}\}$
 $\text{processedCB.SM.leaveState_sentRspCB} : \text{sentRspCB} := \text{sentRspCB} \setminus \{\text{selfATM}\}$
 $\text{performTrans.SM.enterState_rspCB} : \text{rspCB} := \text{rspCB} \cup \{\text{selfATM}\}$
 $\text{recvRspCB.Action1} : \text{atm_acbalA}(\text{selfATM}) := b$

end

Event $\text{withdrawATMOK} \triangleq$

refines withdrawATMOK

any

selfATM contextual instance of class atm
 b
 am

where

$\text{am.type} : am \in \mathbb{N}$
 $\text{selfATM.type} : \text{selfATM} \in \text{atm}$
 $\text{b.type} : b \in \mathbb{N}$
 $\text{performTrans.SM.isin_rspWDOK} : \text{selfATM} \in \text{rspWDOK}$
 $\text{withdrawATMOK.Guard1} : \text{selfATM} \in \text{dom}(\text{atm_card})$
 $\text{withdrawATMOK.Guard2} : \text{selfATM} \in \text{dom}(\text{atm_wdam})$
 $\text{withdrawATMOK.Guard3} : \text{selfATM} \in \text{dom}(\text{atm_acbalA})$
 $\text{withdrawATMOK.Guard5} : \text{atm_wdam}(\text{selfATM}) = am$
 $\text{withdrawATMOK.Guard7} : \text{atm_cashA}(\text{selfATM}) \geq am$
 $\text{withdrawATMOK.Guard4} : \text{atm_acbalA}(\text{selfATM}) = b$

```

then
  performTrans.SM_leaveState_rspWDOK :  $rspWDOK := rspWDOK \setminus \{selfATM\}$ 
  performTrans.SM_enterState_endTrans :  $endTrans := endTrans \cup \{selfATM\}$ 
  withdrawATMOK.Action2 :  $atm\_cashA(selfATM) := atm\_cashA(selfATM) - am$ 
end

Event withdrawATMFail  $\hat{=}$ 

refines withdrawATMFail

any
  selfATM      contextual instance of class atm
  b
where
  selfATM.type :  $selfATM \in atm$ 
  b.type :  $b \in \mathbb{N}$ 
  performTrans.SM_isin_rspWDFail :  $selfATM \in rspWDFail$ 
  withdrawATMFail.Guard1 :  $selfATM \in dom(atm\_card)$ 
  withdrawATMFail.Guard2 :  $selfATM \in dom(atm\_acbalA)$ 
  withdrawATMFail.Guard3 :  $atm\_acbalA(selfATM) = b$ 

then
  performTrans.SM_leaveState_rspWDFail :  $rspWDFail := rspWDFail \setminus \{selfATM\}$ 
  performTrans.SM_enterState_endTrans :  $endTrans := endTrans \cup \{selfATM\}$ 
end

Event checkBalATM  $\hat{=}$ 

refines checkBalATM

any
  selfATM      contextual instance of class atm
  b
where
  selfATM.type :  $selfATM \in atm$ 
  b.type :  $b \in \mathbb{N}$ 
  performTrans.SM_isin_rspCB :  $selfATM \in rspCB$ 
  checkBalATM.Guard1 :  $selfATM \in dom(atm\_card)$ 
  checkBalATM.Guard2 :  $selfATM \in dom(atm\_acbalA)$ 
  checkBalATM.Guard3 :  $atm\_acbalA(selfATM) = b$ 

then
  performTrans.SM_leaveState_rspCB :  $rspCB := rspCB \setminus \{selfATM\}$ 
  performTrans.SM_enterState_endTrans :  $endTrans := endTrans \cup \{selfATM\}$ 
end

Event sendRspWDFail  $\hat{=}$ 

any
  selfATM      contextual instance of class atm
where
  selfATM.type :  $selfATM \in atm$ 
  processedWDFail.SM_isin_processWDFail :  $selfATM \in processWDFail$ 
  sendRspWDFail.Guard1 :  $selfATM \in dom(atm\_cardB)$ 

```

```

    sendRspWDFail.Guard2 :  $selfATM \in dom(atm\_acbalB)$ 
    sendRspWDFail.Guard3 :  $selfATM \in atmB$ 

  then
    processedWDFail.SM_leaveState_processWDFail :  $processWDFail := processWDFail \setminus \{selfATM\}$ 
    processedWDFail.SM_enterState_sentRspWDFail :  $sentRspWDFail := sentRspWDFail \cup \{selfATM\}$ 
    sendRspWDFail.Action1 :  $atmB := atmB \setminus \{selfATM\}$ 
    sendRspWDFail.Action2 :  $atm\_acbalB := \{selfATM\} \triangleleft atm\_acbalB$ 
    sendRspWDFail.Action3 :  $atm\_cardB := \{selfATM\} \triangleleft atm\_cardB$ 
  end

Event sendRspWDOK  $\hat{=}$ 

  any
     $selfATM$       contextual instance of class atm
  where
    selfATM.type :  $selfATM \in atm$ 
    processedWDOK.SM_isin_processWDOK :  $selfATM \in processWDOK$ 
    sendRspWDOK.Guard1 :  $selfATM \in dom(atm\_cardB)$ 
    sendRspWDOK.Guard2 :  $selfATM \in dom(atm\_acbalB)$ 
    sendRspWDOK.Guard3 :  $selfATM \in atmB$ 
  then
    processedWDOK.SM_leaveState_processWDOK :  $processWDOK := processWDOK \setminus \{selfATM\}$ 
    processedWDOK.SM_enterState_sentRspWDOK :  $sentRspWDOK := sentRspWDOK \cup \{selfATM\}$ 
    sendRspWDOK.Action1 :  $atmB := atmB \setminus \{selfATM\}$ 
    sendRspWDOK.Action2 :  $atm\_acbalB := \{selfATM\} \triangleleft atm\_acbalB$ 
    sendRspWDOK.Action3 :  $atm\_cardB := \{selfATM\} \triangleleft atm\_cardB$ 
  end

Event sendRspCB  $\hat{=}$ 

  any
     $selfATM$       contextual instance of class atm
  where
    selfATM.type :  $selfATM \in atm$ 
    processedCB.SM_isin_processCB :  $selfATM \in processCB$ 
    sendRspCB.Guard1 :  $selfATM \in dom(atm\_cardB)$ 
    sendRspCB.Guard2 :  $selfATM \in dom(atm\_acbalB)$ 
    sendRspCB.Guard3 :  $selfATM \in atmB$ 
  then
    processedCB.SM_leaveState_processCB :  $processCB := processCB \setminus \{selfATM\}$ 
    processedCB.SM_enterState_sentRspCB :  $sentRspCB := sentRspCB \cup \{selfATM\}$ 
    sendRspCB.Action1 :  $atmB := atmB \setminus \{selfATM\}$ 
    sendRspCB.Action2 :  $atm\_acbalB := \{selfATM\} \triangleleft atm\_acbalB$ 
    sendRspCB.Action3 :  $atm\_cardB := \{selfATM\} \triangleleft atm\_cardB$ 
  end

END

```