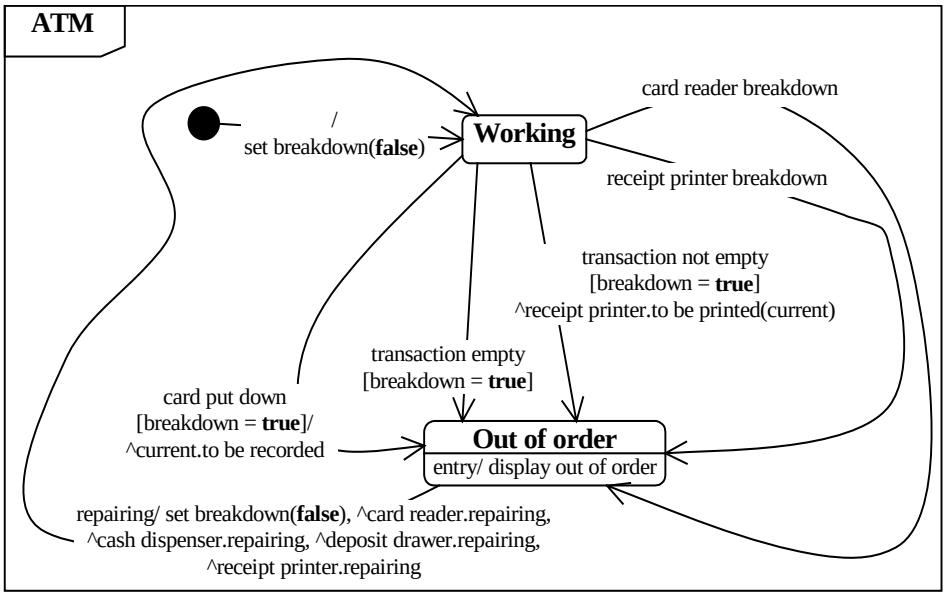
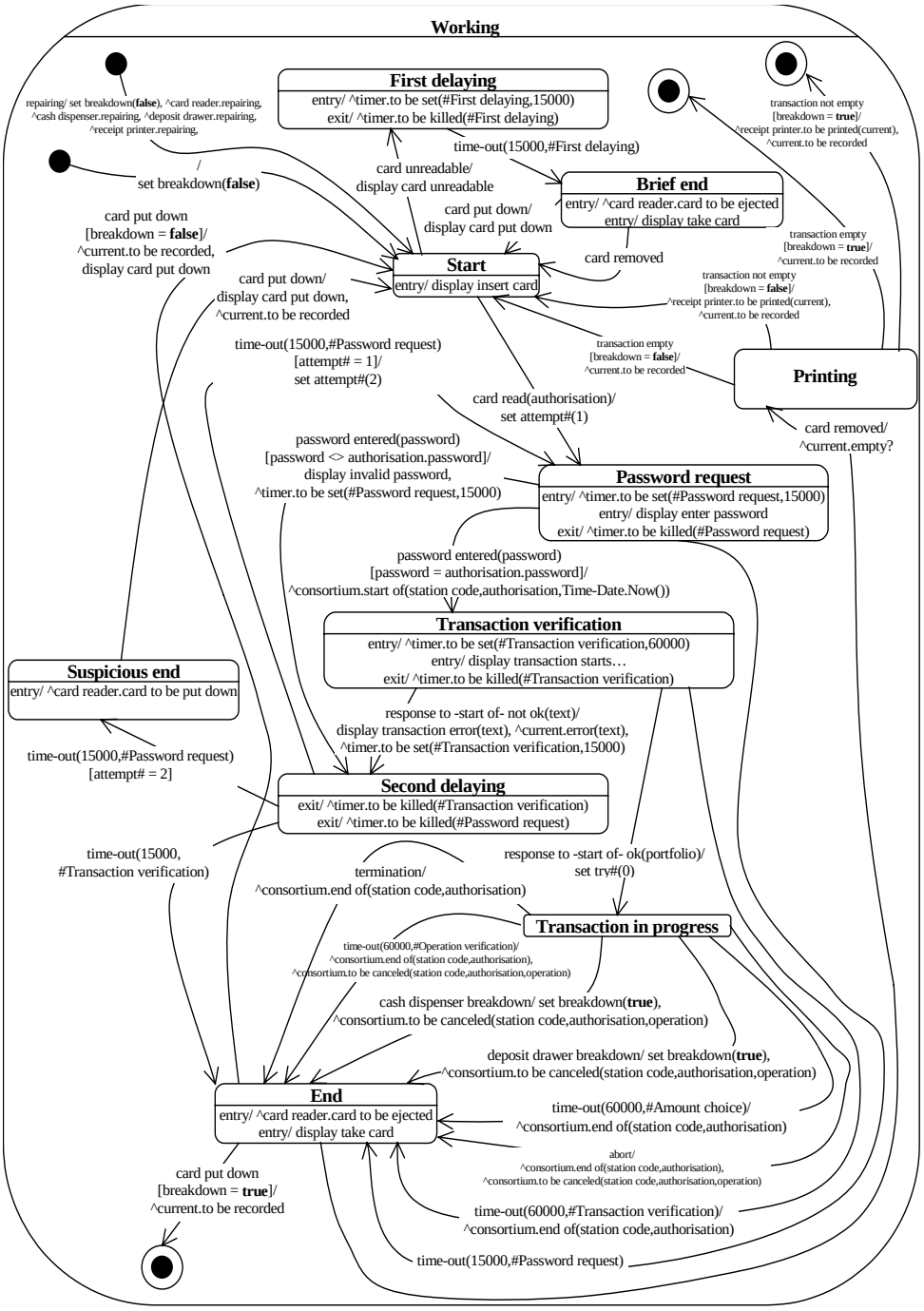
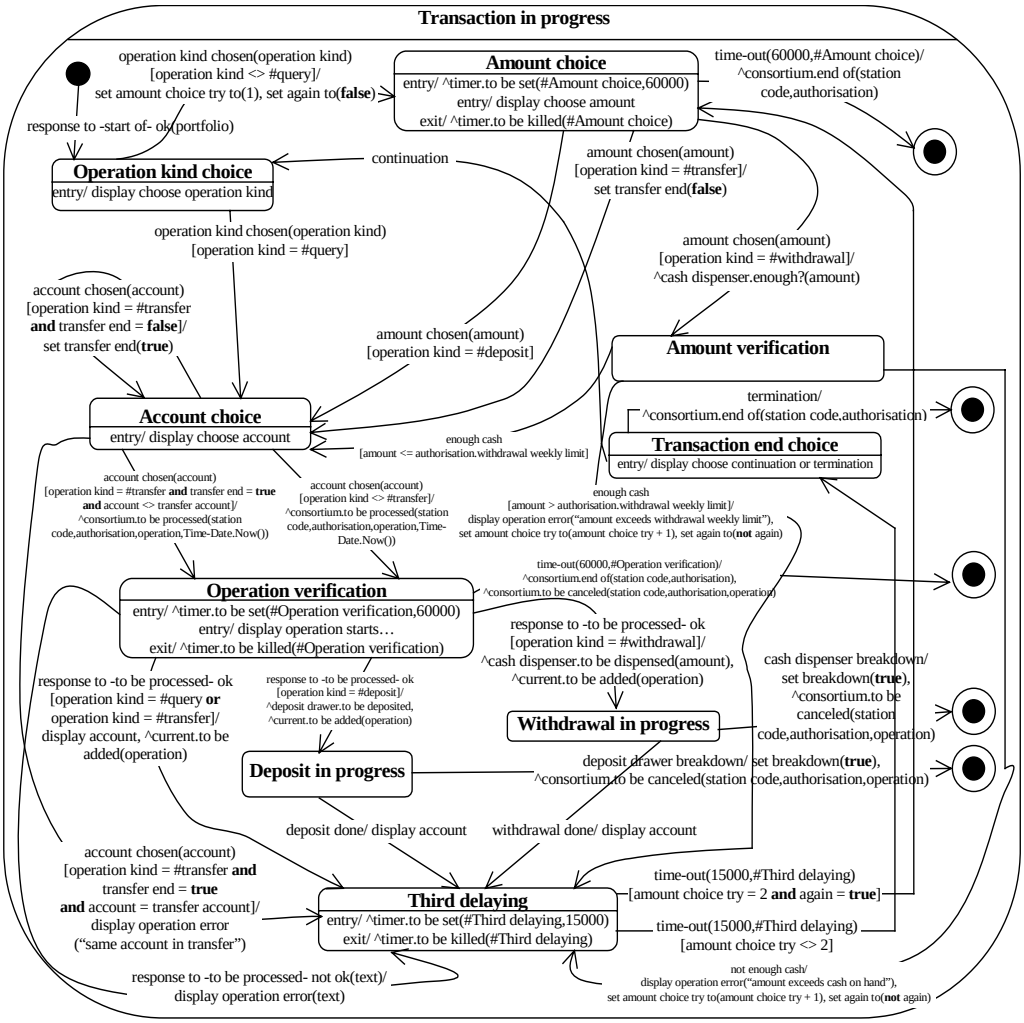

ATM
/
set breakdown(**false**)
Working
card reader breakdown
receipt printer breakdown
transaction not empty
[breakdown = **true**]
^receipt printer.to be printed(current)
transaction empty
[breakdown = **true**]
card put down
[breakdown = **true**]/
^current.to be recorded
Out of order
entry/ display out of order
repairing/ set breakdown(**false**), ^card reader.repairing,
^cash dispenser.repairing, ^deposit drawer.repairing,
^receipt printer.repairing


Working
First delaying
entry/ ^timer.to be set(#First delaying,15000)
exit/ ^timer.to be killed(#First delaying)
repairing/ set breakdown(**false**), ^card reader.repairing, ^cash dispenser.repairing, ^deposit drawer.repairing, ^receipt printer.repairing,
/
set breakdown(**false**)
time-out(15000,#First delaying)
card unreadable/
display card unreadable
Brief end
entry/ ^card reader.card to be ejected
entry/ display take card
card put down/
display card put down
card removed
transaction not empty
[breakdown = **true**]/
^receipt printer.to be printed(current),
^current.to be recorded
transaction empty
[breakdown = **true**]/
^current.to be recorded
card put down
[breakdown = **false**]/
^current.to be recorded,
display card put down
Start
entry/ display insert card
card put down/
display card put down,
^current.to be recorded
transaction not empty
[breakdown = **false**]/
^receipt printer.to be printed(current),
^current.to be recorded
time-out(15000,#Password request)
[attempt# = 1]/
set attempt#(2)
transaction empty
[breakdown = **false**]/
^current.to be recorded
Printing
card read(authorisation)/
set attempt#(1)
card removed/
^current.empty?
password entered(password)
[password <> authorisation.password]/
display invalid password,
^timer.to be set(#Password request,15000)
Password request
entry/ ^timer.to be set(#Password request,15000)
entry/ display enter password
exit/ ^timer.to be killed(#Password request)
password entered(password)
[password = authorisation.password]/
^consortium.start of(station code,authorisation,Time-Date.Now())
Transaction verification
entry/ ^timer.to be set(#Transaction verification,60000)
entry/ display transaction starts…
exit/ ^timer.to be killed(#Transaction verification)
Suspicious end
entry/ ^card reader.card to be put down
response to -start of- not ok(text)/
display transaction error(text), ^current.error(text),
^timer.to be set(#Transaction verification,15000)
time-out(15000,#Password request)
[attempt# = 2]
Second delaying
exit/ ^timer.to be killed(#Transaction verification)
exit/ ^timer.to be killed(#Password request)
time-out(15000,
#Transaction verification)
termination/
^consortium.end of(station code,authorisation)
response to -start of- ok(portfolio)/
set try#(0)
Transaction in progress
time-out(60000,#Operation verification)/
^consortium.end of(station code,authorisation),
^consortium.to be canceled(station code,authorisation,operation)
cash dispenser breakdown/ set breakdown(**true**),
^consortium.to be canceled(station code,authorisation,operation)
deposit drawer breakdown/ set breakdown(**true**),
^consortium.to be canceled(station code,authorisation,operation)
End
entry/ ^card reader.card to be ejected
entry/ display take card
time-out(60000,#Amount choice)/
^consortium.end of(station code,authorisation)
abort/
^consortium.end of(station code,authorisation),
^consortium.to be canceled(station code,authorisation,operation)
card put down
[breakdown = **true**]/
^current.to be recorded
time-out(60000,#Transaction verification)/
^consortium.end of(station code,authorisation)
time-out(15000,#Password request)


Transaction in progress
Amount choice
entry/ ^timer.to be set(#Amount choice,60000)
entry/ display choose amount
exit/ ^timer.to be killed(#Amount choice)
operation kind chosen(operation kind)
[operation kind <> #query]/
set amount choice try to(1), set again to(false)
time-out(60000,#Amount choice)/
^consortium.end of(station
code,authorisation)
response to -start of- ok(portfolio)
Operation kind choice
entry/ display choose operation kind
continuation
amount chosen(amount)
[operation kind = #transfer]/
set transfer end(false)
operation kind chosen(operation kind)
[operation kind = #query]
amount chosen(amount)
[operation kind = #withdrawal]/
^cash dispenser.enough?(amount)
account chosen(account)
[operation kind = #transfer
and transfer end = false]/
set transfer end(true)
amount chosen(amount)
[operation kind = #deposit]
Amount verification
termination/
^consortium.end of(station code,authorisation)
Account choice
entry/ display choose account
enough cash
[amount <= authorisation.withdrawal weekly limit]
Transaction end choice
entry/ display choose continuation or termination
account chosen(account)
[operation kind = #transfer and transfer end = true
and account <> transfer account]/
^consortium.to be processed(station
code,authorisation,operation,Time-Date.Now())
account chosen(account)
[operation kind <> #transfer]/
^consortium.to be processed(station
code,authorisation,operation,Time-
Date.Now())
enough cash
[amount > authorisation.withdrawal weekly limit]/
display operation error("amount exceeds withdrawal weekly limit"),
set amount choice try to(amount choice try + 1), set again to(not again)
time-out(60000,#Operation verification)/
^consortium.end of(station code,authorisation),
^consortium.to be canceled(station code,authorisation,operation)
Operation verification
entry/ ^timer.to be set(#Operation verification,60000)
entry/ display operation starts…
exit/ ^timer.to be killed(#Operation verification)
response to -to be processed- ok
[operation kind = #withdrawal]/
^cash dispenser.to be dispensed(amount),
^current.to be added(operation)
cash dispenser breakdown/
set breakdown(true),
^consortium.to be
canceled(station
code,authorisation,operation)
response to -to be processed- ok
[operation kind = #query or
operation kind = #transfer]/
display account, ^current.to be
added(operation)
response to -to be processed- ok
[operation kind = #deposit]/
^deposit drawer.to be deposited,
^current.to be added(operation)
Withdrawal in progress
Deposit in progress
deposit drawer breakdown/ set breakdown(true),
^consortium.to be canceled(station code,authorisation,operation)
deposit done/ display account
withdrawal done/ display account
account chosen(account)
[operation kind = #transfer and
transfer end = true
and account = transfer account]/
display operation error
("same account in transfer")
Third delaying
entry/ ^timer.to be set(#Third delaying,15000)
exit/ ^timer.to be killed(#Third delaying)
time-out(15000,#Third delaying)
[amount choice try = 2 and again = true]
time-out(15000,#Third delaying)
[amount choice try <> 2]
response to -to be processed- not ok(text)/
display operation error(text)
not enough cash/
display operation error("amount exceeds cash on hand"),
set amount choice try to(amount choice try + 1), set again to(not again)