SOFTWARE ARCHITECTURE LAB

LAB REPORT

PREPARED BY

NAME: RUBI SAHU

ROLL NO: 218CS3324

MTECH 1ND SEM

DEPARTMENT OF COMPUTER SCIENCE &ENGINEERING,

NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA-769008

ODISHA

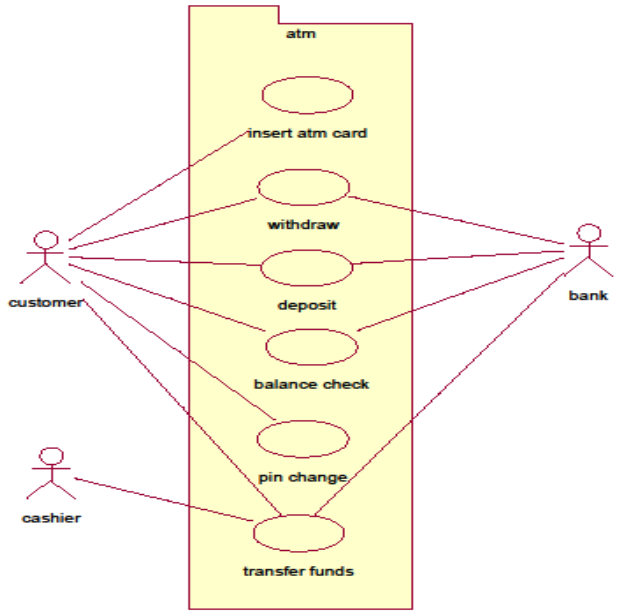
TABLE OF CONTENTS

1 UML DIAGRAMS(RATIONAL ROSE)

1.1 Use case

- A **use case diagram** is a graphic depiction of the interactions among the elements of a system.
- A **use case** is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language as an actor) and a system to achieve a goal. The actor **can** be a human or other external system.

Use case diagram for ATM machine

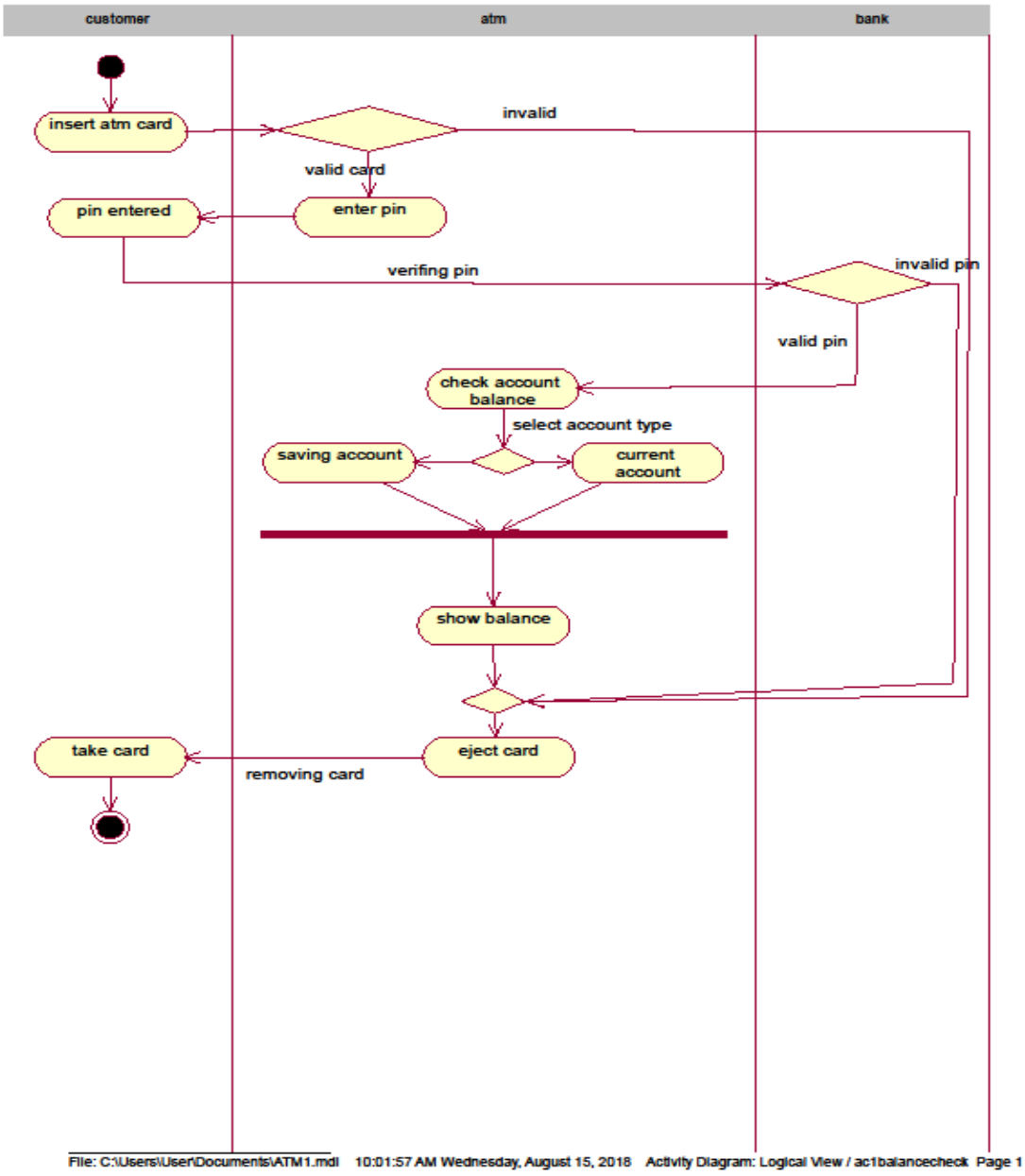


File: C:\Users\User\Documents\ATM1.mdl 11:41:06 AM Saturday, August 11, 2018 Use Case Diagram: Use Case View / Main Page 1

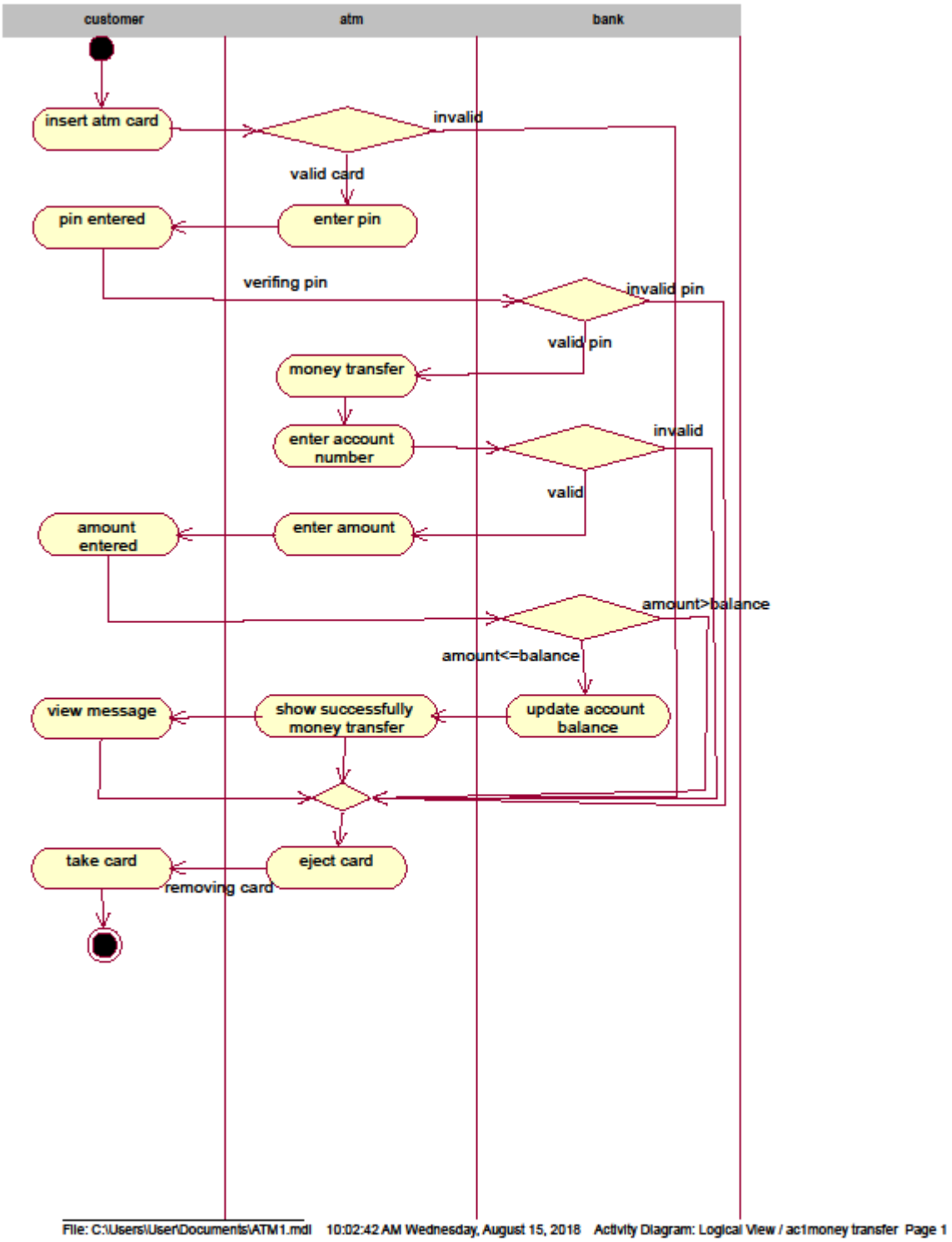
1.1 Activity diagram

Activity diagram is another important **diagram** in UML to describe the dynamic aspects of the system. **Activity diagram** is basically a flowchart to represent the flow from one **activity** to another **activity**.

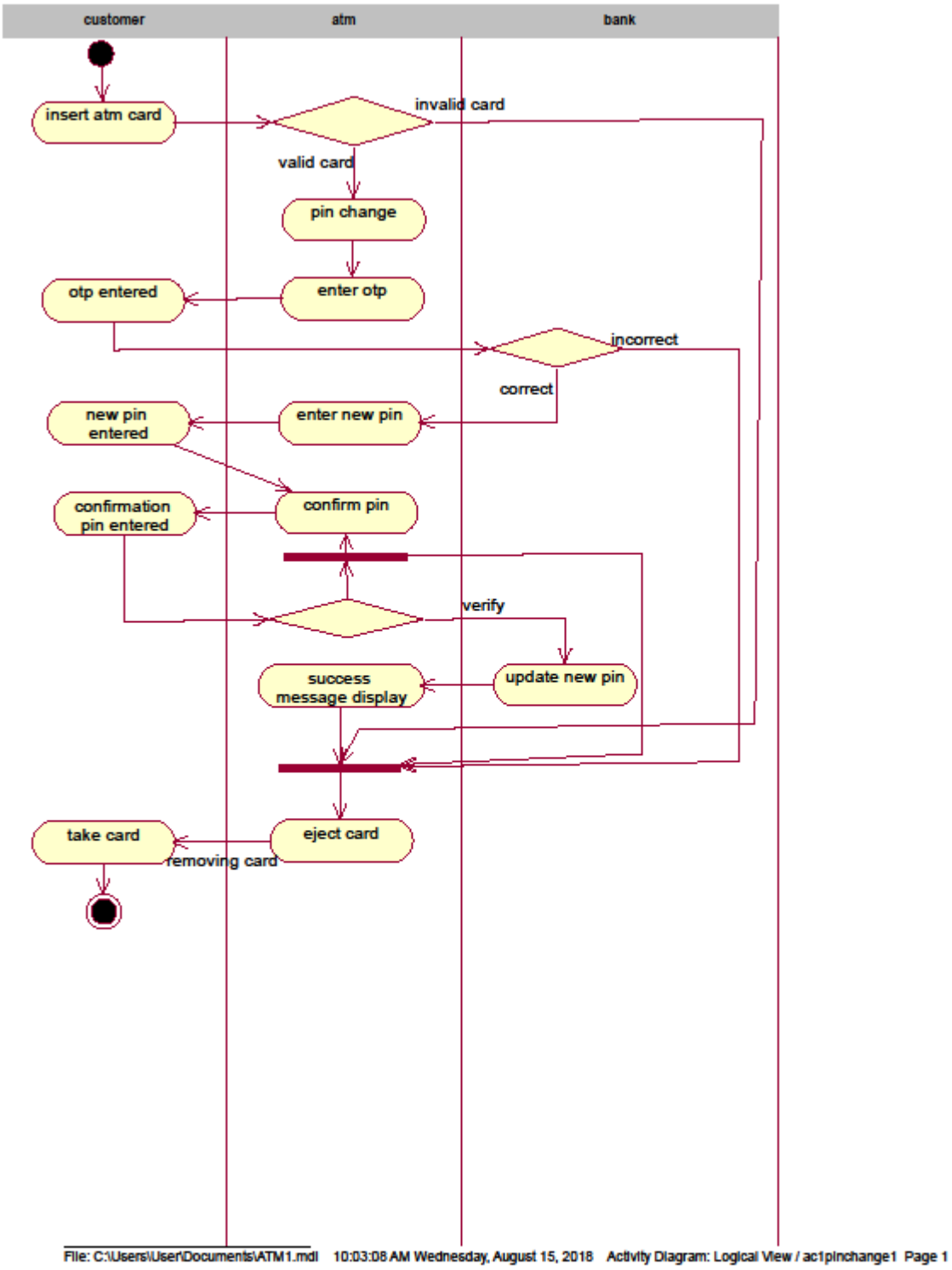
a) Balance check activity diagram



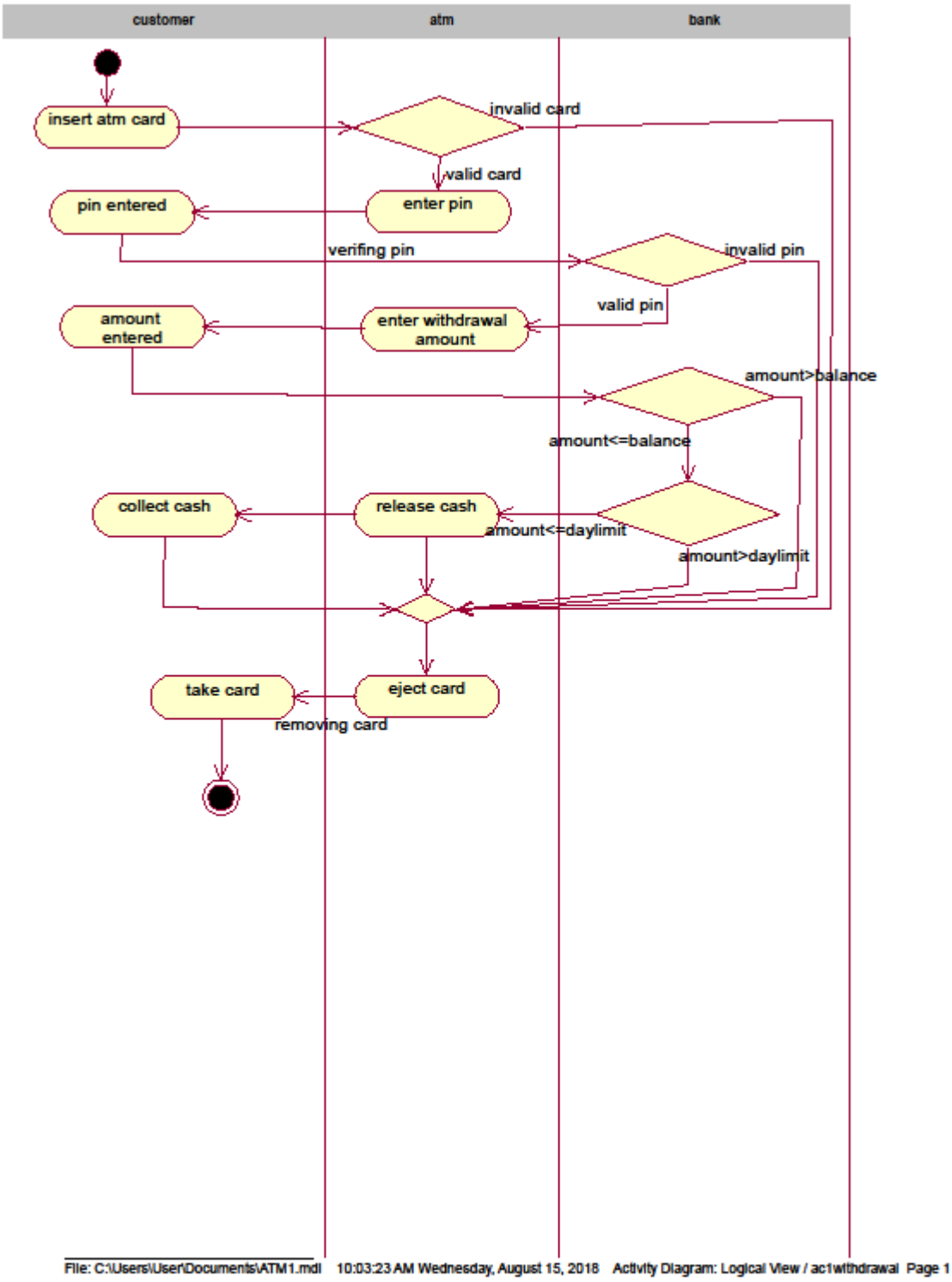
b) Money transfer activity diagram



c) Pin change activity diagram



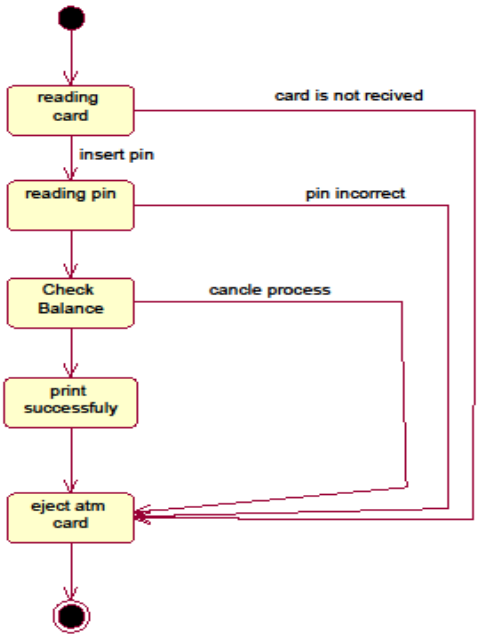
d) Withdraw activity diagram



1.3 State diagram

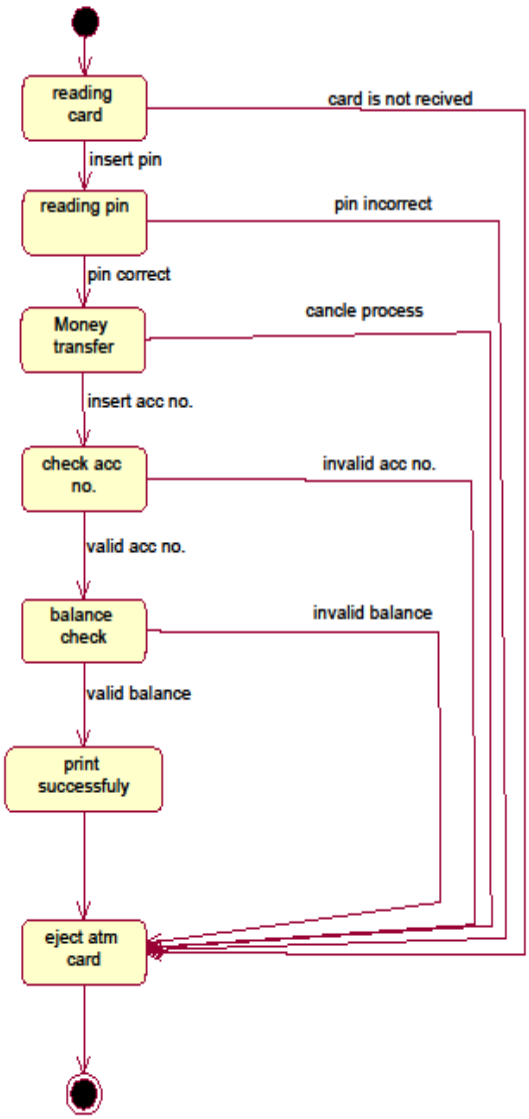
A **state diagram** is a **diagram** used in computer science to describe the behavior of a system considering all the possible **states** of an object when an event occurs. This behavior is represented and analyzed in a series of events that occur in one or more possible **states**.

a) Balance check state diadram



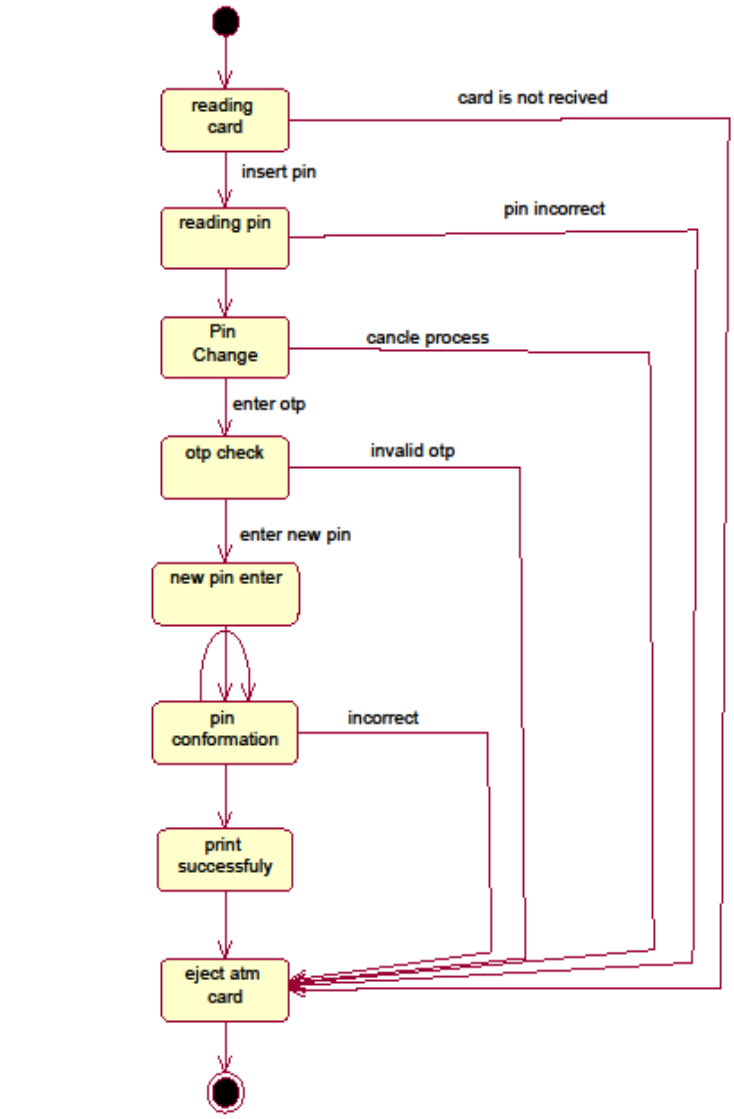
File: C:\Users\User\Documents\ATM1.mdl 10:48:48 AM Wednesday, August 15, 2018 Statechart Diagram: Logical View / st1balance Page 1

b) Money transefor state diagram



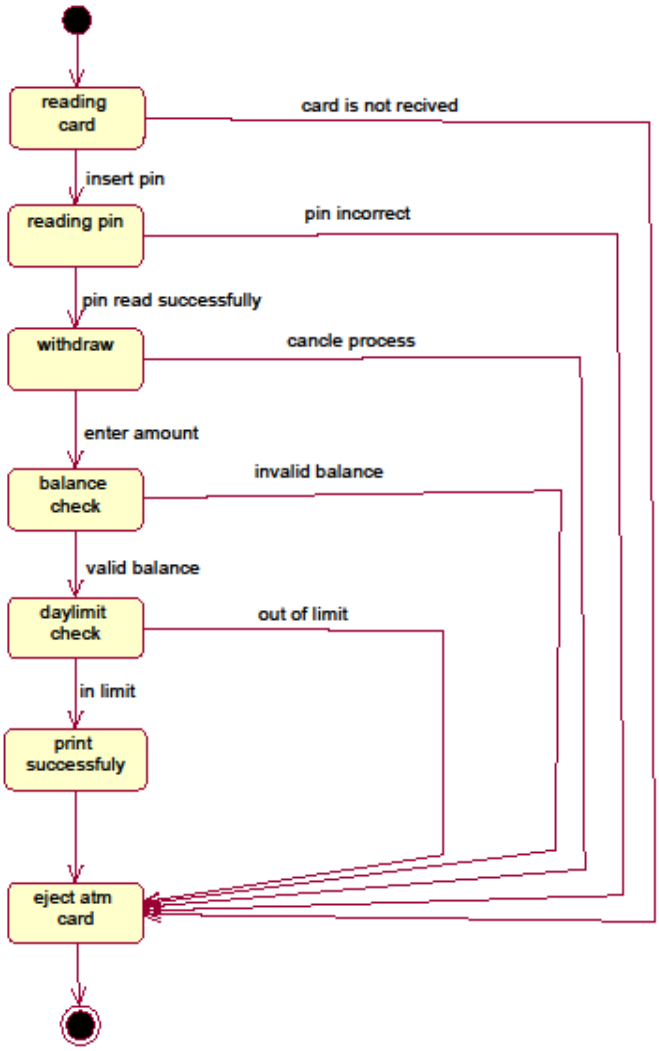
File: C:\Users\User\Documents\ATM1.mdl 10:08:10 AM Wednesday, August 15, 2018 Statechart Diagram: Logical View / st1moneytransfer Page 1

c) Pin change state diagram



File: C:\Users\User\Documents\ATM1.mdl 10:08:59 AM Wednesday, August 15, 2018 Statechart Diagram: Logical View / st1pinchange Page 1

d) Withdram state diagram

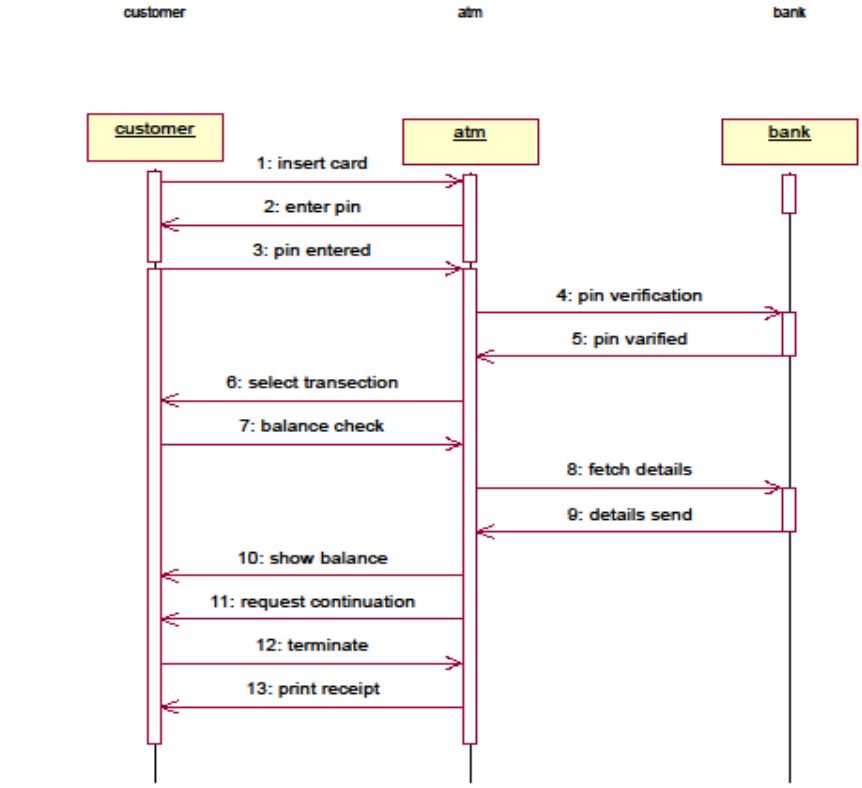


File: C:\Users\User\Documents\ATM1.mdl 10:09:31 AM Wednesday, August 15, 2018 Statechart Diagram: Logical View / st1Withdraw Page 1

1.4Sequence diagram

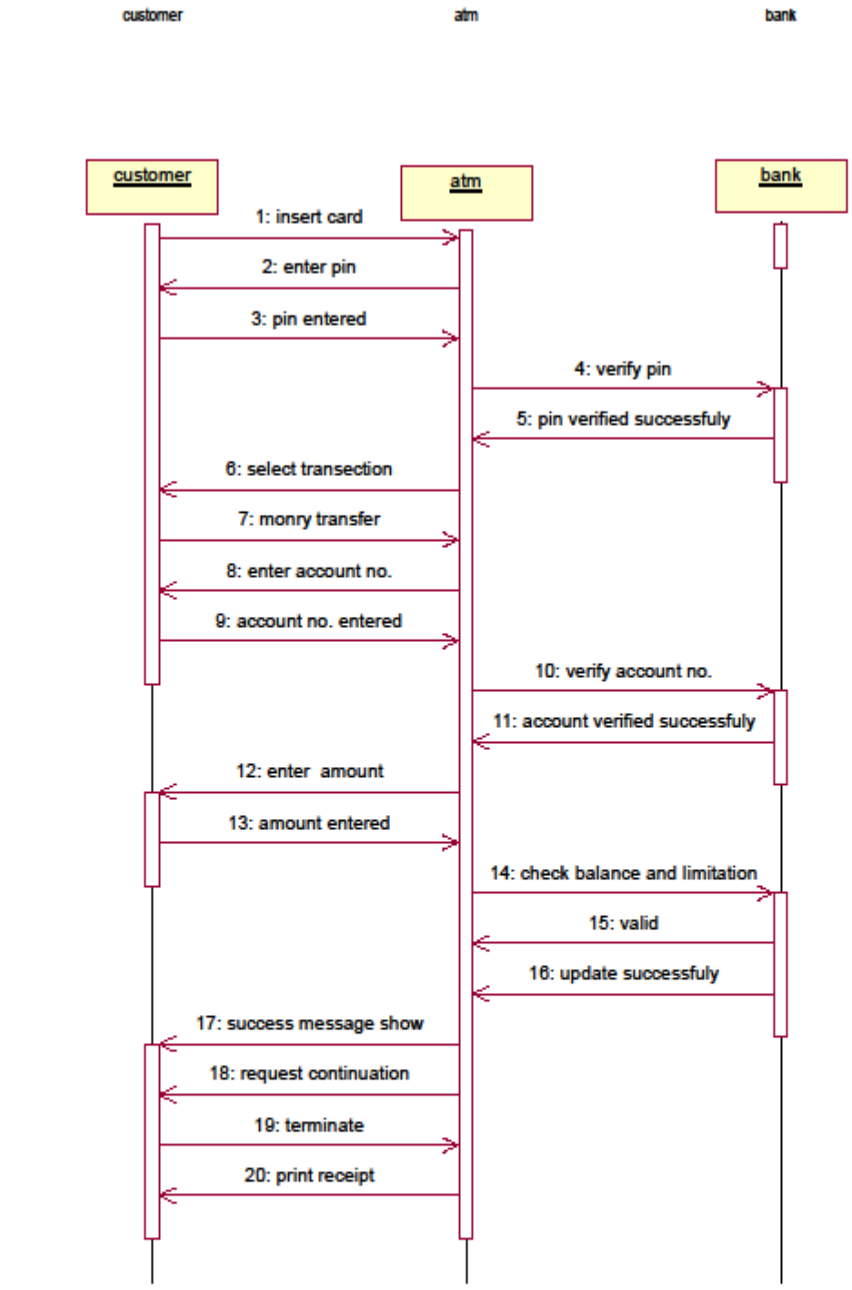
A **sequence diagram** shows object interactions arranged in time **sequence**. It depicts the objects and classes involved in the scenario and the **sequence** of messages exchanged between the objects needed to carry out the functionality of the scenario.

a) Balance check sequence diagram



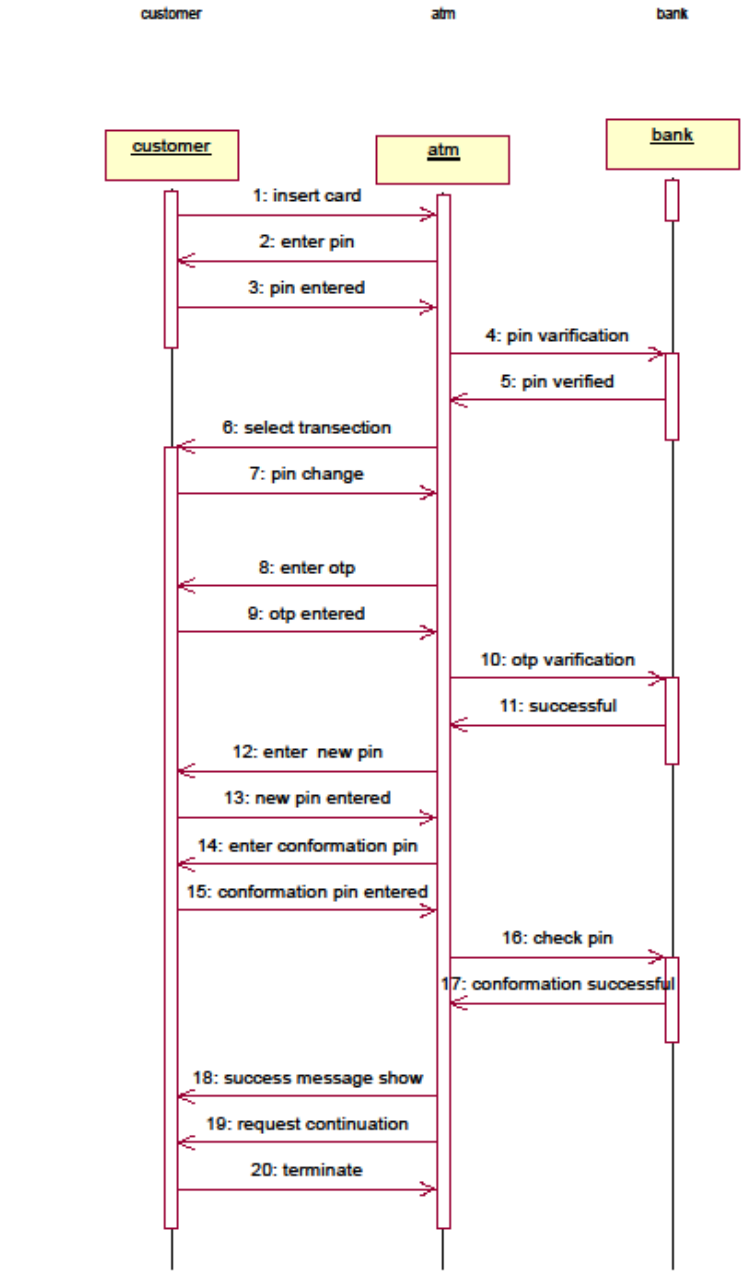
File: C:\Users\User\Documents\ATM1.mdl 9:57:47 AM Wednesday, August 15, 2018 Sequence Diagram: Logical View / se1balance Page 1

b) Money transfer sequence diagram

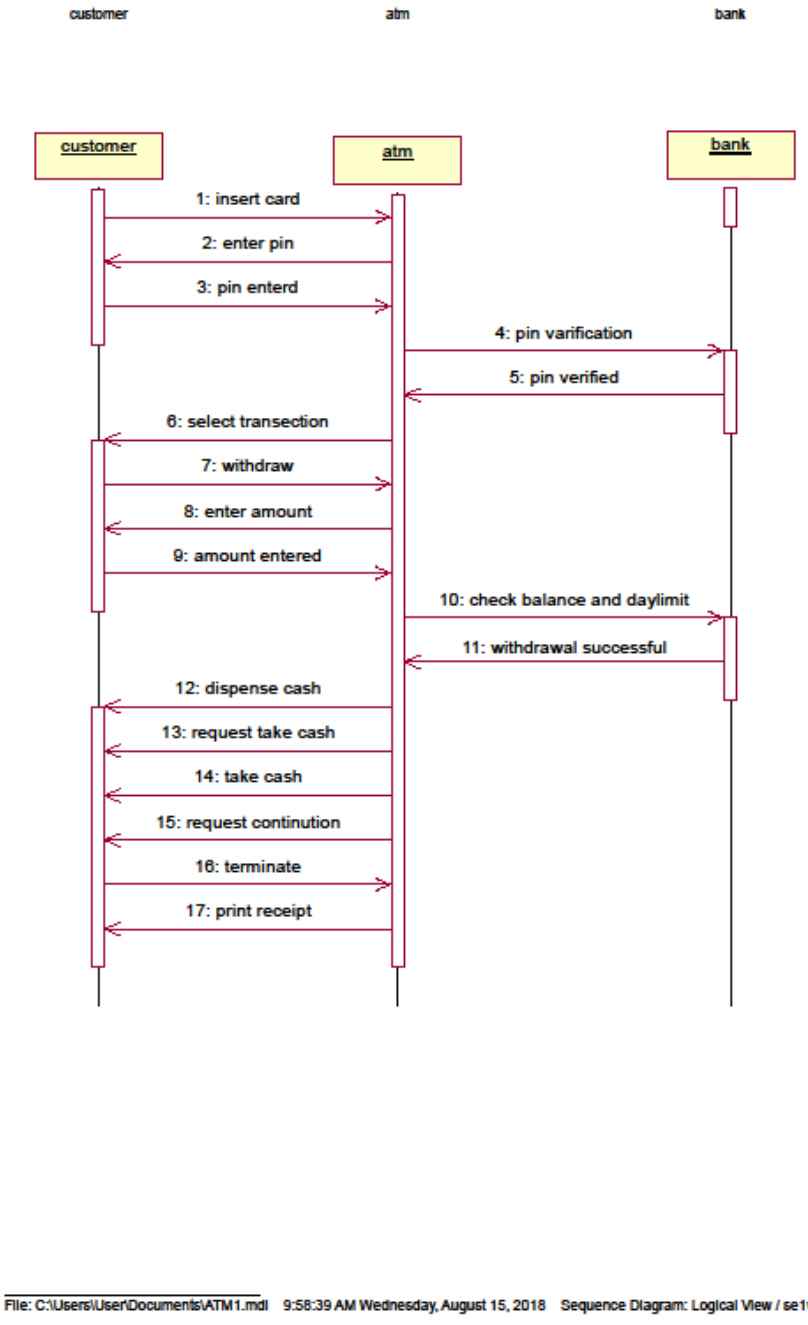


File: C:\Users\User\Documents\ATM1.mdl 11:02:07 AM Wednesday, August 15, 2018 Sequence Diagram: Logical View / se1moneytransfer Page 1

c) Pin change sequence diagram

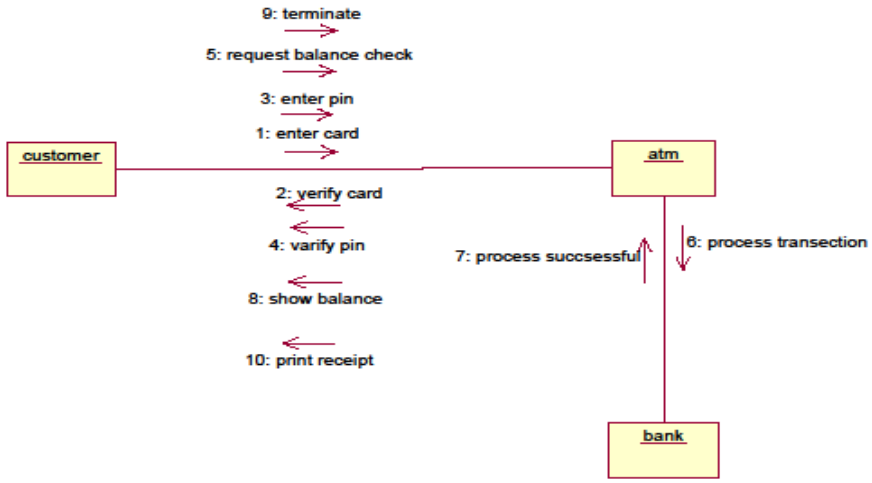


d) Withdraw sequence diagram



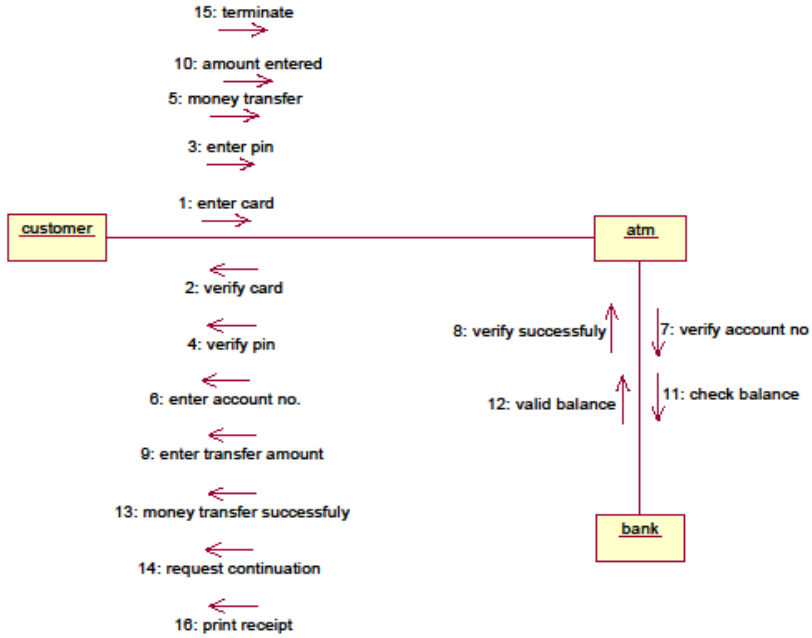
1.5Collaboration diagram

a) Balance check collaboration diagram



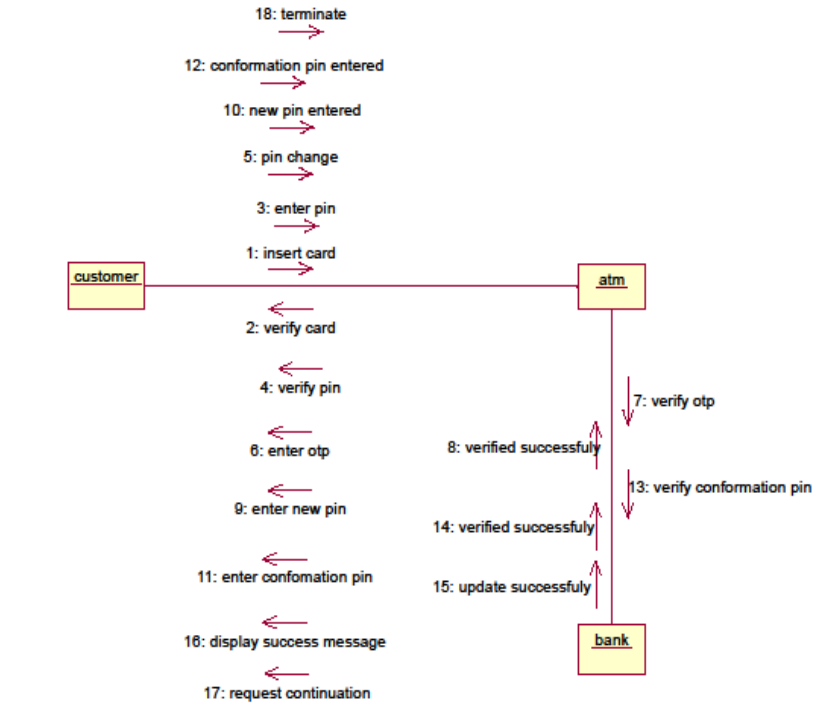
File: C:\Users\User\Documents\ATM1.mdl 9:55:33 AM Wednesday, August 15, 2018 Collaboration Diagram: Logical View / co1balance check
Page 1

b) Money transaction collaboration diagram

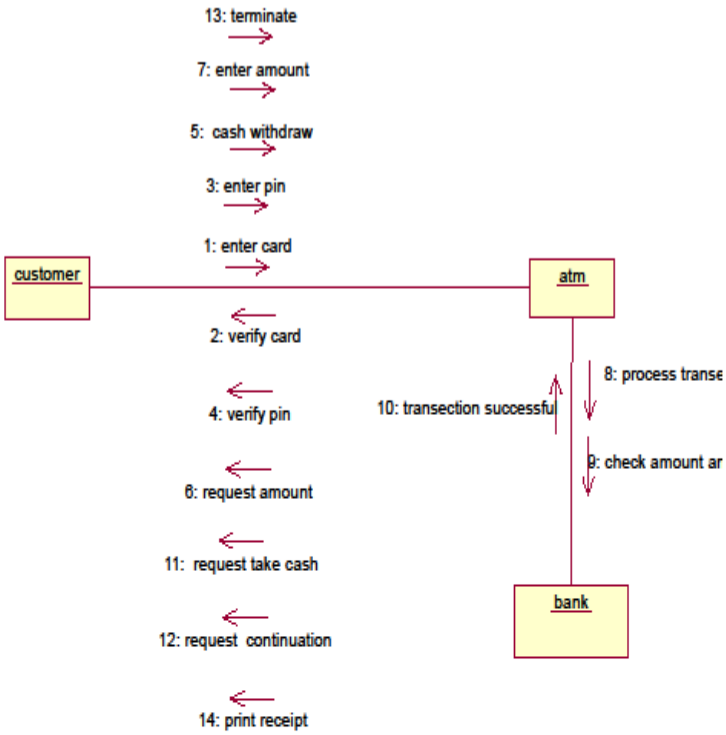


File: C:\Users\User\Documents\ATM1.mdl 9:55:58 AM Wednesday, August 15, 2018 Collaboration Diagram: Logical View / co1moey transfer Page 1

c) Pin change collaboration diagram



d) Withdraw collaboration diagram

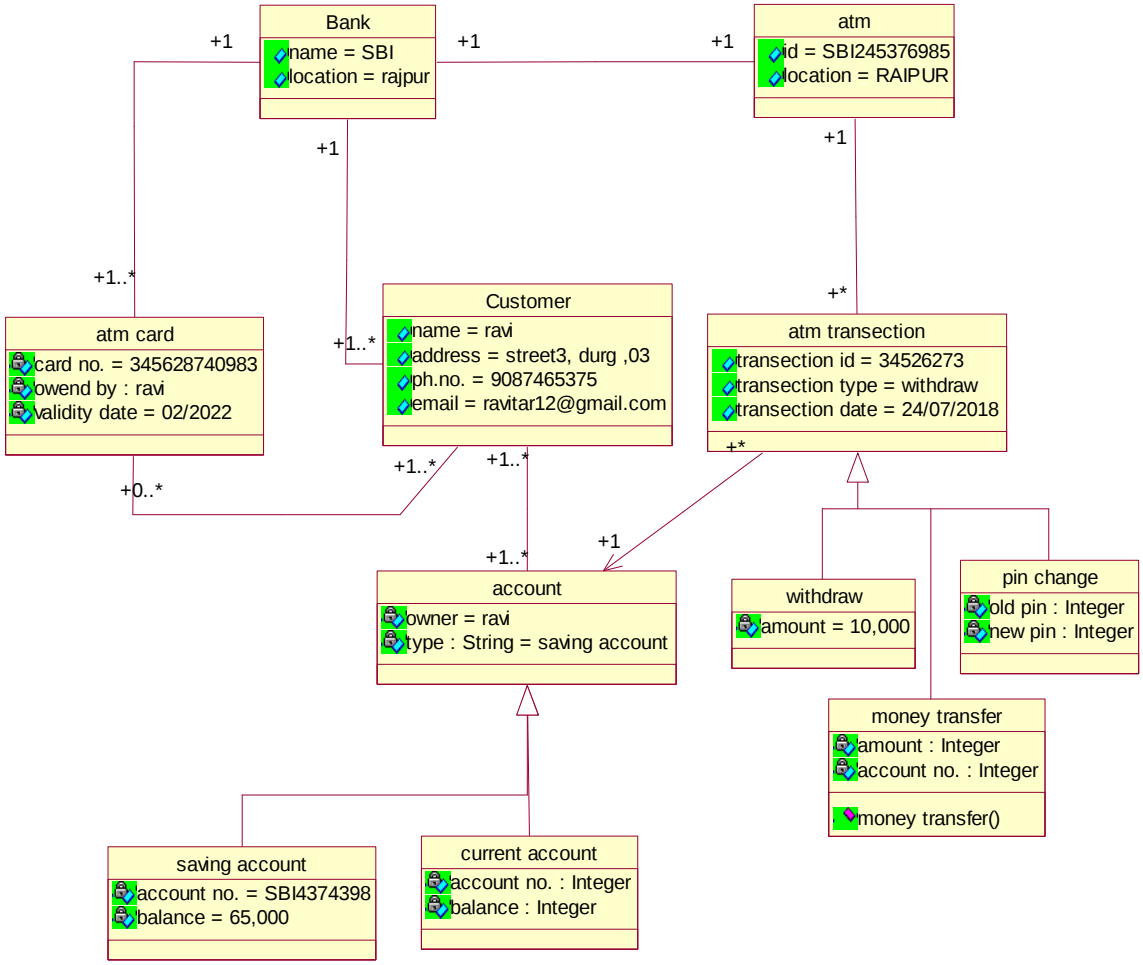


File: C:\Users\User\Documents\ATM1.mdl 9:57:31 AM Wednesday, August 15, 2018 Collaboration Diagram: Logical View / co1withdraw Page 1 (1, 1)

1.6 Class diagram

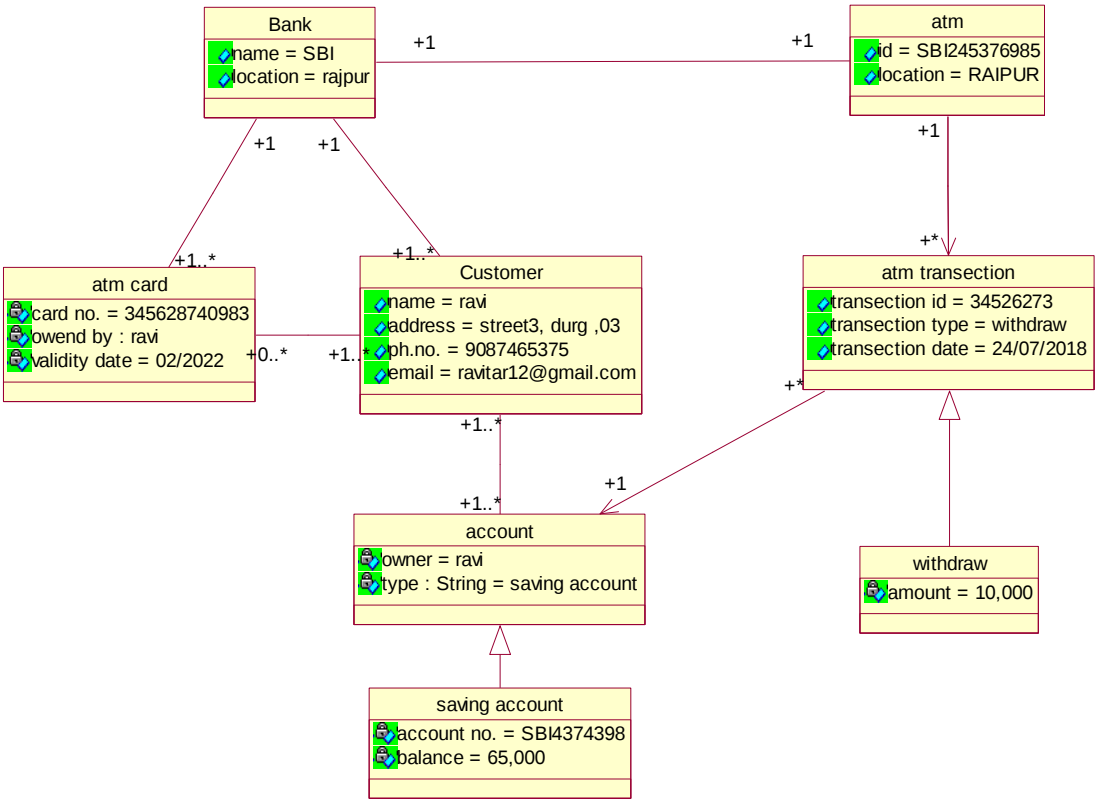
A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's *classes*, their attributes, operations (or methods), and the relationships among objects.

Class diagram for atm machine



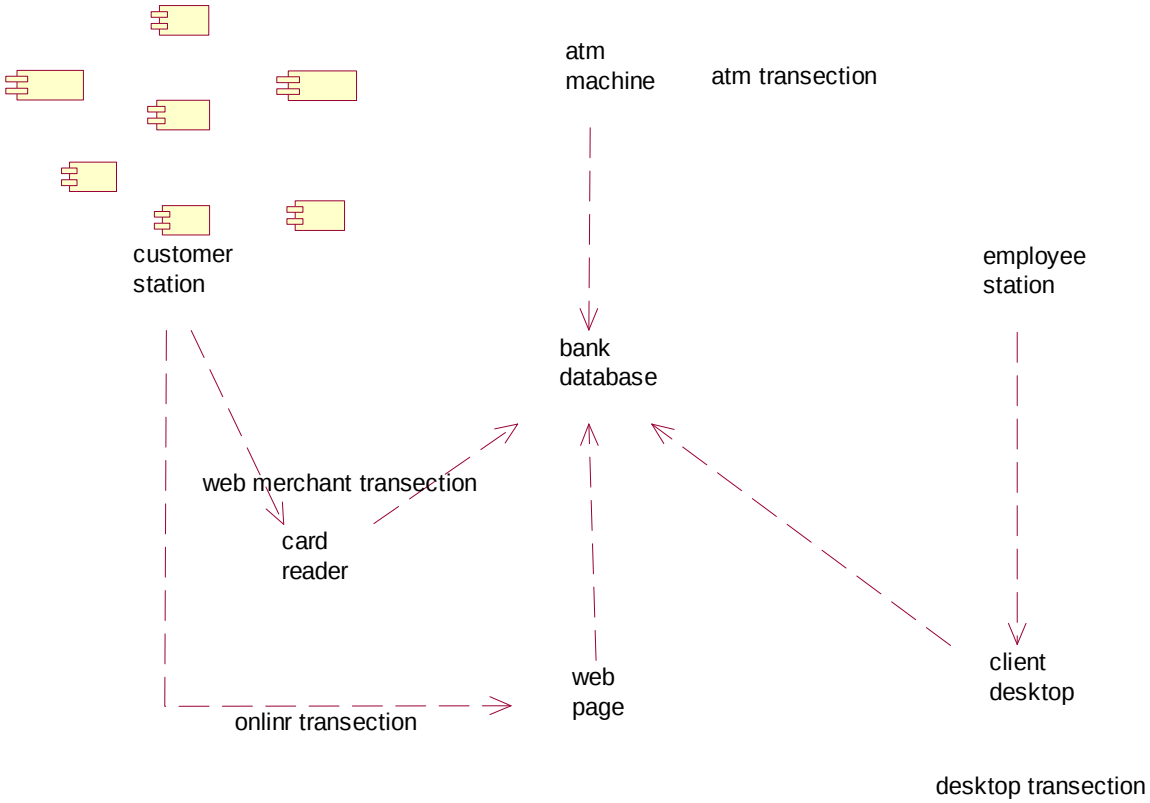
1.7 Object diagram

An **object diagram** is a graph of instances, including **objects** and data values. A static **object diagram** is an instance of a class **diagram**; it shows a snapshot of the detailed state of a system at a point in time.



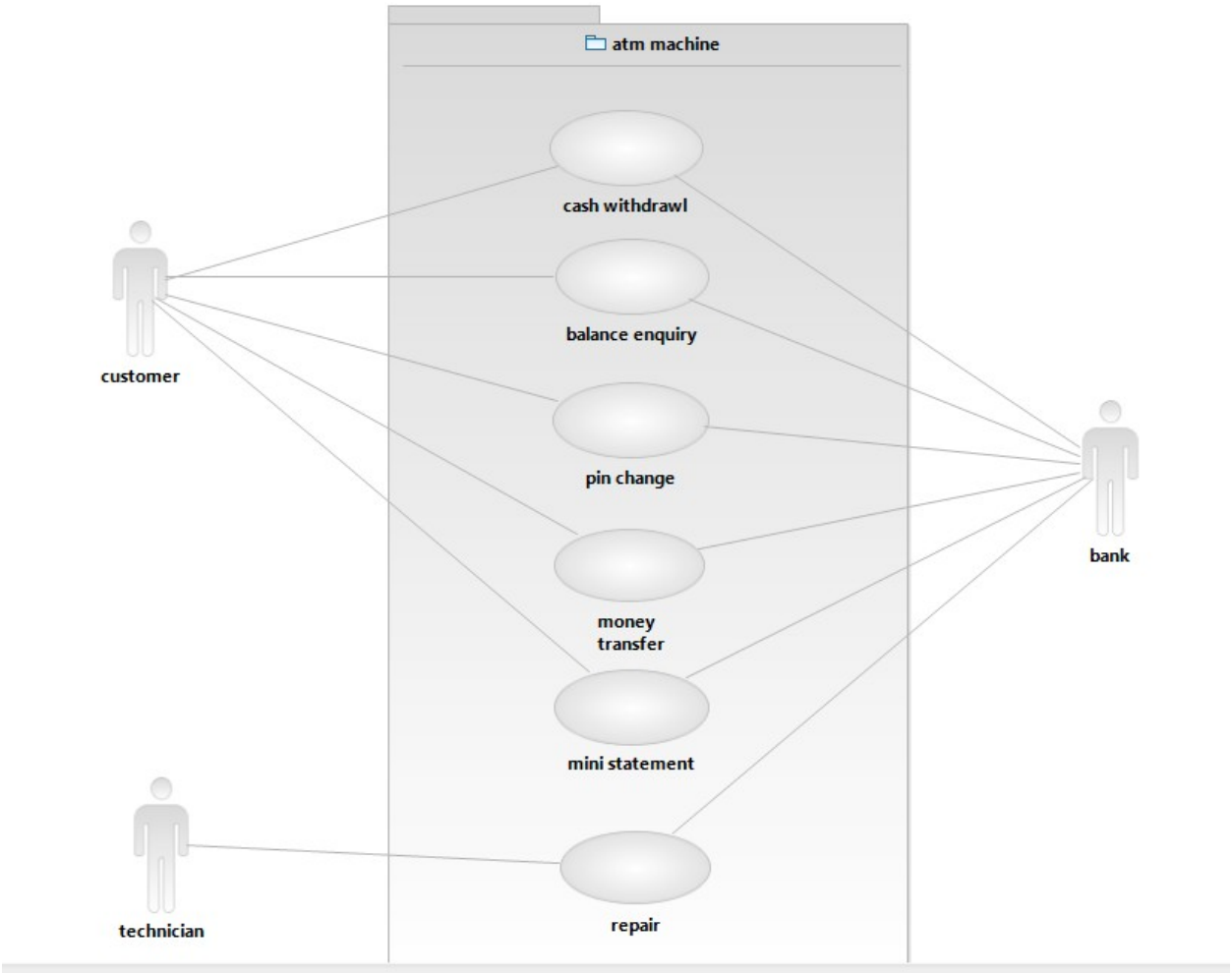
1.8 Component diagram

In Unified Modeling Language (UML), a **component diagram** depicts how **components** are wired together to form larger **components** or software systems. They are used to illustrate the structure of arbitrarily complex systems.



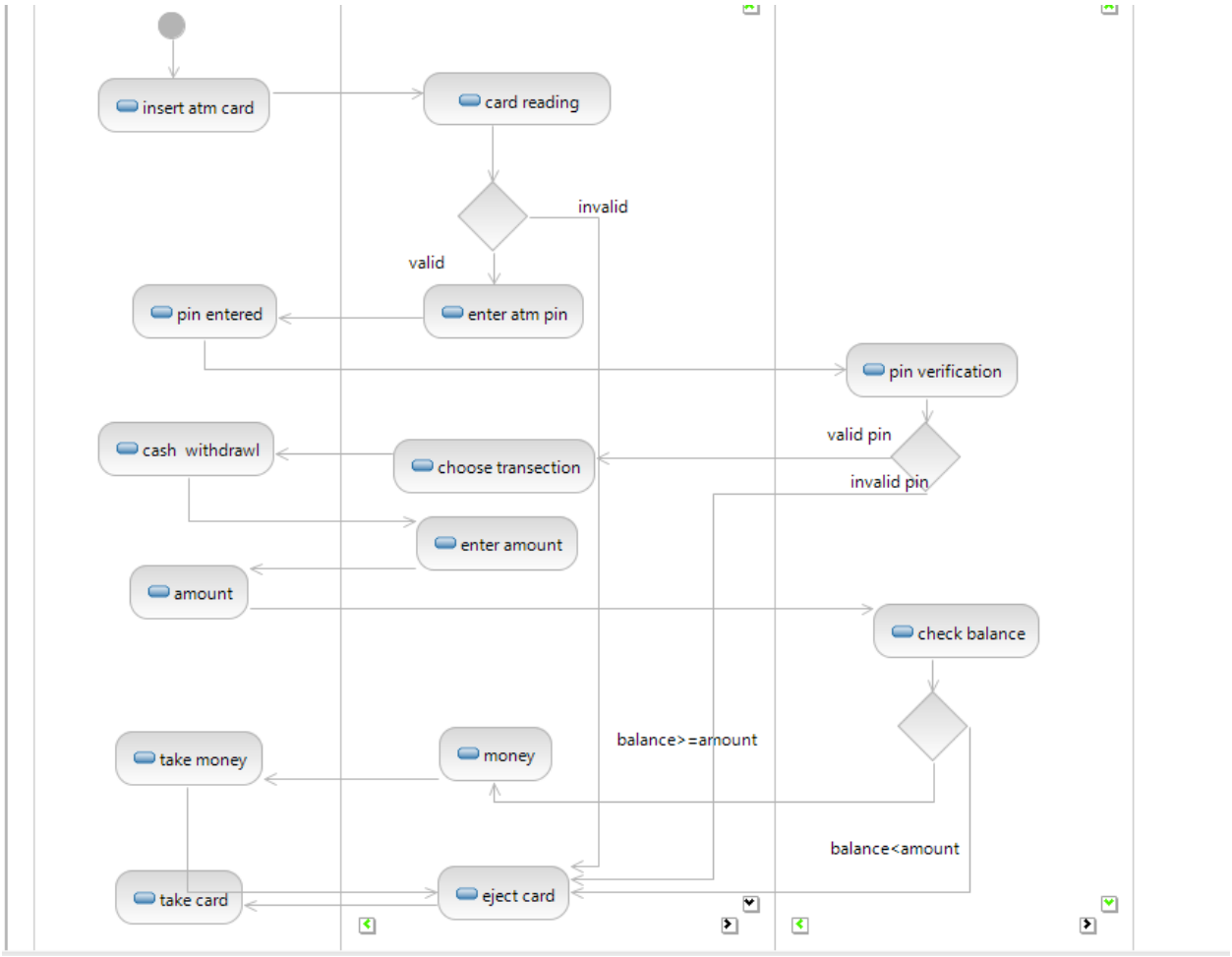
2 UML DIAGRAMS(RATIONAL ROSE)

1. Use case



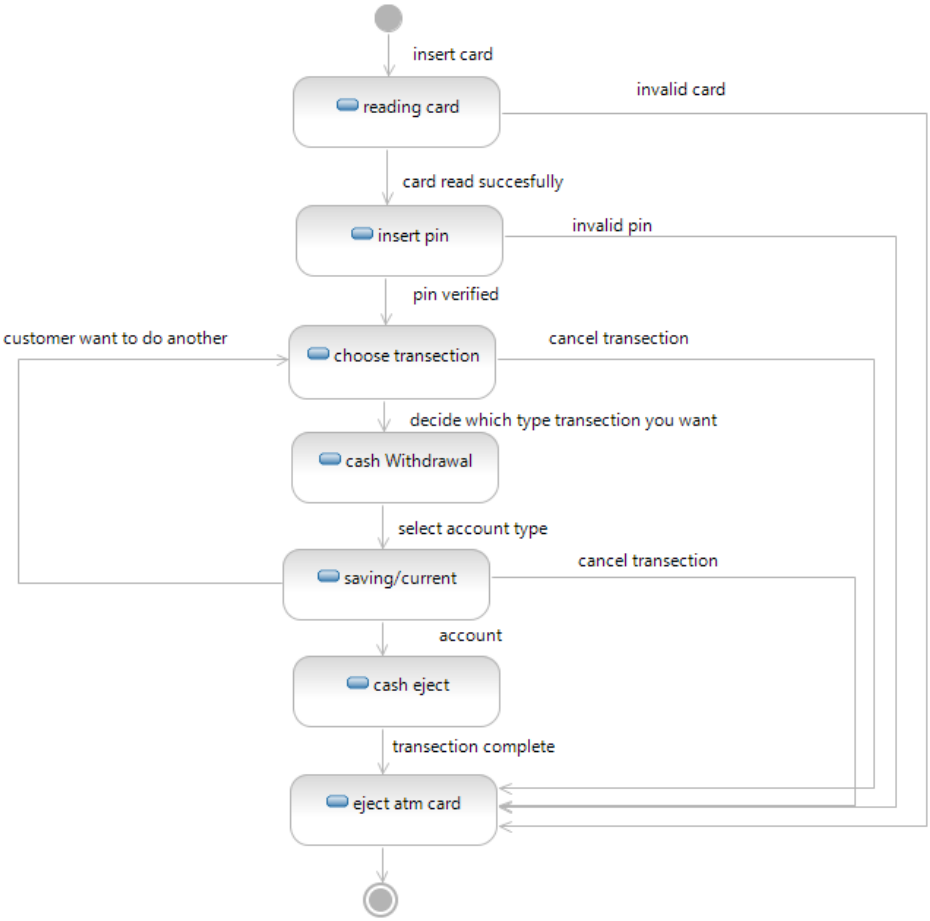
2.Activity diagram.

2.1 cash withdraw diagram



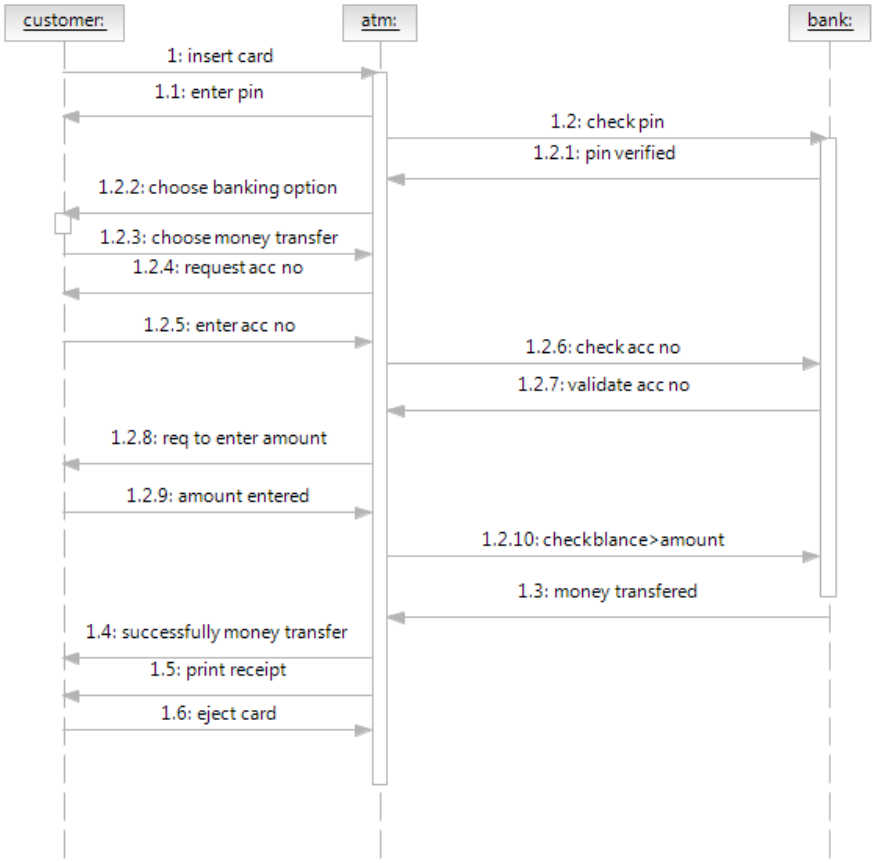
3 State diagram

3.1cash withdraw diagram



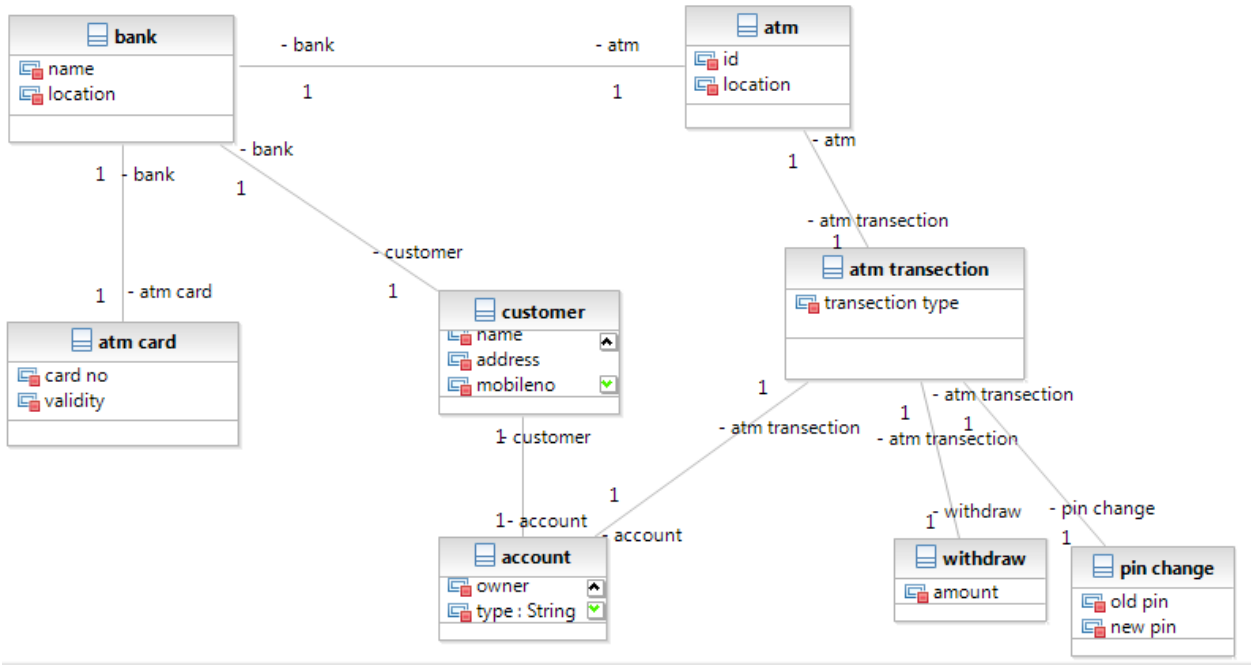
4 Sequence diagram

4.1Money transfer diagram

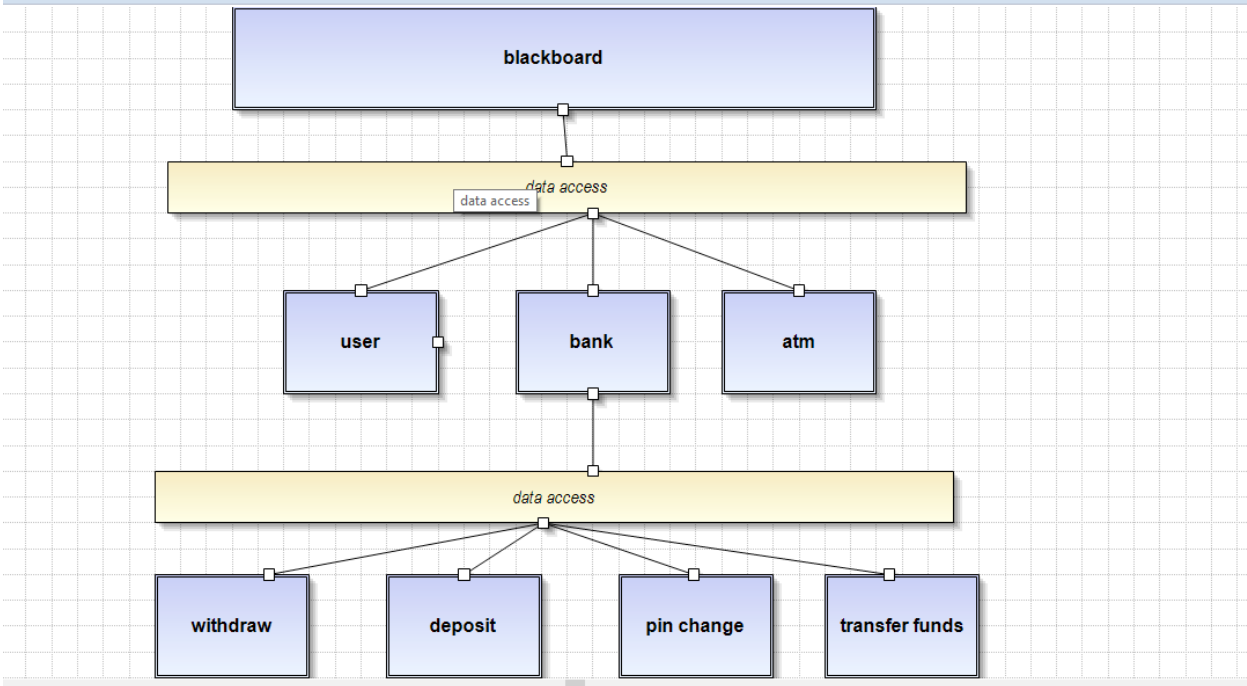


5 Class diagram

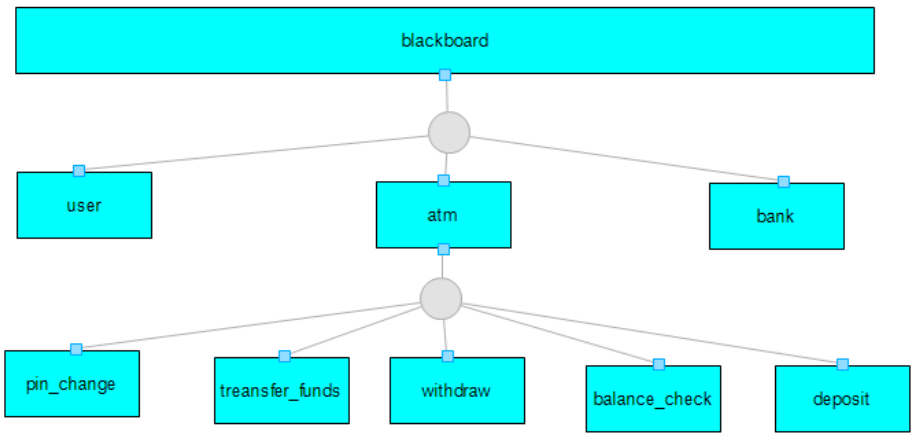
Class diagram for atm machine



4 ATM ARCHITECTURE USING ARCH_STUDIO

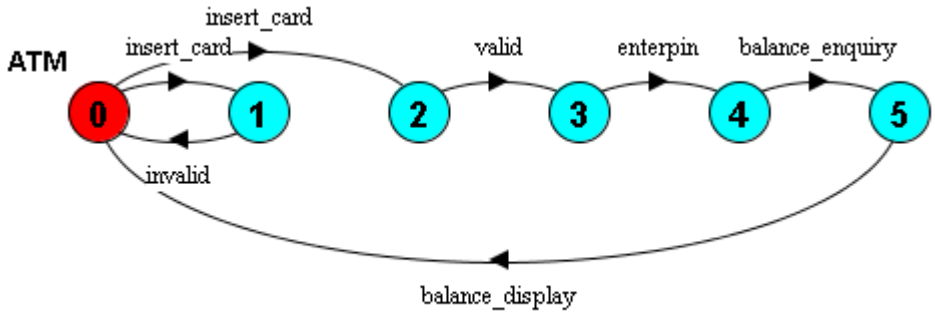


4 ATM ARCHITECTURE USING ACME_STUDIO



5 ATM TRANSITION DIAGRAM USING LTSA

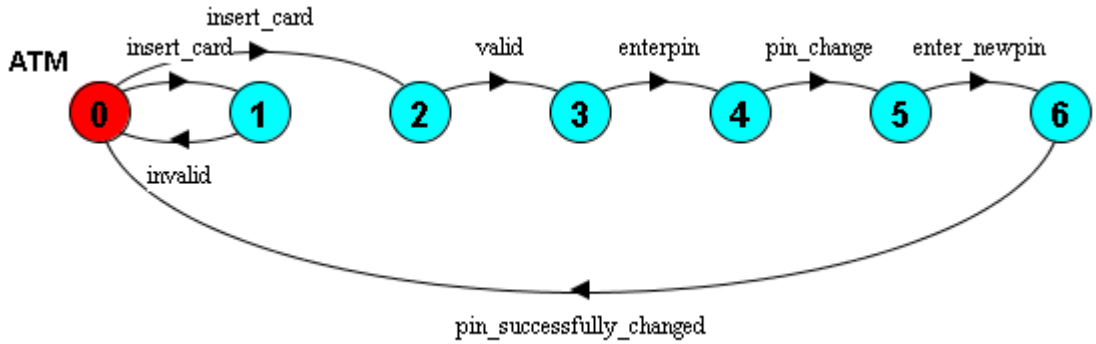
4.1 balance check



Code:

```
ATM =          (insert_card -> valid -> enterpin -> balance_enquiry ->  balance_display ->ATM
      |insert_card -> invalid -> ATM
            ).
menu RUN = {insert_card}
```

4.2 pin change



Code:

ATM = (insert_card -> valid -> enterpin -> pin_change -> enter_newpin -> pin_successfully_changed ->ATM

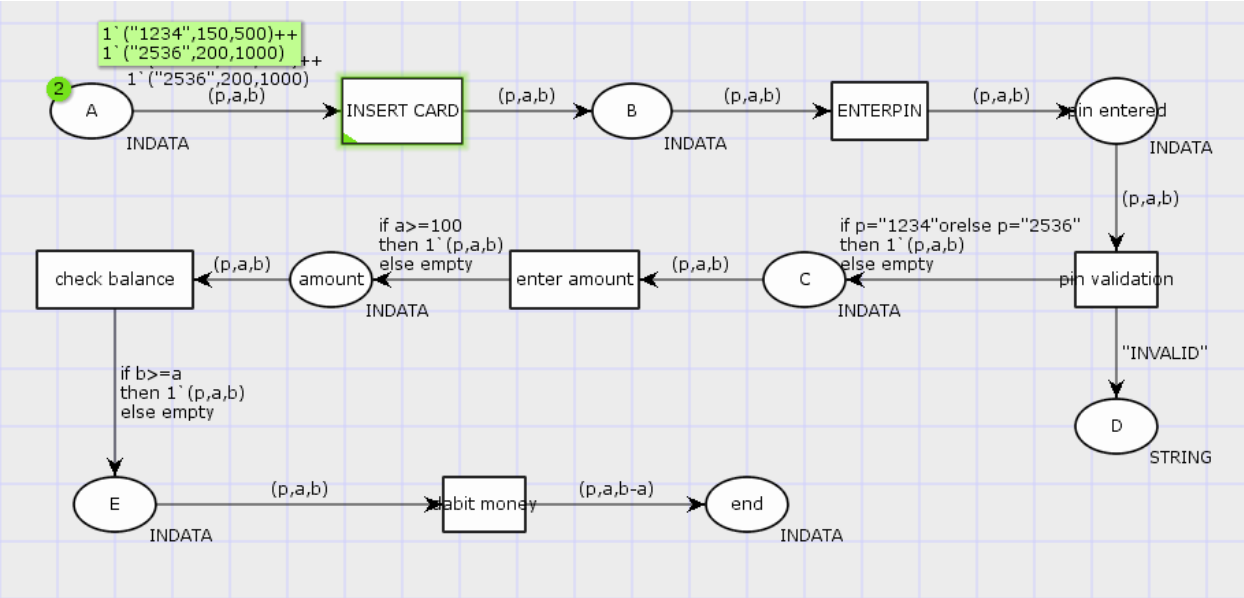
|insert_card -> invalid -> ATM

).

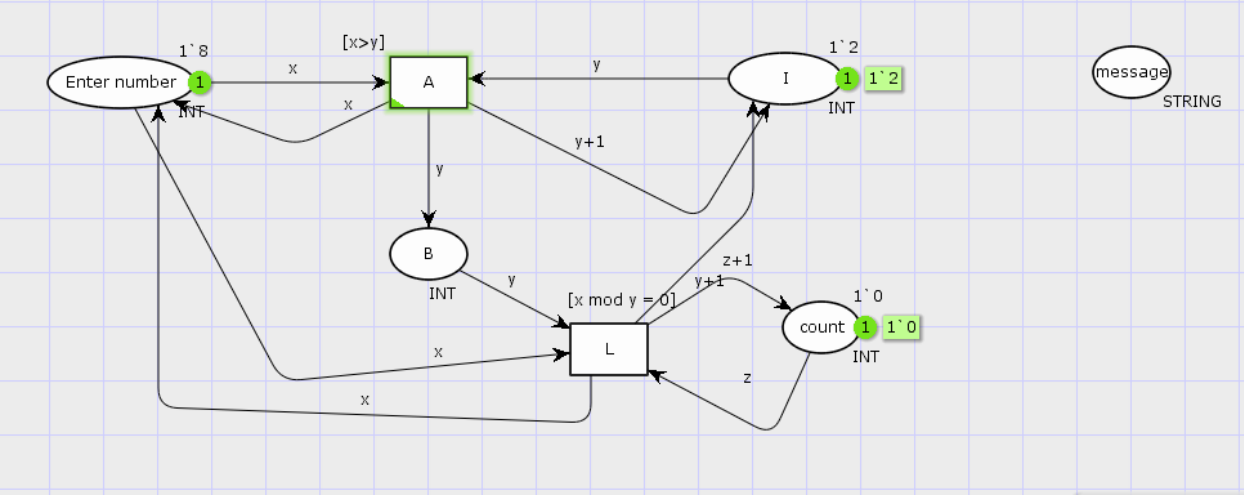
menu RUN = {insert_card}

6 PETRINET DIAGRAM

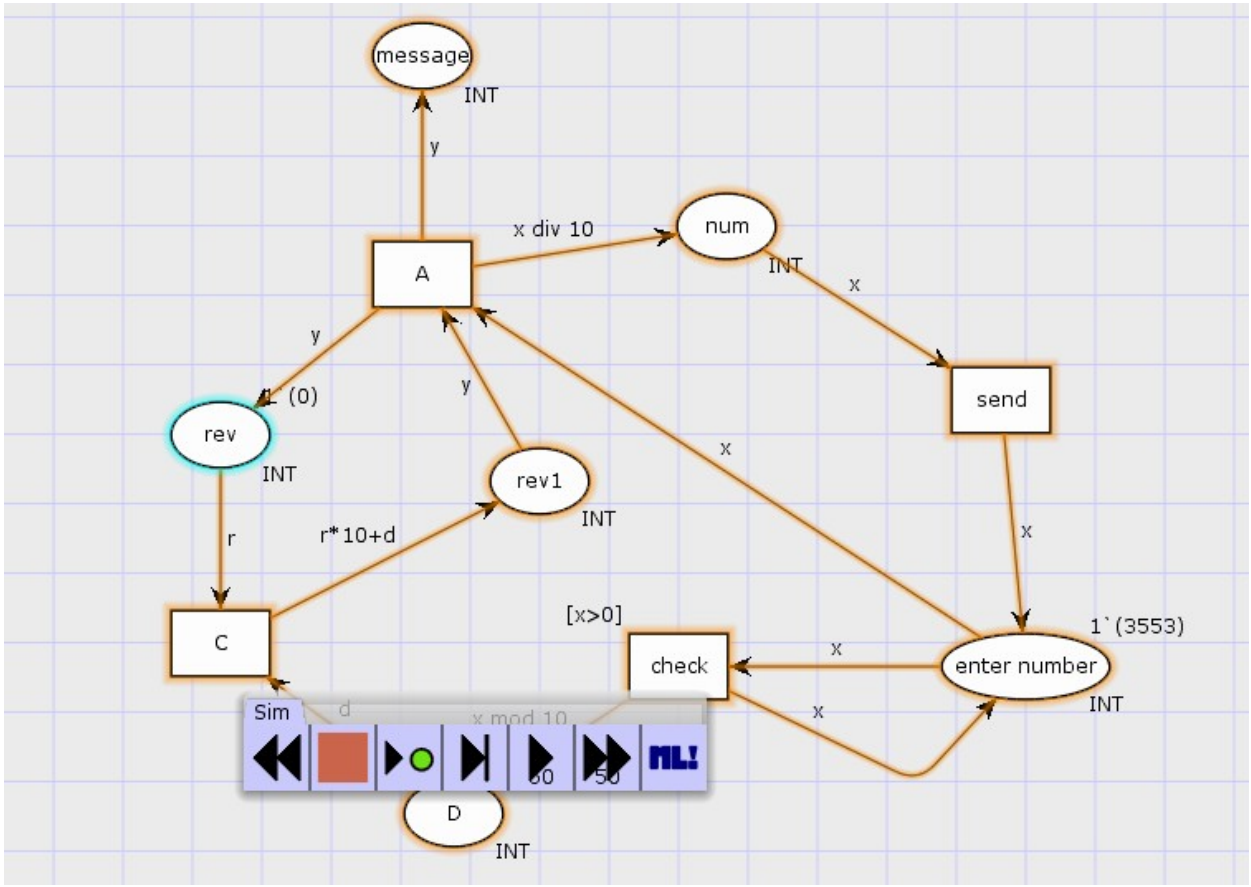
6.1 ATM



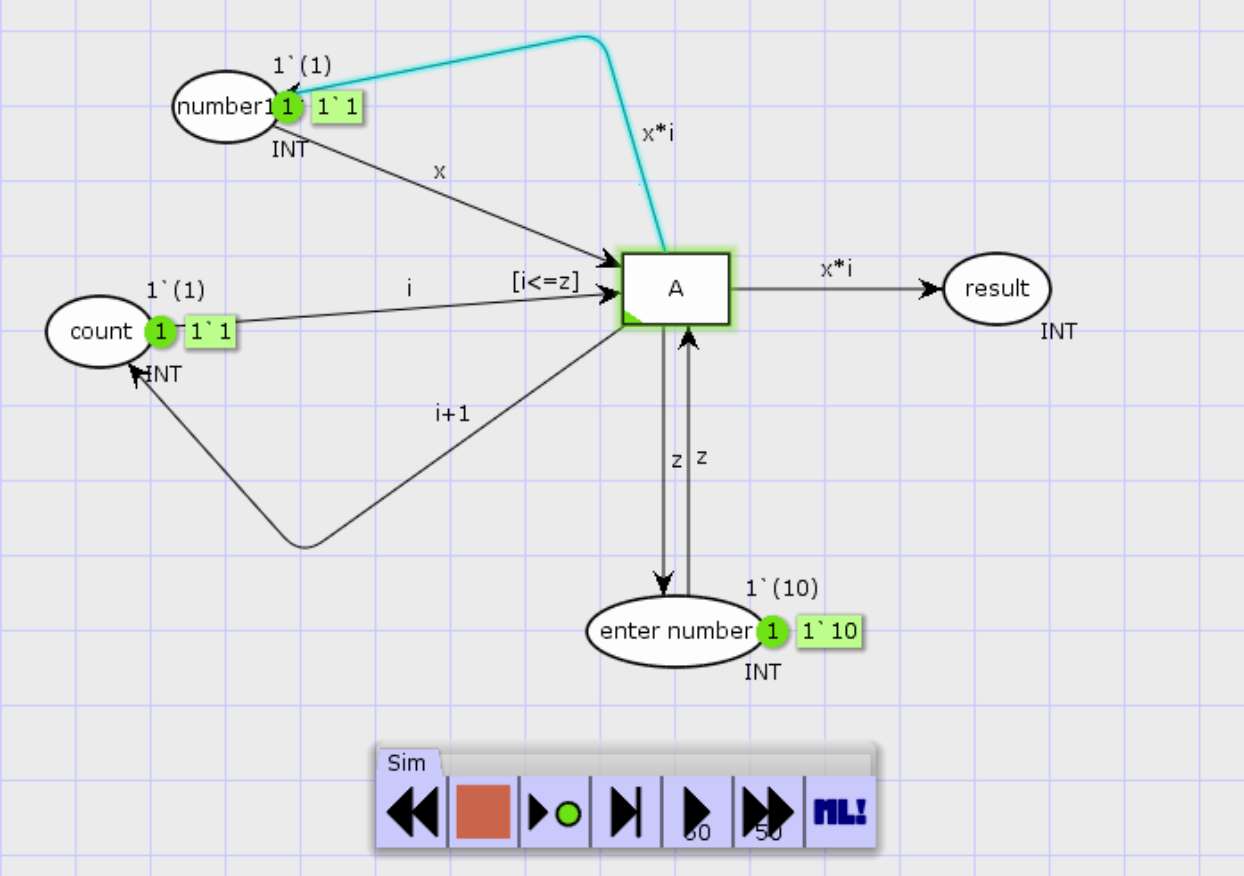
6.2 PRIME NUMBER



6.3PALINDROME OF A NUMBER



6.4 FACTORIAL OF A NUMBER



7 ATM Z-SPECIFICATION

Code:

\documentclass{article}
\usepackage{zed-csp}
\begin{document}
\begin{zed}
CARD ::= cardNo | acctNo | issuingBank | valid
\end{zed}
\begin{zed}
NAME ::= custName | bankName
\end{zed}
When any particular operation perform in ATM then related response generated by ATM system.
\begin{zed}
ATMResponse ::= opSuccess | opFailed | Nil
\end{zed}
For any operation like balance inquiry, cash withdraw it is required to know the status of ATM machine and sufficient balance.
\begin{zed}
STATUS ::= available | busy | idle
\end{zed}

\begin{zed}
RECEIPT ::= receipt
\end{zed}
\begin{zed}
DATE ::= issueDate | expiryDate | todayDate
\end{zed}
ATM system generate following possible types of error messages:
\begin{zed}
ERRORMessage ::= invalidePinNo | invalideCard \\ | insufficientBalance | invalideAmount
\end{zed}
Amount should also declared for cash withdraw operation. Because every bank provides certain restriction on minimum amount and maximum amount. The axiomatic definition for some important constraints. It is required to declared some variables globally.
\begin{axdef}
minAmount: \nat \\
maxAmount: \nat \\
withdrawAmount: \nat \\
accountBalance: \nat \\
pinNo: \nat \\
maxTran: \nat \\
moneyInMachine: \nat
\where
withdrawAmount \leq maxAmount
\end{axdef}
\begin{schema}{Bank}
bankName: NAME \\
card: CARD \\
has: NAME \fun CARD \\
balance: \nat \\
todayDate: DATE
\end{schema}
\begin{schema}{ATM}
balance: \nat \\
maxAmount: \nat \\
todayDate: DATE
\end{schema}
\begin{schema}{CardReader}\\
card?: CARD \\
date:DATE \\
status: STATUS \\
message!: ERRORMessage \\
\where
status = busy \\
date = expiryDate \implies message! = invalideCard
\end{schema}
Schema have two parts first is declaration part and second is predicate part. In the declaration part of schema CardReader, first variable card? represent the input by using ? symbol and variable message? represent for output using ! symbol.
\begin{schema}{BalanceEnquiry}\\
\Xi ATM \\
\Xi Bank \\
response! : ATMResponse \\
accountBalance : \nat \\

receipt! : RECEIPT \\
status : STATUS
\where
status = busy \\
moneyInMachine' = moneyInMachine \\
accountBalance' = accountBalance \\
response! = opSuccess \\
receipt!.amount = accountBalance \\
status' = idle \\
\end{schema}
In BalanceEnqury schema Ξ ATM and Ξ Bank denotes that the state of schemas of ATM and Bank will not change after completing this operation. moneyInMachine' and accountBalance' represent next state of moneyInMachine and accountBalance by using ' operator.
\begin{schema}{CashWithdraw}\\
\Delta ATM \\
\Delta Bank \\
acctNo? : CARD \\
m? : \nat \\
accountBalance : \nat \\
response! : ATMResponse \\
receipt! : RECEIPT \\
status : STATUS
\where
status = busy \\
balance' = balance \oplus acct? \\ \fun balance - m? \\
response! = opSuccess \\
receipt!.amount = m? \\
status' = Idle
\end{schema}
In schema CashWithdraw Δ ATM and Δ Bank represent that after cash withdraw operation the state of ATM and the state of Bank both will change. The operator $\oplus$ used for overwrite operation.
\begin{schema}{FundTransfer}\\
\Delta ATM \\
\Delta Bank \\
acct1?, acct2? : ACCOUNT \\
m? : \nat \\
balance : \nat \\
response! : ATMResponse \\
receipt! : RECEIPT \\
status : STATUS
\where
status = busy \\
balance \geq m? \\
accountBalance' = accountBalance \oplus \\ \fun accountBalance - m? \\
accountBalance' = accountBalance \oplus \\ \fun accountBalance + m? \\
response! = opSuccess \\
receipt!.amount = m? \\
status' = Idle
\end{schema}
\end{document}

Output:

$$CARD ::= cardNo \mid acctNo \mid issuingBank \mid valid$$

$$NAME ::= custName \mid bankName$$

When any particular operation perform in ATM then related response generated by ATM system.

$$ATMResponse ::= opSuccess \mid opFailed \mid Nil$$

For any operation like balance inquiry, cash withdraw it is required to know the status of ATM machine and sufficient balance.

$$STATUS ::= available \mid busy \mid idle$$

$$RECEIPT ::= receipt$$

$$DATE ::= issueDate \mid expiryDate \mid todayDate$$

ATM system generate following possible types of error messages:

$$ERRORMessage ::= invalidePinNo \mid invalideCard \mid insufficientBalance \mid invalideAmount$$

Amount should also declared for cash withdraw operation. Because every bank provides certain restriction on minimum amount and maximum amount. The axiomatic definition for some important constraints. It is required to declared some variables globally.

$$\begin{array}{l} minAmount : \mathbb{N} \\ maxAmount : \mathbb{N} \\ withdrawAmount : \mathbb{N} \\ accountBalance : \mathbb{N} \\ pinNo : \mathbb{N} \\ maxTran : \mathbb{N} \\ moneyInMachine : \mathbb{N} \\ \hline withdrawAmount \leq maxAmount \end{array}$$

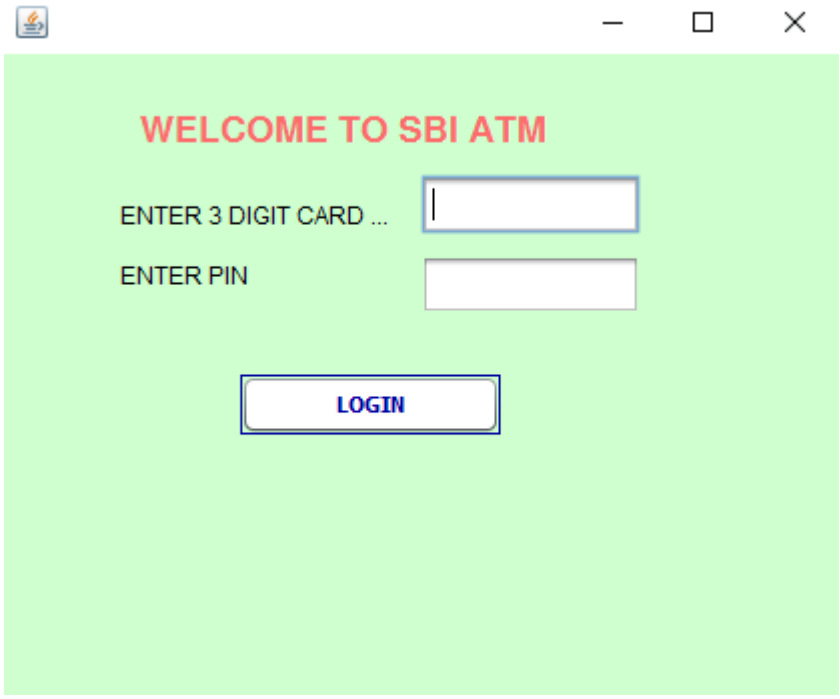
Bank

$$\begin{array}{l} bankName : NAME \\ card : CARD \\ has : NAME \rightarrow CARD \\ balance : \mathbb{N} \\ todayDate : DATE \end{array}$$

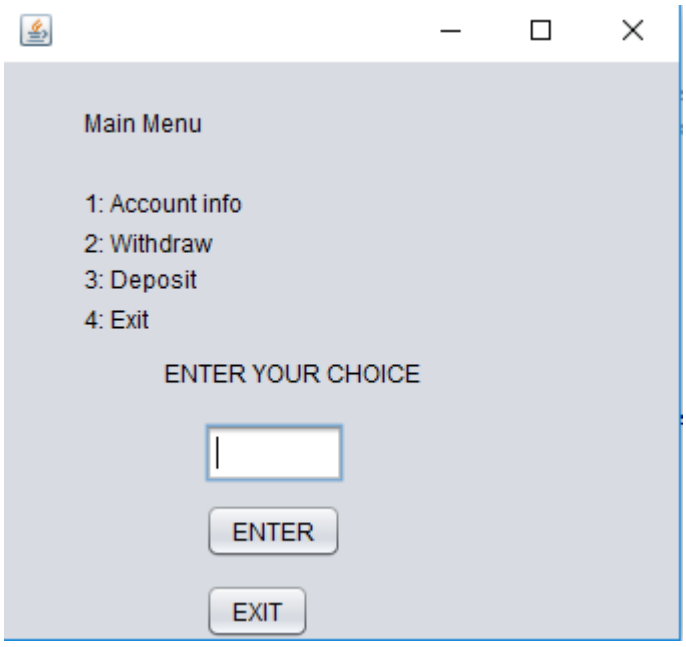
1

8 ATM PROJECT (SNAPSHORT)

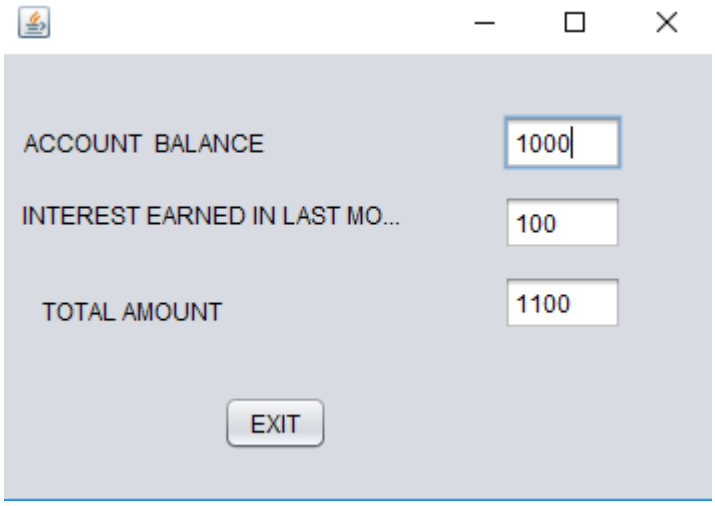
1)login page



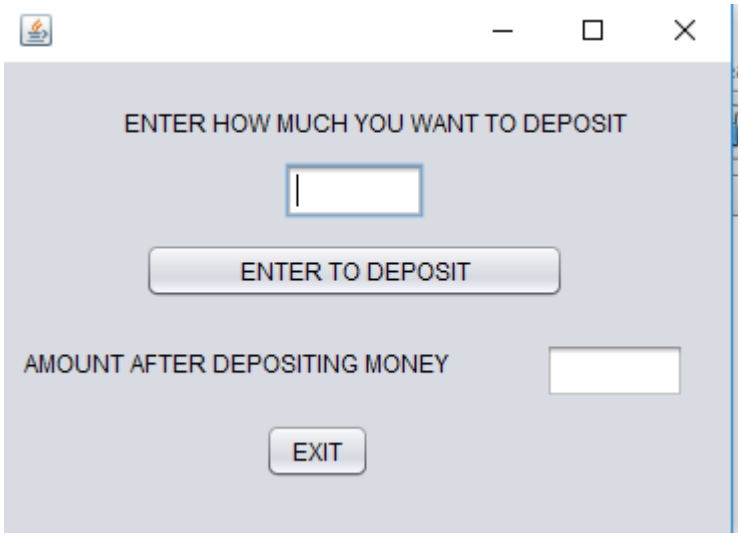
2)main menu



3)balance check page



4)deposit page



5) withdraw page