Software Engineering
Assignment 3
SmartBank

Team Awesome Win

Assumptions

The following are some assumptions we made about the SmartBank system and situation:

- SmartBank is an independent bank that has its own records of its customers including finger print details etc; it does not work on behalf of for example HSBC or Halifax but is a fully functional bank itself.
- A receipt is always printed when doing a topup transaction.
- A withdraw transaction once it has completed returns the card and the session by the customer is ended, this is for security safety reasons where a customer may take their money and forget to take their card, hence the card is returned first then the money to avoid this situation.
- For payment of utility bills, different utility companies have signed-up to SmartBank to allow access to their information allowing customers to pay for the bills they owe to that utility company; SmartBank simply has an interface that connects to the particular utility company's system.
- Once a language has been selected it cannot be changed for the duration of the session by a customer.
- Customers can use voice input, keyboard input, or touchpad input at any time there is no requirement to specify which is being used at a particular time.

Specification

SmartBank currently has an online banking system which they wish to expand to automated teller machines in towns and cities providing the functionality that is currently available online. Customers of SmartBank are first presented with an ATM screen that allows the customer to perform all the available actions. Interaction with the system can be from keyboard input that receives key presses; from a voice recogniser that receives user voice input, or finally from touchpad input that receives user touches. The system features language support for multiple languages including (English, French, Spanish, Portuguese, German, Italian and Chinese) which are displayed to the customer. The user interface is updated with the chosen language; if the language is spoken then the system will check the language. Before customers can use the system, customers are required to verify their credentials which is done by a verifier. This involves a card reader that accepts a customer's card and returns back the PIN number for that card, and a finger print scanner that accepts a finger print and reads the biometric information. All of which is checked by the verifier that uses a SmartBank connection to get customer's account details. If verification was not successful after three attempts then the card is retained. Otherwise a transaction can be made so the system displays a list of transactions, which are the following:

Withdrawing –

One of these transactions is withdraw-transaction allowing customers to withdraw money. The system displays a number of money amounts (£10, £20, £30 or £50) which the customer can select from; alternatively a customer can supply their own desired amount instead. The system then validates the chosen amount by ensuring that the amount is greater or equal to £10, but is less than £250 per day and that the amount is a multiple of £10. When the amount is chosen the customer's account is checked for sufficient funds. If the customer has sufficient funds the system displays the new account balance and a cash dispenser counts the required money, places the money in the money tray and then opens the money tray, once the customer has removed the money the cash dispenser closes the money tray. The customer is then asked if they require a receipt to be printed in A4 or printed in A5, for this the system has a receipt printer which can print a receipt by setting the receipt paper size and setting the receipt information. Afterwards they are returned to the main menu or alternatively if they don't have enough funds the system displays an insufficient funds message and they are returned to the main menu. For all parts of this functionality there is an option to display information that explains what operations can be done.

Depositing –

Deposit-transaction is the next transaction which allows customers to deposit money; this can be done in multiple ways. The first way is to input a cheque amount for this the system features a cheque reader which accepts cheques and then checks that the cheque is valid if it is not it is rejected otherwise the total amount is obtained. The second way is to input a money amount for this the system features a money collector which accepts money notes and then checks that the money note is valid if it is not it is rejected otherwise the total amount is obtained. The third is to input a coin amount for this the system features a coin collector which accepts coins and then checks that the coin is valid if it is not it is rejected otherwise the total amount is obtained. After the customer has deposited the money the system displays the new account balance. The customer is then asked if they require a receipt to be printed in A4 or printed in A5 using the aforementioned receipt printer, afterwards they are returned to the main menu. For all parts of this functionality there is an option to display information that explains what operations can be done.

View Account Information –

View-account-information-transaction is the next transaction this allows a customer to get their current account balance, get previous transactions, search for previous transactions and get their sort code. For any of these there are options to print an A5 receipt or print an A4 receipt using the aforementioned receipt printer. Once they have finished a customer can return to the main menu. For all parts of this functionality there is an option to display information that explains what operations can be done.

Topup Mobile Phone –

Topup-mobile-phone-transaction is the next transaction this allows a customer to purchase a mobile phone topup. The system displays a list of mobile providers (3, Vodafone etc) for the customer to choose from. The system then requests the customer to input their mobile phone number which is validated by using a mobile phone provider connection which can verify the mobile phone number once the mobile phone provider has been set. After the mobile phone number has been validated the system displays a selection of topup amounts (£10, £15, £20 etc) but customers can supply a topup amount if the amount they require is not there. The system validates the amount by ensuring it is greater or equal to £10 but is below £100 per day and is also a multiple of £5. When the topup amount has been chosen the customer's account is checked for sufficient funds. If the customer does not have sufficient funds they are returned to the main menu. Otherwise the system displays the new account balance, then the customer is then asked if they require a receipt to be printed in A4 or printed in A5 using the aforementioned receipt printer, finally they are returned to the main menu. For all parts of this functionality there is an option to display information that explains what operations can be done.

Pay Bills –

The final transaction possible is pay-bills-transaction which allows a customer to pay a bill. The system displays the outstanding bills that the customer has, this is done using a utility connection which gets the outstanding bills and they can then choose the bills they wish to pay. As always the customer's account is checked for sufficient funds. If the customer does not have sufficient funds they are returned to the main menu. Otherwise the system displays the new account balance, then the customer is asked if they require a receipt to be printed in A4 or printed in A5 using the aforementioned receipt printer, finally they are returned to the main menu. For all parts of this functionality there is an option to display information that explains what operations can be done.

All of these transactions use a SmartBank connection which establishes a connection to the SmartBank databases commits the transaction when ready and if necessary reverses the transaction to ensure data integrity.

Functional Requirements

1. The system must provide customers with different means of interaction and communication.
1.1 The system must be able to process and accept commands from keyboard-input.
1.2 The system must be able to process and accept commands from voice-activation.
1.3 The system must provide visual messages detailing instructions and/or information for each separate screen.
1.4 The system could provide audio alerts such as beeps to acquire customers' attention.
1.5 The system could provide audio instructions for each separate screen.

2. The system must allow customers to specify which language they wish to use.
2.1 The system must provide ability for customers to choose their language via voice-activation.
2.2 The system must show a message if the spoken language can not be recognised.
2.3 The system must provide ability for customers to choose their language from a list of options.
2.4 The system must provide support for the English language.
2.5 The system must provide support for the Spanish language.
2.6 The system must provide support for the French language.
2.7 The system must provide support for the Portuguese language.
2.8 The system must provide support for the German language.
2.9 The system must provide support for the Italian language.
2.10 The system must provide support for the Chinese language.

3. The system must allow customers to print off a receipt detailing any usage of the system.
3.1 The system must allow a customer to print off a receipt detailing their account balance.
3.2 The system must allow a customer to print off a receipt detailing a withdrawal.
3.3 The system must allow a customer to print off a receipt detailing a deposit.
3.4 The system must allow a customer to print off a receipt detailing the purchase of a mobile phone top-up.
3.5 The system must allow a customer to print off a receipt detailing the payment of a utility bill.
3.6 The system must allow a customer to print off receipts for the aforementioned types that happened in the past.
3.7 The system should allow a customer to select any number of these aforementioned types so that they can be combined into a single receipt to be printed off.
3.8 The system should allow the customer to specify the size of paper used for the receipt.

4. The system must allow customers to purchase a mobile phone top-up.
4.1 The system must allow customers to select their mobile phone provider.
4.2 The system must allow customers to specify their mobile phone number.
4.3 The system must provide a menu where a customer can select from vouchers worth £10, £15, £20 or £40.
4.4 The system must provide an option for customers to specify their own topup amount.
4.5 The system must restrict the exact amount to multiples of £5.
4.6 The system must restrict the minimum amount for a topup to £10.
4.7 The system must restrict the maximum amount for a topup to £100 per day.
4.8 The system must show an inadequate funds message if a customer tries to purchase a topup which has a value greater than that of their current account balance.

5. The system must allow access for customers to their personal private user account.
5.1 The system must request and provide ability for customers to enter their PIN number.
5.2 The system must request and provide ability for customers to give their finger print.
5.3 The system must only allow access to their user account after their PIN number and finger print have been verified.
5.4 The system must allow customers to view any information about their account, such as recent usages, current statues, sort code etc.

6. The system must allow customers to withdraw money from their bank account.
6.1 The system must provide a menu where a customer can select from options of withdrawing £10, £20, £30 or £50.
6.2 The system must provide an option for customers to specify their own withdraw amount.
6.3 The system must restrict the exact amount to multiples of £10.
6.4 The system must restrict the minimum amount to be withdrawn to £10.
6.5 The system must restrict the maximum amount to be withdrawn to £250 per day.
6.6 The system must ensure that customers do not withdraw a greater amount than that of their current account balance.
6.7 The system must show an inadequate funds message if a customer tries to withdraw more money than they have in their account.

7. The system must allow customers to deposit money into their back account.
7.1 The system must allow customers to deposit money with a cheque.
7.2 The system must allow customers to deposit money using money notes.
7.3 The system must allow customers to deposit money using coins.

8. The system must allow customers to be able to pay their utility bills.
8.1 The system must present customers with a list of outstanding bills, detailing the company, type and the cost of the bill.
8.2 The system must allow customers to choose from the list the bills they wish pay.
8.3 The system must show an inadequate funds message if a customer tries to pay a bill that has a value greater than that of their account.
8.4 The system should present customers with previous paid bills to allow quicker access to bills they commonly wish to pay using the system.

Non-Functional Requirements

Efficiency (Performance)

1. The system must respond within a reasonable amount of time to ensure that the customer does not believe the system has crashed.

1.1 The system must identify customers' chosen language by voice recognition or alert the user that the language could not be recognised within 10 seconds.

1.2 The system must show the updated balance after a deposit within 5 seconds.

1.3 The system must show the updated balance after a withdraw or inform the customer it was not possible within 5 seconds.

1.4 The system must show the updated balance after a mobile phone top-up voucher has been purchased within 5 seconds.

1.5 The system must show the update balance and bill payment confirmation within 30 seconds.

Efficiency (Space)

2. The system must have sufficient space to be able to support up to a million customers.

2.1 The system must be able to record and save details of all deposits made by customers.

2.2 The system must be able to record and save details of all withdraws made by customers.

2.3 The system must be able to record and save details of all mobile phone top-up vouchers purchased by customers.

2.4 The system must be able to record and save details of all bill payments by customers.

Usability

3 The system must be easy to use and navigate for all types of customers.

3.1 The system must have a simple graphical user interface with as few options per screen as possible.

3.2 The system's screens must have a consistent look and feel that is not confusing and flows in a logical manner.

3.3 The system must take no longer than 3 minutes for less experienced customers to understand it's most basic form.

3.4 The system should provide an option at every screen for on-screen help that explains the operations that can be done on that particular screen.

3.5 The system should provide access to more advanced features for customers who have more experience.

Reliability

4 The system must achieve 1000 hours MTBF (mean time between failures).

5 The system must be available 99% of the time 24/7.

Interoperability -

6 The system must correctly integrate with the Smart Bank databases for customer accounts and transaction information.

7 The system must correctly integrate with the Smart Bank biometrics database and the separate sub-system that handles verification of customers' finger-prints.

8 The system must correctly integrate with the interfaces provided by the utility companies.

Legislative (Privacy) -

9 The system must provide a high-level of security at all stages to protect the privacy of customers' data.

9.1 The system must only allow access to a customer's account once provided with the correct PIN number.

9.2 The system must only allow access to a customer's account once a customer's finger-print has been supplied and correctly verified.

9.3 The system must ensure that all operations (depositing, withdrawing, purchasing mobile phone top-up, paying bills) are secured.

9.4 The system must provide secure connection between other sub-systems including the customer's record database, the biometric database and the biometric subsystem.

9.5 The system should retain the card and lock-out a customer's account if after three attempts the customer has not entered the correct PIN number for their account.

9.6 The system should retain the card and lock-out a customer's account if after three attempts the customer's finger-print has not been verified.

10 The system must conform to the laws laid out in the European Data Protection Act.

Legislative (Safety) –

11 The system must be safe in terms of operation and should detect if there are fingers present in the withdrawing money section and inserting money, coin and cheques.

Ethical -

12 The system must correctly store and protect customer's personal information to ensure that it is properly secure and is not lost or tampered with.

13 The system must correctly store and protect customer's biometric data to ensure that it is properly secure and is not lost or tampered with.

Delivery -

14 The system must be implemented and delivered by the end of the 2011.

Implementation Constraints (Language) -

15 The system should be written in Java.

Implementation Constraints (Design Methods) -

16 The system should be completed using an object-orientated approach.

Implementation Constraints (Maintainability)-

17 The structure of the code should be written in a clear and simplistic manner conforming to good design practices.

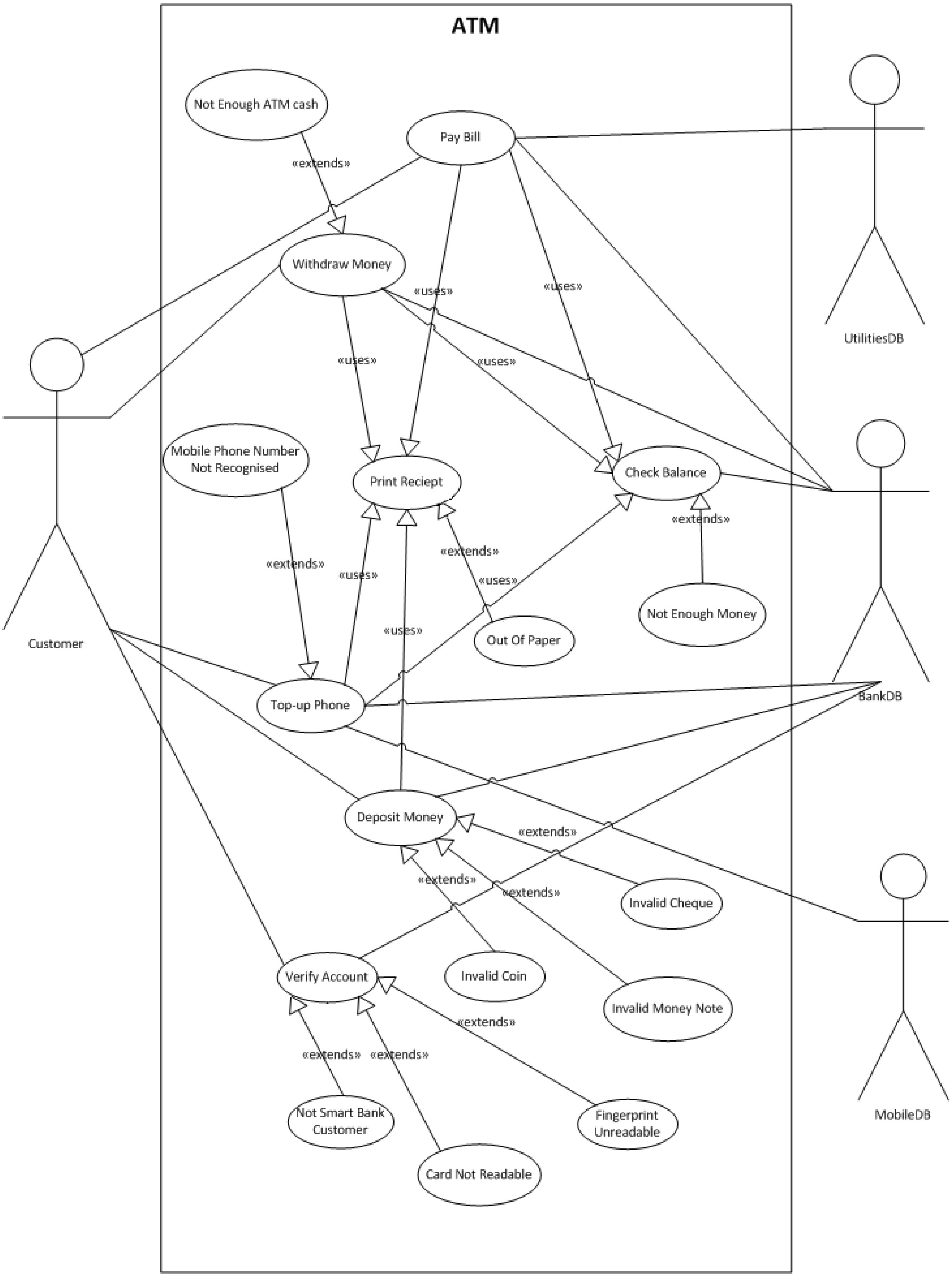
17.1 The system should have low coupling between its different modules.

17.2 The system's modules should have high cohesion.

Standards –

18 The system should conform to the agreed standard for Smart Bank systems in particular their online system and biometric system.

Use Case Diagram



Use Cases Descriptions

Top-up Phone

Precondition

1. The Customer has selected a language to use.
2. The Customer is a member of SmartBank.
3. The Customer has verified their account with finger-print and PIN number.
4. The Customer has placed their credit card into the ATM machine.

Flow Of Events

1. The use case starts when the Customer selects top-up from the list of transaction options.
2. The system asks the Customer for their mobile phone network provider which can be selected from a list or spoken using voice-recognition.
   2a. If the mobile phone network provider could not be recognised a message is displayed and the Customer is asked to choose one again.
   2b. If the Customer chooses to exit from the transaction they are asked if they would like to make a different transaction if so they are returned to the transaction options list otherwise their account card is returned and the system returns to an idle state ready for the next Customer.
3. The system asks the Customer for the mobile phone number which can be typed via keyboard input or using voice recognition.
   3a. If the mobile phone number could not be recognised a message is displayed and the Customer is asked to supply the mobile phone number again.
   3b. If the mobile phone number is not a valid mobile phone number a message is displayed and the Customer is asked to supply the mobile phone number again.
   3c. If the mobile phone number does not match a record with the MobilePhoneDB then a message is displayed and the Customer is asked to supply the mobile phone number again.
   3d. If the Customer chooses to exit from the transaction they are asked if they would like to make a different transaction if so they are returned to the transaction options list otherwise their account card is returned and the system returns to an idle state ready for the next Customer.
4. The system asks the Customer for a top-up amount which can be selected from a list of options, or an amount chosen via keypad input or using voice-recognition.
   4a. If the amount could not be recognised a message is displayed and the Customer is asked to supply an amount again.
   4b. If the amount is not a valid amount because it is not between £10 and £100 or it is not a multiple of £5 then the Customer is asked to supply an amount again.
   4c. If the Customer chooses to exit from the transaction they are asked if they would like to make a different transaction if so they are returned to the transaction options list otherwise their account card is returned and the system returns to an idle state ready for the next Customer.
5. The system then checks this chosen amount against then SmartBankDB record for this Customer's current account balance.
   5a. If the Customer does not have sufficient funds for the top-up a message is displayed and they are requested to supply a different top-up amount.
6. The system once it has verified that the Customer has sufficient funds it requests SmartBankDB to subtract the top-up amount from the Customer's current account balance.
   6a. If there is an error connecting to the SmartBankDB or some other error then a message is shown, the transaction is cancelled and reversed, the Customer's account card is returned and the system goes into an idle state ready for the next Customer.
7. The system then requests that MobilePhoneDB update the Customer's mobile phone credit for that particular mobile phone number
   7a. If there is an error connecting to the MobilePhoneDB or some other error then a message is shown, the transaction is cancelled and reversed, the Customer's account card is returned and the system goes into an idle state ready for the next Customer.

8. The system shows a confirmation message that the top-up mobile phone transaction was successful.
9. The system then asks if the Customer would like an A5 or an A4 receipt for this mobile phone transaction.
   9a. If the Customer selects or says "A5" an A5 receipt is printed for this transaction.
   9b. If the Customer selects or says "A4" an A4 receipt is printed for this transaction.
   9c. If the Customer selects or says "No" then a transaction receipt is not printed and the system carries on.
   9d. If the system could not recognise the command from the Customer (not "A5", "A4" or "No") then a message is displayed and they are requested for another reply.
10. The system finally asks the Customer if they would like to make another transaction.
   10a. If the Customer selects or says "Yes" then they are returned to the transaction options list.
   10b. If the Customer selects or says "No" then the Customer's account card is returned and the system goes into an idle state ready for the next Customer.
   10c. If the system could not recognise the command from the Customer (not "Yes" or "No") then a message is displayed and they are requested for another reply.

Post Conditions

1. The system has updated the Customer's account balance if a transaction was successfully completed using the SmartBankDB.
2. The system has updated the Customer's mobile phone credit amount if a transaction was successfully completed using the MobilePhoneDB.
3. The system has ejected the Customer's account card if no other transactions were required.
4. The system is in an idle state if no other transactions were required ready for the next Customer.

Actors –

The actor primarily involved in this use case is Customer who initiates the use case by interacting with the ATM system, also involved are the actors SmartBankDB and MobilePhoneDB. The SmartBankDB is updated after a top-up has been purchased to reflect the Customer's new account balance, and is also used to check a Customer's balance. The MobilePhoneDB is also updated after a top-up has been purchased to reflect the Customer's new credit amount.

Scenarios –

Below is a selection of scenarios for this particular use case they detail what can happen and how it happens, for obvious reasons not every single possible combination is detailed.

Scenario 1

Customer James chooses Top-up Mobile Phone from the ATM transaction options and is requested to supply a mobile phone network provider, he selects Vodafone from the list of options. He then is requested to supply his mobile phone number which he does by keying in the numbers 01455 123456 the system confirms that this is a valid format for a mobile phone number and then checks with MobilePhoneDB to ensure that the mobile phone number exists which it does. The system then requests that he supplies a top-up amount either from a list of amounts or specifying own amount. He chooses to specify his own amount and keys in 90 the system validates that this is a correct amount and proceeds to checking with the SmartBankDB that he has sufficient funds. His current account balance is only £60 so the system reports that he has insufficient funds as a message and he is returned to the ATM transaction options.

Scenario 2

Customer Claire chooses Top-up Mobile Phone from the ATM transaction options and is requested to supply a mobile phone network provider, she says "tmobile" and the system identifies the network as T-Mobile and asks if this is correct which she confirms it is. She then is requested to supply her mobile phone number which she does by keying in the numbers 01455 634572 the system confirms that this is a valid format for a mobile phone number and then checks with MobilePhoneDB to ensure that the mobile phone number exists which it does. The system then requests that she supplies a top-up amount either from a list of amounts or specifying own amount. She says "thirty five" and the system identifies the amount as £35 and asks if this is correct which she confirms it is. The system also validates that this is a valid amount and proceeds to checking with the SmartBankDB that she has sufficient funds. Her current account balance is £24,000 so a confirmation message is shown. Her current account balance is updated to £23,965 by the SmartBankDB and her credit amount is updated with an added £35 by the MobilePhoneDB. Finally the system requests if she wants a receipt for the transaction in either A5 or A4 which she doesn't, and also if she wants to make another transaction which she doesn't so her card is returned, and the system returns to an idle state ready for the next Customer.

Scenario 3

Customer Vicky chooses Top-up Mobile Phone from the ATM transaction options and is requested to supply a mobile phone network provider, she selects O2 from the list of options. She then is requested to supply her mobile phone number which she does by saying the numbers 01455 989761 the system confirms that this is a valid format for a mobile phone number and then checks with MobilePhoneDB to ensure that the mobile phone number exists which it does. The system then requests that she supplies a top-up amount either from a list of amounts or specifying own amount. She chooses to specify her own amount and keys in 123 the system validates that this is not a correct amount and requests that she specifies a different amount she keys in 20 and the system validates that this is a correct amount. The system then proceeds to checking with the SmartBankDB that she has sufficient funds. Her current account balance is £100 so a confirmation message is shown. Her current account balance is updated to £80 by the SmartBankDB and her credit amount is updated with an added £20 by the MobilePhoneDB. Finally the system requests if she wants a receipt for the transaction in either A5 or A4 so she choose A5 and the system prints out the receipt, the system then asks her if she wants to make another transaction which she doesn't so her card is returned, and the system returns to an idle state ready for the next Customer.

Scenario 4

Customer Clive chooses Top-up Mobile Phone from the ATM transaction options and is requested to supply a mobile phone network provider, he selects Vodafone from the list of options. He then is requested to supply his mobile phone number which he does by keying in the numbers 01722 454545 the system confirms that this is a valid format for a mobile phone number and then checks with MobilePhoneDB to ensure that the mobile phone number exists which it does. The system then requests that he supplies a top-up amount either from a list of amounts or specifying own amount. He says "twenty" and the system replies that it could not recognise the amount, so he instead presses £20 from the list of options. The system then proceeds to checking with the SmartBankDB that he has sufficient funds. His current account balance is £14,020 so a confirmation message is shown. His current account balance is updated to £14,000 by the SmartBankDB and his credit amount is updated with an added £20 by the MobilePhoneDB. Finally the system requests if he wants a receipt for the transaction in either A5 or A4 which he doesn't, and also if he wants to make another transaction which he does, so the system goes to the ATM transaction options.

Scenario 5

Customer Alicia chooses Top-up Mobile Phone from the ATM transaction options and is requested to supply a mobile phone network provider, he selects BT from the list of options. He is then requested to supply his mobile phone number which he does by keying in the numbers 09191 134673 the system confirms that this is a valid format for a mobile phone number and then checks with MobilePhoneDB to ensure that the mobile phone number exists which it does not. A message is displayed saying that the number could not be recognised by the mobile phone network provider and that Customer Alicia should supply a different number. She chooses to exit from the transaction where the system asks if she would like to make another transaction which she does not so her account card is returned, and the system returns to an idle state ready for the next Customer.

Withdraw Money

Precondition

1. The Customer has selected a language to use.
2. The Customer is a member of SmartBank.
3. The Customer has verified their account with finger-print and PIN number.
4. The Customer has placed their credit card into the ATM machine.

Flow Of Events

1. The use case starts when the Customer selects withdraw money from the transaction options list.
2. The system asks the Customer for a withdraw amount which can be selected from a list of options, or an amount chosen via keypad input or using voice-recognition.
   2a. If the amount could not be recognised a message is displayed and the Customer is asked to supply an amount again.
   2b. If the amount is not a valid amount because it is not between £10 and £250 per day or it is not a multiple of £10 then the Customer is asked to supply an amount again.
   2c. If the amount is not available because the ATM does not have enough money or money to total that amount the Customer is asked to supply an amount again.
   2d. If the Customer chooses to exit from the transaction they are asked if they would like to make a different transaction if so they are returned to the transaction options list otherwise their account card is returned and the system returns to an idle state ready for the next Customer.
3. The system then checks this chosen amount against then SmartBankDB record for this Customer's current account balance.
   3a. If the Customer does not have sufficient funds for the withdraw amount a message is displayed and they are requested to supply a different withdraw money amount.
4. The system once it has verified that the Customer has sufficient funds it requests SmartBankDB to subtract the amount from the Customer's current account balance.
   4a. If there is an error connecting to the SmartBankDB or some other error then a message is shown, the transaction is cancelled and reversed, the Customer's account card is returned and the system goes into an idle state ready for the next Customer.
5. The system shows a confirmation message that the withdraw money transaction was successful.
6. The system then counts up money notes to the value of the withdraw amount and places it in the cash dispenser.
7. The system then asks if the Customer would like an A5 or an A4 receipt for this mobile phone transaction.
   7a. If the Customer selects or says "A5" an A5 receipt is printed for this transaction.
   7b. If the Customer selects or says "A4" an A4 receipt is printed for this transaction.
   7c. If the Customer selects or says "No" then a transaction receipt is not printed and the system carries on.
   7d. If the system could not recognise the command from the Customer (not "A5", "A4" or "No") then a message is displayed and they are requested for another reply.

8. The system finally asks the Customer if they would like to make another transaction.
   8a. If the Customer selects or says "Yes" then they are returned to the transaction options list.
   8b. If the Customer selects or says "No" then the Customer's account card is returned and the system goes into an idle state ready for the next Customer.
   8c. If the system could not recognise the command from the Customer (not "Yes" or "No") then a message is displayed and they are requested for another reply.

Post Conditions

1. The system has updated the Customer's account balance if the transaction was successfully completed using the SmartBankDB.
2. The system has ejected the Customer's account card if the transaction was not successful or the Customer chose not to make anymore transactions.
3. The system has placed the Customer's chosen withdraw money amount in the cash dispenser if the transaction was successfully completed.
4. The system is now in an idle state ready for the next Customer.

Actors –

The actor primarily involved in this use case is Customer who initiates the use case by interacting with the ATM system, also involved is the SmartBankDB which is used to check that the Customer has sufficient funds and to update the Customer's account balance if the transaction was successfully completed.

Scenarios –

Below is a selection of scenarios for this particular use case they detail what can happen and how it happens, for obvious reasons not every single possible combination is detailed.

Scenario 1-

Customer Becki chooses to withdraw option from the ATM transaction options. The customer is requested to enter the amount that they wish to withdraw from the list of options that have been provided. Becki decides to use the voice activation devise to enter the amount, which is £40 to withdraw from her account. The system checks the SmartBankDB record to ensure that Becki has at least £40 in her bank account. The system confirms that Becki has £40 in her account and the system shows a message saying that the transaction has been completed and their money will be dispensed. The money is that was requested is dispensed to the front of the ATM. The system then asked the customer whether they would like to print a receipt. Becki selects not to print of a receipt. The system then asks the customer would like to carry out another transaction which Becki selects not to.

Scenario 2-

Customer Lauren chooses to withdraw options from the ATM. Lauren selects to withdraw £300 and is presented with a message which says that they need to select another amount between £10 and £250. Lauren then selects to withdrawn £250 from her account. The system checks the SmartBankDB record to ensure that Lauren has enough money in her account. The system confirms that she does and the system shows a message that says that the transaction has been completed and their money will be dispensed. The money is dispensed to the front of the ATM and the system then asks whether they would like to print a receipt. Lauren decides against this option and when the system asks whether she would like to carry out another transaction, she's selects no.

Scenario 3-

Customer Pamela chooses the withdraw option from the ATM. Once the options showing the values which are available to withdraw, Pamela decided that she no longer wanted to withdraw money from her account. She selects the cancel option and the system asks whether she would like to select another option. Pamela selected no and the system returned her card.

Scenario 4-

Customer Jade chooses to withdraw option from the ATM. Jade selects to withdraw £100 from her account. The system checks the SmartBankDB record to ensure that Jade has enough money in her account. The system reports back to Jade that she doesn’t have enough funds in her account to withdraw the amount that she has selected and asks her to select another amount. Jade decided that she no longer wanted to withdraw money from her account. She selects the cancel option and the system asks whether she would like to select another option. Pamela selected no and the system returned her card.

Scenario 5-

Customer Karen chooses the withdraw option from the ATM. Karen selects to withdraw £5. The system presents Karen with a message saying that this amount was not recognised and could she please supply another amount between £10 and £250 in multiples of 10. Karen selects to withdraw £10 from her account. The system checks the SmartBankDB record to ensure that she has enough money from her account which she has and the system displays a message stating that the money is being dispensed. The system deposits the amount to the front panel of the ATM. The system displays a message asking whether the she wants to have a receipt printed, which she selects yes. The system presents the options in which the receipt can be printed which Karen picks A5. The system asks whether she would like to select another option. Karen selected no and the system returned her card.

Noun-Verb Analysis

Below is the noun-verb analysis for the SmartBank ATM system specification, it aims to find appropriate classes by analysing the nouns and to find appropriate methods for these classes by analysis the verbs. This will not give a complete specification of the classes and methods required to implement the system but it helps drive the process especially when coupled with Responsibility-Driven Analysis, so both will be used to create a first-cut class diagram and then a fully-complete class diagram with any added classes or methods that are required through further analysis.

The specification is written out again and the nouns are in bold type and the verbs are in italics after the specification there is a table with the word/phrase whether it was accepted or not and a very short justification as to why.

Specification

SmartBank currently has an **online banking system** which they wish to *expand* to **automated teller machines** in **towns** and **cities** *providing the functionality* that is currently available online. **Customers** of **SmartBank** are first *presented* with an **ATM screen** that *allows* the **customer** to *perform all the available actions*. Interaction with the **system** can be from **keyboard input** that *receives key presses*; from a **voice recogniser** that *receives user voice input*, or finally from **touchpad input** that *receives user touches*. The **system** features **language support** for multiple languages including (**English**, **French**, **Spanish**, **Portuguese**, **German**, **Italian** and **Chinese**) which are *displayed to the customer*. The *user interface is updated with the chosen language*; if the language is spoken then the **system** will *check the language*. Before **customers** can use the **system, customers** are required to **verify their credentials** which is done by a **verifier**. This involves a **card reader** that *accepts a customer's card* and *returns back the PIN numbe*r for that card, and a **finger print scanner** that *accepts a finger print* and *reads the biometric information*. All of which is *checked* by the **verifier** that uses a **SmartBank connection** to *get customer's account details.* If verification was not successful after three attempts then the *card is retained*. Otherwise a **transaction** can be made so the **system** *displays a list of transactions*, which are the following:

Withdrawing –

One of these **transactions** is **withdraw-transaction** *allowing customers to withdraw money.* The **system** *displays a number of money amounts* (£10, £20, £30 or £50) which the **customer** can *select* from; alternatively a **customer** can *supply their own desired amount* instead. The **system** then *validates the chosen amount* by ensuring that the amount is greater or equal to £10, but is less than £250 per day and that the amount is a multiple of £10. When the amount is *chosen* the **customer's account** is *checked for sufficient funds*. If the **customer** has sufficient funds the **system** *displays the new account balance* and a **cash dispenser** *counts the required money*, *places the money in the money tray* and then *opens the money tray*, once the **customer** has *removed the money* the **cash dispenser** *closes the money tray*. The **customer** is then *asked if they require a receipt* to be *printed in A4* or *printed in A5*, for this the **system** has a **receipt printer** which can *print a receipt* by *setting the receipt paper size* and *setting the receipt information.* Afterwards they are *returned to the main menu* or alternatively if they don't have enough funds the **system** *displays an insufficient funds message* and they are *returned to the main menu*. For all parts of this functionality there is an option to *display information* that explains what operations can be done.

Depositing –

Deposit-transaction is the next transaction which *allows customers to deposit money*; this can be done in multiple ways. The first way is to *input a cheque amount* for this the **system** features a **cheque reader** which *accepts cheques* and then *checks that the cheque is valid* if it is not it is *rejected* otherwise the *total amount is obtained*. The second way is to *input a money amount* for this the **system** features a **money collector** which *accepts money notes* and then *checks that the money note is valid* if it is not it is *rejected* otherwise the *total amount is obtained*. The third is to *input a coin amount* for this the **system** features a **coin collector** which *accepts coins* and then *checks that the coin is valid* if it is not it is *rejected* otherwise the *total amount is obtained*. After the **customer** has *deposited the money* the **system** *displays the new account balance*. The **customer** is then *asked if they require a receipt* to be *printed in A4* or *printed in A5* using the aforementioned **receipt printer**, afterwards they are *returned to the main menu*. For all parts of this functionality there is an option to *display information* that explains what operations can be done.

View Account Information –

View-account-information-transaction is the next transaction this *allows a customer to get their current account balance*, *get previous transactions*, *search for previous transactions* and *get their sort code*. For any of these there are options to *print an A5 receipt* or *print an A4 receipt* using the aforementioned **receipt printer**. Once they have finished a **customer** can *return to the main menu*. For all parts of this functionality there is an option to *display information* that explains what operations can be done.

Topup Mobile Phone –

Topup-mobile-phone-transaction is the next **transaction** this allows a **customer** to *purchase a mobile phone topup*. The **system** *displays a list of mobile providers* (3, Vodafone etc) for the **customer** to *choose from*. The **system** then requests the **customer** to *input their mobile phone number* which is *validated* by using a **mobile phone provider connection** which can *verify the mobile phone number* once the *mobile phone provider has been set*. After the mobile phone number has been *validated* the **system** *displays a selection of topup amounts* (£10, £15, £20 etc) but **customers** can *supply a topup amount* if the amount they require is not there. The **system** *validates the amount* by ensuring it is greater or equal to £10 but is below £100 per day and is also a multiple of £5. When the topup amount has been chosen the **customer's account** is *checked for sufficient funds*. If the **customer** does not have sufficient funds they are *returned to the main menu*. Otherwise the **system** *displays the new account balance,* then the **customer** is then *asked if they require a receipt* to be *printed in A4* or *printed in A5* using the aforementioned **receipt printer**, finally they are *returned to the main menu*. For all parts of this functionality there is an option to *display information* that explains what operations can be done.

Pay Bills –

The final **transaction** possible is **pay-bills-transaction** which allows a **customer** to *pay a bill*. The **system** *displays the outstanding bills* that the **customer** has, this is done using a **utility connection** which *gets the outstanding bills* and they can then *choose the bills* they wish to pay. As always the **customer's account** is *checked for sufficient funds*. If the **customer** does not have sufficient funds they are *returned to the main menu*. Otherwise the **system** *displays the new account balance,* then the **customer** is *asked if they require a receipt* to be *printed in A4* or *printed in A5* using the aforementioned **receipt printer**, finally they are *returned to the main menu*. For all parts of this functionality there is an option to *display information* that explains what operations can be done.

All of these **transactions** use a SmartBank connection which *establishes a connection* to the SmartBank databases *commits the transaction* when ready and if necessary *reverses the transaction* to ensure data integrity.

Word/Phrase	**Accepted**	**Reason**
SmartBank	No	Company rather than about the system itself.
Online banking system	No	Refers to a different system that SmartBank has.
Expand	No	Irrelevant
Automated teller machines	No	This is far too general to be a class.
Towns	No	Irrelevant to system classes.
Cities	No	Irrelevant to system classes.
Providing the functionality	No	Irrelevant
Customers	No	They are actors of the system.
SmartBank	No	Duplicate
Presented	No	Too general
ATM screen	**ATMScreen**	**This is the user interface that the customer will see and hence should have a class of its own.**
Allows	No	Too general
Customer	No	Duplicate
Peform all the avaliable actions	No	Too general
System	No	Far too general to be used as a class.
Keyboard input	**KeyboardInput**	**This is a component of the system that allows user interaction and hence should have a class of its own.**
Receive key presses	**receivedKeyPress**	**This is an action of the keyboard input and hence should be a method of that class.**
Voice recogniser	**VoiceRecogniser**	**This is a component of the system that allows user interaction and hence should have a class of its own.**
Receives user voice input	**receivedVoiceInput**	**This is an action of the voice recogniser and hence should be a method of the class.**
Touchpad input	**TouchpadInput**	**This is a component of the system that allows user interaction and hence should have a class of its own.**
Receives user touches	**recievedTouchInput**	**This is an action of the touchpad input and hence should be a method of the class.**
System	No	Duplicate
Language support	**LanguageSupport**	**This is suitable as a class for all of the support for languages.**
English	No	Refers to a particular language supported rather than a specific component.
French	No	Refers to a particular language supported rather than a specific component.
Spanish	No	Refers to a particular language supported rather than a specific component.
Portuguese	No	Refers to a particular language supported rather than a specific component.
German	No	Refers to a particular language supported rather than a specific component.
Italian	No	Refers to a particular language supported rather than a specific component.
Chinese	No	Refers to a particular language supported rather than a specific component.

Displayed to the customer	**displayAvaliableLanguages**	**This is a particular action of the language support and hence becomes a method.**
User interface is updated with the chosen language	**updateUI**	**This is a particular action of the ATM screen and hence becomes a method.**
Customers	No	Duplicate
System	No	Duplicate
Verify their account	**verifyAccount**	**This is a particular action that needs to take place for a customer to use the system and hence becomes a method of ATMScreen.**
Verifier	**Verifier**	**This is a particular component of the system that allows customer's account to be verified and hence should be a class of its own.**
Card reader	**Card Reader**	**This is a particular component of the system that accepts customer's account cards and reads the data and hence should have a class of its own.**
Accepts a customer's card	**acceptCard**	**This is a particular action of the card reader and hence becomes a method.**
Returns back the PIN number	**returnPINNumber**	**This is a particular action of the card reader and hence becomes a method.**
Finger print scanner	**FingerPrintScanner**	**This is a particular component of the system that accepts a customer's finger print to be verified and hence should be a class of its own.**
Accepts a finger print	**acceptFingerPrint**	**This is a particular action of the finger print scanner and hence becomes a method.**
Reads the biometric information	**readBiometricInformation**	**This is a particular action of the finger print scanner and hence becomes a method.**
Checked	No	Too general
Verifier	No	Duplicate
SmartBank Connection	**SmartBankConnection**	**This is a particular connection to the SmartBank databases and hence becomes a class.**
Get customer's account details	**getAccountDetails**	**This is a particular action of the smart bank connection and hence becomes a method.**
Card is retained	**retainCard**	**This is a particular action of the ATM screen and hence becomes a method.**
Transaction	No	Too general
System	No	Duplicate
Displays a list of transactions	**displayTransactionOptions**	**This is a particular action of the system and becomes a method of ATMScreen.**
Transactions	No	Far too general to be a class
Withdraw-transaction	**Withdraw**	**This can be used for a class representing a withdraw transaction**
Allowing customers to withdraw money	No	Too general to be a method
System	No	Duplicate
Displays a number of money amounts	**displayMoneyAmounts**	**This is a particular action of the withdraw transaction and hence becomes a method.**

Customer	No	Duplicate
Select	No	Too general to be a method
Customer	No	Duplicate
Supply their own desired amount	**supplyMoneyAmount**	**This is a particular action of the withdraw transaction and hence becomes a method.**
System	No	Duplicate
Validates chosen amount	**validateChosenAmount**	**This is a particular action of the withdraw transaction and hence becomes a method**
Chosen	No	Too general to be a method
Customer's account	No	Too general
Checked for sufficient funds	**checkSufficientFunds**	**This is a particular action of withdraw transaction and hence becomes a method.**
System	No	Duplicate
Displays the new account balance	**displayNewAccountBalance**	**This is a particular action of withdraw transaction and hence becomes a method.**
Cash dispenser	**Cash Dispenser**	**This can be used for a class representing a cash dispenser.**
Counts the required money	**countMoney**	**This is a particular action of cash dispenser and hence becomes a method.**
Places the money in the tray	**placeMoneyInTray**	**This is a particular action of cash dispenser and hence becomes a method.**
Opens the money tray	**openMoneyTray**	**This is a particular action of cash dispenser and hence becomes a method.**
Customer	No	Duplicate
Removed the money	No	Refers to a customer action not an action of the system
Cash Dispenser	No	Duplicate
Customer	No	Duplicate
Closes the money tray	**Yes**	**This is a particular action of cash dispenser and becomes a method.**
Asked if they require a receipt	**askIfRequireReceipt**	**This is a particular action of withdraw transaction and hence becomes a method.**
Printed in A4	**printA4Receipt**	**This is a particular action of withdraw transaction and hence becomes a method.**
Printed in A5	**printA5Receipt**	**This is a particular action of withdraw transaction and hence becomes a method.**
System	No	Duplicate
Receipt printer	**ReceiptPrinter**	**This can be used for a class representing a cash dispenser.**
Print a receipt	**printReceipt**	This is a particular action of receipt printer and hence becomes a method.
Setting the receipt paper size	**setReceiptPaperSize**	**This is a particular action of receipt printer and hence becomes a method.**
Setting the receipt information	**setReceiptInformation**	**This is a particular action of receipt printer and hence becomes a method.**
Returned to main menu	**returnToMainMenu**	**This is a particular action of withdraw transaction and hence becomes a method.**
System	No	Duplicate

Displays an insufficient funds message	**displayInsufficientFundsMessage**	**This is a particular action of withdraw transaction and hence becomes a method.**
Returned to main menu	**returnToMainMenu**	**This is a particular action of withdraw transaction and hence becomes a method.**
Display information	**displayInformation**	**This is a particular action of withdraw transaction and hence becomes a method.**
Deposit-transaction	**Deposit**	**This can be used for a class representing a deposit transaction.**
Transaction	No	Duplicate
Allows customers to deposit money	No	Too general to be a method
Input a cheque amount	**inputChequeAmount**	**This is a particular action of the deposit transaction and hence becomes a method.**
System	No	Duplicate
Cheque reader	**ChequeReader**	**This can be used for a class representing a cheque reader.**
Accepts cheques	**acceptCheque**	**This is a particular action of the cheque reader and hence becomes a method.**
Checks that the cheque is valid	**checkCheque**	**This is a particular action of the cheque reader and hence becomes a method.**
Rejected	**rejectCheque**	**This is a particular action of the cheque reader and hence becomes a method.**
Total amount is obtained	**getTotalAmount**	**This is a particular action of the cheque reader and hence becomes a method.**
Input a money amount	**inputMoneyAmount**	**This is a particular action of the deposit transaction and hence becomes a method.**
System	No	Duplicate
Money collector	**MoneyCollector**	**This can be used for a class representing a money collector.**
Accepts money notes	**acceptMoneyNote**	**This is a particular action of the money collector and hence becomes a method.**
Checks that the money note is valid	**checkMoneyNote**	**This is a particular action of the money collector and hence becomes a method.**
Rejected	**rejectMoney**	**This is a particular action of the money collector and hence becomes a method.**
Total amount is obtained	**getTotalAmount**	**This is a particular action of the money collector and hence becomes a method.**
Input a coin amount	**inputCoinAmount**	**This is a particular action of deposit transaction and hence becomes a method.**
System	No	Duplicate
Coin collector	**CoinCollector**	**This can be used for a class representing a coin collector.**
Accepts coins	**acceptCoin**	**This is a particular action of the coin collector and hence becomes a method.**

Checks that the coin is valid	**checkCoinValid**	**This is a particular action of the coin collector and hence becomes a method.**
Rejected	**rejectCoin**	**This is a particular action of the coin collector and hence becomes a method.**
Total amount is obtained	**getTotalAmount**	**This is a particular action of the coin collector and hence becomes a method.**
Customer	No	Duplicate
Deposited the money	No	Too general
System	No	Duplicate
Displays the new account balance	**displayNewAccountBalance**	**This is a particular action of the deposit transaction and hence becomes a method.**
Customer	No	Duplicate
Asked if they require a receipt	**askIfRequireAReceipt**	**This is a particular action of the deposit transaction and hence becomes a method.**
Printed in A4	**printA4Receipt**	**This is a particular action of the deposit transaction and hence becomes a method.**
Printed in A5	**printA5Receipt**	**This is a particular action of the deposit transaction and hence becomes a method.**
Receipt printer	No	Duplicate
Returned to the main menu	**returnToMainMenu**	**This is a particular action of the deposit transaction and hence becomes a method.**
Display information	**displayInformation**	**This is a particular action of the deposit transaction and hence becomes a method.**
View-account-information-transaction	**ViewAccountInformation**	**This can be used for a class representing a view account information transaction.**
Transaction	No	Duplicate
Allows a customer to get their current account balance	**getCurrentAccountBalance**	**This is a particular action of the view account information transaction and hence becomes a method.**
Get previous transactions	**getPreviousTransactions**	**This is a particular action of the view account information transaction and hence becomes a method.**
Search for previous transactions	**searchPreviousTransactions**	**This is a particular action of the view account information transaction and hence becomes a method.**
Get their sort code	**getSortCode**	**This is a particular action of the view account information transaction and hence becomes a method.**
Print an A5 receipt	**printA5Receipt**	**This is a particular action of the view account information transaction and hence becomes a method.**
Print an A4 receipt	**printA4Receipt**	**This is a particular action of the view account information transaction and hence becomes a method.**
Receipt printer	No	Duplicate
Customer	No	Duplicate
Return to the main menu	**returnToMainMenu**	**This is a particular action of the view account information transaction and hence becomes a method.**

Display information	**displayInformation**	**This is a particular action of the view account information transaction and hence becomes a method.**
Topup-mobile-phone-transaction	**TopupMobilePhone**	**This can be used for a class representing a topup mobile phone transaction.**
Transaction	No	Duplicate
Customer	No	Duplicate
Purchase a mobile phone topup	No	Far too general
System	No	Duplicate
Displays a list of mobile phone providers	**displayMobilePhoneProviders**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Customer	No	Duplicate
Choose from	**setMobilePhoneProvider**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
System	No	Duplicate
Customer	No	Duplicate
Input their mobile phone number	**setMobilePhoneNumber**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Validate	**validateMobilePhoneNumber**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Mobile phone provider connection	**MobilePhoneProviderConnection**	**This can be used for a class representing a mobile phone provider connection.**
Verify the mobile phone number	**verifyMobilePhoneNumber**	**This is a particular action of the mobile phone provider connection and hence becomes a method.**
Mobile phone number has been set	**setMobilePhoneNumber**	**This is a particular action of the mobile phone provider connection and hence becomes a method.**
Validated	No	Duplicate
System	No	Duplicate
Displays a selection of topup amounts	**displayTopupAmounts**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Customers	No	Duplicate
Supply a topup amount	**setToupAmount**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
System	No	Duplicate
Validates the amount	**validateTopupAmount**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Customer's account	No	Too general
Checked for sufficient funds	**checkForSufficientFunds**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Customer	No	Duplicate
Returned to the main menu	**returnToMainMenu**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
System	No	Duplicate
Displays the new account balance	**displayNewAccountBalance**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**

Customer	No	Duplicate
Asked if they require a receipt	**askIfRequireReceipt**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Printed in A4	**printA4Receipt**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Printed in A5	**printA5Receipt**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Receipt printer	No	Duplicate
Returned to the main menu	No	Duplicate
Display information	**displayInformation**	**This is a particular action of the topup mobile phone transaction and hence becomes a method.**
Transaction	No	Duplicate
Pay-bills-transaction	**PayBills**	**This can be used to represent the pay bills transaction.**
Customer	No	Duplicate
Pay a bill	No	Far too general
System	No	Duplicate
Displays outstanding bills	**displayOutstandingBills**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Customer	No	Duplicate
Utility connection	**UtilityConnection**	**This can be used to represent a utility connection.**
Gets the outstanding bills	**getOutstandingBills**	**This is a particular action of the utility connection and hence becomes a method.**
Choose the bills	**chooseBill**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Customer's account	No	Duplicate
Checked for sufficient funds	**checkForSufficientFunds**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Customer	No	Duplicate
Returned to the main menu	**returnToMainMenu**	**This is a particular action of the pay bills transaction and hence becomes a method.**
System	No	Duplicate
Displays the new account balance	**displayNewAccountBalance**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Customer	No	Duplicate
Asked if they require a receipt	**askIfRequireReceipt**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Printed in A4	**printA4Receipt**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Printed in A5	**printA5Receipt**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Receipt printer	No	Duplicate
Returned to the main menu	No	Duplicate

Display information	**displayInformation**	**This is a particular action of the pay bills transaction and hence becomes a method.**
Transactions	No	Duplicate
SmartBank connection	No	Duplicate
Establishes a connection	**establishConnection**	**This is a particular action of the SmartBank Connection and hence becomes a method.**
Commits the transaction	**commitTransaction**	**This is a particular action of the SmartBank Connection and hence becomes a method.**
Reverses the transaction	**reverseTransaction**	**This is a particular action of the SmartBank Connection and hence becomes a method.**

Responsibility Driven Analysis

Using the knowledge gained from the specification, functional requirements, non-functional requirements and noun-verb analysis the following is a responsibility driven analysis of the system identifying the class, responsibilities and collaborators. This when combined with all the aforementioned parts will be aid the construction of the first cut class diagram and eventually the fully developed class diagram.

CardReader	
Responsibilities	**Collaborators**
The responsibilities of this class are to accept a customer's bank account card and to read the details of the card for PIN verification.	

SmartBankConnection	
Responsibilities	**Collaborators**
The responsibility of this class is to provide an interface to the SmartBank databases for account verification and for the various transactions provided by the ATM.	

ReceiptPrinter	
Responsibilities	**Collaborators**
The responsibility of this class is to prints out the receipt in different sizes including A4 or A5 for the different transactions provided by the ATM.	

FingerPrintScanner	
Responsibilities	**Collaborators**
The responsibility of this class is to accept a customer's finger-print and return the biometric information so it can be used for customer verification.	

CashDispenser	
Responsibilities	**Collaborators**
The responsibility of this class is to dispense the amount of money chosen during a withdraw transaction.	

ChequeReader	
Responsibilities	**Collaborators**
The responsibility of this class is to read and analyse cheques inserted by a customer rejecting a cheque if it is not valid, otherwise returning the amount stated on the cheque.	

MoneyCollector	
Responsibilities	**Collaborators**
The responsibility of this class is to accept money notes inserted by a customer rejecting invalid notes, otherwise returning the amount of the money note.	

CoinCollector	
Responsibilities	**Collaborators**
The responsibility of this class is to accept money coins inserted by a customer rejecting invalid coins, otherwise returning the amount of the coin inserted.	

LanguageSupport	
Responsibilities	**Collaborators**
The responsibility of this class is to validate a customer’s chosen language, to return all the languages that are supported, provide information for ATMScreen for different languages for the user interface and to also interpret customer requests using speech recognition later on in a customer’s session when they are performing transactions.	

Verifier	
Responsibilities	**Collaborators**
The responsibility of this class is to verify the customer’s credentials including PIN number and biometric information and to retain the customer's account card if verification has not been successful after three attempts.	SmartBankConnection FingerPrintScanner CardReader

Transaction	
Responsibilities	**Collaborators**
The responsibility of this class is to provide the common functionality of a transaction it is not used solely on it’s own but it used by real transactions which extend the class.	SmartBankConnection LanguageSupport

Withdraw	
Responsibilities	**Collaborators**
The responsibility of this class is to provide all of the functionality required to allow a customer to withdraw money.	Transaction CashDispenser ReceiptPrinter ATMScreen

Deposit	
Responsibilities	**Collaborators**
The responsibility of this class is to provide all of the functionality required to allow a customer to deposit money.	Transaction ChequeReader CoinCollector MoneyCollector ReceiptPrinter ATMScreen

TopupMobilePhone	
Responsibilities	**Collaborators**
The responsibility of this class is to provide all of the functionality required to allow a customer to topup their mobile phone.	Transaction MobilePhoneProviderConnection ReceiptPrinter ATMScreen

PayBills	
Responsibilities	**Collaborators**
The responsibility of this class is to provide all of the functionality required to allow a customer to pay their bills.	Transaction UtilityConnection ReceiptPrinter SmartBankConnection

ViewAccountInformation	
Responsibilities	**Collaborators**
The responsibility of this class is to provide all of the functionality required to allow a customer to view account information and view previous transactions.	Transaction ReceiptPrinter SmartBankConnection

MobilePhoneProviderConnection	
Responsibilities	**Collaborators**
The responsibility of this class is to accept a mobile phone provider and to establish a connection to that mobile phone provider’s interface (which in turn will connect to the mobile phone provider’s database) allowing for checking of the customer’s mobile phone number, then finally updating the credit amount of the customer by passing the topup amount to the interface.	

UtilityConnection	
Responsibilities	**Collaborators**
The responsibility of this class it to accept a utility company and to establish a connection to that utility company’s interface (which in turn will connect to the mobile phone provider’s database) allowing for the payment of a utility bill with the bill reference being passed. It is also responsible for getting the outstanding bills for a customer.	

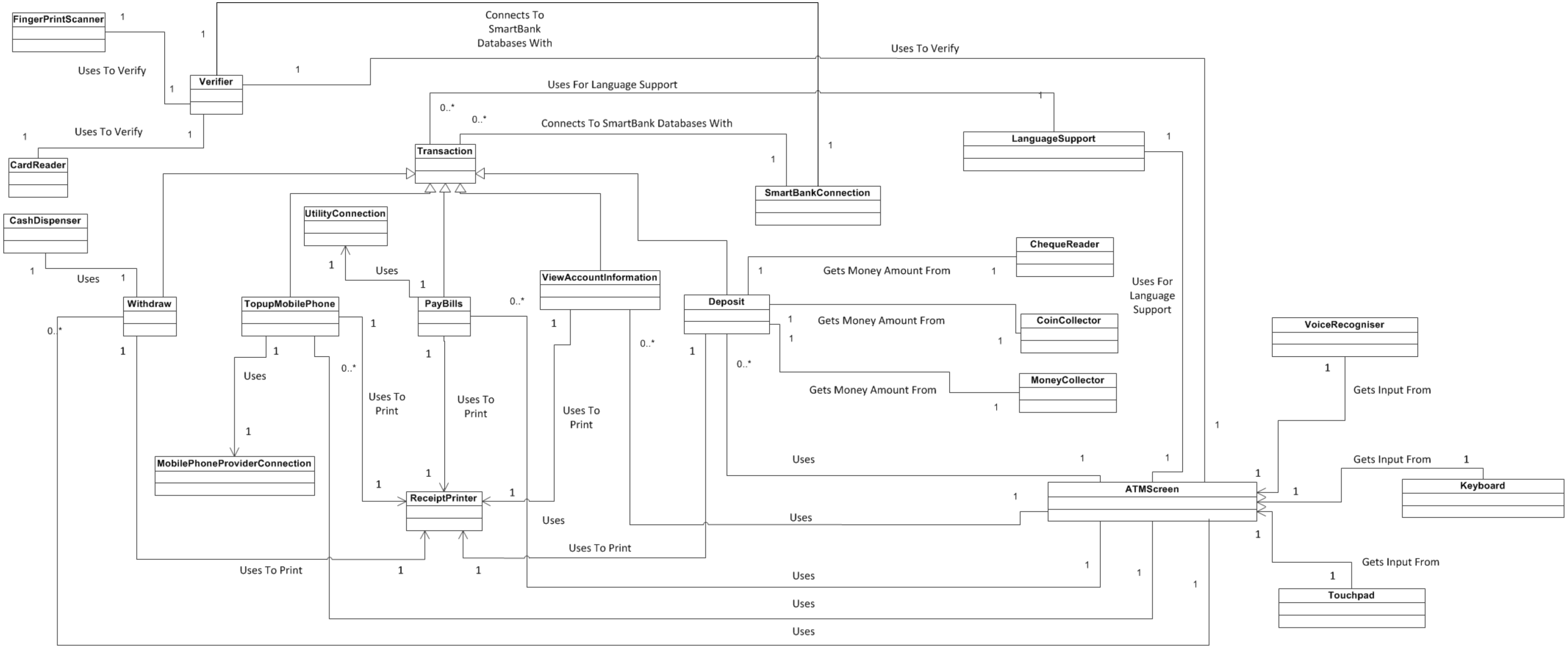
ATMScreen	
Responsibilities	**Collaborators**
The responsibility of this class is to provide access to the functionality of the ATM machine providing a graphical user interface using many of the other classes.	Verifier LanguageSupport VoiceRecogniser KeyboardInput TouchpadInput Withdraw TopupMobilePhone PayBills ViewAccountInformation Deposit

VoiceRecogniser	
Responsibilities	**Collaborators**
The responsibility of this class is to accept voice input from the customer and pass it to the ATMScreen.	

KeyboardInput	
Responsibilities	**Collaborators**
The responsibility of this class is to accept keyboard input from the customer and pass it to the ATMScreen.	

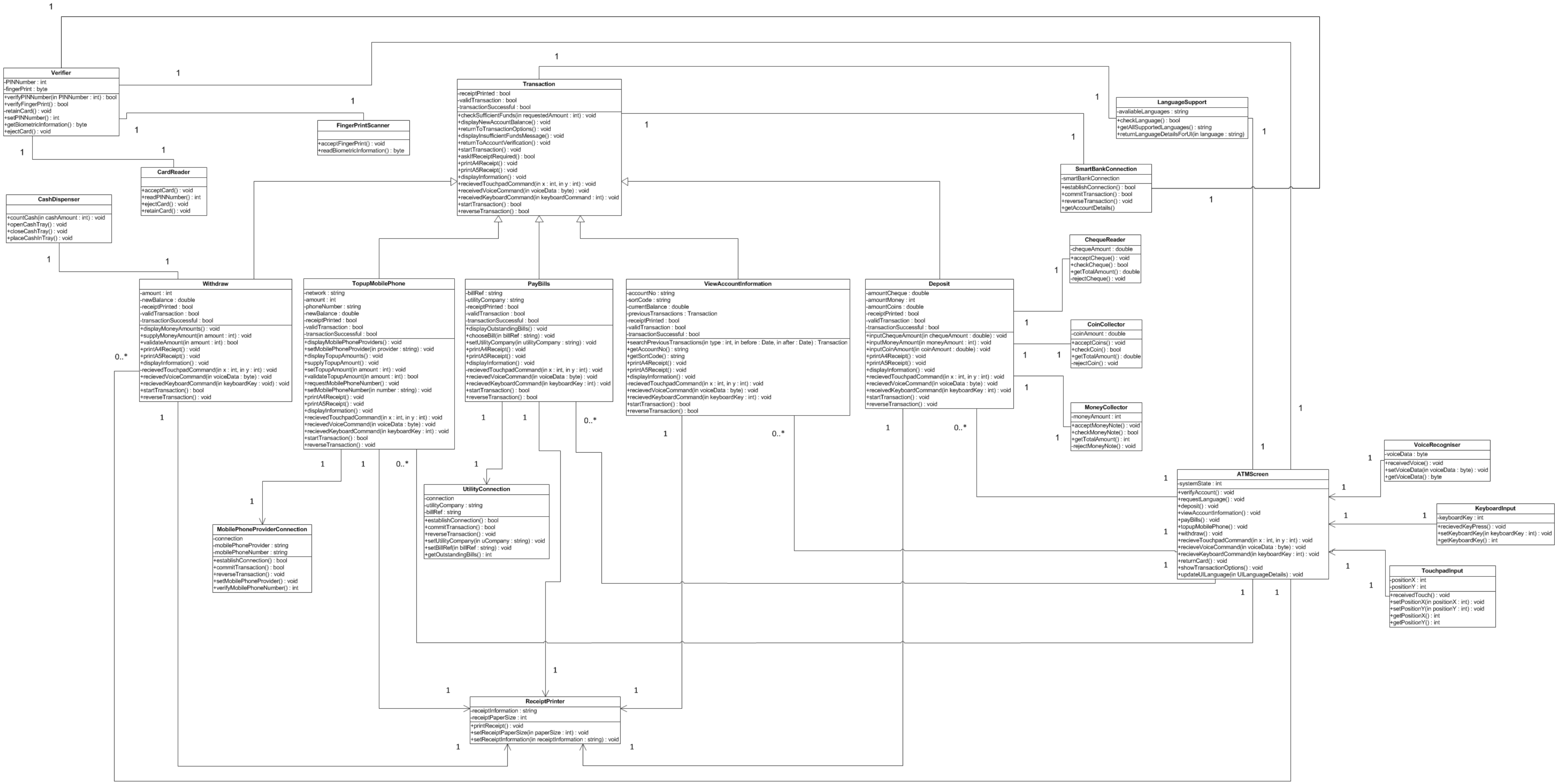
TouchpadInput	
Responsibilities	**Collaborators**
The responsibility of this class is to accept touchpad input from the customer and pass it to the ATMScreen.	

First Cut Class Diagram

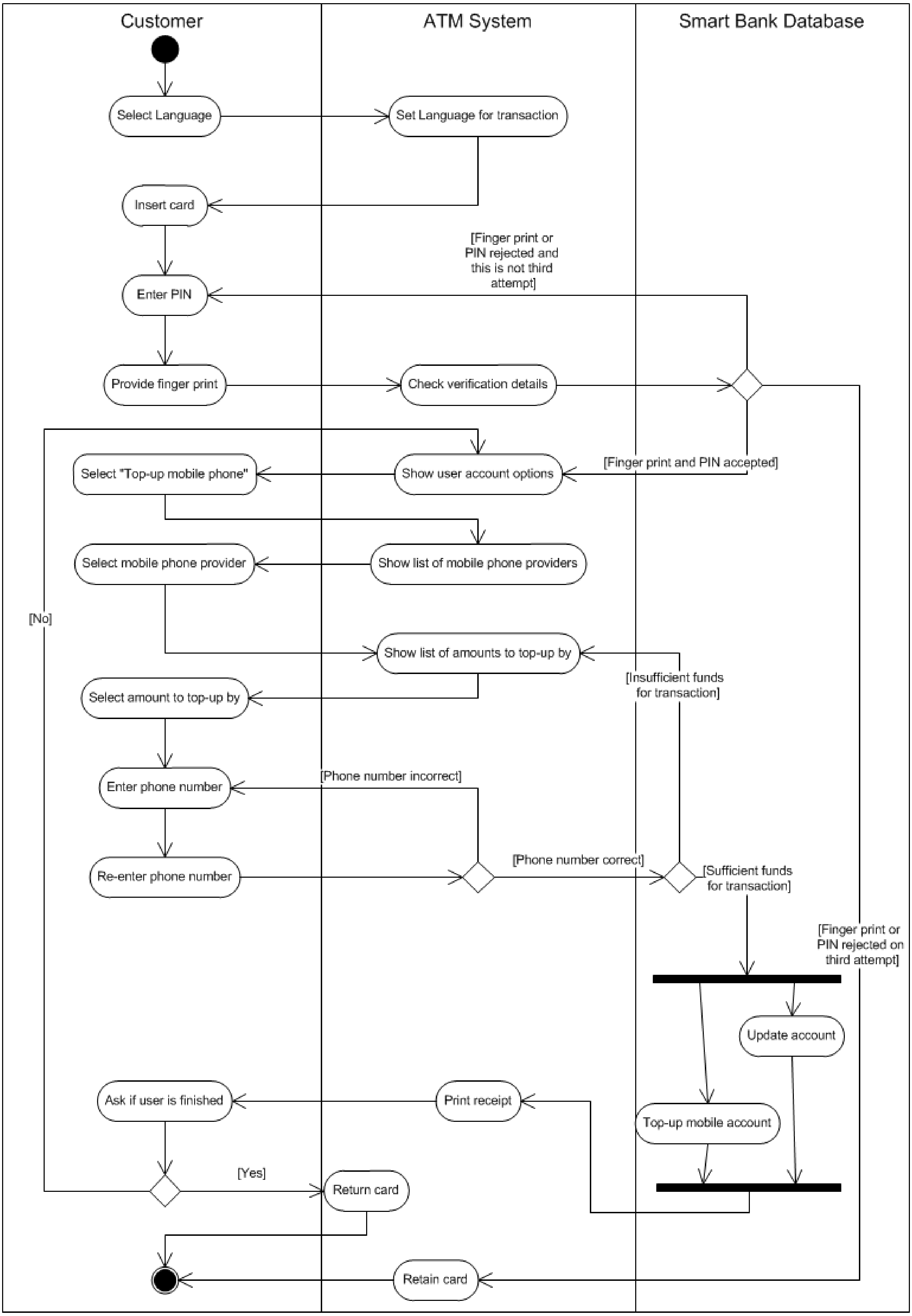


Using the first cut class diagram we set out to create a more complete class diagram with methods and attributes.

Class Diagram

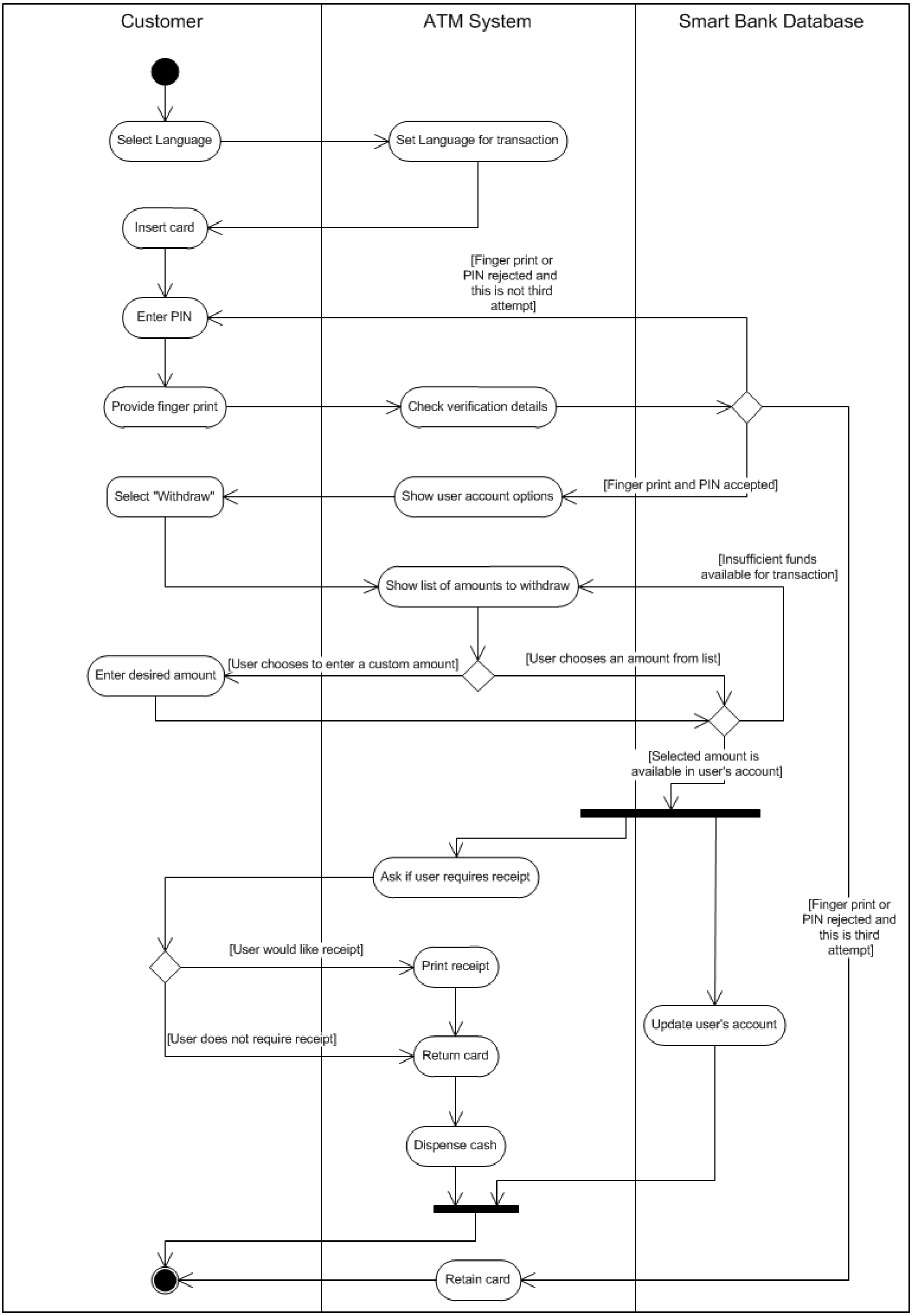

Verifier
-PINNumber : int
-fingerPrint : byte
+verifyPINNumber(in PINNumber : int) : bool
+verifyFingerPrint() : bool
-retainCard() : void
+setPINNumber() : int
+getBiometricInformation() : byte
+ejectCard() : void
FingerPrintScanner
+acceptFingerPrint() : void
+readBiometricInformation() : byte
CardReader
+acceptCard() : void
+readPINNumber() : int
+ejectCard() : void
+retainCard() : void
CashDispenser
+countCash(in cashAmount : int) : void
+openCashTray() : void
+closeCashTray() : void
+placeCashInTray() : void
Transaction
-receiptPrinted : bool
-validTransaction : bool
-transactionSuccessful : bool
+checkSufficientFunds(in requestedAmount : int) : void
+displayNewAccountBalance() : void
+returnToTransactionOptions() : void
+displayInsufficientFundsMessage() : void
+returnToAccountVerification() : void
+startTransaction() : void
+askIfReceiptRequired() : bool
+printA4Receipt() : void
+printA5Receipt() : void
+displayInformation() : void
+recievedTouchpadCommand(in x : int, in y : int) : void
+receivedVoiceCommand(in voiceData : byte) : void
+receivedKeyboardCommand(in keyboardCommand : int) : void
+startTransaction() : bool
+reverseTransaction() : bool
LanguageSupport
-avaliableLanguages : string
+checkLanguage() : bool
+getAllSupportedLanguages() : string
+returnLanguageDetailsForUI(in language : string)
SmartBankConnection
-smartBankConnection
+establishConnection() : bool
+commitTransaction() : bool
+reverseTransaction() : void
+getAccountDetails()
ChequeReader
-chequeAmount : double
+acceptCheque() : void
+checkCheque() : bool
+getTotalAmount() : double
-rejectCheque() : void
Withdraw
-amount : int
-newBalance : double
-receiptPrinted : bool
-validTransaction : bool
-transactionSuccessful : bool
+displayMoneyAmounts() : void
+supplyMoneyAmount(in amount : int) : void
+validateAmount(in amount : int) : bool
+printA4Reciept() : void
+printA5Receipt() : void
+displayInformation() : void
-recievedTouchpadCommand(in x : int, in y : int) : void
+recievedVoiceCommand(in voiceData : byte) : void
+recievedKeyboardCommand(in keyboardKey : void) : void
+startTransaction() : bool
+reverseTransaction() : void
TopupMobilePhone
-network : string
-amount : int
-phoneNumber : string
-newBalance : double
-receiptPrinted : bool
-validTransaction : bool
-transactionSuccessful : bool
+displayMobilePhoneProviders() : void
+setMobilePhoneProvider(in provider : string) : void
+displayTopupAmounts() : void
+supplyTopupAmount() : void
+setTopupAmount(in amount : int) : void
+validateTopupAmount(in amount : int) : bool
+requestMobilePhoneNumber() : void
+setMobilePhoneNumber(in number : string) : void
+printA4Receipt() : void
+printA5Receipt() : void
+displayInformation() : void
+recievedTouchpadCommand(in x : int, in y : int) : void
+recievedVoiceCommand(in voiceData : byte) : void
+recievedKeyboardCommand(in keyboardKey : int) : void
+startTransaction() : bool
+reverseTransaction() : void
PayBills
-billRef : string
-utilityCompany : string
-receiptPrinted : bool
-validTransaction : bool
-transactionSuccessful : bool
+displayOutstandingBills() : void
+chooseBill(in billRef : string) : void
+setUtilityCompany(in utilityCompany : string) : void
+printA4Receipt() : void
+printA5Receipt() : void
+displayInformation() : void
-recievedTouchpadCommand(in x : int, in y : int) : void
+recievedVoiceCommand(in voiceData : byte) : void
+recievedKeyboardCommand(in keyboardKey : int) : void
+startTransaction() : bool
+reverseTransaction() : bool
ViewAccountInformation
-accountNo : string
-sortCode : string
-currentBalance : double
-previousTransactions : Transaction
-receiptPrinted : bool
-validTransaction : bool
-transactionSuccessful : bool
+searchPreviousTransactions(in type : int, in before : Date, in after : Date) : Transaction
+getAccountNo() : string
+getSortCode() : string
+printA4Receipt() : void
+printA5Receipt() : void
+displayInformation() : void
-recievedTouchpadCommand(in x : int, in y : int) : void
+recievedVoiceCommand(in voiceData : byte) : void
+recievedKeyboardCommand(in keyboardKey : int) : void
+startTransaction() : bool
+reverseTransaction() : bool
Deposit
-amountCheque : double
-amountMoney : int
-amountCoins : double
-receiptPrinted : bool
-validTransaction : bool
-transactionSuccessful : bool
+inputChequeAmount(in chequeAmount : double) : void
+inputMoneyAmount(in moneyAmount : int) : void
+inputCoinAmount(in coinAmount : double) : void
+printA4Receipt() : void
+printA5Receipt() : void
+displayInformation() : void
+recievedTouchpadCommand(in x : int, in y : int) : void
+recievedVoiceCommand(in voiceData : byte) : void
+receivedKeyboardCommand(in keyboardKey : int) : void
+startTransaction() : void
+reverseTransaction() : void
CoinCollector
-coinAmount : double
+acceptCoins() : void
+checkCoin() : bool
+getTotalAmount() : double
-rejectCoin() : void
MoneyCollector
-moneyAmount : int
+acceptMoneyNote() : void
+checkMoneyNote() : bool
+getTotalAmount() : int
-rejectMoneyNote() : void
VoiceRecogniser
-voiceData : byte
+receivedVoice() : void
+setVoiceData(in voiceData : byte) : void
+getVoiceData() : byte
ATMScreen
-systemState : int
+verifyAccount() : void
+requestLanguage() : void
+deposit() : void
+viewAccountInformation() : void
+payBills() : void
+topupMobilePhone() : void
+withdraw() : void
+recieveTouchpadCommand(in x : int, in y : int) : void
+recieveVoiceCommand(in voiceData : byte) : void
+recieveKeyboardCommand(in keyboardKey : int) : void
+returnCard() : void
+showTransactionOptions() : void
+updateUILanguage(in UILanguageDetails) : void
KeyboardInput
-keyboardKey : int
+recievedKeyPress() : void
+setKeyboardKey(in keyboardKey : int) : void
+getKeyboardKey() : int
TouchpadInput
-positionX : int
-positionY : int
+receivedTouch() : void
+setPositionX(in positionX : int) : void
+setPositionY(in positionY : int) : void
+getPositionX() : int
+getPositionY() : int
MobilePhoneProviderConnection
-connection
-mobilePhoneProvider : string
-mobilePhoneNumber : string
+establishConnection() : bool
+commitTransaction() : bool
+reverseTransaction() : void
+setMobilePhoneProvider() : void
+verifyMobilePhoneNumber() : int
UtilityConnection
-connection
-utilityCompany : string
-billRef : string
+establishConnection() : bool
+commitTransaction() : bool
+reverseTransaction() : void
+setUtilityCompany(in uCompany : string) : void
+setBillRef(in billRef : string) : void
+getOutstandingBills() : int
ReceiptPrinter
-receiptInformation : string
-receiptPaperSize : int
+printReceipt() : void
+setReceiptPaperSize(in paperSize : int) : void
+setReceiptInformation(in receiptInformation : string) : void
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
0..*
1
1
1
1
1
1
1
0..*
1
0..*
1
0..*
1
1
1
1
1
0..*
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1

Activity Diagram 1 (Topup Mobile Phone)



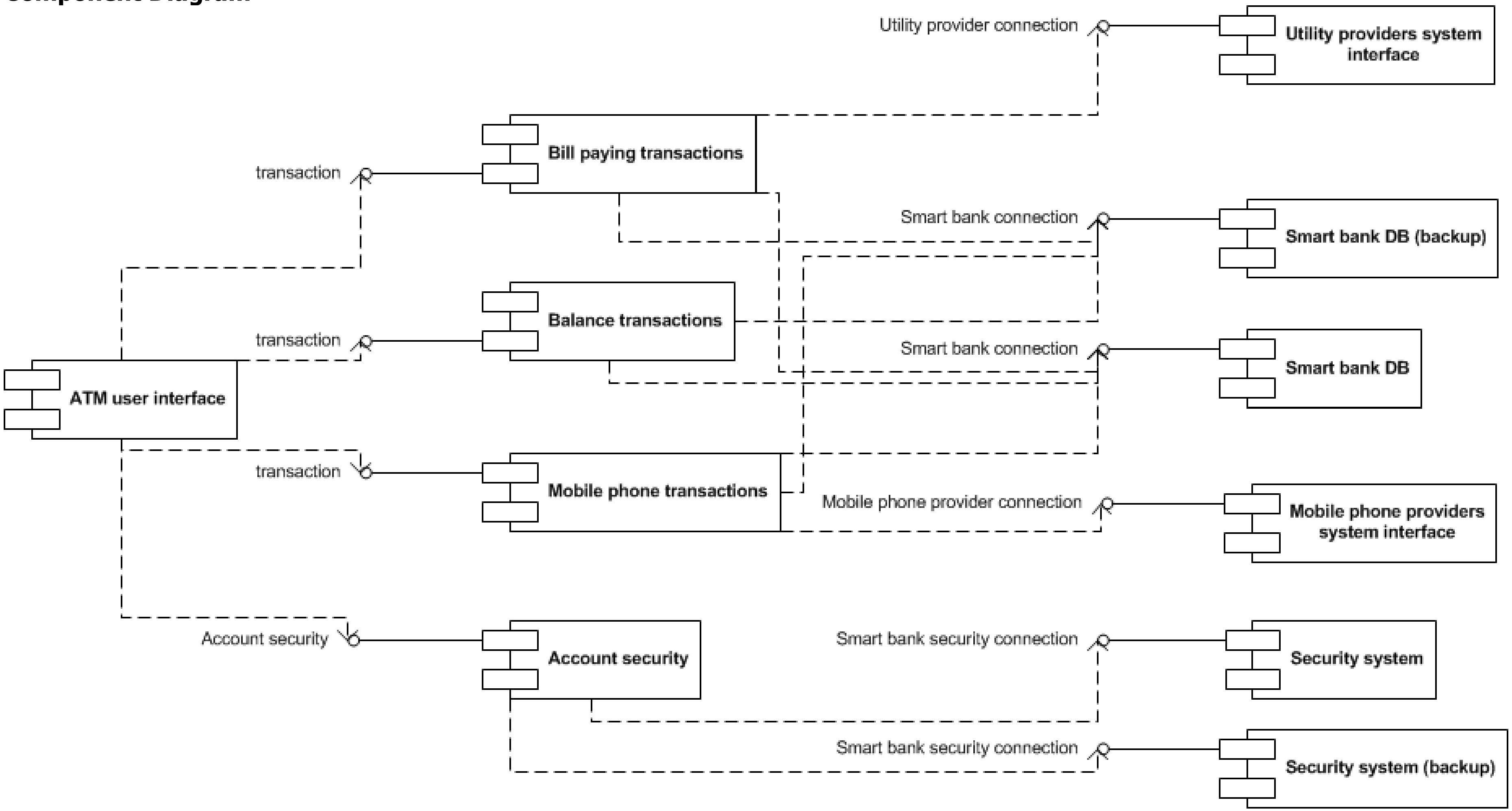
Customer
ATM System
Smart Bank Database
Select Language
Set Language for transaction
Insert card
[Finger print or PIN rejected and this is not third attempt]
Enter PIN
Provide finger print
Check verification details
Select "Top-up mobile phone"
Show user account options
[Finger print and PIN accepted]
Select mobile phone provider
Show list of mobile phone providers
[No]
Show list of amounts to top-up by
[Insufficient funds for transaction]
Select amount to top-up by
[Phone number incorrect]
Enter phone number
[Phone number correct]
Re-enter phone number
[Sufficient funds for transaction]
[Finger print or PIN rejected on third attempt]
Update account
Ask if user is finished
Print receipt
Top-up mobile account
[Yes]
Return card
Retain card

Activity Diagram 2 (Withdraw Money)

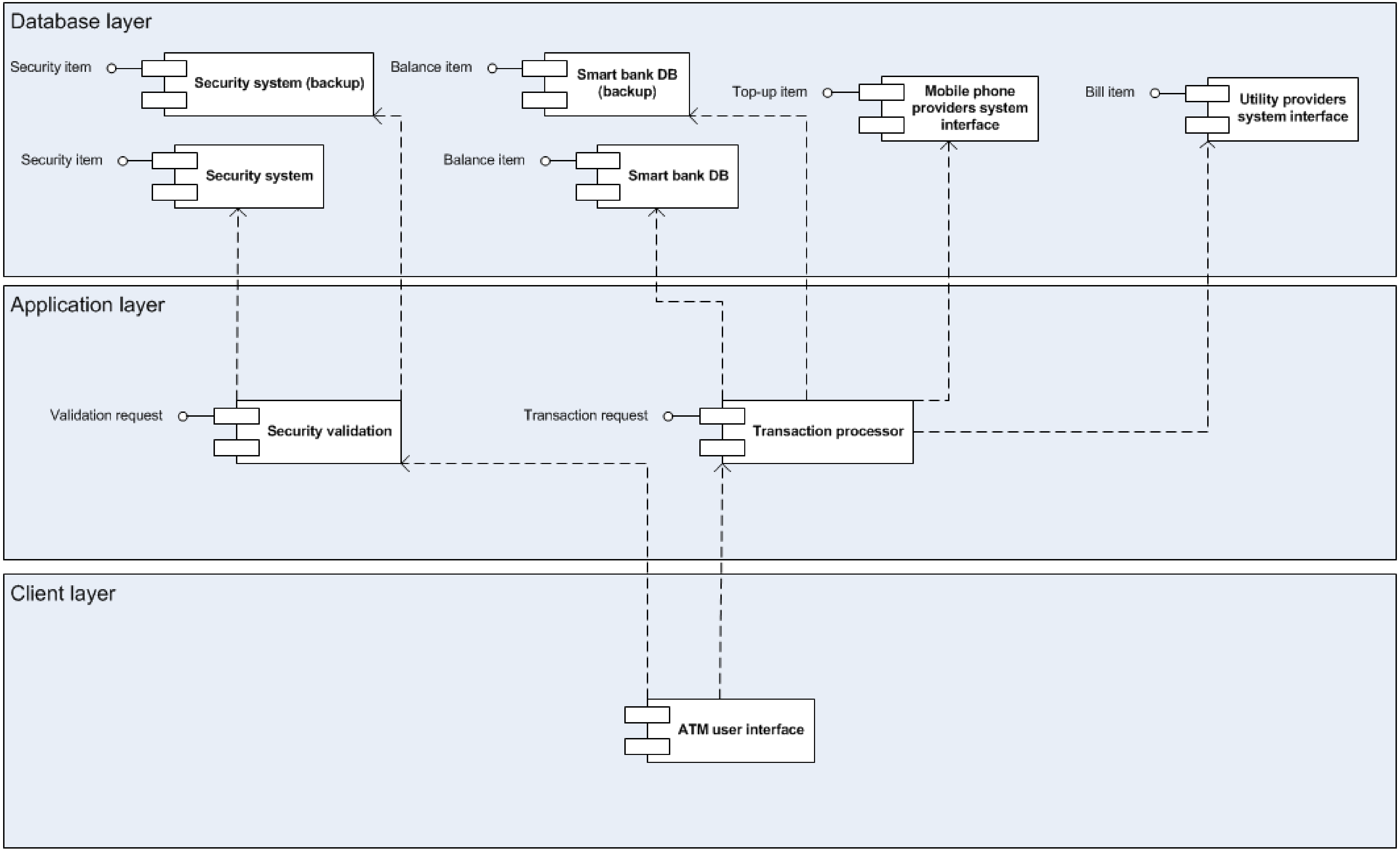


Customer
ATM System
Smart Bank Database
Select Language
Set Language for transaction
Insert card
[Finger print or PIN rejected and this is not third attempt]
Enter PIN
Provide finger print
Check verification details
Select "Withdraw"
Show user account options
[Finger print and PIN accepted]
Show list of amounts to withdraw
[Insufficient funds available for transaction]
Enter desired amount
[User chooses to enter a custom amount]
[User chooses an amount from list]
[Selected amount is available in user's account]
Ask if user requires receipt
[Finger print or PIN rejected and this is third attempt]
[User would like receipt]
Print receipt
Update user's account
[User does not require receipt]
Return card
Dispense cash
Retain card

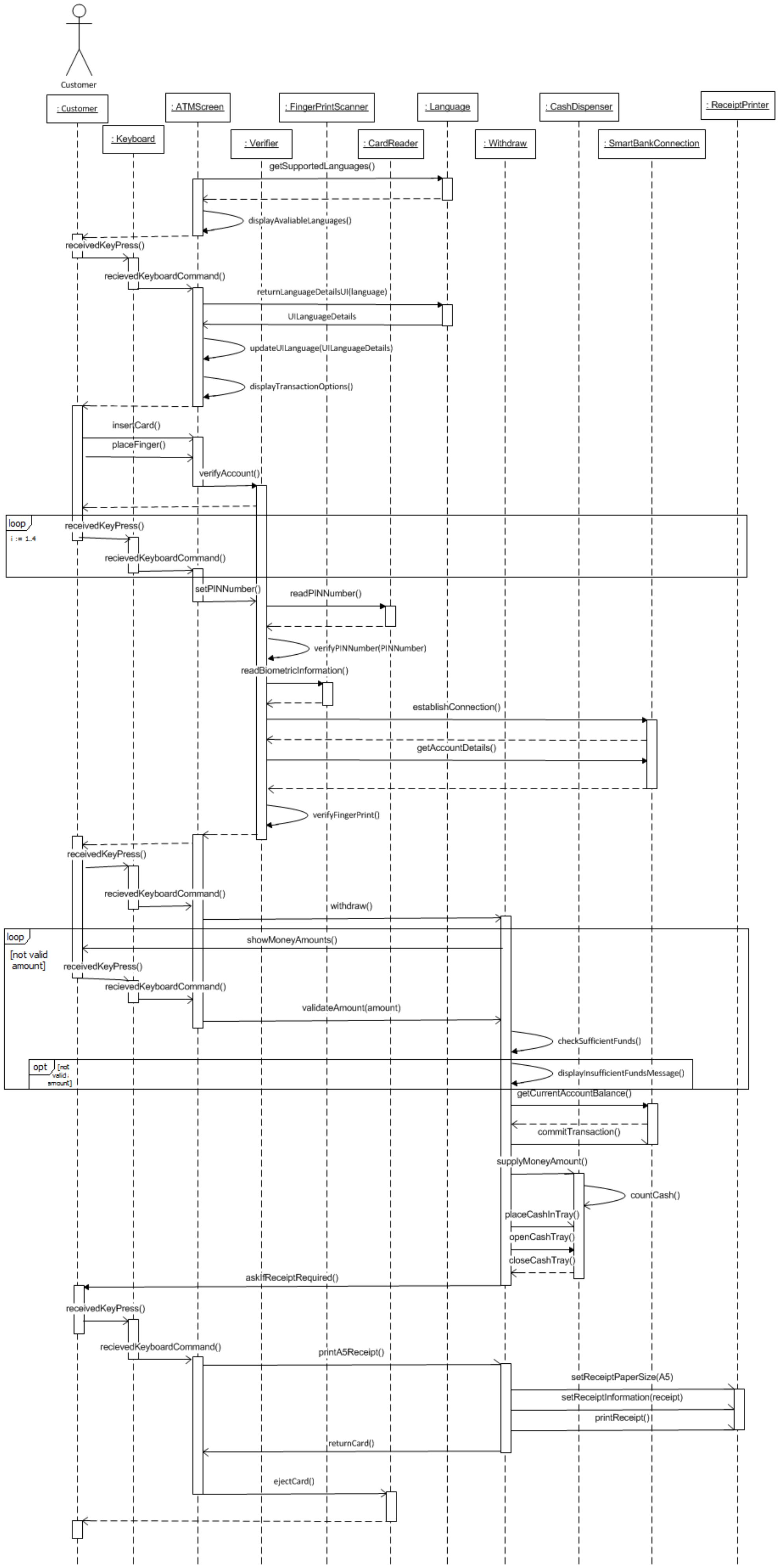
Component Diagram



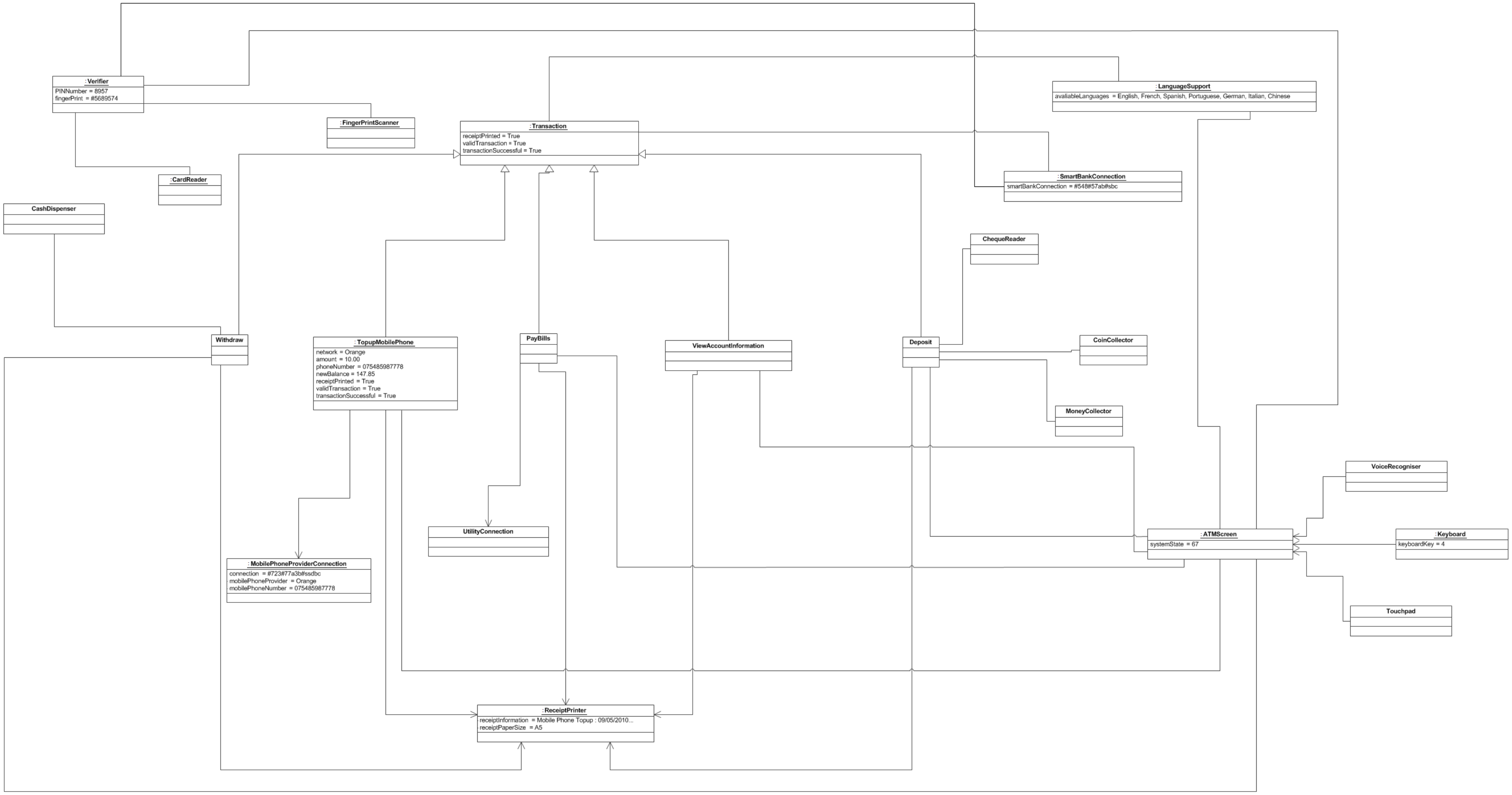
Deployment Diagram



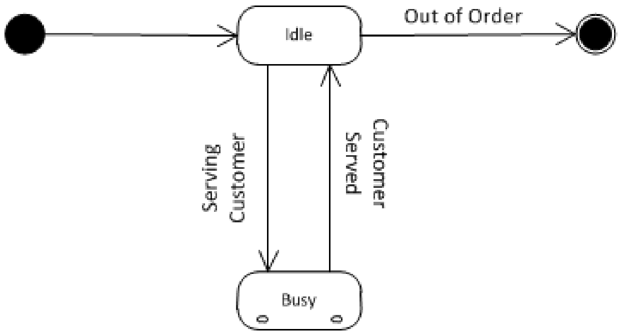
Sequence Diagram (Withdraw Money)

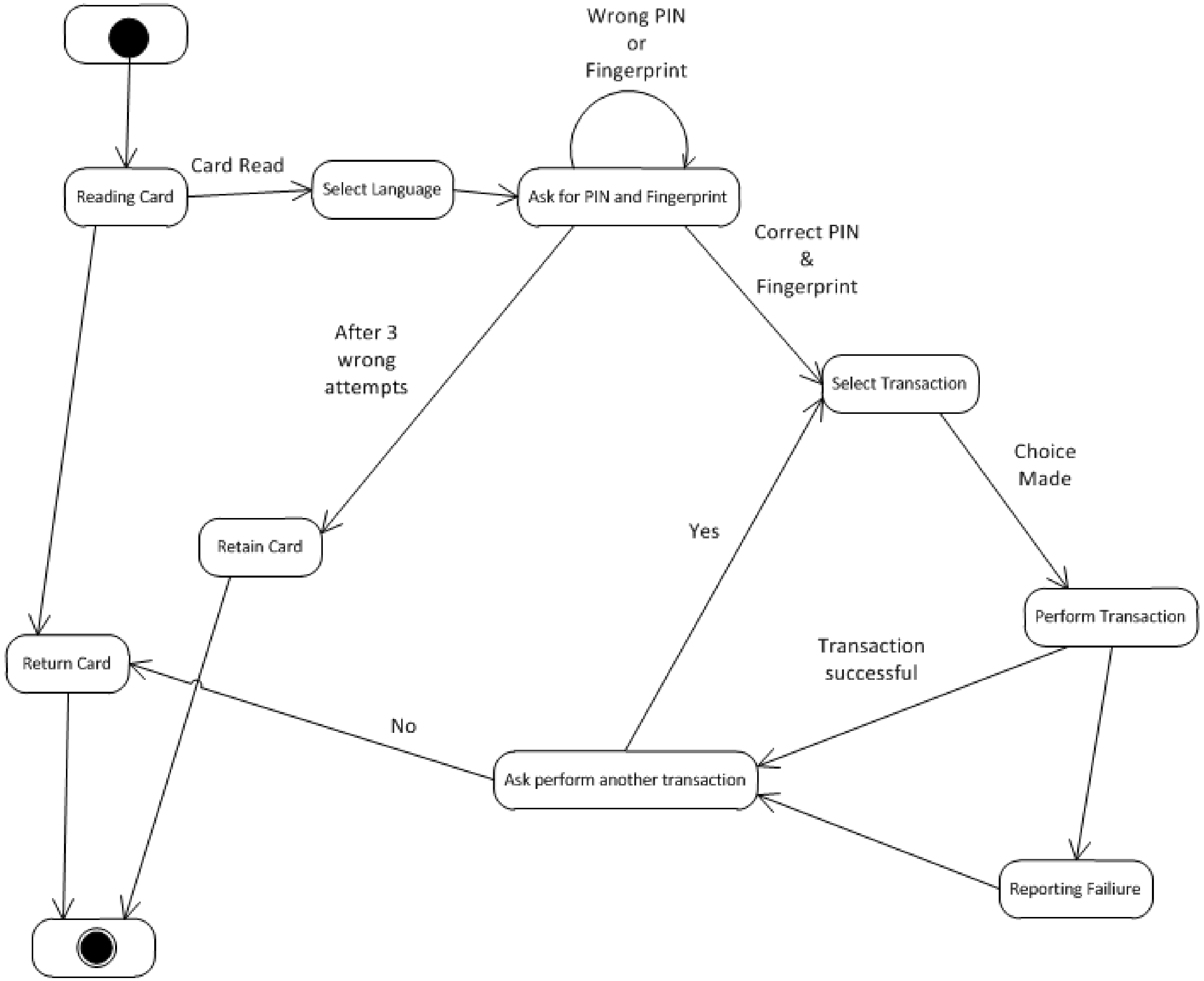


Object Diagram (Topup Mobile Phone – Other Classes Are Shown As Well But Are Not Active)



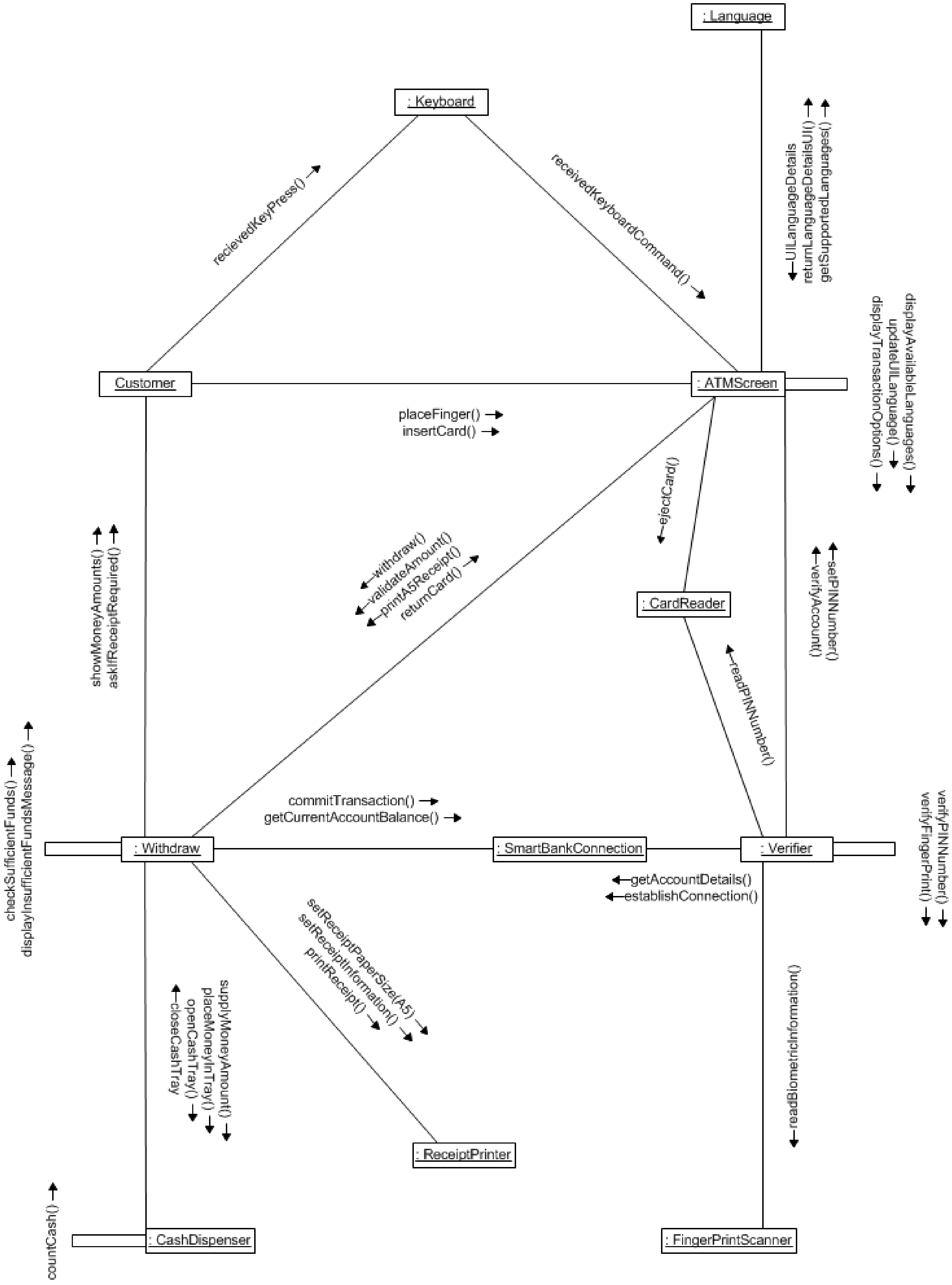
State Diagram (ATM System Itself)





Reference: Rami Bahsoon
Engineering OO Software with Unified Modelling Language (UML)
http://www.cs.bham.ac.uk/~rzb/Software%20Engineering%20-UML2.pdf

Collaboration Diagram (Withdraw Money)



: Language
: Keyboard
recievedKeyPress()
receivedKeyboardCommand()
UILanguageDetails
returnLanguageDetailsUI()
getSupportedLanguages()
displayAvailableLanguages()
updateUILanguage()
displayTransactionOptions()
Customer
: ATMScreen
placeFinger()
insertCard()
ejectCard()
withdraw()
validateAmount()
printA5Receipt()
returnCard()
setPINNumber()
verifyAccount()
: CardReader
readPINNumber()
showMoneyAmounts()
askIfReceiptRequired()
checkSufficientFunds()
displayInsufficientFundsMessage()
commitTransaction()
getCurrentAccountBalance()
: Withdraw
: SmartBankConnection
: Verifier
getAccountDetails()
establishConnection()
verifyPINNumber()
verifyFingerPrint()
setReceiptPaperSize(A5)
setReceiptInformation()
printReceipt()
supplyMoneyAmount()
placeMoneyInTray()
openCashTray()
closeCashTray
readBiometricInformation()
: ReceiptPrinter
countCash()
: CashDispenser
: FingerPrintScanner