ATM User Interface Design
Requirements

A statement of requirements for a portion of the system may be extracted from the use cases and other Inception documents. A brief narrative is useful for identifying the problem domain objects (concepts).

A bank customer is able to access his or her account using an automatic teller machine. To be able to use an ATM a customer must first register an account number and a passcode number. The customer’s information is then added to a list of registered users. The ATM user interface consists of a keypad, a display window, a selection of choice options, and a help screen that displays instructions for completing an ATM transaction. Users are asked to enter their account number from the keypad followed by their passcode. If the customer is a valid user, instructions are given for choosing a transaction. During a transaction, the user’s account is accessed and updated. Upon completion of a transaction, the user may elect to make another transaction or to quit.

This prototype simulates the actual machine that includes a card reader and buttons implemented in the hardware.
Requirements

Identify the noun phrases and verb phrases in the narrative

A bank customer is able to access his or her account using an automatic teller machine. To be able to use an ATM a customer must first register an account number and a passcode number. The customer’s information is then added to a list of registered users. The ATM user interface consists of a keypad, a display window, a selection of choice options, and a help screen that displays instructions for completing an ATM transaction. Users are asked to enter their account number from the keypad followed by their passcode. If the customer is a valid user, instructions are given for choosing a transaction. During a transaction, the user’s account is accessed and updated. Upon completion of a transaction, the user may elect to make another transaction or to quit.
List of Noun Phrases

Bank customer  Concept
Account  Concept
Automatic teller machine  Redundant, same as ATM
ATM  Concept
Account number  Attribute
Passcode number  Attribute
Customer information  Concept
List of registered users  Concept
ATM user interface  Concept
Keypad  Concept
Display window  Concept
Help screen  Concept
Selection of choice options  Concept
Transaction  Concept

But in this example the ATM relates only to the user interface – same as ATM

Instructions

Attribute
Text strings
Associations

Customer *accesses* Account (via ATM)

Customer Information *is added to* List of registered users

ATM *consists of* Keypad, Display, Help screen, Selection of choice options

Account is accessed *by* (ATM)

ATM *verifies* Customer Information

*Implicit*

Customer *has* Account

Keypad *has* Buttons

Choice Selection *uses* Buttons

Draw a Domain Model relating these Concepts
State Transition Diagram

- **Welcome**
  - do: enter acct. #
  - Digit | append digit
  - Enter
  - Clear | clear display

- **Validate**
  - do: enter passcode
  - [valid] Enter
  - [invalid] Enter

- **Select**
  - Check Bal | balance
  - Enter
  - Withdraw
  - Deposit

- **Deposit**
  - do: enter amt
  - Enter | clear display
  - Clear | clear display

- **Withdraw**
  - do: enter amt.
  - Enter | clear display
  - Clear | clear display

- **Another**
  - Clear | clear display
Implementation of a Prototype

To provide an illustration of the ATM user interface, we will implement an Applet that uses stubs to represent the Account, UserList, and CustomerInfo objects.

The prototype shows a working model of the user interface with the message sequence that the customer will encounter. The purpose of the prototype is to examine the human factors issues relating to the interface and to make certain that there are no deadlocks or partial cycles and that each transaction will properly terminate and return the system to the welcome state.
Test Plan for the User Interface Prototype

1. Specify the paths through the system to be tested
   Welcome screen (ask for Account Number) (DIGITS + ENTER) →
   Passcode State (Ask for Passcode) (DIGITS + ENTER) →
   Select State (Show Options) (DEPOSIT or WITDRAW + ENTER) →
   Deposit State or Withdraw State (Key-in amount and press ENTER) →
   Other State (Ask if user wants to make another transaction) (CLEAR = NO) →
   Goodbye screen (if answer is NO) (CLEAR) →
   Welcome screen is returned

Alternative paths from Select State
   User aborts transaction by pressing CLEAR → Goodbye (CLEAR) → Welcome
   User selects check balance (BALANCE) → Other State (CLEAR = NO) →
   Goodbye screen (CLEAR) → Welcome
Test Plan for User Interface Prototype

Other Alternate Paths

After a Deposit or Withdrawl make another transaction

Deposit (ENTER) → Other State(ENTER = YES) → Select screen (BALANCE) → Other State (CLEAR = NO) → Goodbye screen (CLEAR) → Welcome

Test that pressing ENTER before keying in the amount of the deposit or withdrawal will generate the appropriate error message recovery sequence.

2. When debugging and testing the prototype, keep a log of corrections and changes made in the code.

You should NOT record simple errors like missing semicolons, missing parentheses, or misspelled identifiers, but errors that lead to changes in the code like adding the boolean hold flag to ensure that the goodbye screen would display after a balance check.

Changes to the code made during debugging and testing can modify or override design decisions and should be documented for later review.
Results of Testing

Errors Detected

• Account numbers and passcodes that are 9 digits or longer can cause integer overflow errors.

• If ENTER is pressed before an amount is indicated when performing a withdrawal, then ENTER must be pressed twice after an amount has been entered before the correct screen is displayed.
# Plan for the Next Iteration

**Start Date:** Sept. 9, 2004  
**Due Date:** Sept. 23, 2004

<table>
<thead>
<tr>
<th>Artifact</th>
<th>Comment</th>
</tr>
</thead>
</table>
| **Use cases:**            | **Withdrawal**  
|                           | **Deposit**  
|                           | **Check Balance**  
|                           | Use cases describe the main sequence scenario and lists alternate sequence scenarios (to be developed in next iteration). Fully dressed use cases are begun for each of these goals. |
| **Actor-Goal List**       | Identify the stakeholders (and external systems) and make a list of the goals that each might have.                                     |
| **Use case diagrams**     | From the Actor-Goal List, identify use cases and the actors that participate in each. Display the system boundary.                        |
| **State Model Diagram**   | Add error states for incorrect passcode – do not accept passcodes with more than 6 digits.                                            |
| **Prototype**             | Construct a table for verifying user accounts/passcode. Construct a database for user accounts and incorporate these databases into the ATM simulation. |
Homework

Complete the artifacts listed in the iteration Plan. In particular:

1. Write use cases for Withdraw, Deposit, and Check Balance (as described in the comments.)

2. Construct an Actor-Goal List and Use Case Diagram.

3. Construct a Table for verifying account number/passcode pair. Assume that Account numbers are pre-assigned and let the passcodes be established at that time. (You may add the ability for a user to change his/her passcode as an option in this iteration.

4. Construct a database for user accounts. (A structure of your own creation holding account information and providing methods for performing transactions will suffice here.)

5. Incorporate the databases into the prototype system. Add checking of account number/passcode for valid user (and don’t allow passcode numbers greater than 6 digits).

6. Write a test plan for testing the prototype and perform test.