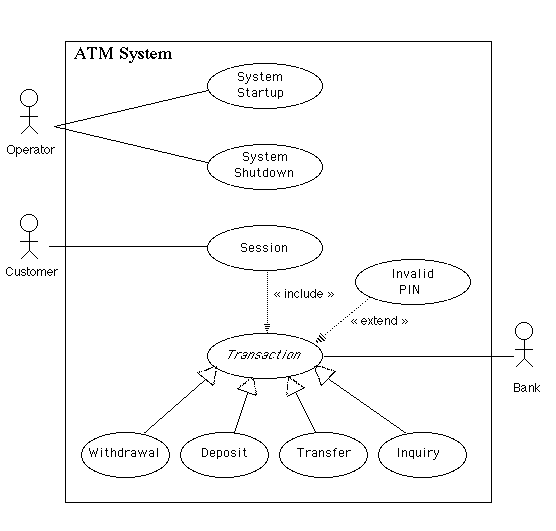
**Use Cases for Example ATM System**



**(Click on a use case above to go to the flow of events for that use case)**

**Flows of Events for Individual Use Cases**

**System Startup Use Case**

The system is started up when the operator turns the operator switch to the "on" position. The operator will be asked to enter the amount of money currently in the cash dispenser, and a connection to the bank will be established. Then the servicing of customers can begin.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Startup)

**System Shutdown Use Case**

The system is shut down when the operator makes sure that no customer is using the machine, and then turns the operator switch to the "off" position. The connection to the bank will be shut down. Then the operator is free to remove deposited envelopes, replenish cash and paper, etc.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Shutdown)

**Session Use Case**

A session is started when a customer inserts an ATM card into the card reader slot of the machine. The ATM pulls the card into the machine and reads it. (If the reader cannot read the card due to improper insertion or a damaged stripe, the card is ejected, an error screen is displayed, and the session is aborted.) The customer is asked to enter his/her PIN, and is then allowed to perform one or more transactions, choosing from a menu of possible types of transaction in each case. After each transaction, the customer is asked whether he/she would like to perform another. When the customer is through performing transactions, the card is ejected from the machine and the session ends. If a transaction is aborted due to too many invalid PIN entries, the session is also aborted, with the card being retained in the machine.

The customer may abort the session by pressing the Cancel key when entering a PIN or choosing a transaction type.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Session)

**Transaction Use Case**

*Note: Transaction is an abstract generalization. Each specific concrete type of transaction implements certain operations in the appropriate way. The flow of events given here describes the behavior common to all types of transaction. The flows of events for the individual types of transaction (withdrawal, deposit, transfer, inquiry) give the features that are specific to that type of transaction.*

A transaction use case is started within a session when the customer chooses a transaction type from a menu of options. The customer will be asked to furnish appropriate details (e.g. account(s) involved, amount). The transaction will then be sent to the bank, along with information from the customer's card and the PIN the customer entered.

If the bank approves the transaction, any steps needed to complete the transaction (e.g. dispensing cash or accepting an envelope) will be performed, and then a receipt will be printed. Then the customer will be asked whether he/she wishes to do another transaction.

If the bank reports that the customer's PIN is invalid, the Invalid PIN extension will be performed and then an attempt will be made to continue the transaction. If the customer's card is retained due to too many invalid PINs, the transaction will be aborted, and the customer will not be offered the option of doing another.

If a transaction is cancelled by the customer, or fails for any reason other than repeated entries of an invalid PIN, a screen will be displayed informing the customer of the reason for the failure of the transaction, and then the customer will be offered the opportunity to do another.

The customer may cancel a transaction by pressing the Cancel key as described for each individual type of transaction below.

All messages to the bank and responses back are recorded in the ATM's log.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Transaction)

**Withdrawal Transaction Use Case**

A withdrawal transaction asks the customer to choose a type of account to withdraw from (e.g. checking) from a menu of possible accounts, and to choose a dollar amount from a menu of possible amounts. The system verifies that it has sufficient money on hand to satisfy the request before sending the transaction to the bank. (If not, the customer is informed and asked to enter a different amount.) If the transaction is approved by the bank, the appropriate amount of cash is dispensed by the machine before it issues a receipt. (The dispensing of cash is also recorded in the ATM's log.)

A withdrawal transaction can be cancelled by the customer pressing the Cancel key any time prior to choosing the dollar amount.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Withdrawal)

**Deposit Transaction Use Case**

A deposit transaction asks the customer to choose a type of account to deposit to (e.g. checking) from a menu of possible accounts, and to type in a dollar amount on the keyboard. The transaction is initially sent to the bank to verify that the ATM can accept a deposit from this customer to this account. If the transaction is approved, the machine accepts an envelope from the customer containing cash and/or checks before it issues a receipt. Once the envelope has been received, a second message is sent to the bank, to confirm that the bank can credit the customer's account - contingent on manual verification of the deposit envelope contents by an operator later. (The receipt of an envelope is also recorded in the ATM's log.)

A deposit transaction can be cancelled by the customer pressing the Cancel key any time prior to inserting the envelope containing the deposit. The transaction is automatically cancelled if the customer fails to insert the envelope containing the deposit within a reasonable period of time after being asked to do so.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Deposit)

**Transfer Transaction Use Case**

A transfer transaction asks the customer to choose a type of account to transfer from (e.g. checking) from a menu of possible accounts, to choose a different account to transfer to, and to type in a dollar amount on the keyboard. No further action is required once the transaction is approved by the bank before printing the receipt.

A transfer transaction can be cancelled by the customer pressing the Cancel key any time prior to entering a dollar amount.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Transfer)

**Inquiry Transaction Use Case**

An inquiry transaction asks the customer to choose a type of account to inquire about from a menu of possible accounts. No further action is required once the transaction is approved by the bank before printing the receipt.

An inquiry transaction can be cancelled by the customer pressing the Cancel key any time prior to choosing the account to inquire about.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "Inquiry)

**Invalid PIN Extension**

An invalid PIN extension is started from within a transaction when the bank reports that the customer's transaction is disapproved due to an invalid PIN. The customer is required to re-enter the PIN and the original request is sent to the bank again. If the bank now approves the transaction, or disapproves it for some other reason, the original use case is continued; otherwise the process of re-entering the PIN is repeated. Once the PIN is successfully re-entered, it is used for both the current transaction and all subsequent transactions in the session. If the customer fails three times to enter the correct PIN, the card is permanently retained, a screen is displayed informing the customer of this and suggesting he/she contact the bank, and the entire customer session is aborted.

If the customer presses Cancel instead of re-entering a PIN, the original transaction is cancelled.

[[ Interaction Diagram ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html" \l "InvalidPIN)

[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Requirements.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/InitialFunctionalTests.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 1996, 1997, 1998, 2000, 2001, 2002, 2004 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**Initial Functional Test Cases for Example ATM System**

The following initial test cases can be identified early in the design process as a vehicle for checking that the implementation is basically correct. *No attempt has been made at this point to do thorough testing, including all possible errors and boundary cases.* That needs to come later. These cases represent an initial check that the functionality specified by the use cases is present.

Some writers would argue for developing test cases like these *in place of* use cases. Here, they are presented as a vehicle for "fleshing out" the use cases, not as a substitute for them.

| **Use Case** | **Function Being Tested** | **Initial System State** | **Input** | **Expected Output** |
| --- | --- | --- | --- | --- |
| System Startup | System is started when the switch is turned "on" | System is off | Activate the "on" switch | System requests initial cash amount |
| System Startup | System accepts initial cash amount | System is requesting cash amount | Enter a legitimate amount | System is on |
| System Startup | Connection to the bank is established | System has just been turned on | Perform a legitimate inquiry transaction | System output should demonstrate that a connection has been established to the Bank |
| System Shutdown | System is shut down when the switch is turned "off" | System is on and not servicing a customer | Activate the "off" switch | System is off |
| System Shutdown | Connection to the Bank is terminated when the system is shut down | System has just been shut down |  | Verify from the bank side that a connection to the ATM no longer exists |
| Session | System reads a customer's ATM card | System is on and not servicing a customer | Insert a readable card | Card is accepted; System asks for entry of PIN |
| Session | System rejects an unreadable card | System is on and not servicing a customer | Insert an unreadable card | Card is ejected; System displays an error screen; System is ready to start a new session |
| Session | System accepts customer's PIN | System is asking for entry of PIN | Enter a PIN | System displays a menu of transaction types |
| Session | System allows customer to perform a transaction | System is displaying menu of transaction types | Perform a transaction | System asks whether customer wants another transaction |
| Session | System allows multiple transactions in one session | System is asking whether customer wants another transaction | Answer yes | System displays a menu of transaction types |
| Session | Session ends when customer chooses not to do another transaction | System is asking whether customer wants another transaction | Answer no | System ejects card and is ready to start a new session |
| Transaction | Individual types of transaction will be tested below |  |  |  |
| Transaction | System handles an invalid PIN properly | A readable card has been entered | Enter an incorrect PIN and then attempt a transaction | The Invalid PIN Extension is performed |
| Withdrawal | System asks customer to choose an account to withdraw from | Menu of transaction types is being displayed | Choose Withdrawal transaction | System displays a menu of account types |
| Withdrawal | System asks customer to choose a dollar amount to withdraw | Menu of account types is being displayed | Choose checking account | System displays a menu of possible withdrawal amounts |
| Withdrawal | System performs a legitimate withdrawal transaction properly | System is displaying the menu of withdrawal amounts | Choose an amount that the system currently has and which is not greater than the account balance | System dispenses this amount of cash; System prints a correct receipt showing amount and correct updated balance; System records transaction correctly in the log (showing both message to the bank and approval back) |
| Withdrawal | System verifies that it has sufficient cash on hand to fulfill the request | System has been started up with less than the maximum withdrawal amount in cash on hand; System is requesting a withdrawal amount | Choose an amount greater than what the system currently has | System displays an appropriate message and asks customer to choose a different amount |
| Withdrawal | System verifies that customer's balance is sufficient to fulfill the request | System is requesting a withdrawal ammount | Choose an amount that the system currently has but which is greater than the account balance | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Withdrawal | A withdrawal transaction can be cancelled by the customer any time prior to choosing the dollar amount | System is displaying menu of account types | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Withdrawal | A withdrawal transaction can be cancelled by the customer any time prior to choosing the dollar amount | System is displaying menu of dollar amounts | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Deposit | System asks customer to choose an account to deposit to | Menu of transaction types is being displayed | Choose Deposit transaction | System displays a menu of account types |
| Deposit | System asks customer to enter a dollar amount to deposit | Menu of account types is being displayed | Choose checking account | System displays a request for the customer to type a dollar amount |
| Deposit | System asks customer to insert an envelope | System is displaying a request for the customer to type a dollar amount | Enter a legitimate dollar amount | System requests that customer insert an envelope |
| Deposit | System performs a legitimate deposit transaction properly | System is requesting that customer insert an envelope | Insert an envelope | System accepts envelope; System prints a correct receipt showing amount and correct updated balance; System records transaction correctly in the log (showing message to the bank, approval back, and acceptance of the envelope) |
| Deposit | A deposit transaction can be cancelled by the customer any time prior to inserting an envelope | System is displaying menu of account types | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Deposit | A deposit transaction can be cancelled by the customer any time prior to inserting an envelope | System is requesting customer to enter a dollar amount | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Deposit | A deposit transaction can be cancelled by the customer any time prior to inserting an envelope | System is requesting customer to insert an envelope | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Deposit | A deposit transaction is cancelled automatically if an envelope is not inserted within a reasonable time | System is requesting customer to insert an envelope | Wait for the request to time out | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Transfer | System asks customer to choose an account to transfer from | Menu of transaction types is being displayed | Choose Transfer transaction | System displays a menu of account types specifying transfer from |
| Transfer | System asks customer to choose an account to transfer to | Menu of account types to transfer from is being displayed | Choose checking account | System displays a menu of account types specifying transfer to |
| Transfer | System asks customer to enter a dollar amount to transfer | Menu of account types to transfer to is being displayed | Choose savings account | System displays a request for the customer to type a dollar amount |
| Transfer | System performs a legitimate transfer transaction properly | System is displaying a request for the customer to type a dollar amount | Enter a legitimate dollar amount | System prints a correct receipt showing amount and correct updated balance; System records transaction correctly in the log (showing both message to the bank and approval back) |
| Transfer | A transfer transaction can be cancelled by the customer any time prior to entering dollar amount | System is displaying menu of account types specifying transfer from | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Transfer | A transfer transaction can be cancelled by the customer any time prior to entering dollar amount | System is displaying menu of account types specifying transfer to | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Transfer | A transfer transaction can be cancelled by the customer any time prior to entering dollar amount | System is requesting customer to enter a dollar amount | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Inquiry | System asks customer to choose an account to inquire about | Menu of transaction types is being displayed | Choose Inquiry transaction | System displays a menu of account types |
| Inquiry | System performs a legitimate inquiry transaction properly | System is displaying menu of account types | Choose checking account | System prints a correct receipt showing correct balance; System records transaction correctly in the log (showing both message to the bank and approval back) |
| Inquiry | An inquiry transaction can be cancelled by the customer any time prior to choosing an account | System is displaying menu of account types | Press "Cancel" key | System displays an appropriate message and offers customer the option of choosing to do another transaction or not. |
| Invalid PIN Extension | Customer is asked to reenter PIN |  | Enter an incorrect PIN; Attempt an inquiry transaction on the customer's checking account | Customer is asked to re-enter PIN |
| Invalid PIN Extension | Correct re-entry of PIN is accepted | Request to re-enter PIN is being displayed | Enter correct PIN | Original transaction completes successfully |
| Invalid PIN Extension | A correctly re-entered PIN is used for subsequent transactions | An incorrect PIN has been re-entered and transaction completed normally | Perform another transaction | This transaction completes successfully as well |
| Invalid PIN Extension | Incorrect re-entry of PIN is not accepted | Request to re-enter PIN is being displayed | Enter incorrect PIN | An appropriate message is displayed and re-entry of the PIN is again requested |
| Invalid PIN Extension | Correct re-entry of PIN on the second try is accepted | Request to re-enter PIN is being displayed | Enter incorrect PIN the first time, then correct PIN the second time | Original transaction completes successfully |
| Invalid PIN Extension | Correct re-entry of PIN on the third try is accepted | Request to re-enter PIN is being displayed | Enter incorrect PIN the first time and second times, then correct PIN the third time | Original transaction completes successfully |
| Invalid PIN Extension | Three incorrect re-entries of PIN result in retaining card and aborting transaction | Request to re-enter PIN is being displayed | Enter incorrect PIN three times | An appropriate message is displayed; Card is retained by machine; Session is terminated |

[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/UseCases.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/AnalysisClasses.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2004 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**Analysis Classes**

An initial reading of the use cases suggests that the following will be part of the system.

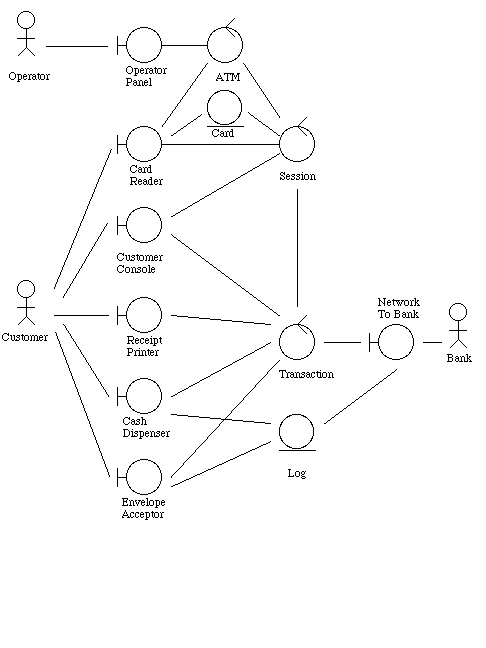
* A controller object representing the ATM itself (managing the boundary objects listed below.)
* Boundary objects representing the individual component parts of the ATM:   
  + Operator panel.
  + Card reader.
  + Customer console, consisting of a display and keyboard.
  + Network connection to the bank.
  + Cash dispenser.
  + Envelope acceptor.
  + Receipt printer.

* Controller objects corresponding to use cases. (Note: class ATM can handle the Startup and Shutdown use cases itself, so these do not give rise to separate objects here.)   
  + Session
  + Transaction (abstract generalization, responsible for common features, with concrete specializations responsible for type-specific portions)

* An entity object representing the information encoded on the ATM card inserted by customer.
* An entity object representing the log of transactions maintained by the machine.

This leads to the following diagram of analysis classes:

**Click on a class icon to go to links to various kinds of information about it**



[Link to 
previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/InitialFunctionalTests.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2000, 2001, 2002 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**CRC Cards for ATM Example**

Using CRC cards to assign responsibiities to various classes for the tasks required by the various use cases leads to the creation of the following cards.

The following links can be used to go directly to the CRC cards for the various classes:

* [Class ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ATM)
* Boundary/entity objects - component parts of the ATM  
  + [Class CardReader](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CardReader)
  + [Class CashDispenser](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CashDispenser)
  + [Class CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)
  + [Class EnvelopeAcceptor](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#EnvelopeAcceptor)
  + [Class Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Log)
  + [Class NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#NetworkToBank)
  + [Class OperatorPanel](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#OperatorPanel)
  + [Class ReceiptPrinter](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ReceiptPrinter)

* Controller objects corresponding to the various use cases  
  + [Class Session](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Session)
  + [Class Transaction](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transaction)
  + [Class Withdrawal](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Withdrawal)
  + [Class Deposit](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Deposit)
  + [Class Transfer](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transfer)
  + [Class Inquiry](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Inquiry)

* Entity objects found necessary when assigning responsiblities to other objects  
  + [Class Balances](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Balances)
  + [Class Card](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Card)
  + [Class Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)
  + [Class Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt)
  + [Class Status](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Status)

**Class ATM**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Start up when switch is turned on | [OperatorPanel](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#OperatorPanel)  [CashDispenser](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CashDispenser)  [NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#NetworkToBank) |
| Shut down when switch is turned off | [NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#NetworkToBank) |
| Start a new session when card is inserted by customer | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Session](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Session) |
| Provide access to component parts for sessions and transactions |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "ATM)

**Class CardReader**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Tell ATM when card is inserted | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ATM) |
| Read information from card | [Card](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Card) |
| Eject card |  |
| Retain card |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "CardReader)

**Class CashDispenser**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Keep track of cash on hand, starting with initial amount |  |
| Report whether enough cash is available |  |
| Dispense cash | [Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Log) |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "CashDispenser)

**Class CustomerConsole**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Display a message |  |
| Display a prompt, accept a PIN from keyboard |  |
| Display a prompt and menu, accept a choice from keyboard |  |
| Display a prompt, accept a dollar amount from keyboard |  |
| Respond to cancel key being pressed by customer |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "CustomerConsole)

**Class EnvelopeAcceptor**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Accept envelope from customer; report if timed out or cancelled | [Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Log) |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "EnvelopeAcceptor)

**Class Log**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Log messages sent to bank |  |
| Log responses from bank |  |
| Log dispensing of cash |  |
| Log receiving an envelope |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Log)

**Class NetworkToBank**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Initiate connection to bank at startup |  |
| Send message to bank and wait for response | [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)  [Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Log)  [Balances](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Balances)  [Status](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Status) |
| Terminate connection to bank at shutdown |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "NetworkToBank)

**Class OperatorPanel**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Inform ATM of changes to state of switch | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ATM) |
| Allow operator to specify amount of initial cash |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "OperatorPanel)

**Class ReceiptPrinter**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Print receipt | [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt) |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "ReceiptPrinter)

**Class Session**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Perform session use case | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ATM) [CardReader](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CardReader)  [Card](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Card) [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Transaction](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transaction) |
| Update PIN value if customer has to re-enter it |  |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Session)

**Abstract Class Transaction**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Allow customer to choose a type of transaction | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ATM) [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Withdrawal](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Withdrawal)  [Deposit](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Deposit)  [Transfer](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transfer)  [Inquiry](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Inquiry) |
| Perform Transaction Use Case | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ATM) [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Withdrawal](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Withdrawal)  [Deposit](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Deposit)  [Transfer](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transfer)  [Inquiry](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Inquiry)  [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)  [NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#NetworkToBank)  [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt)  [ReceiptPrinter](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ReceiptPrinter) |
| Perform invalid PIN extension | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Session](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Session)  [CardReader](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CardReader) |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Transaction)

**Class Withdrawal**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Perform operations peculiar to withdrawal transaction use case | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [CashDispenser](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CashDispenser)  [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)  [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt) |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Withdrawal)

**Class Deposit**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Perform operations peculiar to deposit transaction use case | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)  [EnvelopeAcceptor](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#EnvelopeAcceptor)  [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt) |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Deposit)

**Class Transfer**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Perform operations peculiar to transfer transaction use case | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)  [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt) |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Transfer)

**Class Inquiry**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Perform operations peculiar to inquiry transaction use case | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#CustomerConsole)  [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Message)  [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Receipt) |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Inquiry)

**Class Balances**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Represent account balance information returned by bank |  |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Balances)

**Class Card**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Represent information encoded on customer's ATM card |  |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Card)

**Class Message**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Represent information to be sent over network to bank |  |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Message)

**Class Receipt**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Represent information to be printed on a receipt |  |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Receipt)

**Class Status**

|  |  |
| --- | --- |
| **Responsibilities** | **Collaborators** |
| Represent transaction status information returned by bank |  |
|  |  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Status)

[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/AnalysisClasses.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassDiagram.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2000, 2001, 2002 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**Class Diagram for Example ATM System**

Shown below is the class diagram for the ATM system. The basic structure of the class diagram arises from the responsibilities and relationships discovered when doing the CRC cards and Interaction Diagrams. (If a class uses another class as a collaborator, or sends a message to an object of that class during an Interaction, then there must either be an association linking objects of those classes, or linking the "sending" class to an object which provides access to an object of the "receiving" class.)

In the case of the ATM system, one of the responsibilities of the ATM is to provide access to its component parts for Session and Transaction objects; thus, Session and Transaction have associations to ATM, which in turn has associations to the classes representing the individual component parts. (Explicit "uses" links between Session and Transaction, on the one hand, and the component parts of the ATM, on the other hand, have been omitted from the diagram to avoid making it excessively cluttered.)

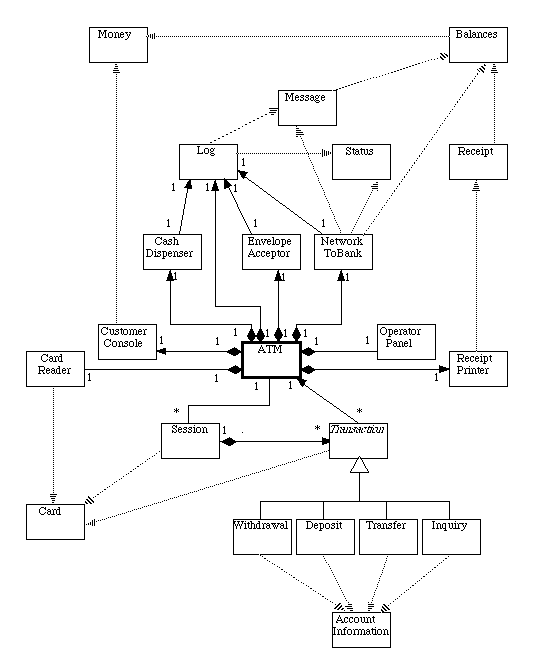
The need for the various classes in the diagram was discovered at various points in the design process.

* Some classes were discovered when doing analysis (see the [Analysis Class Diagram](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/AnalysisClasses.html) developed earlier.)
* Some classes were discovered when doing CRC cards  
  + Message - used to represent a message to the bank.
  + Receipt - used to encapsulate information to be printed on a receipt.
  + Status - used to represent return value from message to the bank.
  + Balances - used to record balance information returned by the bank.
* Some classes were discovered when doing detailed design or writing code  
  + Money - used to represent money amounts, in numerous places.
  + AccountInformation - contains names of various types of accounts customer can choose from

That is, OO design is not a "waterfall" process - discoveries made when doing detailed design and coding can impact overall system design.

To prevent the diagram from becoming overly large, only the name of each class is shown - the attribute and behavior "compartments" are shown in the detailed design, but are omitted here.

**Click on a class icon for links to further information about it**



[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Statecharts.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

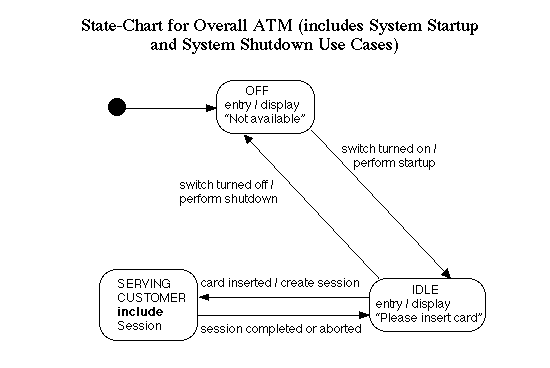
[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

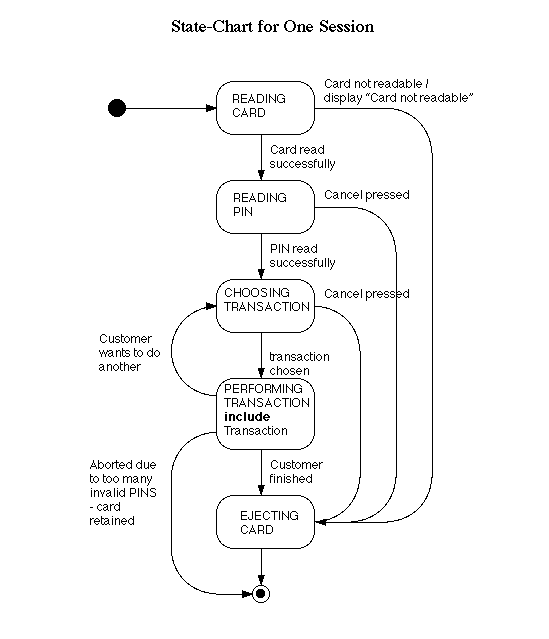
**Copyright © 2000, 2001, 2002 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

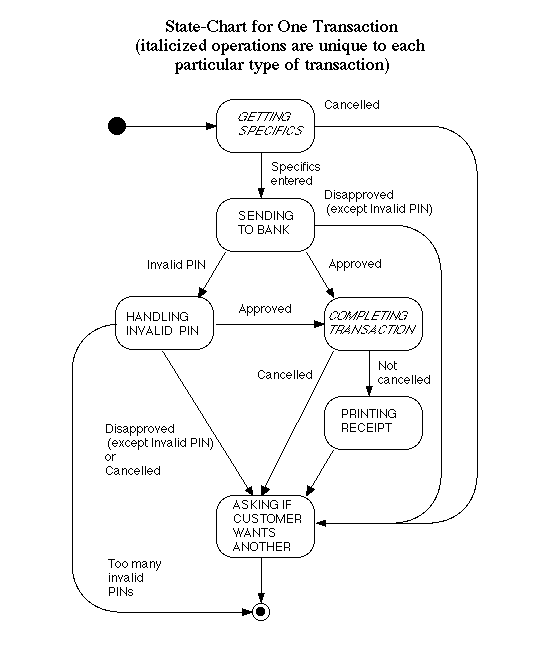
**State Charts for Example ATM System**

Three of the objects we have identified have behavior that is sufficiently complex to warrant developing a State Chart for them. (These are the objects that were identified as the major controller objects.)

* [The object representing the machine itself](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Statecharts.html#ATM) (responsible for the System Startup and Shutdown use cases)
* [Objects representing a customer session (one per session)](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Statecharts.html#Session) (responsible for the Session use case)
* [Objects representing an individual transaction (one per transaction)](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Statecharts.html#Transaction) (responsible for the Transaction use case, use cases for the specific types of transaction, and Invalid PIN extension).

  
[[ CRC Card for ATM ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#ATM) [[ Interaction diagram for Startup ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Startup) [[ Interaction diagram for Shutdown ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Shutdown)

  
[[ CRC Card for Session ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Session) [[ Interaction diagram for Session ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Session)

  
[[ CRC Card for Transaction ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transaction) [[ Interaction diagram for Transaction ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Transaction)  
[[ CRC Card for Withdrawal ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Withdrawal) [[ Interaction diagram for portion unique to Withdrawal ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Withdrawal)  
[[ CRC Card for Deposit ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Deposit) [[ Interaction diagram for portion unique to Deposit ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Deposit)  
[[ CRC Card for Transfer ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Transfer) [[ Interaction diagram for portion unique to Transfer ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Transfer)  
[[ CRC Card for Inquiry ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/CRCCards.html#Inquiry) [[ Interaction diagram for portion unique to Inquiry ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Inquiry)  
[[ Interaction diagram for Invalid PIN Extension ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#InvalidPIN)

[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassDiagram.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

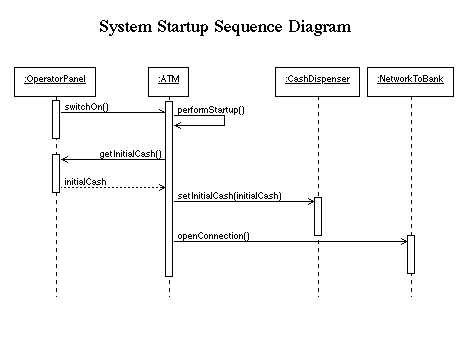
[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

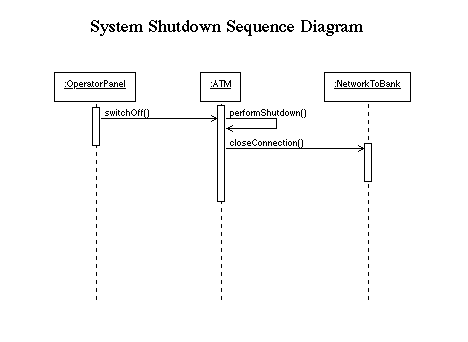
**Copyright © 2000, 2001, 2002 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

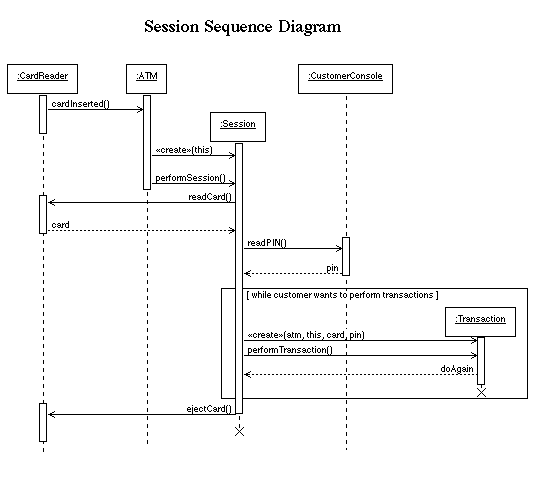
**Interaction Diagrams for Example ATM System**

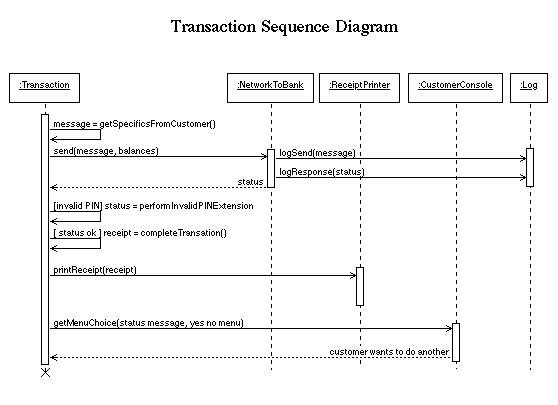
UML defines two types of Interaction Diagram: the Sequence Diagram and the Collaboration Diagram. In order to illustrate both types, the major use cases are documented using Sequence Diagrams, and the specific subcases of transaction (withdrawal, etc.) and the Invalid PIN Extension are documented using Collaboration Diagrams. (The major reason for using two different types of diagram is pedagogical - to illustrate each type.)

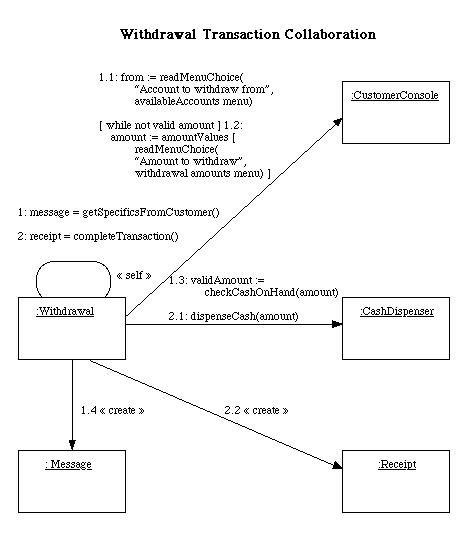
* [Sequence Diagram for System Startup Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Startup)
* [Sequence Diagram for System Shutdown Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Shutdown)
* [Sequence Diagram for Session Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Session)
* [Sequence Diagram for Transaction Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Transaction) (Since transaction is abstract, this gives the overall flow of a transaction. See the interactions below for specific concrete cases.)
* [Collaboration realizing specifics of Withdrawal Transaction Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Withdrawal)
* [Collaboration realizing specifics of Deposit Transaction Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Deposit)
* [Collaboration realizing specifics of Transfer Transaction Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Transfer)
* [Collaboration realizing specifics of Inquiry Transaction Use Case](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#Inquiry)
* [Collaboration realizing Invalid PIN Extension](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html#InvalidPIN)

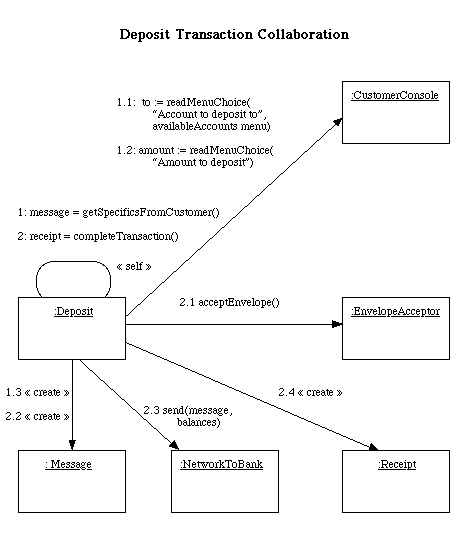


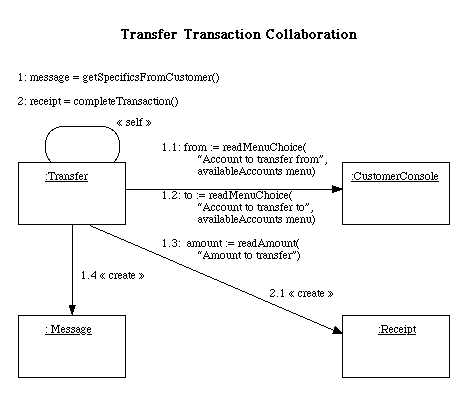


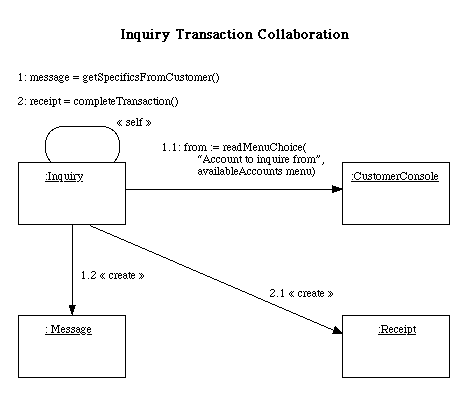


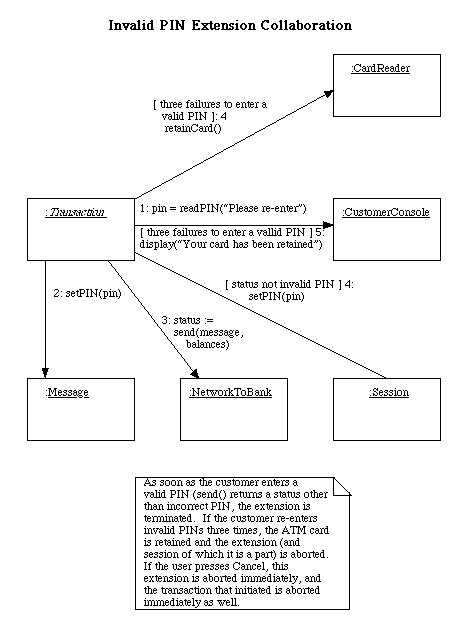












[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Statecharts.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2000, 2002 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**Detailed Design**

A major task of detailed design is to spell out, in detail, the attributes and methods needed by each class (the second and third "compartments" of the representation for each class in a class diagram.)

The methods needed by each class are implicit in the responsibilities assigned to the class in the CRC cards, and become explicit in the Interaction Diagrams. A responsibility listed for a class on its CRC card generally maps into a method or methods in the detailed design. Likewise, any time an object belonging to a given class is shown as the recipient of a message in either a Sequence or Collaboration Diagram, the class needs a corresponding method. Many of the needed attributes are also either explicitly or implicitly present in the diagrams; the need for others becomes evident as the code for the class is being written. (Thus detailed design and coding are a "round trip" process - detailed design dictates coding, and coding leads to elaboration of the detailed design.)

In designing this system, a few key design decisions were made:

* The class ATM is made an active class - that is, the ATM object has its own thread. Using the Java thread facllity leads to defining a run() method in this class whose body is executed by the ATM's thread. The fact that class ATM is active is indicated in class diagrams by enclosing it in a heavier outline.
* Certain signals initiate computation - e.g. the signals from the operator console when the state of the switch is changed, or from the card reader when a card is inserted. In the GUI simulation of the ATM, these signals are sent by the "actionPerformed()" method of the appropriate GUI button; in a real ATM they would be sent by the physical components themselves, which might then also need to be active classes. (Note: this forms an exception to the rule that a responsibility on a CRC card translates into a method in the design - in this case the class sends a signal, rather than receiving it, so it does not need a method directly corresponding to the responsibility.)
* The Transaction hierarchy consists of the abstract class Transaction and four concrete subclasses (Withdrawal, Deposit, Transfer and Inquiry). The class Transaction has a "virtual constructor" called makeTransaction() which asks the customer to choose a transaction type and then constructs and returns an object of the appropriate subclass. The Transaction class is made responsible for carrying out the Transaction use case and the Invalid PIN extension; for the former, it makes use of abstract methods getSpecificsFromCustomer() and completeTransaction() which are implemented concretely by each subclass.
* The class Receipt is abstract. The completeTransaction() method of each kind of transaction creates a concrete instance that contains the information relevant to that kind of transaction.
* The class Status is abstract. The send() method of NetworkToBank constructs a concrete instance that contains the information appropriate to the response received from the bank to a particular message.

In the design below, each class is developed in isolation. No attempt is made to connect each class to related classes as in the class diagram, because the resulting picture would not fit on a displayable web page. Therefore, this detailed design should be viewed in conjunction with the [Class Diagram](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassDiagram.html) developed earlier.

* [Detailed design for class ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#ATM)
* Component parts of the ATM  
  + [Detailed design for class CardReader](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#CardReader)
  + [Detailed design for class CashDispenser](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#CashDispenser)
  + [Detailed design for class CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#CustomerConsole)
  + [Detailed design for class EnvelopeAcceptor](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#EnvelopeAcceptor)
  + [Detailed design for class Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Log)
  + [Detailed design for class NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#NetworkToBank)
  + [Detailed design for class OperatorPanel](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#OperatorPanel)
  + [Detailed design for class ReceiptPrinter](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#ReceiptPrinter)
* [Detailed design for class Session](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Session)
* The Transaction class hierarchy  
  + [Detailed design for class Transaction](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Transaction)
  + [Detailed design for class Withdrawal](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Withdrawal)
  + [Detailed design for class Deposit](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Deposit)
  + [Detailed design for class Transfer](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Transfer)
  + [Detailed design for class Inquiry](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Inquiry)
* Classes representing banking concepts, used by the above  
  + [Detailed design for class AccountInformation](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#AccountInformation)
  + [Detailed design for class Balances](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Balances)
  + [Detailed design for class Card](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Card)
  + [Detailed design for class Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Message)
  + [Detailed design for class Money](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Money)
  + [Detailed design for class Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Receipt)
  + [Detailed design for class Status](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html#Status)

|  |
| --- |
| **ATM** |
|  |
| - id: int |
| - place: String |
| - bankName: String |
| - bankAddress: InetAddress |
| - cardReader: CardReader |
| - cashDispenser: CashDispenser |
| - customerConsole: CustomerConsole |
| - envelopeAcceptor: EnvelopeAcceptor |
| - log: Log |
| - networkToBank: NetworkToBank |
| - operatorPanel: OperatorPanel |
| - receiptPrinter: ReceiptPrinter |
| - state: int |
| - switchOn: boolean |
| - cardInserted: boolean |
| - OFF\_STATE: final int |
| - IDLE\_STATE: final int |
| - SERVING\_CUSTOMER\_STATE: final int |
|  |
| + ATM(id: int, place: String, bankName: String, bankAddress: InetAddress) |
| + run() |
| + switchOn() |
| + switchOff |
| + cardInserted() |
| + getID(): int |
| + getPlace(): String |
| + getBankName(): String |
| + getCardReader(): CardReader |
| + getCashDispenser(): CashDispenser |
| + getCustomerConsole(): CustomerConsole |
| + getEnvelopeAcceptor(): EnvelopeAcceptor |
| + getLog(): Log |
| + getNetworkToBank(): NetworkToBank |
| + getOperatorPanel(): OperatorPanel |
| + getReceiptPrinter(): ReceiptPrinter |
| - performStartup() |
| - performShutdown() |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#ATM)

|  |
| --- |
| **CardReader** |
|  |
| - atm: ATM |
|  |
| + CardReader(atm: ATM) |
| + readCard(): Card |
| + ejectCard() |
| + retainCard() |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#CardReader)

|  |
| --- |
| **CashDispenser** |
|  |
| - log: Log |
| - cashOnHand: Money |
|  |
| + CashDispenser(log: Log) |
| + setInitialCash(initialCash: Money) |
| + checkCashOnHand(amount: Money): boolean |
| + dispenseCash(amount: Money) |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#CashDispenser)

|  |
| --- |
| **CustomerConsole** |
|  |
|  |
| + CustomerConsole() |
| + display(message: String) |
| + readPIN(prompt: String): int throws Cancelled |
| + readMenuChoice(prompt: String, menu: String []): int throws Cancelled |
| + readAmount(prompt: String): Money throws Cancelled |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#CustomerConsole)

|  |
| --- |
| **EnvelopeAcceptor** |
|  |
| - log: Log |
|  |
| + EnvelopeAcceptor(log: Log) |
| + acceptEnvelope() throws Cancelled |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#EnvelopeAcceptor)

|  |
| --- |
| **Log** |
|  |
|  |
| + Log() |
| + logSend(message: Message) |
| + logResponse(status: Status) |
| + logCashDispensed(amount: Money) |
| + logEnvelopeAccepted() |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Log)

|  |
| --- |
| **NetworkToBank** |
|  |
| - log: Log |
| - bankAddress: InetAddress |
|  |
| + NetworkToBank(log: Log, bankAddress: InetAddress) |
| + openConnection() |
| + closeConnection() |
| + sendMessage(message: Message, out balances: Balances): Status |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#NetworkToBank)

|  |
| --- |
| **OperatorPanel** |
|  |
| - atm: ATM |
|  |
| + OperatorPanel(atm: ATM) |
| + getInitialCash(): Money |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#OperatorPanel)

|  |
| --- |
| **ReceiptPrinter** |
|  |
|  |
| + ReceiptPrinter() |
| + printReceipt(receipt: Receipt) |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#ReceiptPrinter)

|  |
| --- |
| **Session** |
|  |
| - atm: ATM |
| - pin: int |
| - state: int |
| - READING\_CARD\_STATE: final int |
| - READING\_PIN\_STATE: final int |
| - CHOOSING\_TRANSACTION\_STATE: final int |
| - PERFORMING\_TRANSACTION\_STATE: final int |
| - EJECTING\_CARD\_STATE: final int |
| - FINAL\_STATE: final int |
|  |
| + Session(atm: ATM)> |
| + performSession() |
| + setPIN(int pin) |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Session)

|  |
| --- |
| ***Transaction*** |
|  |
| # atm: ATM |
| # session: Session |
| # card: Card |
| # pin: int |
| # serialNumber: int |
| # message: Message |
| # balances: Balances |
| - TRANSACTION\_TYPES\_MENU: final String [] |
| - nextSerialNumber: int |
| - state: int |
| - GETTING\_SPECIFICS\_STATE: final int |
| - SENDING\_TO\_BANK\_STATE: final int |
| - INVALID\_PIN\_STATE: final int |
| - COMPLETING\_TRANSACTION\_STATE: final int |
| - PRINTING\_RECEIPT\_STATE: final int |
| - ASKING\_DO\_ANOTHER\_STATE: final int |
|  |
| # Transaction(atm: ATM, session: Session, card: Card, pin: int) |
| + makeTransaction(atm: ATM, session: Session, card: Card, pin: int): Transaction throws Cancelled |
| + performTransaction(): boolean throws CardRetained |
| + performInvalidPinExtension(): Status throws Cancelled, CardRetained |
| + getSerialNumber(): int |
| # *getSpecificsFromCustomer(): Message throws Cancelled* |
| # *completeTransaction(): Receipt throws Cancelled* |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Transaction)

|  |
| --- |
| **Withdrawal** |
|  |
| - from: int |
| - amount: Money |
|  |
| + Withdrawal(atm: ATM, session: Session, card: Card, pin: int) |
| # getSpecificsFromCustomer(): Message throws Cancelled |
| # completeTransaction(): Receipt |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Withdrawal)

|  |
| --- |
| **Deposit** |
|  |
| - to: int |
| - amount: Money |
|  |
| + Deposit(atm: ATM, session: Session, card: Card, pin: int) |
| # getSpecificsFromCustomer(): Message throws Cancelled |
| # completeTransaction(): Receipt throws Cancelled |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Deposit)

|  |
| --- |
| **Transfer** |
|  |
| - from: int |
| - to: int |
| - amount: Money |
|  |
| + Transfer(atm: ATM, session: Session, card: Card, pin: int) |
| # getSpecificsFromCustomer(): Message throws Cancelled |
| # completeTransaction(): Receipt |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Transfer)

|  |
| --- |
| **Inquiry** |
|  |
| - from: int |
|  |
| + Inquiry(atm: ATM, session: Session, card: Card, pin: int) |
| # getSpecificsFromCustomer(): Message throws Cancelled |
| # completeTransaction(): Receipt |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Inquiry)

|  |
| --- |
| **AccountInformation** |
|  |
| + ACCOUNT\_NAMES: final String[] |
| + ACCOUNT\_ABBREVIATIONS: final String [] |
|  |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "AccountInformation)

|  |
| --- |
| **Balances** |
|  |
| - total: Money |
| - available: Money |
|  |
| + Balances() |
| + setBalances(total: Money, available: Money) |
| + getTotal(): Money |
| + getAvailable(): Money |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Balances)

|  |
| --- |
| **Card** |
|  |
| - number: int |
|  |
| + Card(number: int) |
| + getNumber(): int |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Card)

|  |
| --- |
| **Message** |
|  |
| + WITHDRAWAL: final int |
| + INITIATE\_DEPOSIT: final int |
| + COMPLETE\_DEPOSIT: final int |
| + TRANSFER: final int |
| + INQUIRY: final int |
| - messageCode: int |
| - card: Card |
| - pin: int |
| - serialNumber: int |
| - fromAccount: int |
| - toAccount: int |
| - amount: int |
|  |
| + Message(messageCode: int, cardNumber: Card, pin: int, serialNumber: int, fromAccount: int, toAccount: int, amount: Money) |
| + toString(): String |
| + setPIN(pin: int) |
| + getMessageCode(): int |
| + getCard(): Card |
| + getPIN(): int |
| + getSerialNumber(): int |
| + getFromAccount(): int |
| + getToAccount(): int |
| + getAmount(): Money |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Message)

|  |
| --- |
| **Money** |
|  |
| - cents: long |
|  |
| + Money(dollars: int) |
| + Money(dollars: int, cents: int) |
| + Money(toCopy: Money) |
| + toString(): String |
| + add(amountToAdd: Money) |
| + subtract(amountToSubtract: Money) |
| + lessEqual(compareTo: Money): boolean |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html" \l "Money)

|  |
| --- |
| ***Receipt*** |
|  |
| - headingPortion: String [] |
| # detailsPortion(): String [] |
| - balancesPortion: String [] |
|  |
| # Receipt(atm: ATM, card: Card, transaction: Transaction, balances: Balances) |
| + getLines(): Enumeration |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Receipt)

|  |
| --- |
| ***Status*** |
|  |
|  |
| + toString(): String |
| + *isSuccess(): boolean* |
| + *isInvalidPIN(): boolean* |
| + *getMessage(): String* |

[[ Links for this class ]](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/ClassLinks.html#Status)

[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Interactions.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Package.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2000, 2001, 2002 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**Package Diagram for Example ATM System**

The package diagram shows how the various classes are grouped into packages. There are two "top-level" classes - ATMMain and ATMApplet - which allow the system to be run (respectively) as an application or as an Applet. (Only one of the two would be instantiated in any particular use of the system.)

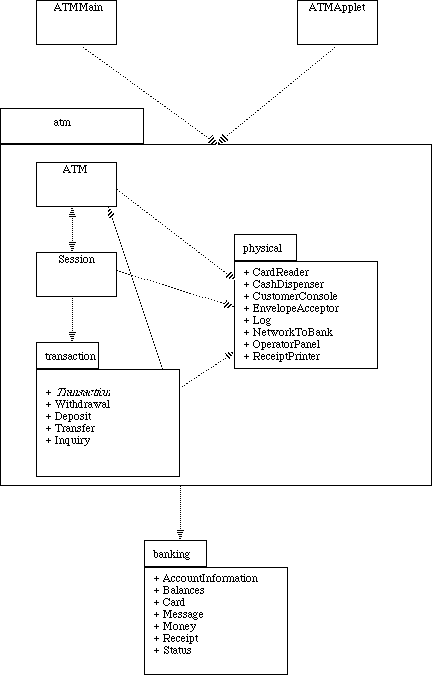
Each of these classes, in turn, depends on the package atm which contains the class ATM that represents the system as a whole, and the class Session that represents one session. ATM depends on Session, and vice versa - since the ATM creates Sessions, and each Session, in turn, uses the ATM to interact with the customer.

The subpackage transaction contains the classes used to represent individual transactions that a customer initiates. The class Session depends on the transaction package because it creates individual transaction objects. These, in turn, again depend on the ATM to interact with the customer.

The subpackage physical contains the classes that represent the various physical components of the ATM. For the purposes of this simulation, these are simulated by a GUI. A real ATM would have quite different classes in this package - classes that actually control its physical components. The class ATM makes use of these components, and Session and the various kinds of transaction gain access to them through ATM to actually perform the needed operations.

Finally, the package banking contains classes that represent the banking enterprise itself and the information communicated back and forth between the ATM and the bank - i.e. classes which might be the same in a totally different implementation of an ATM that interacts with the same bank.

This is, of course, a simulation. However, most of the code that is specific to the simulation resides in the package physical, plus the two top-level classes. Presumably, the other classes and packages might be similar in a real system.



[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/DetailedDesign.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Code.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2001 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**

**Code for Example ATM System**

This page contains links both to the actual Java code that implements the ATM simulation, and to javadoc documentation generated from the code. The code is organized into a collection of packages, as shown in the [Package Diagram](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Package.html).

In addition to the classes and packages that have been discussed throughout the design, the GUI simulation makes use of a package (called simulation) that contains the details of the GUI, including the code that does the animation. This code takes the place of an actual hardware ATM, but has not been otherwise documented in this series of web pages. The design of the simulation package is discussed briefly, and its code may be accessed [here](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/SimulationDetails.html) if you want to see it.

|  |  |  |
| --- | --- | --- |
| **Package** | **Class - source code** | **Class - javadoc** |
|  |  |  |
| (Top-level) | [ATMMain.java](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/ATMMain.java) | (No javadoc for this class) |
| (Top-level) | [ATMApplet.java](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/ATMApplet.java) | (No javadoc for this class) |
|  |  |  |
| atm | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/ATM.java) | [ATM](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/ATM.html) |
|  | [Session](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/Session.java) | [Session](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/Session.html) |
|  |  |  |
| atm.physical | [CardReader](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/CardReader.java) | [CardReader](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/CardReader.html) |
|  | [CashDispenser](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/CashDispenser.java) | [CashDispenser](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/CashDispenser.html) |
|  | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/CustomerConsole.java) | [CustomerConsole](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/CustomerConsole.html) |
|  | [EnvelopeAcceptor](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/EnvelopeAcceptor.java) | [EnvelopeAcceptor](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/EnvelopeAcceptor.html) |
|  | [Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/Log.java) | [Log](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/Log.html) |
|  | [NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/NetworkToBank.java) | [NetworkToBank](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/NetworkToBank.html) |
|  | [OperatorPanel](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/OperatorPanel.java) | [OperatorPanel](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/OperatorPanel.html) |
|  | [ReceiptPrinter](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/physical/ReceiptPrinter.java) | [ReceiptPrinter](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/physical/ReceiptPrinter.html) |
|  |  |  |
| atm.transaction | [Transaction](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/transaction/Transaction.java) | [Transaction](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/transaction/Transaction.html) |
|  | [Withdrawal](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/transaction/Withdrawal.java) | [Withdrawal](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/transaction/Withdrawal.html) |
|  | [Deposit](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/transaction/Deposit.java) | [Deposit](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/transaction/Deposit.html) |
|  | [Transfer](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/transaction/Transfer.java) | [Transfer](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/transaction/Transfer.html) |
|  | [Inquiry](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/atm/transaction/Inquiry.java) | [Inquiry](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/atm/transaction/Inquiry.html) |
|  |  |  |
| banking | [AccountInformation](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/AccountInformation.java) | [AccountInformation](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/AccountInformation.html) |
|  | [Balances](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/Balances.java) | [Balances](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/Balances.html) |
|  | [Card](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/Card.java) | [Card](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/Card.html) |
|  | [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/Message.java) | [Message](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/Message.html) |
|  | [Money](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/Money.java) | [Money](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/Money.html) |
|  | [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/Receipt.java) | [Receipt](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/Receipt.html) |
|  | [Status](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/source/banking/Status.java) | [Status](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/javadoc/banking/Status.html) |

[Link to previous page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Package.html)[Link to next page](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Executable.html)

[Page of links for non frames-enabled browsers.](http://www.math-cs.gordon.edu/courses/cs211/ATMExample/Links.html)

[Valid XHTML 1.0!](http://validator.w3.org/check/referer)

**Copyright © 2001 - Russell C. Bjork. Permission for non-commercial reproduction for educational use is hereby granted; all other rights are reserved.**