Finding classes

- Choosing classes is first step in defining essence of problem
- If you can recognize an abstraction, you’ve found a candidate class
- If you can formulate a statement of purpose for a candidate class, it’s more likely to be included in the design
Finding classes

- Carefully read requirements specification or description of design goals
- Discuss what the system should do:
  • expected inputs
  • desired responses
- Look for noun phrases (nouns, nouns modified by adjectives) in spec
Finding Classes

- Having identified nouns, change plural nouns to singular form & make preliminary list
- 3 categories will emerge from this list:
  - obvious classes
  - obvious nonsense
  - not sure
- Your candidate classes will emerge from the first & last categories
Example specification

An automatic teller machine (ATM) performs various financial transactions (deposits, withdrawals, and balance inquiries) in response to user requests. The user is able to use the machine if the 4-digit PIN code s/he types in to the ATM keypad matches the code embedded in the magnetic strip on his/her card, which is read by the machine. If the user enters an invalid code, an error message is displayed, and the user is given another chance to enter the code correctly; a second incorrect code results in the user’s card being retained by the machine. Once a valid code is entered, the user may access his/her account for transactions. When a balance inquiry is requested, the machine prints the information on a receipt. When a deposit is requested, the machine receives the deposit envelope and the amount specified is added to the user’s account balance. When a withdrawal is requested, the account balance is checked to ensure that sufficient funds are available, and, if so, the machine dispenses cash and the account is debited by the withdrawal amount.
**Nouns & noun phrases**

An automatic teller machine (ATM) performs various financial transactions (deposits, withdrawals, and balance inquiries) in response to user requests. The user is able to use the machine if the 4-digit PIN code s/he types in to the ATM keypad matches the code embedded in the magnetic strip on his/her card, which is read by the machine. If the user enters an invalid code, an error message is displayed, and the user is given another chance to enter the code correctly; a second incorrect code results in the user’s card being retained by the machine. Once a valid code is entered, the user may access his/her account for transactions. When a balance inquiry is requested, the machine prints the information on a receipt. When a deposit is requested, the machine receives the deposit envelope and the amount specified is added to the user’s account balance. When a withdrawal is requested, the account balance is checked to ensure that sufficient funds are available, and, if so, the machine dispenses cash and the account is debited by the withdrawal amount.
<table>
<thead>
<tr>
<th><strong>Preliminary List</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Automatic teller machine</strong></td>
</tr>
<tr>
<td><strong>financial transaction</strong></td>
</tr>
<tr>
<td><strong>withdrawal</strong></td>
</tr>
<tr>
<td><strong>user request</strong></td>
</tr>
<tr>
<td><strong>machine</strong></td>
</tr>
<tr>
<td><strong>ATM keypad</strong></td>
</tr>
<tr>
<td><strong>magnetic strip</strong></td>
</tr>
<tr>
<td><strong>invalid code</strong></td>
</tr>
<tr>
<td><strong>chance</strong></td>
</tr>
<tr>
<td><strong>valid code</strong></td>
</tr>
<tr>
<td><strong>transaction</strong></td>
</tr>
<tr>
<td><strong>receipt</strong></td>
</tr>
</tbody>
</table>
Initial elimination phase

- Distill redundant terms down to the single term that best describes the concept
- Eliminate noun phrases that describe things outside the system
- Eliminate nouns that are standins for verbs
Redundant terms

- Automatic teller machine, ATM, machine: **ATM**
- Financial transaction, transaction: **Transaction**
- 4-digit PIN code, code, valid code, invalid code, incorrect code: **PINcode**
- Account balance, sufficient funds: **Balance**
- Withdrawal amount, amount specified: **Amount**
Eliminate terms from outside system

- User is not part of the ATM, so can be eliminated
- Likewise the deposit envelope, cash, receipt, card, and the card’s magnetic strip are inputs and outputs, not parts of the machine
Eliminate verb standins

- “Chance” falls into this category - this noun comes from the phrase “… another chance to enter the code …” - it’s really a standin for “enter,” a verb
Narrowed list

ATM
transaction
deposit
withdrawal
balance inquiry
ATM keypad
user request

account
amount specified
information
balance
PINcode
error message
Choosing candidate classes

- The guidelines on the next several slides provide suggestions for teasing out the classes in a specification
- This process should be done in a systematic fashion, but is more art than science
- At this stage, it is much better to have too many candidate classes than too few
Choosing candidate classes

- Model physical objects
- Model conceptual entities that form a cohesive abstraction
- Model categories of classes as individual, specific classes - don’t try to set up super/subclass relationships at this stage
- Model values of attributes, not attributes themselves
Model physical objects

- ATM is a physical object, representing the machine as a whole
- Likewise, the ATM keypad is a physical object
Model conceptual entities

- “Error message,” “user request” and “information” are terms that suggest communication between the machine and the user.
- This suggests some sort of Communication class, which can be used to accept requests and convey information.
- Account is also a cohesive abstraction, as is PINcode.
Model categories

- The most apparent category is Transaction, with subtypes deposit, withdrawal, and balance inquiry
- Each subtype, as well as the category itself, can be modelled as a class
Model values, not attributes

- “Amount specified” is an attribute of transactions deposit and withdrawal
- “Balance” is an attribute of account
Second revised class list

ATM account
transaction deposit
withdrawal balance inquiry
PINcode ATM keypad
communication
Recording candidate classes: CRC cards

- CRC stands for Class-Responsibility-Collaboration
- Create one card for each class
- On the front of the card, write the class name
- On the back, write a brief description of the purpose of the class
- Cards serve as object surrogates - easy to manipulate, arrange, and discard as needed
## CRC examples

<table>
<thead>
<tr>
<th>Class</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account</td>
<td>represents bank customer’s account</td>
</tr>
<tr>
<td>Keypad</td>
<td>group of keys for user input</td>
</tr>
<tr>
<td>Transaction</td>
<td>performs requested financial transaction &amp; updates user’s account</td>
</tr>
</tbody>
</table>
Finding abstract classes

- Abstract classes spring from set of classes that share useful attribute (implying shared behavior)
- Identify abstract superclass candidates by grouping related classes - once group is identified, name superclass to represent it
- Should create as many superclasses as possible - reduces duplication of effort
Identifying missing classes

- Once identified, extending categories can help in finding missing classes
- For example, the Communication class, we can identify subclasses Message, which just provides information to the user, and Menu, which provides information and waits for a response
Identifying missing classes

- Classes may be found by looking at descriptions of existing classes - for example, keypad is described as “a group of keys” - but what is a key?

- Classes may be missing because the spec was imprecise - for example, our spec doesn’t mention a display device, but such a device is clearly necessary
Classes -ends-