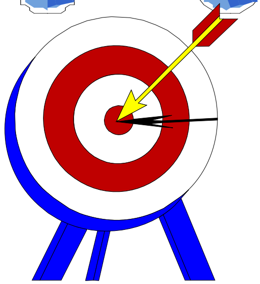
Object-Oriented Systems Development:

Using the Unified Modeling Language

Chapter 7:

Object Analysis: Classification



Goals

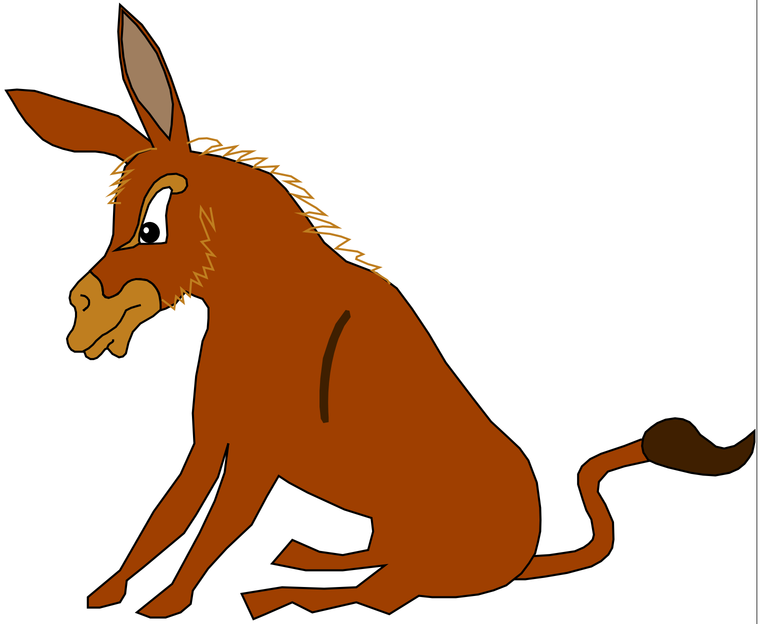
- The concept of classification.
- How to identify classes with the noun phrase approach.
- How to identify classes with the common class patterns approach.

Goals (Con't)

- How to identify classes and object behavior analyzed by sequence/collaboration modeling.
- Class responsibilities collaboration (CRC) approach.

… Intelligent classification is intellectually hard work, and it best comes about through an incremental and iterative process

Booch

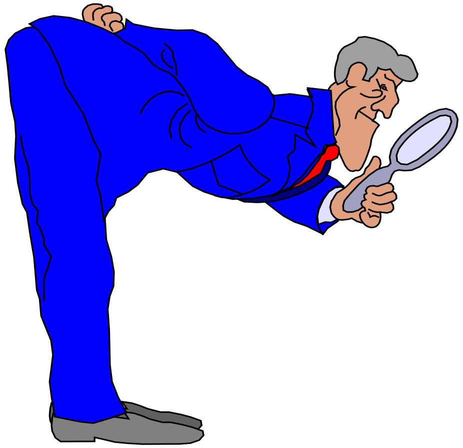


..There is no such thing as the perfect class structure, nor the right set of objects. As in any engineering discipline, our design choice is compromisingly shaped by many competing factors.

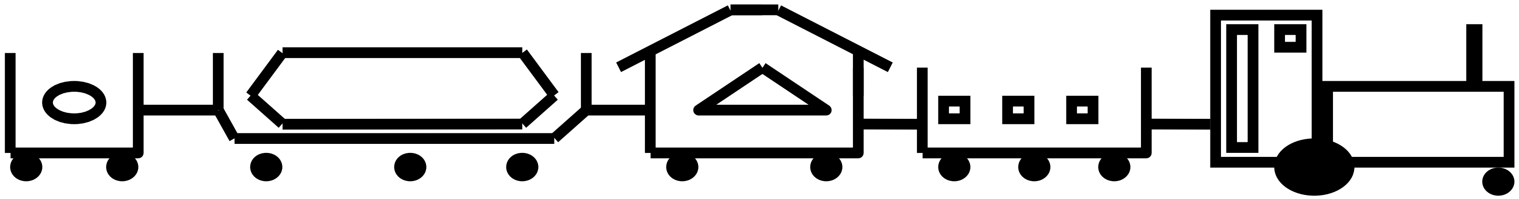
Booch

Classification Theory

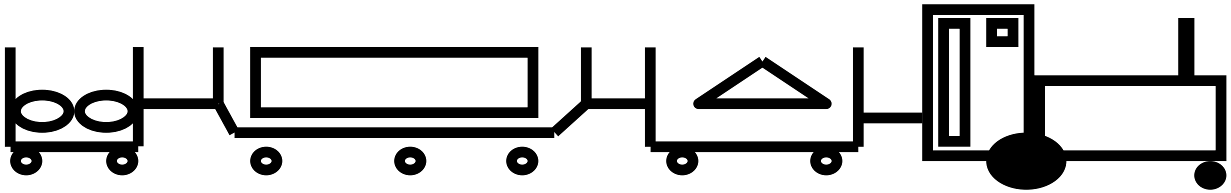
- Classification is the process of checking to see if an object belongs to a category or a class and it is regarded as a basic attribute of human nature.



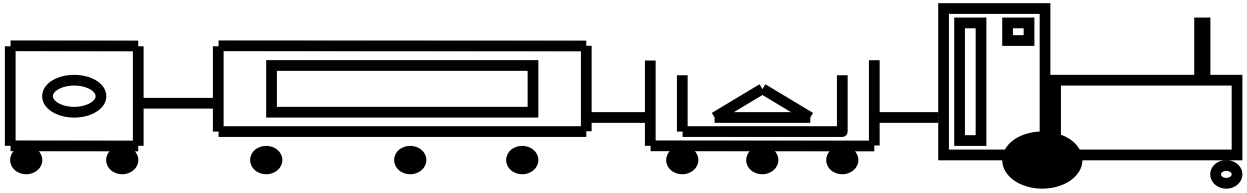
A



B

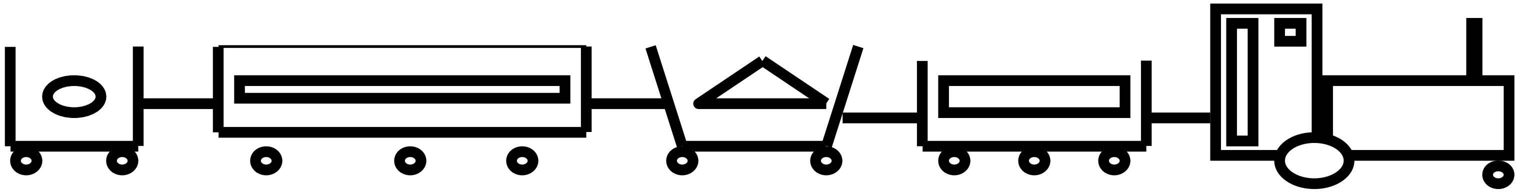


C

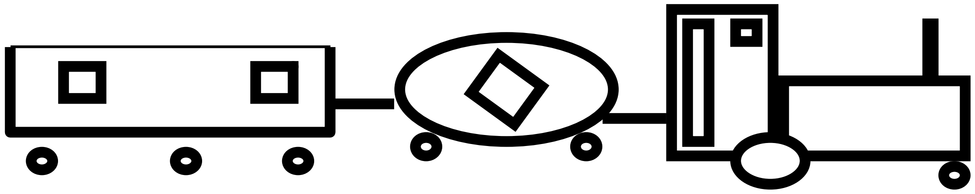


D

E



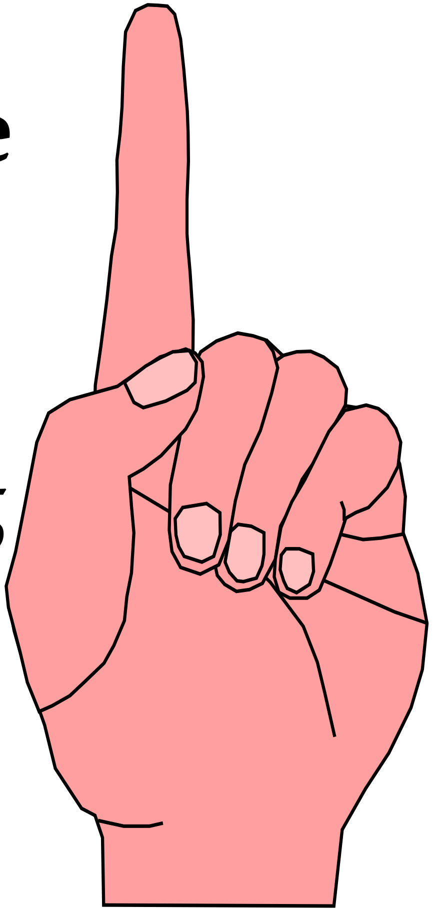
F



Point To Remember

Two Issues

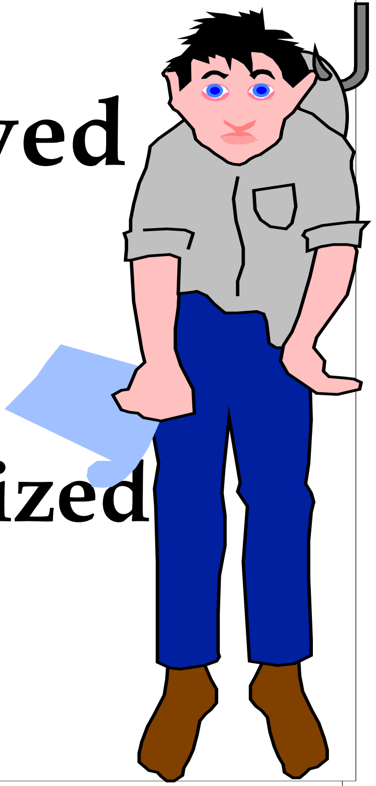
- A class is a specification of structure, behavior, and the description of an object.
- Classification is more concerned with identifying classes than identifying the individual objects in a system.

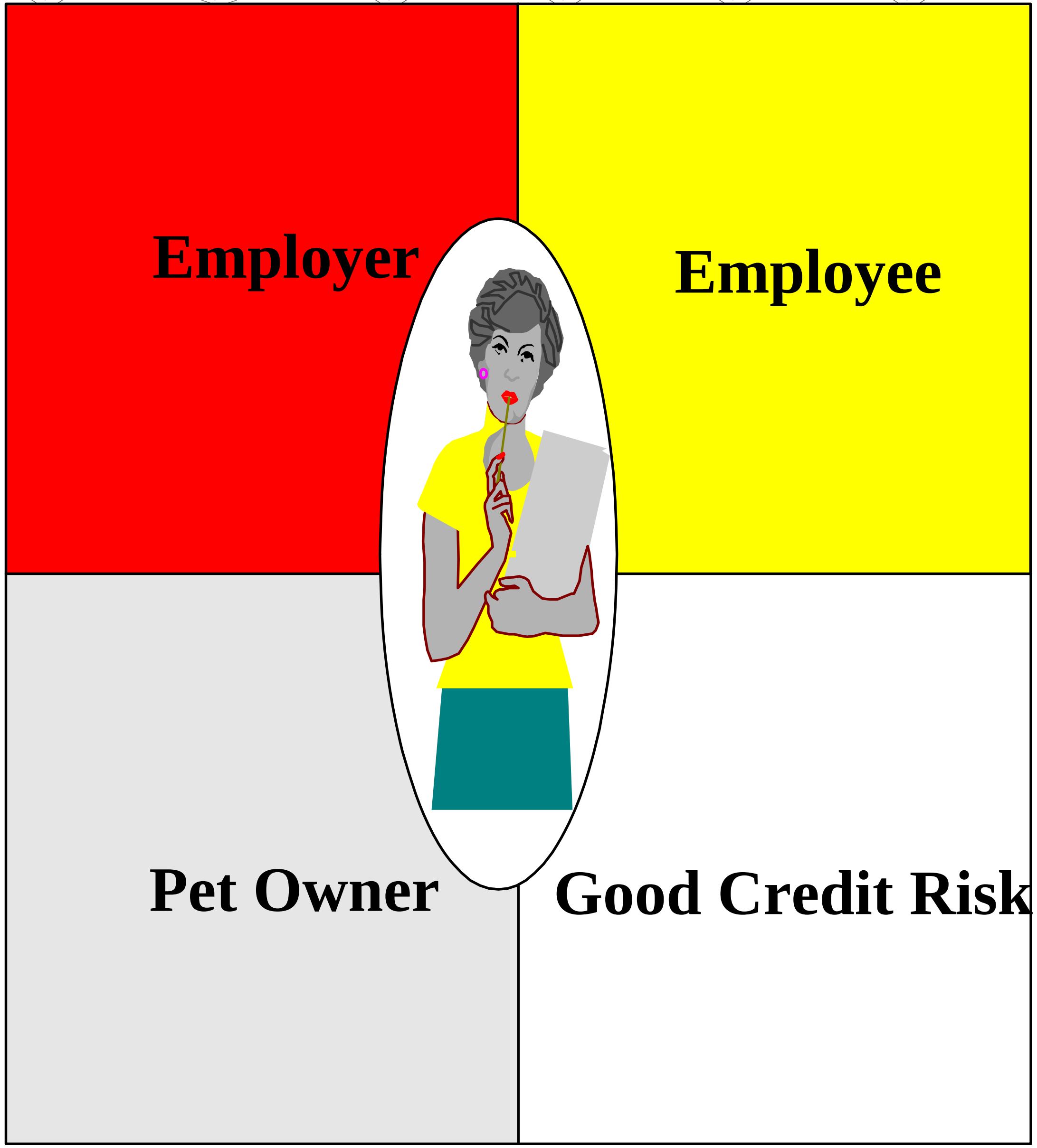


The Challenge of Classification

- Intelligent classification is intellectually hard work and may seem rather arbitrary.
- Martin and Odell have observed in object-oriented analysis and design, that

  “In fact, an object can be categorized in more than one way.”





Employer

Employee

Pet Owner

Good Credit Risk

Approaches for Identifying Classes

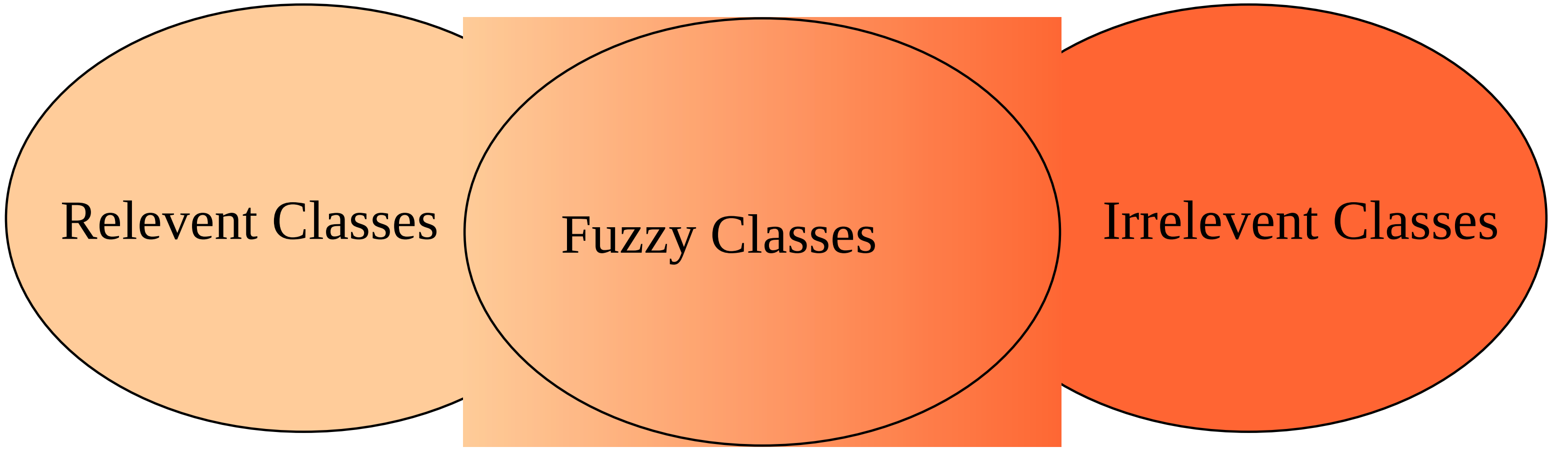
- **The noun phrase approach.**
- **The common class patterns approach.**
- **The use-case driven approach.**
- **The class responsibilities collaboration (CRC) approach.**

Noun Phrase Approach

- **Using this method, you have to read through the Use cases, interviews, and requirements specification carefully, looking for noun phrases.**

Noun Phrase Strategy (Con't)

- **Change all plurals to singular and make a list, which can then be divided into three categories.**



Noun Phrase Strategy (Con't)

- **It is safe to scrap the Irrelevant Classes.**
- **You must be able to formulate a statement of purpose for each candidate class; if not, simply eliminate it.**
- **You must then select candidate classes from the other two categories.**

Guidelines For Identifying Classes

- The followings are guidelines for selecting classes in your application:
- Look for nouns and noun phrases in the problem statement.
- Some classes are implicit or taken from general knowledge.

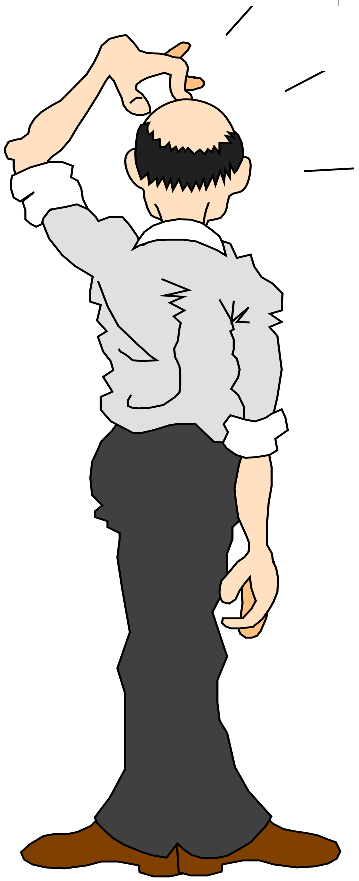
Guidelines For Identifying Classes (Con't)

- All classes must make sense in the application domain.
- Avoid computer implementation classes, defer it to the design stage.
- Carefully choose and define class names.

Guidelines For Refining Classes

Redundant Classes:

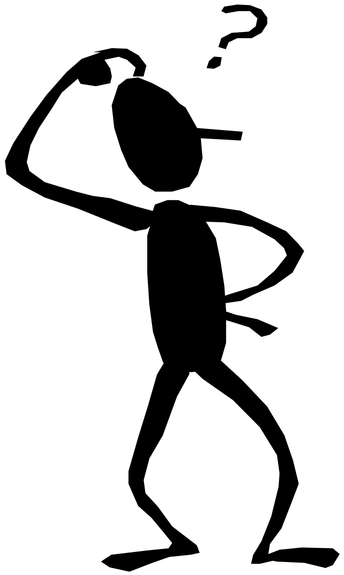
- Do not keep two classes that express the same information.
- If more than one word is being used to describe the same idea, select the one that is the most meaningful in the context of the system.



Guidelines For Refining Classes (Con't)

Adjective Classes:

- **Does the object represented by the noun behave differently when the adjective is applied to it?**



Guidelines For Refining Classes (Con't)

- **If the use of the adjective signals that the behavior of the object is different, then make a new class.**
- **For example, If *Adult Membership* and *Youth Membership* behave differently, than they should be classified as different classes.**

Guidelines For Refining Classes (Con't)

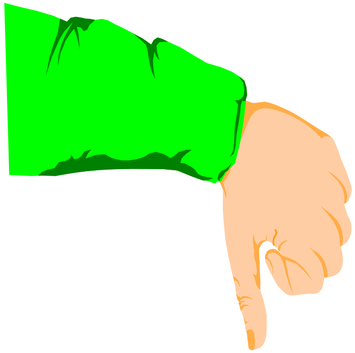
Attribute Classes:

- Tentative objects which are used only as values should be defined or restated as attributes and not as a class.
- For example the demographics of Membership are not classes but attributes of the Membership class.

Guidelines For Refining Classes (Con't)

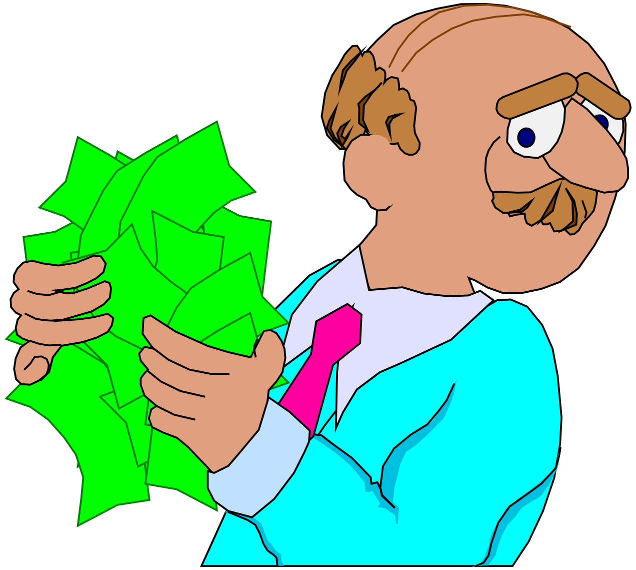
Irrelevant Classes:

- Each class must have a purpose and every class should be clearly defined and necessary.
- If you cannot come up with a statement of purpose, simply eliminate the candidate class.



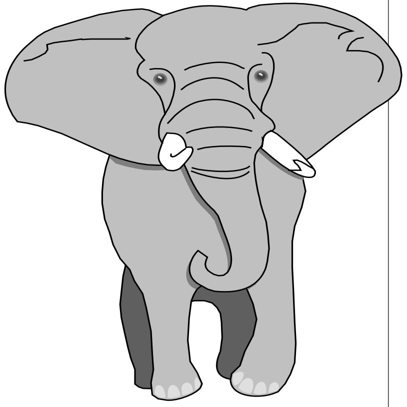
Common Class Patterns Approach

- **This approach is based on the knowledge-base of the common classes that have been proposed by various researchers.**



Candidate Classes - Events

- These are points in time that must be recorded and remembered.
- Things happen, usually to something else, at a given date and time, or as a step in an ordered sequence.
- For example order which is an event that must be remembered.



Candidate Classes - Organization

- The organizational units that people belong to.
- For example, accounting department might be considered as a potential class.

Candidate Classes - People and Person (Roles and Roles Played)

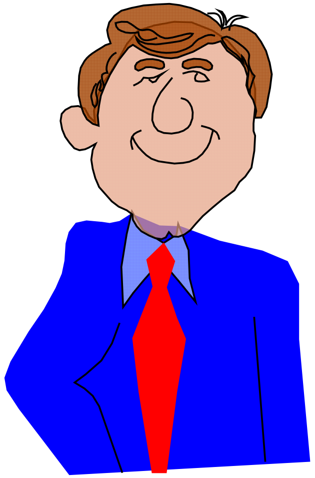
- **The different roles users play in interacting with the application.**

Candidate Classes - People (Con’t)

- **It can be divided into two types (Coad & Yourdon):**
- **Those representing users of the system, such as an operator, or a clerk;**

Candidate Classes - People (Con’t)

- **Those people who do not use the system but about whom information is kept.**
  - **Some examples are Client, Employee, Teacher, Manager.**

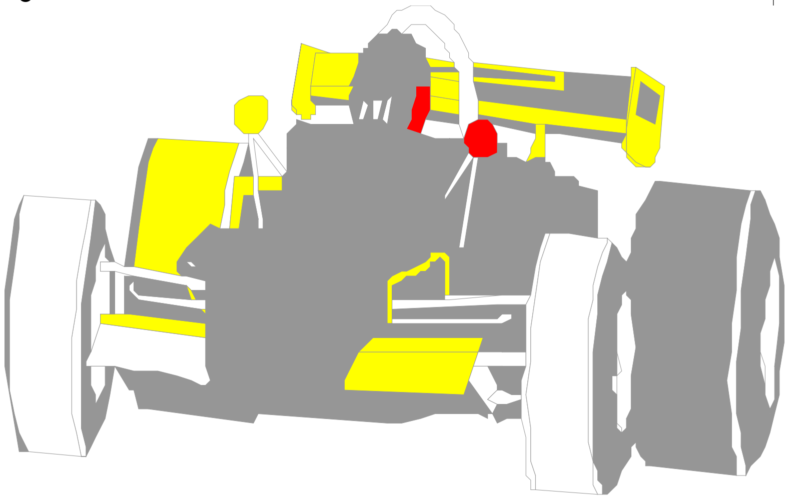


Candidate Classes - Places

- **These are physical locations, such as buildings, stores, sites or offices that the system must keep information about.**

Candidate Classes - Tangible Things and Devices

- Physical objects, or group of objects, that are tangible, and devices with which the application interacts.
- For example, cars, pressure sensors.



Candidate Classes - Concepts

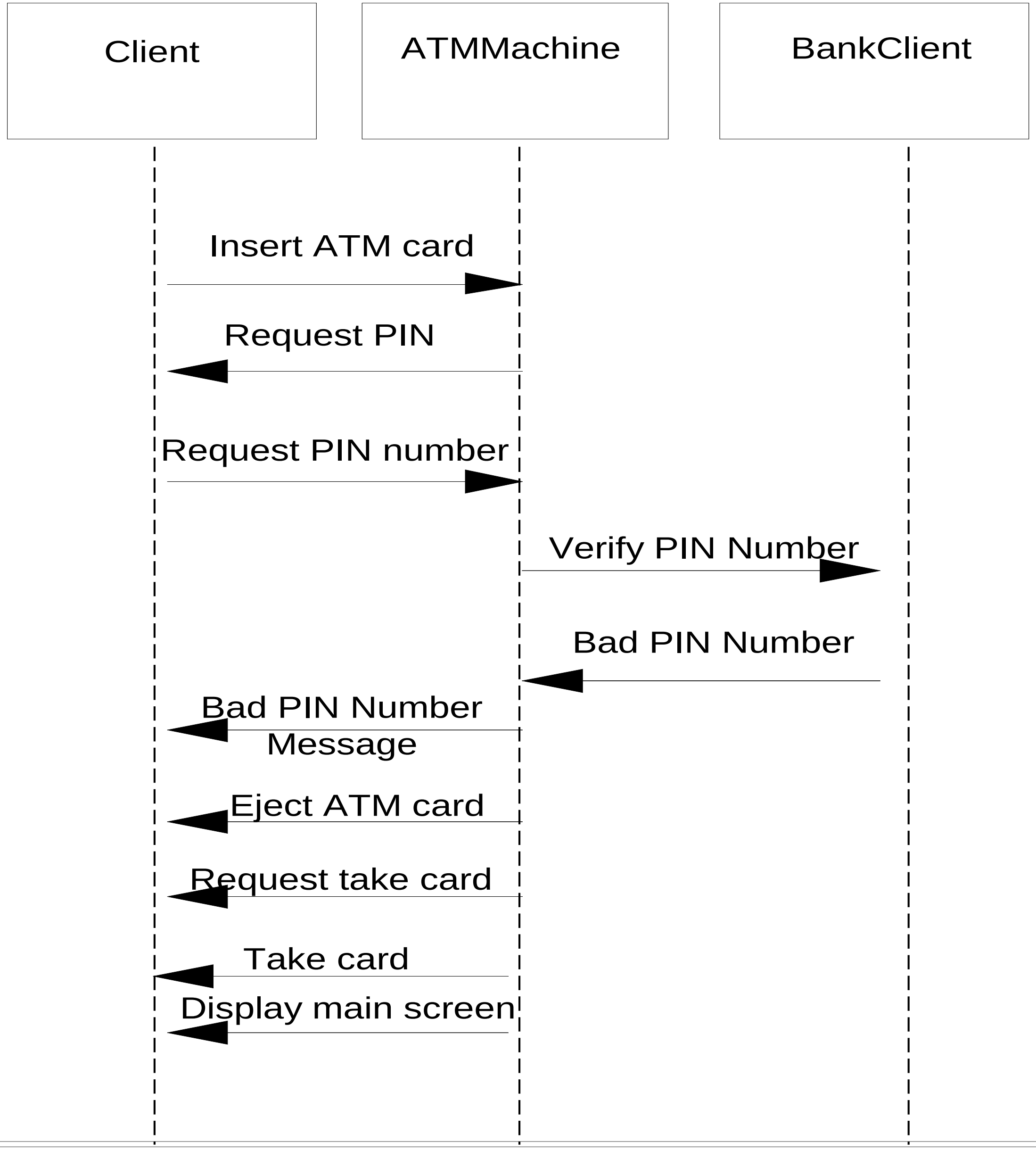
- **Concepts are principles or ideas not tangible but used to organize or keep track of business activities and/or communications.**

Use-case Driven Approach

- **Once the system has been described in terms of its scenarios, we can examine the textual description or steps of each scenario to determine what objects are needed for the scenario to occur.**

Use-case Driven Approach

- To identify objects of a system and their behaviors, the lowest level of executable use cases is further analyzed with a sequence and collaboration diagram pair.
- By walking through the steps, you can determine what objects are necessary for the steps to take place.



Client

ATMMachine

BankClient

Insert ATM card

Request PIN

Request PIN number

Verify PIN Number

Bad PIN Number

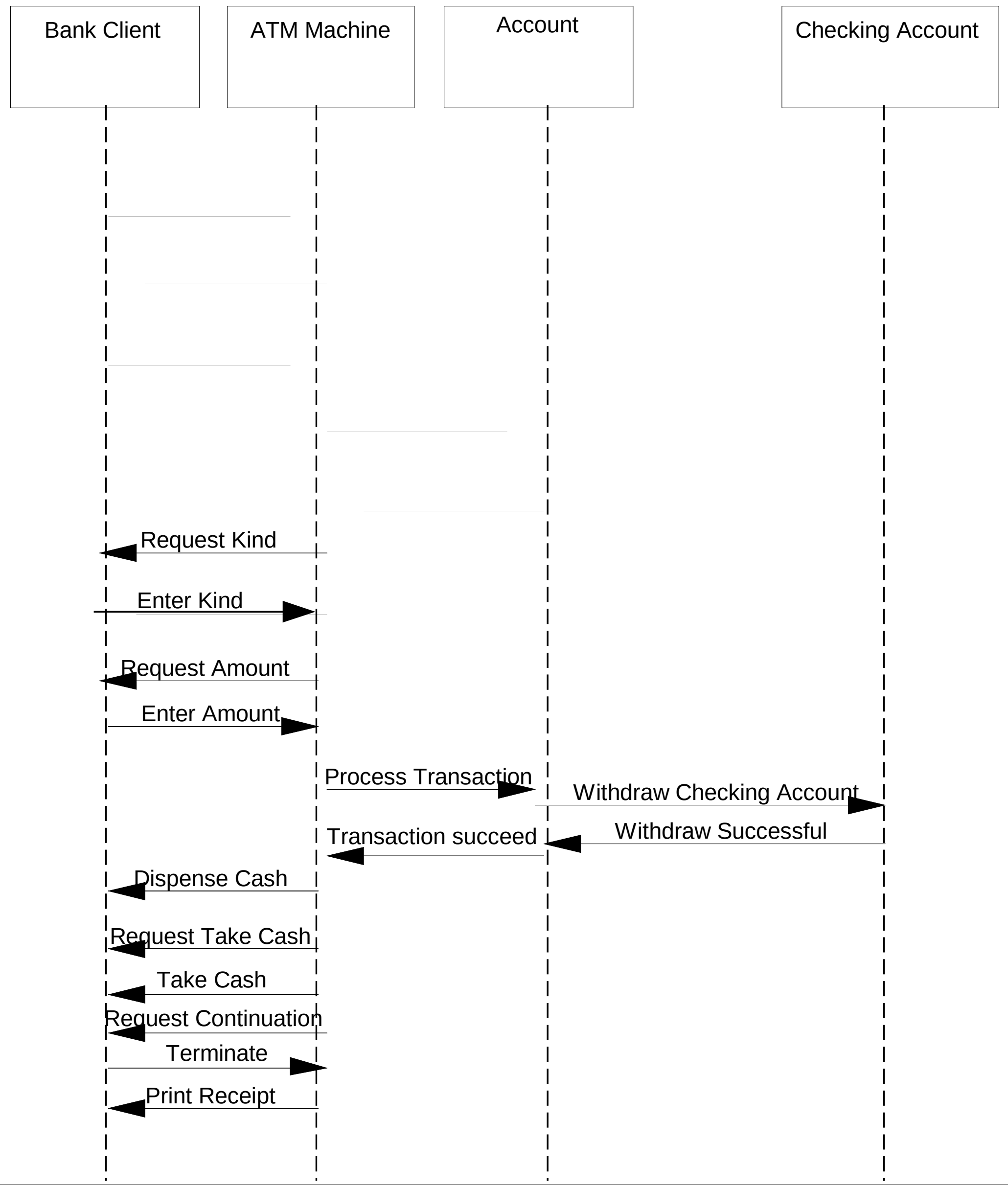
Bad PIN Number Message

Eject ATM card

Request take card

Take card

Display main screen



Bank Client

ATM Machine

Account

Checking Account

Request Kind

Enter Kind

Request Amount

Enter Amount

Process Transaction

Withdraw Checking Account

Transaction succeed

Withdraw Successful

Dispense Cash

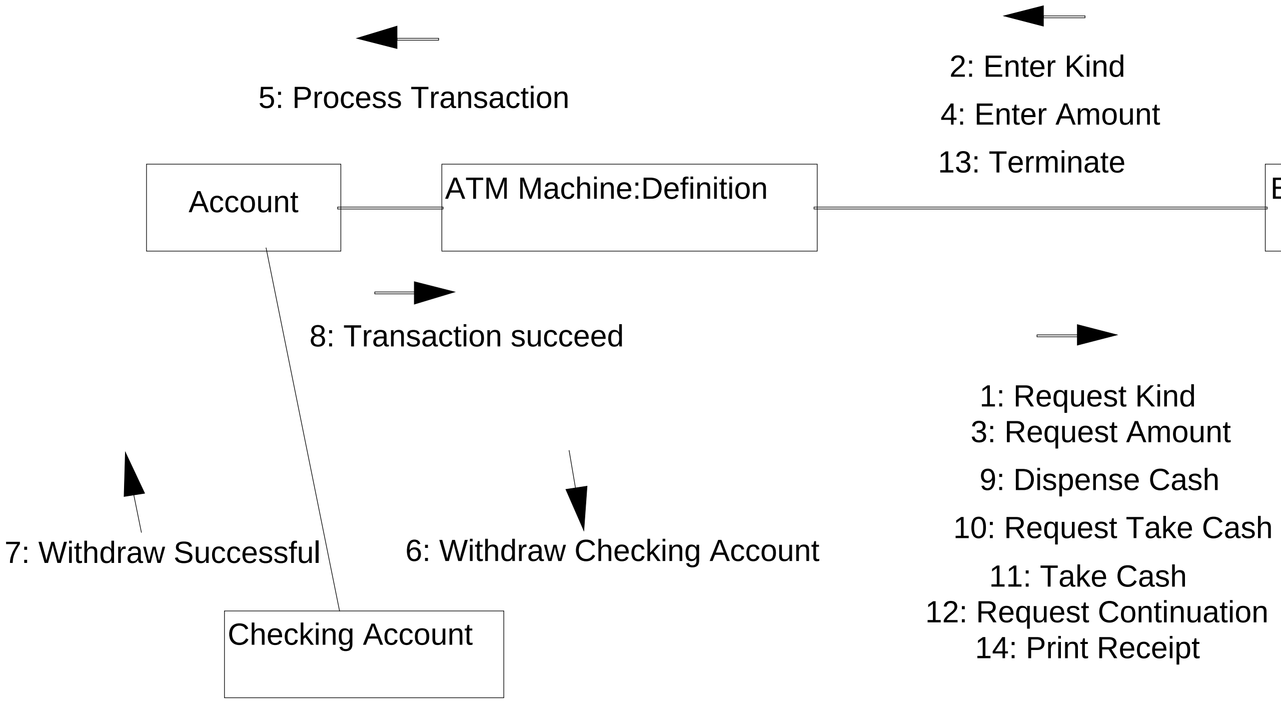
Request Take Cash

Take Cash

Request Continuation

Terminate

Print Receipt



5: Process Transaction

2: Enter Kind

4: Enter Amount

13: Terminate

Account

ATM Machine:Definition

Bank Client

8: Transaction succeed

1: Request Kind

3: Request Amount

9: Dispense Cash

10: Request Take Cash

11: Take Cash

12: Request Continuation

14: Print Receipt

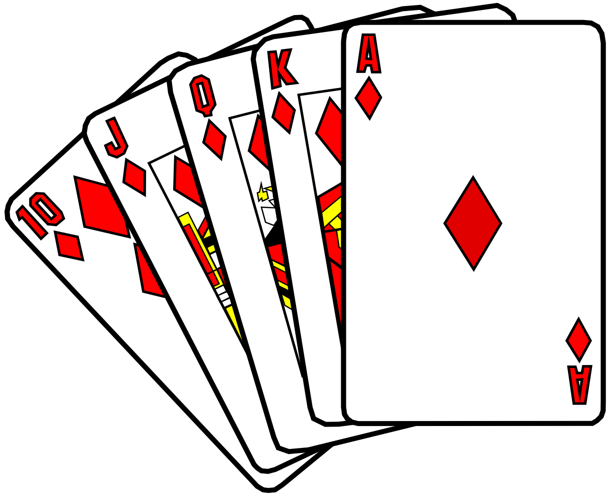
7: Withdraw Successful

6: Withdraw Checking Account

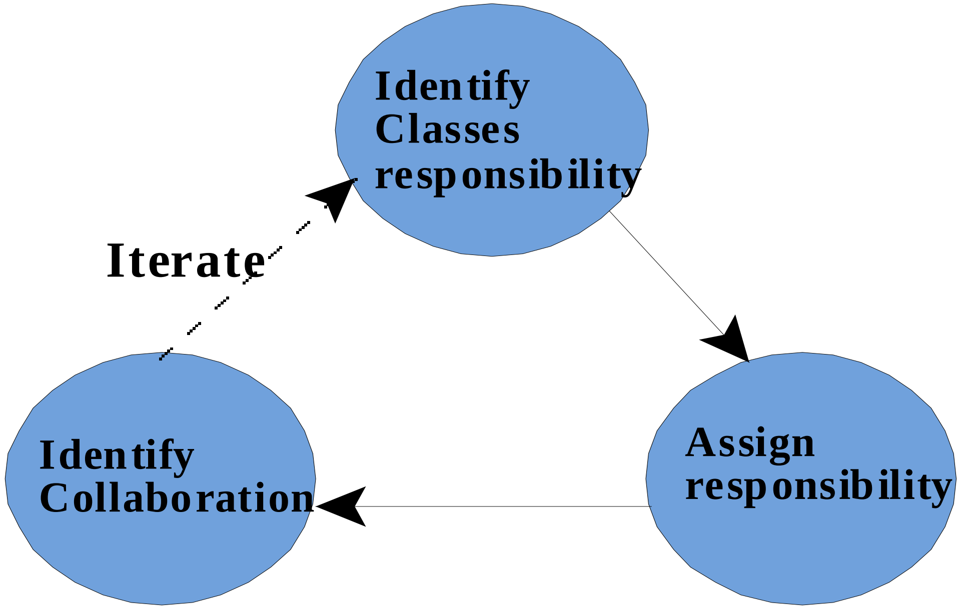
Checking Account

CRC Cards

- CRC stands for Class, Responsibilities and Collaborators developed by Cunningham, Wilkerson and Beck.
- CRC can be used for identifying classes and their responsibilities.



Process of the CRC Technique

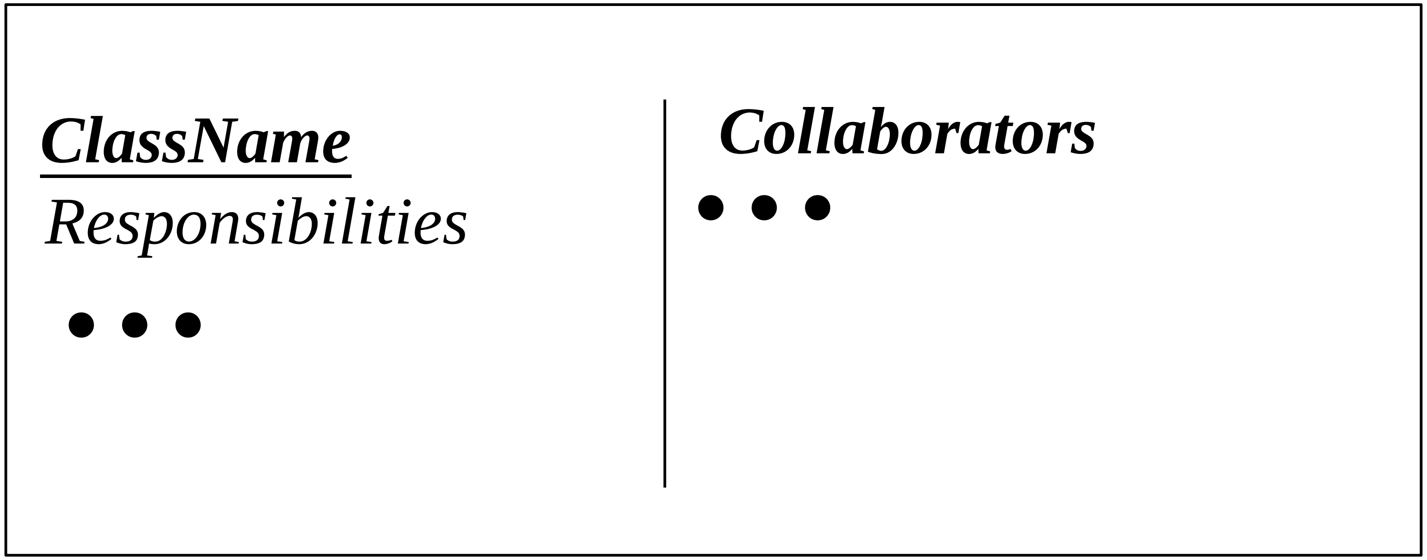


Collaborations

- An object can accomplish either a certain responsibility itself, or it may require the assistance of other objects.
- If it requires an assistance of other objects, it must collaborate with those objects to fulfill its responsibility.

CRC Cards (Con't)

- **CRC cards are 4" x 6" index cards. All the information for an object is written on a card.**



ClassName

Responsibilities

● ● ●

Collaborators

● ● ●

CRC Cards (Con’t)

- **CRC starts with only one or two obvious cards.**
- **If the situation calls for a responsibility not already covered by one of the objects:**
  - **Add, or**
  - **Create a new object to address that responsibility.**

Guidelines for Naming Classes

- **The class should describe a single object, so it should be the singular form of noun.**
- **Use names that the users are comfortable with.**
- **The name of a class should reflect its intrinsic nature.**

Guidelines for Naming Classes (Con't)

- By the convention, the class name must begin with an upper case letter.
- For compound words, capitalize the first letter of each word - for example, LoanWindow.

Summary

- Finding classes is not easy.
- The more practice you have, the better you get at identifying classes.
- There is no such thing as the "right set of classes."
- Finding classes is an incremental and iterative process.

Summary (Con't)

- **Unless you are starting with a lot of domain knowledge, you are probably missing more classes than you will eliminate.**
- **Naming a class is also an important activity.**
- **The class should describe a single object, so it should be a singular noun or an adjective and a noun.**