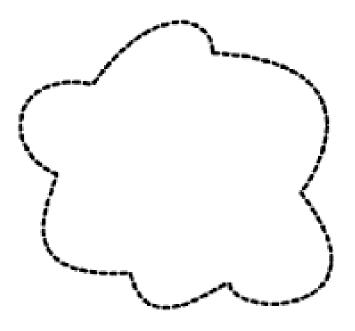
Object Oriented Analysis

Class

- A Class is a software template that defines the methods and variables to be included in a particular kind of Object.
- Is a blue print used to create objects. As it is a blue print, at runtime it will not occupy any memory.
- Examples : Humans, machine



Class icon as a cloud / amorphous blob

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

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

Challenge of Classification

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

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

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, we have to read through the Use cases, interviews, and requirements specification carefully, looking for noun phrases.
- Change all plurals to singular and make a list, which can then be divided into three categories.

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

The followings are the guidelines for electing classes in any application:

- Look for nouns and noun phrases in the problem statement.
- Some classes are implicit or taken from general knowledge.
- 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.

- 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.
- Adjective Classes: Does the object represented by the noun behave differently when the adjective is applied to it?

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

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

- 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.

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.

Class Name Collaborators Responsibilities

Order			
Check items are in stock	Order Line		
Determine the price	Order Line		
Check for valid	Customer		
payment			
Dispatch to delivery			
address			

TApplication	Flip	TList	Flip	
Superclasses: TEventHandler		Superclasses: TObject		
Subclasses:		Subclasses:		
Responsibilities:	Collaborators:	Responsibilities:	Collaborators:	
EventLoop	TWindow	Initialize		
DoMenuCommand	TDocument, TWindo	Insert		
SetupMemis	TWindow	Delete		
DoMouseCommand	TWindow	EachItemDo		
		Free		
TBox		TObject		
Superclasses: TShape		Superclasses:		
Subclasses:		Subclasses: TEventHandler, TList, TShape		
Responsibilities:	Collaborators:	Responsibilities:	Collaborators:	
Initialize		Free		
Read				
Write				
Draw				
			I	

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.
- 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.

Use-case Driven Approach

- To identify objects of a system and their behaviours, the lowest level of executable use cases is further analysed 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.

Data Acquisition: Weather Monitoring System

Weather Monitoring Station Requirements

- Wind speed and direction
- Temperature
- Barometric pressure
- Humidity
- Wind Chill
- Dew point temperature
- Temperature trend
- Barometric pressure trend

Weather Monitoring Station Requirements

- TimeDate
- Temperature Sensor
- Pressure Sensor
- Humidity Sensor
- Wind Speed Sensor
- Wind Direction Sensor
- Keypad
- LCD Device
- Timer
- Display Manager

A-Part-of Relationship - Aggregation

- A-part-of relationship, also called aggregation, represents the situation where a class consists of several component classes.
- This does not mean that the class behaves like its parts.
- Example: A car consists of many other classes, one of them is a radio, but a car does not behave like a radio.

- Two major properties of a-part-of relationship are:
- Transitivity: If A is part of B and B is part of C, then A is part of C.
 - Example: A carburetor is part of an engine and an engine is part of a car; therefore, a carburetor is part of a car.
- Antisymmetry: If A is part of B, then B is not part of A.
 - Example: An engine is part of a car, but a car is not part of an engine.

Where responsibilities for certain behaviour must reside?

- Does the part class belong to problem domain?
- Is the part class within the system's responsibilities?
- Does the part class capture more than a single value? (If it captures only a single value, then simply include it as an attribute with the whole class)
- Does it provide a useful abstraction in dealing with the problem domain?

Responsibility

- How am I going to be used?
- How am I going to collaborate with other classes?
- How am I described in the context of this system's responsibility?
- What do I need to know?
- What state information do I need to remember over time?
- What states can I be in?

Key Points

- 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.

Key Points

- 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.
- The A-Part-of Structure is a special form of association.
- Every class is responsible for storing certain information from domain knowledge

Reference

 http://plato.acadiau.ca/courses/Busi/IntroBus/ /CASEMETHOD.html