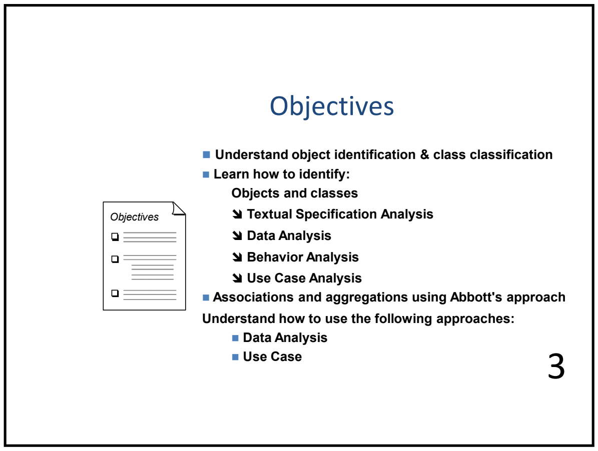
Advanced Object-Oriented Analysis & Design



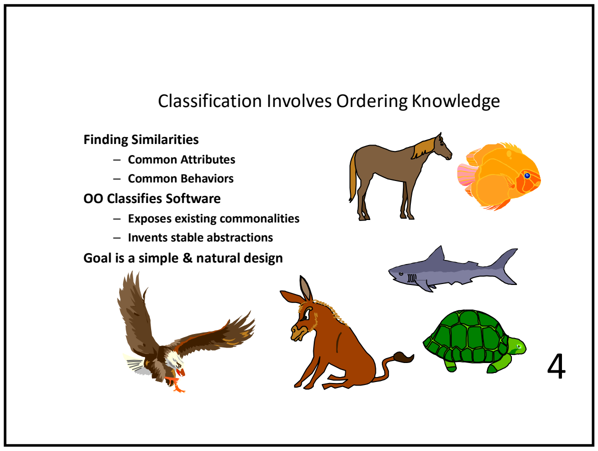
Objectives

Objectives

- Understand object identification & class classification
- Learn how to identify:
  - Objects and classes
    - ➘ Textual Specification Analysis
    - ➘ Data Analysis
    - ➘ Behavior Analysis
    - ➘ Use Case Analysis
- Associations and aggregations using Abbott's approach

Understand how to use the following approaches:

- Data Analysis
- Use Case



Classification Involves Ordering Knowledge

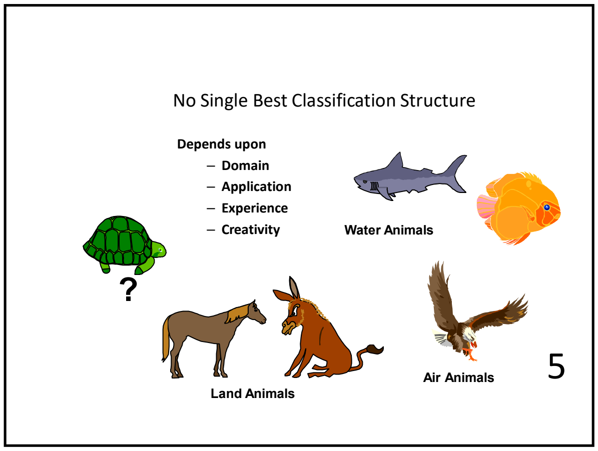
Finding Similarities

- Common Attributes
- Common Behaviors

OO Classifies Software

- Exposes existing commonalities
- Invents stable abstractions

Goal is a simple & natural design

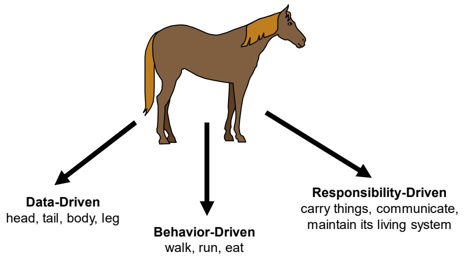


No Single Best Classification Structure

Depends upon

- Domain
- Application
- Experience
- Creativity

Same Object Can Be Perceived from Several Perspectives



Data-Driven
head, tail, body, leg

Behavior-Driven
walk, run, eat

Responsibility-Driven
carry things, communicate, maintain its living system

6

Object/Class Identification Techniques

- ➲**Textual Specification Analysis**
- ➲ **Data Analysis**
- ➲ **Behavior Analysis**
- ➲ **Use-Case Analysis**
- ➲ **Responsibility Analysis**

7

Textual Analysis Approach

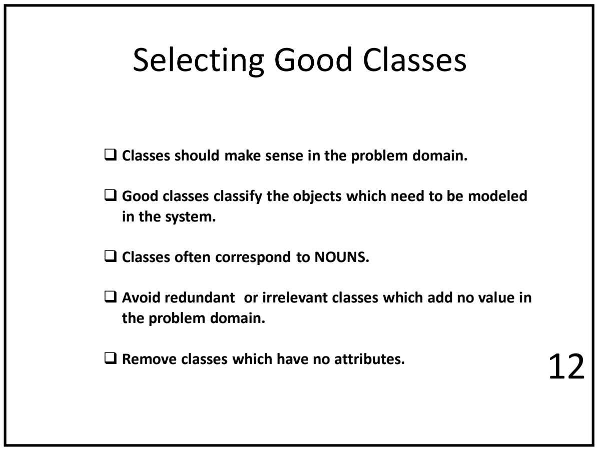
- **Abbott's Noun Approach**
  - **Use noun, pronoun, and noun phrases to identify abstract objects and classes.**
  - **Use singular proper nouns (e.g., sensor number 5) and nouns of direct reference (e.g., the fifth sensor) to identify abstract objects.**
  - **Use plural and common (e.g., sensor) nouns to identify classes.**
  - **Use verbs and predicate phrases (e.g., are simultaneously activated) to identify the associated operations.**
- **Comments**
  - **This approach is the oldest approach**

8

Textual Analysis Approach

- **Benefits:**
  - **Easy for beginners to use**
  - **Abbott's mapping should usually work**
  - **Can be used with pre-existing textual requirements specifications**
  - **Does not require a complete paradigm shift**
- **Risks**
  - **Indirect**
  - **Many software engineers are weak in grammar**
  - **English is vague, Examples**
    - Some nouns can be used as verbs and vis versa
    - Some words (e.g., purchase, record) can be used as both nouns and verbs
  - **Assumes user's requirements are coherent, complete and correct**
  - **No tool support**

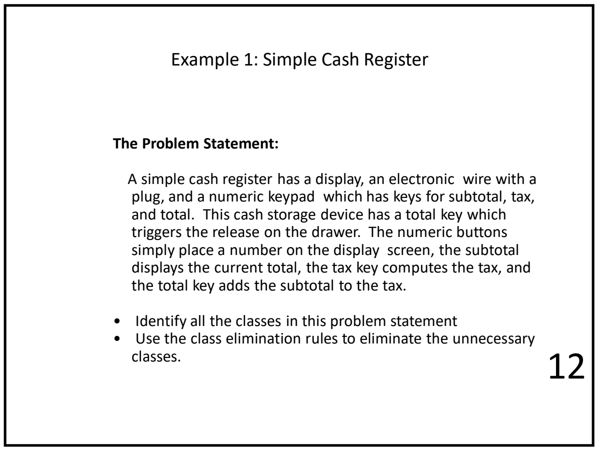
10



Selecting Good Classes

- ❑ **Classes should make sense in the problem domain.**
- ❑ **Good classes classify the objects which need to be modeled in the system.**
- ❑ **Classes often correspond to NOUNS.**
- ❑ **Avoid redundant or irrelevant classes which add no value in the problem domain.**
- ❑ **Remove classes which have no attributes.**

12



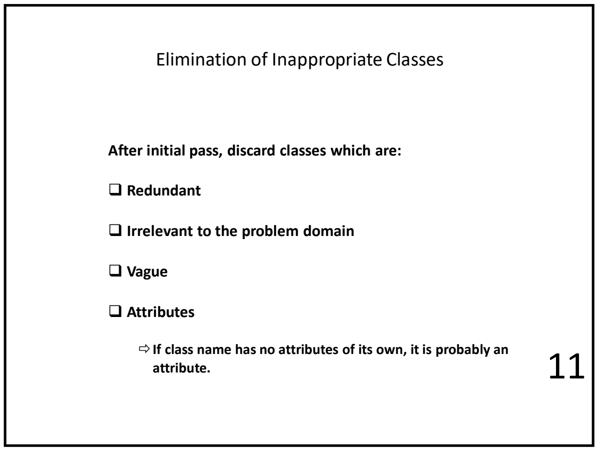
Example 1: Simple Cash Register

The Problem Statement:

A simple cash register has a display, an electronic wire with a plug, and a numeric keypad which has keys for subtotal, tax, and total. This cash storage device has a total key which triggers the release on the drawer. The numeric buttons simply place a number on the display screen, the subtotal displays the current total, the tax key computes the tax, and the total key adds the subtotal to the tax.

- Identify all the classes in this problem statement
- Use the class elimination rules to eliminate the unnecessary classes.

12

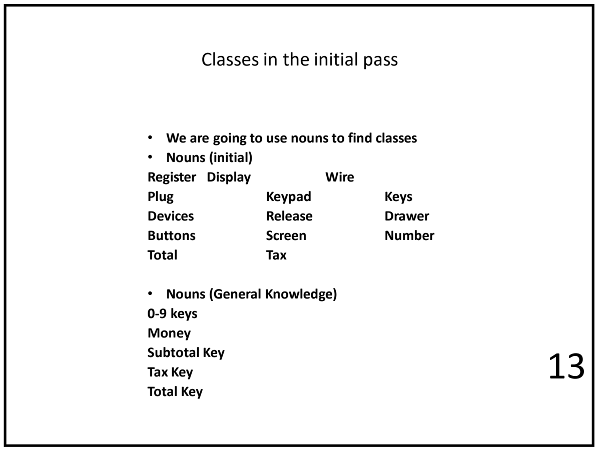


Elimination of Inappropriate Classes

After initial pass, discard classes which are:

- ❑ **Redundant**
- ❑ **Irrelevant to the problem domain**
- ❑ **Vague**
- ❑ **Attributes**
  - ⇨ **If class name has no attributes of its own, it is probably an attribute.**

11



Classes in the initial pass

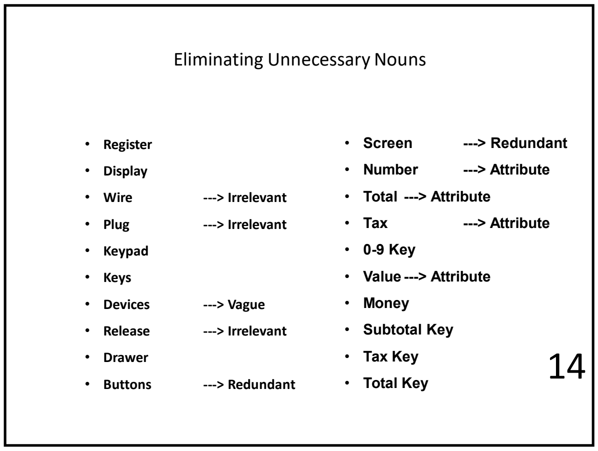
- **We are going to use nouns to find classes**
- **Nouns (initial)**

Register Display Wire
Plug Keypad Keys
Devices Release Drawer
Buttons Screen Number
Total Tax

- **Nouns (General Knowledge)**

0-9 keys
Money
Subtotal Key
Tax Key
Total Key

13

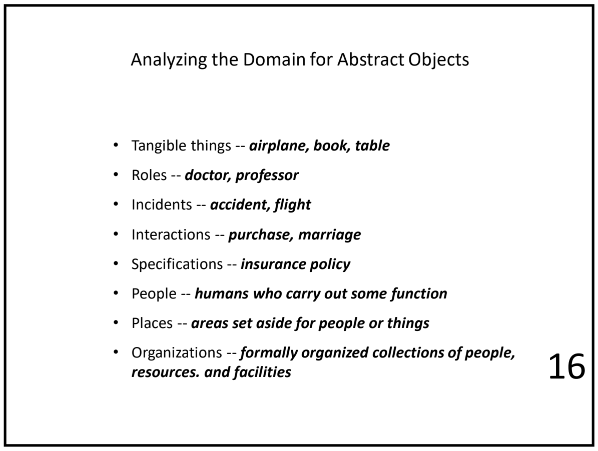


Eliminating Unnecessary Nouns

- **Register**
- **Display**
- **Wire ---> Irrelevant**
- **Plug ---> Irrelevant**
- **Keypad**
- **Keys**
- **Devices ---> Vague**
- **Release ---> Irrelevant**
- **Drawer**
- **Buttons ---> Redundant**

- **Screen ---> Redundant**
- **Number ---> Attribute**
- **Total ---> Attribute**
- **Tax ---> Attribute**
- **0-9 Key**
- **Value ---> Attribute**
- **Money**
- **Subtotal Key**
- **Tax Key**
- **Total Key**

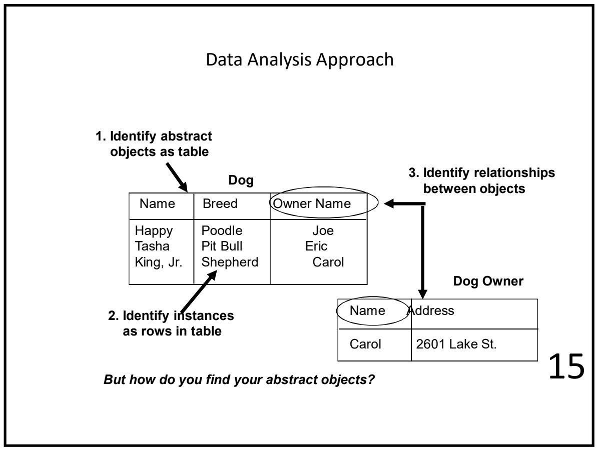
14



Analyzing the Domain for Abstract Objects

- Tangible things -- ***airplane, book, table***
- Roles -- ***doctor, professor***
- Incidents -- ***accident, flight***
- Interactions -- ***purchase, marriage***
- Specifications -- ***insurance policy***
- People -- ***humans who carry out some function***
- Places -- ***areas set aside for people or things***
- Organizations -- ***formally organized collections of people, resources. and facilities***

16



Data Analysis Approach

1. Identify abstract objects as table
2. Identify instances as rows in table
3. Identify relationships between objects

Dog

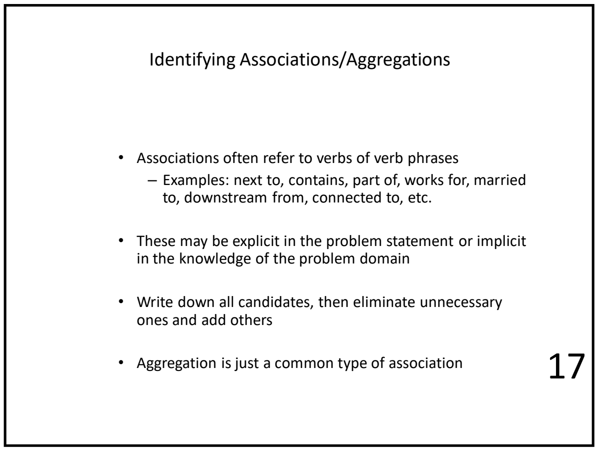
Name	Breed	Owner Name
Happy Tasha King, Jr.	Poodle Pit Bull Shepherd	Joe Eric Carol

Dog Owner

Name	Address
Carol	2601 Lake St.

But how do you find your abstract objects?

15



Identifying Associations/Aggregations

- Associations often refer to verbs of verb phrases
  - Examples: next to, contains, part of, works for, married to, downstream from, connected to, etc.
- These may be explicit in the problem statement or implicit in the knowledge of the problem domain
- Write down all candidates, then eliminate unnecessary ones and add others
- Aggregation is just a common type of association

17