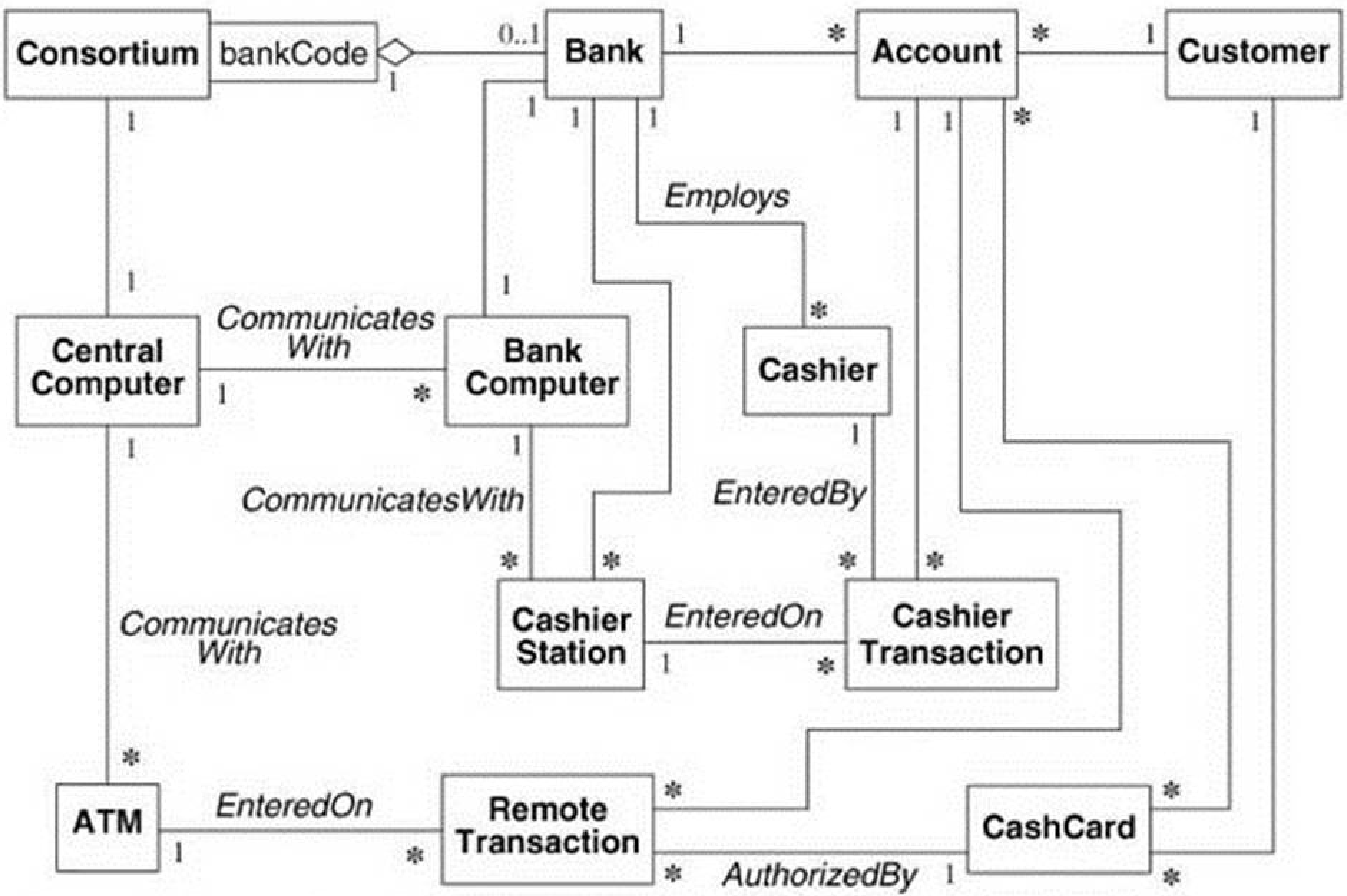


Figure 12.9 Initial class diagram for ATM system.

Add Attributes

- Attributes usually correspond to nouns followed by possessive phrases

  Ex: the color of the car

- Less likely to be fully described in the problem statement

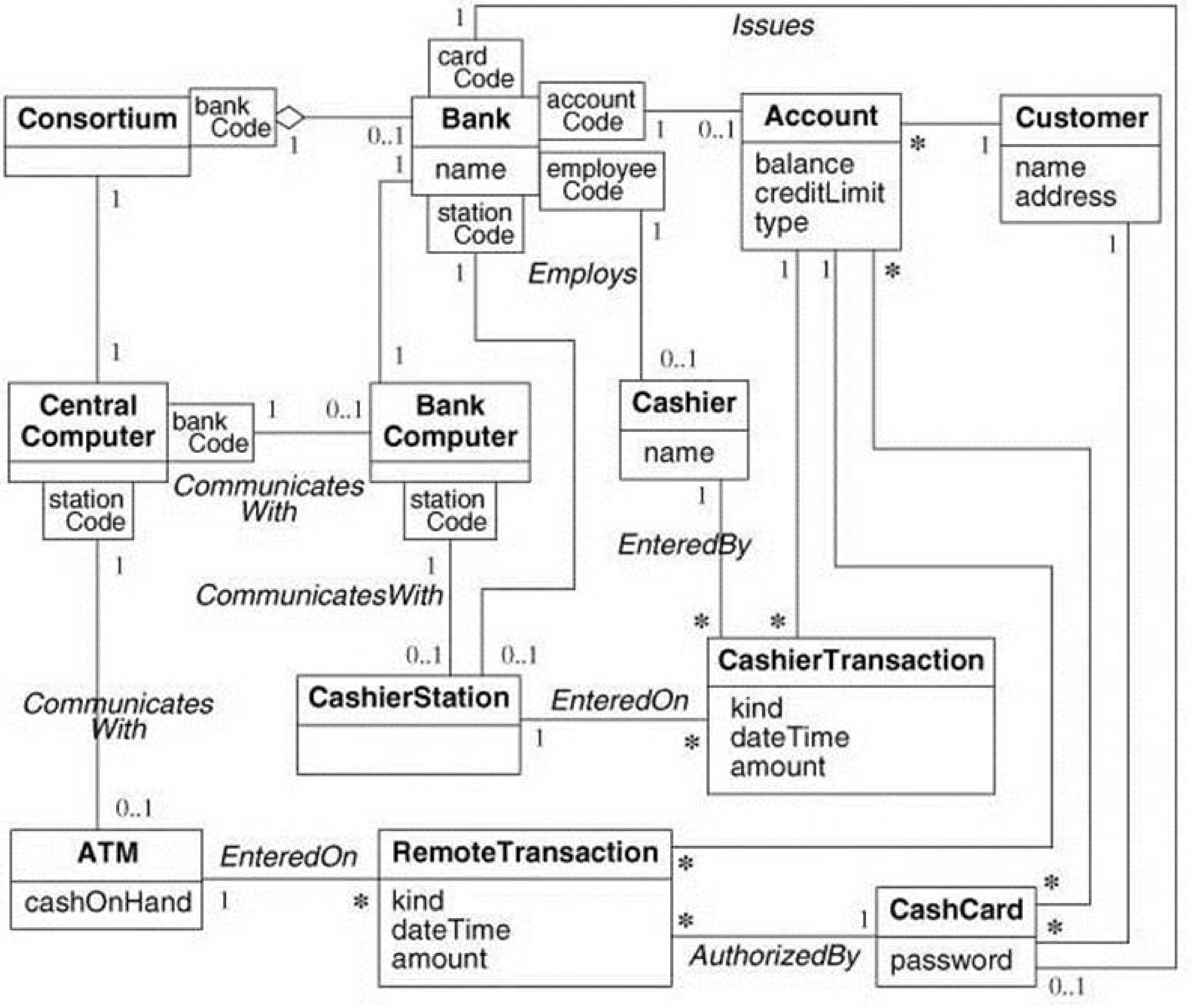


Figure 12.10 ATM class model with attributes.

Refine with inheritance

- Bottom up generalization
- Top down specialization
- Generalization vs. enumeration
- Multiple inheritance
- Similar associations

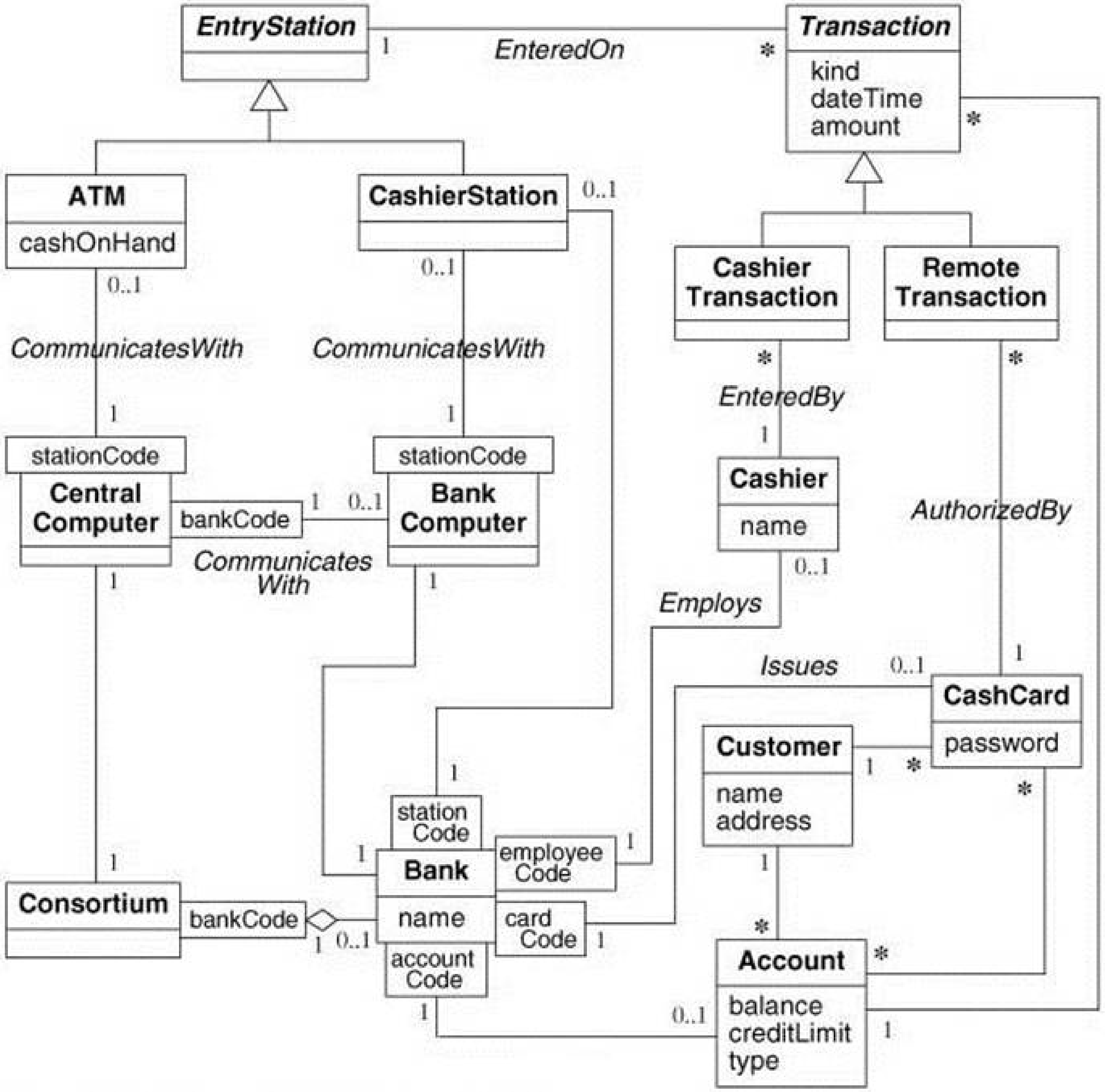


Figure 12.11 ATM class model with attributes and inheritance.

Object Diagram

- A **class diagram** describes the **types of objects** in the system and the various kinds of static relationships that exist among them
- An **object diagram** shows a set of objects and their relationships at a point in time
- It is a snapshot of the objects in a system at a point in time.
- Because it shows instances rather than classes, an object diagram is often called an **instance diagram**.

- *An object is a* concept, abstraction or thing with identity that has meaning for an application

Blaha & Rumbaugh

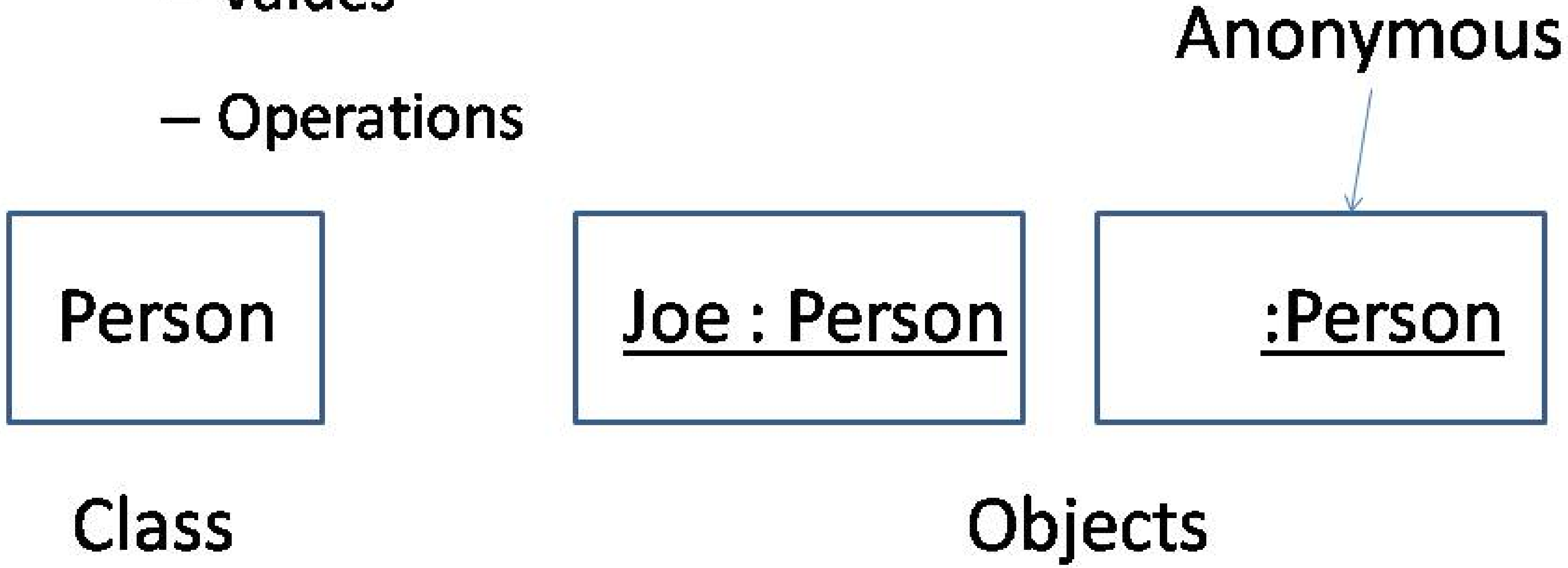
- An object has state, exhibits some well-defined behavior, and has a unique identity

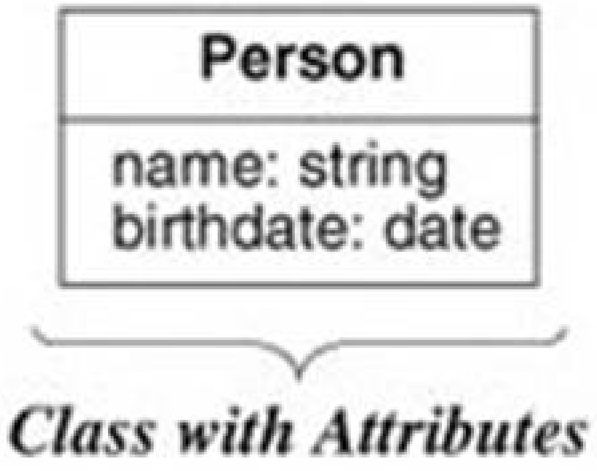
Booch

- Objects appear as proper nouns in problem description
- Some objects have real-world counterparts
- Some are conceptual entities
- An object is an instance of a class

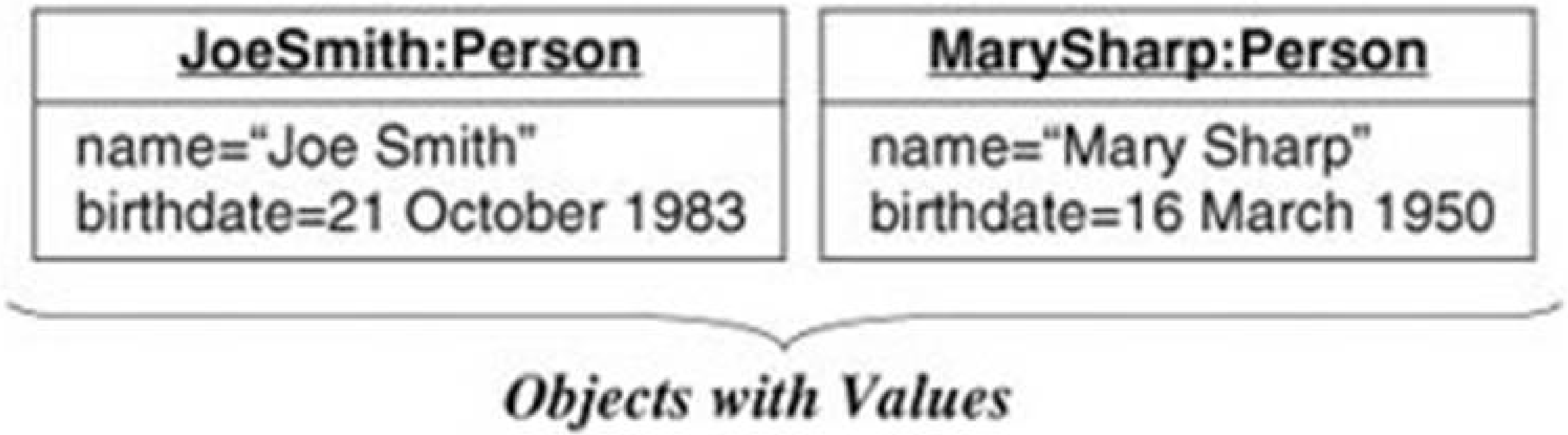
Terms and Concepts

- *Object is shown by a rectangle*
- *Objects have …*
  - Names - Used to distinguish one object from another
  - Values
  - Operations

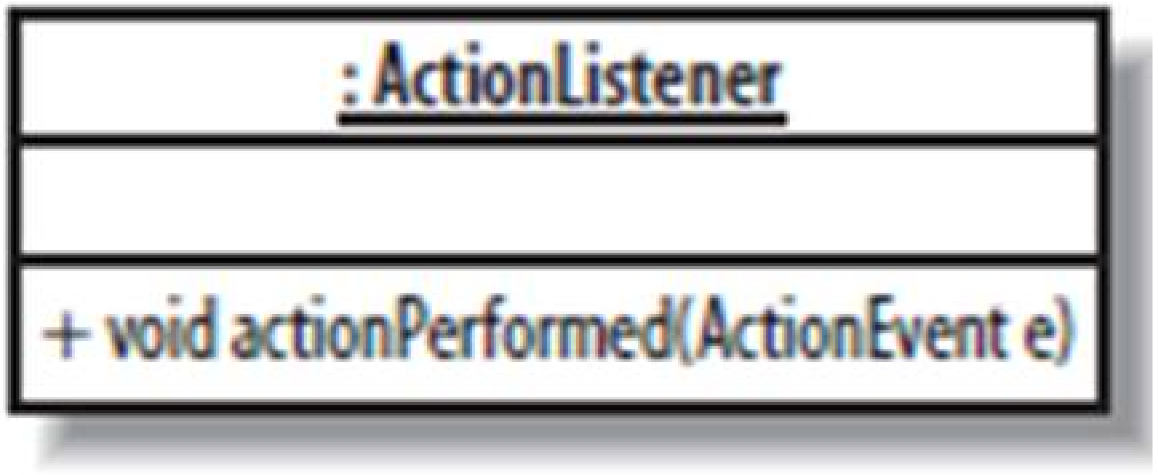




Class with Attributes



Objects with Values



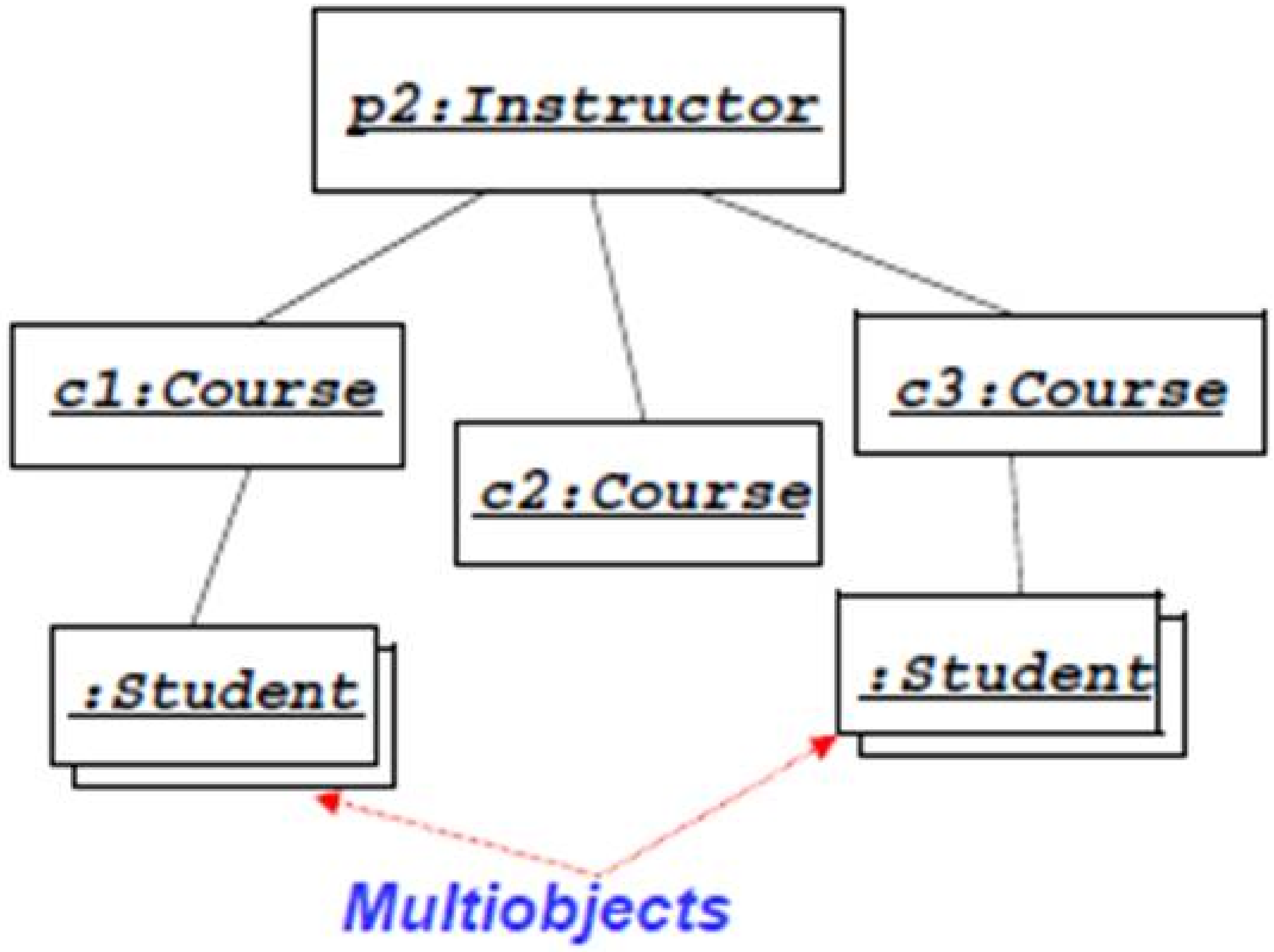
Object with method

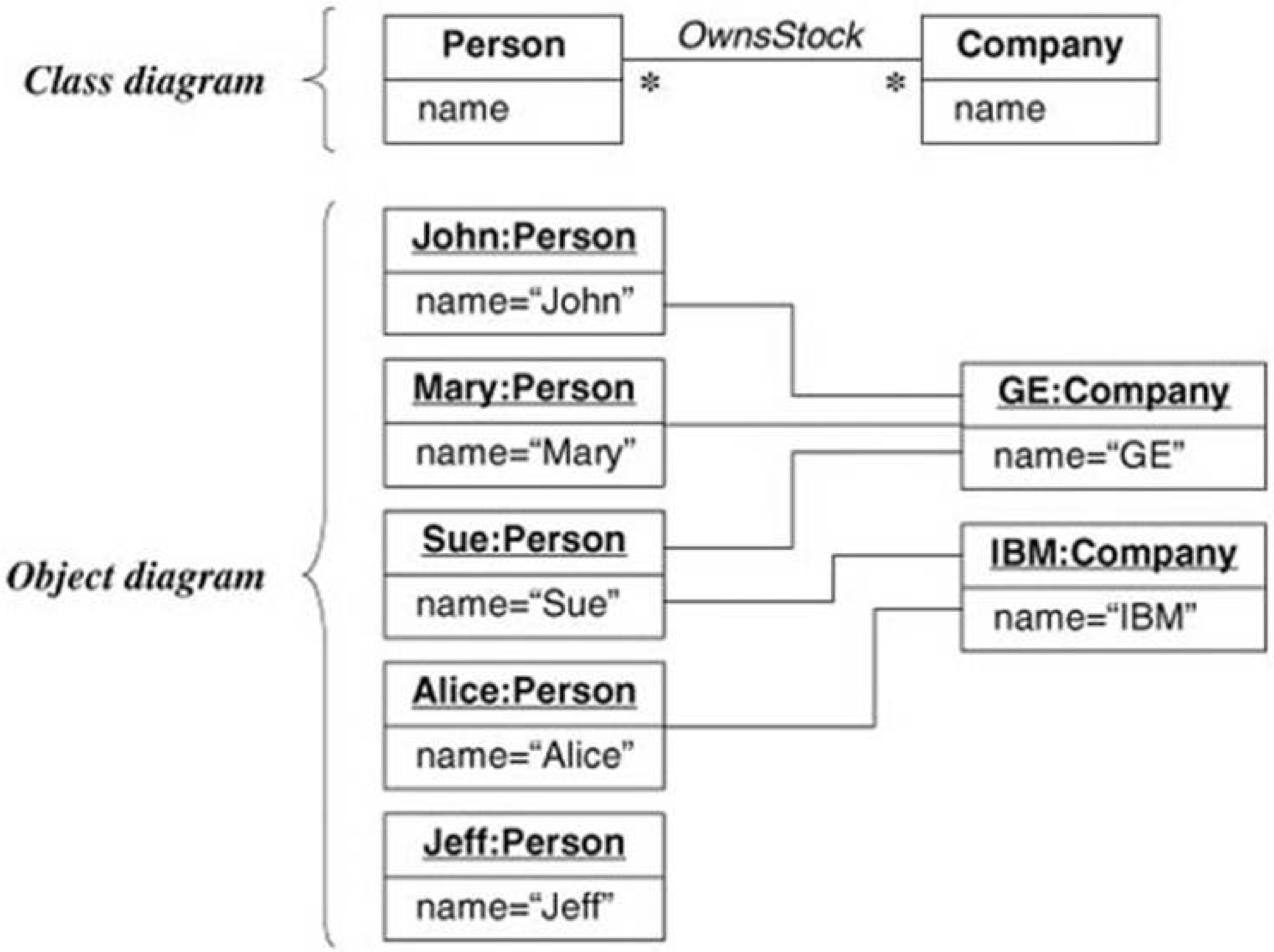
State

- An object has state
- It encompasses all the (**usually static**) properties of the object plus the current (**usually dynamic**) values of each of these properties.
- These properties include the attributes and associations of the object, as well as its aggregate parts.
- An object's state is dynamic.
- It specifies the values at a given moment in time

Multiobjects

- A multiobject is a set of objects, with an undefined number of elements

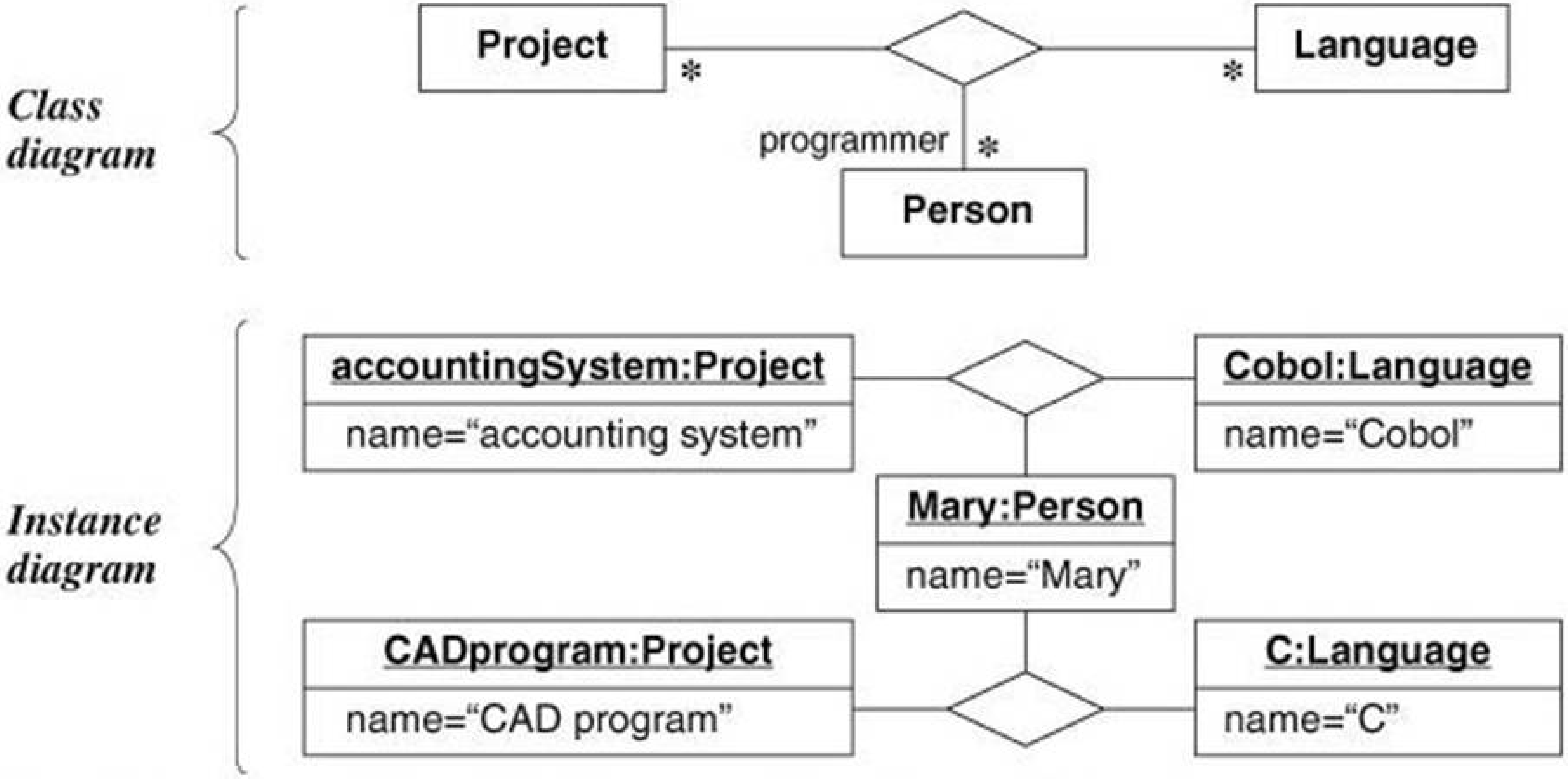




Many-to-many association. An association describes a set of potential links in the same way that a class describes a set of potential objects.

Object Diagrams

- Object Diagrams Commonly contain
  - Objects
  - Links
- A link is an instance of an association
- A Links is a physical or conceptual connection among objects
- A link can be written as a tuple – a list of objects



Ternary association and links. An n-ary association can have association end names, just like a binary association.

When to Use Object Diagrams

- Object diagrams model the static design view of the system
- An object diagram represents one static frame in the dynamic storyboard represented by an interaction diagram.
- Object diagrams are used to visualize, specify, construct, and document the existence of certain instances in the system, together with their relationships to one another.

Modeling an Object Structure

- Identify some function or behavior (of the part) of the system you are modeling
- Identify the classes, interfaces, and other elements that participate in this collaboration and their relationships
- Consider one scenario that walks through this
- Freeze that scenario at a moment in time, and render each object that participates in it
- Expose the state and attribute values of each such object, as necessary, to understand the scenario.
- Expose the links among these objects

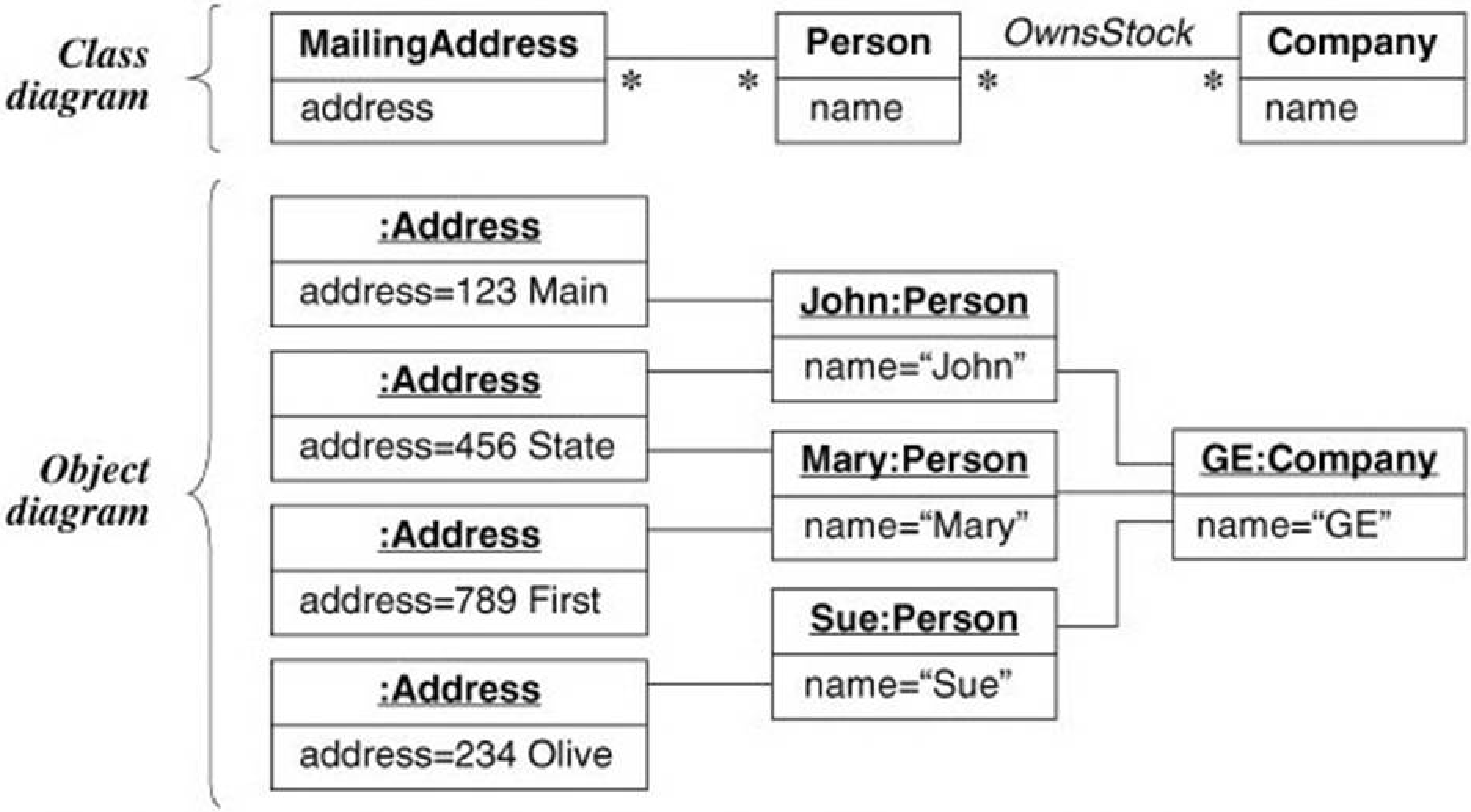
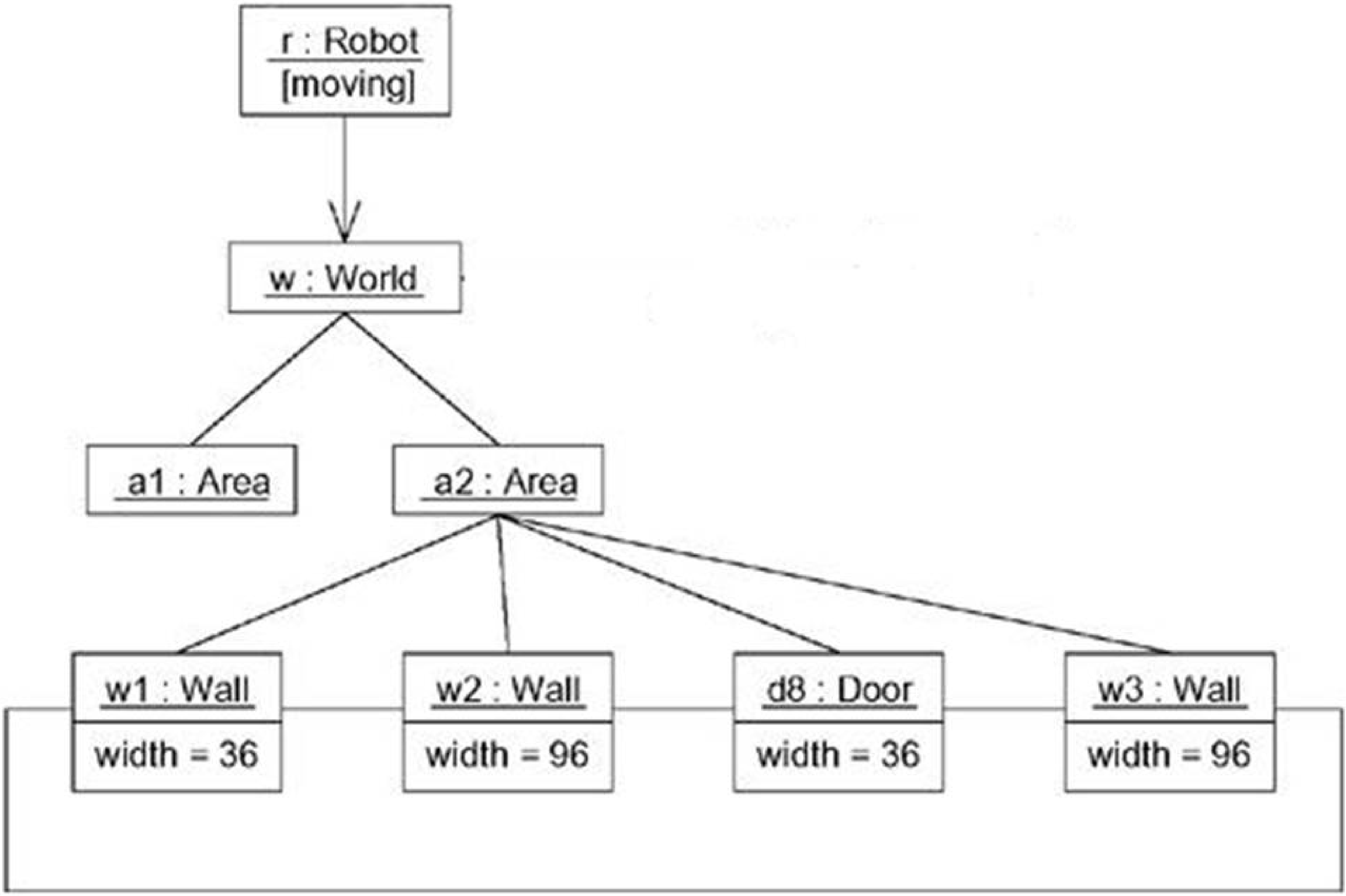


Figure 3.28 A sample model and examples. Traversal of multiple associations can yield a bag.


r : Robot
[moving]
w : World
a1 : Area
a2 : Area
w1 : Wall
width = 36
w2 : Wall
width = 96
d8 : Door
width = 36
w3 : Wall
width = 96

Use Cases

- A *use case is …*
  - a high-level description of how the product (system) will be used.
  - describes the behavior of the system (or part of it) as seen by the end users, analysts and testers
  - shows the interactions between the users of a system and the system itself
- Use cases are independent of their realization
- A use case is a set of scenarios

- A **scenario** is a sequence of steps describing an interaction between a user and a system.
- “Buy a Product” scenario on a web-based on-line store

The customer browses the catalog and adds desired items to the shopping basket. When the customer wishes to pay, the customer describes the shipping and credit card information and confirms the sale. The system checks the authorization on the credit card and confirms the sale both immediately and with a follow-up e-mail.

- What if credit card authorization fails
- There is no need to capture the shipping and credit card information for a regular customer
- All these scenarios are different yet similar.
- In all these three scenarios, the user has the same goal: to buy a product.
- This user goal is the key to use cases
- A **use case is a set of scenarios tied together** by a common user goal.