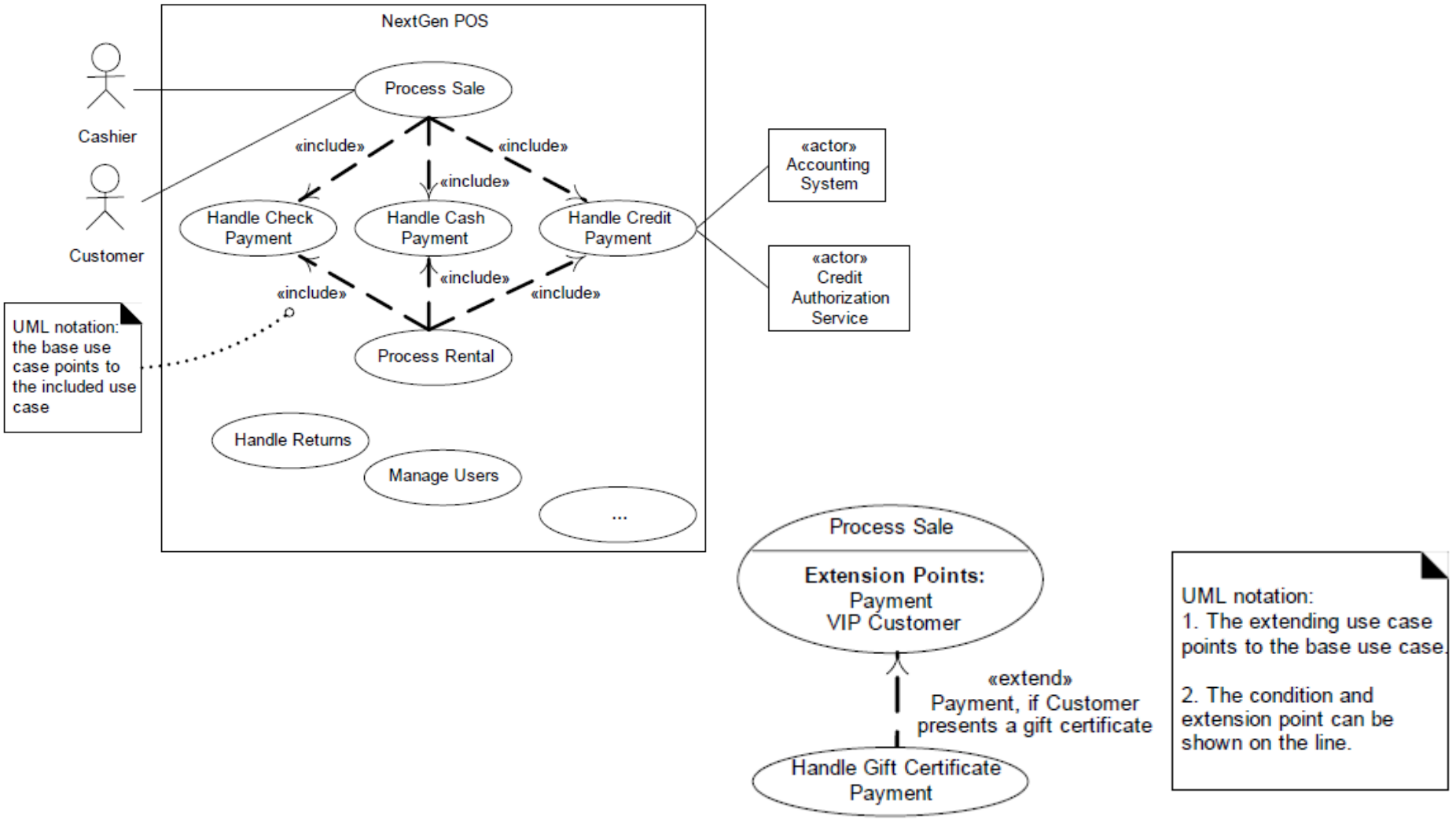
Domain Models (Conceptual Modeling)

Software Design and Analysis

CSCI 2040

Use Cases – Include vs Extend



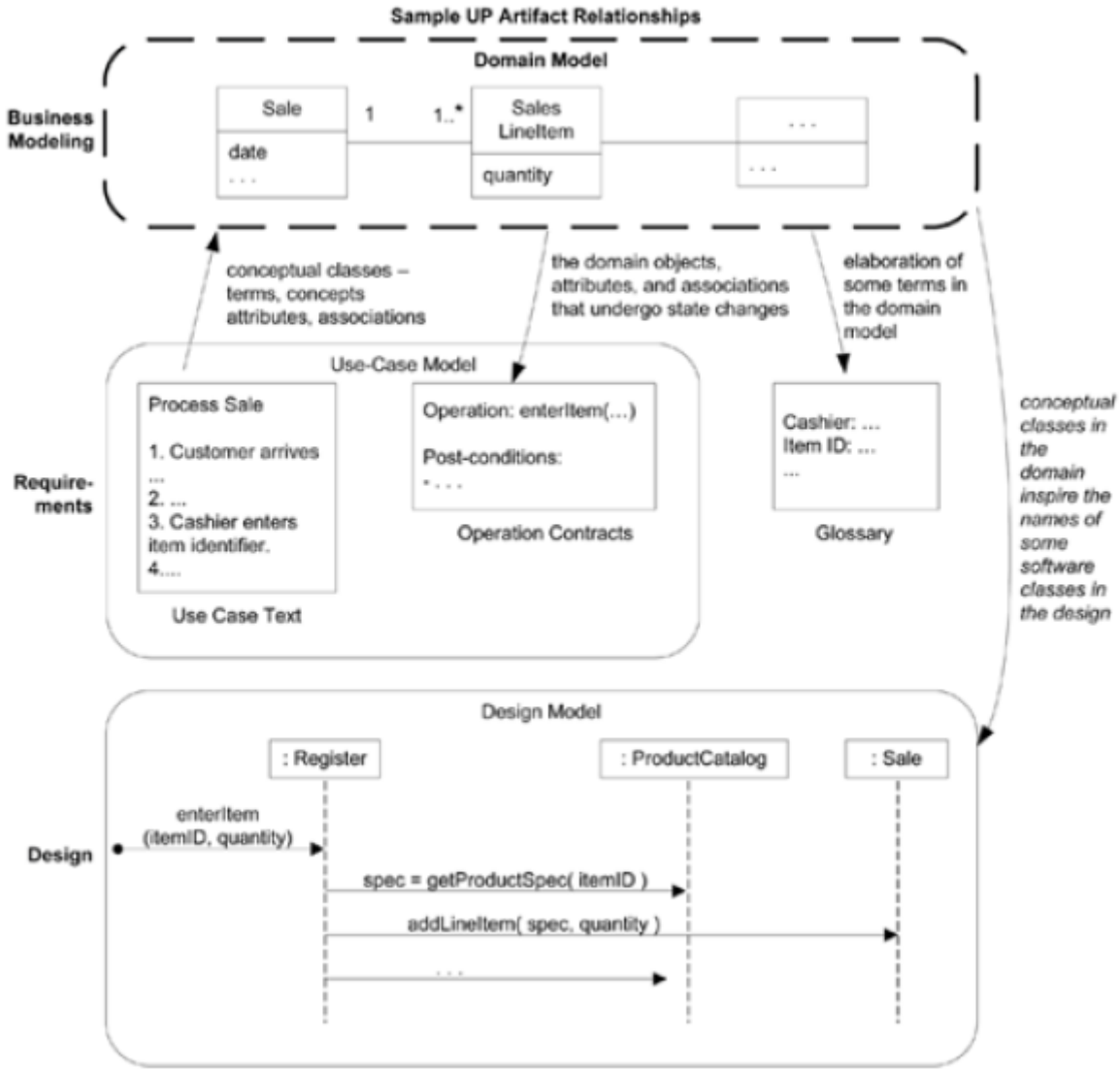
Objectives

- Identify conceptual classes related to the current iteration.
- Create an initial domain model.
- Model appropriate attributes and associations.

Introduction – Domain Model

- A **domain model** is the most important model in object oriented *analysis*.
- It illustrates noteworthy concepts in a domain.
- It can act as a source of inspiration for designing some software objects.
- As with all things in an agile modeling and UP spirit, a domain model is optional.

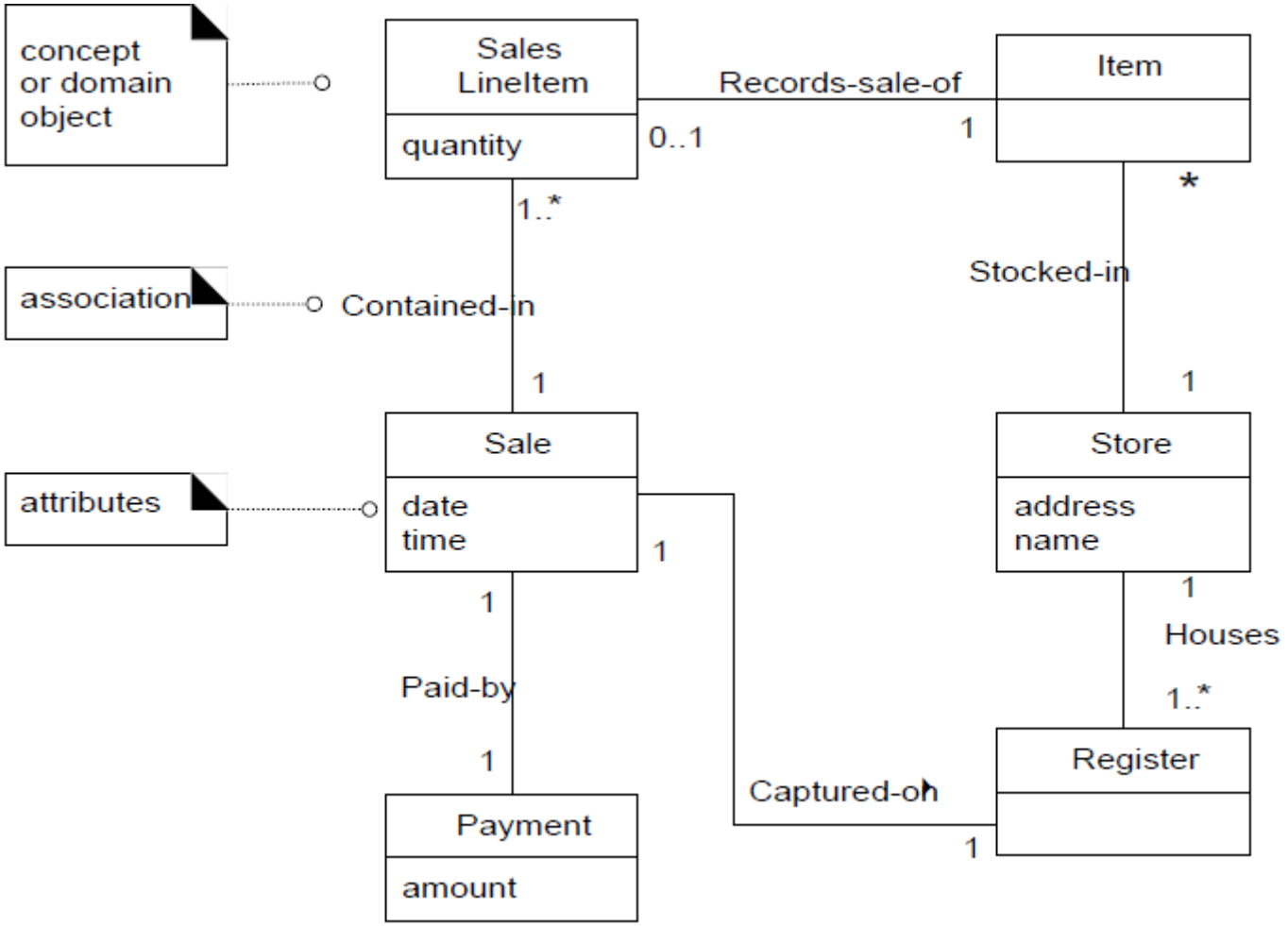
Partial Domain Model



UML Class Diagram

- **UML class diagram** consists of:
  - **Conceptual classes**, e.g., Payment and Sale
  - **Relationships (association)** between classes, e.g., Payment is related to Sale
  - **Attributes**, e.g., Sale has a date and time
- Avoid a waterfall-mindset big-modeling effort to make a thorough analysis.

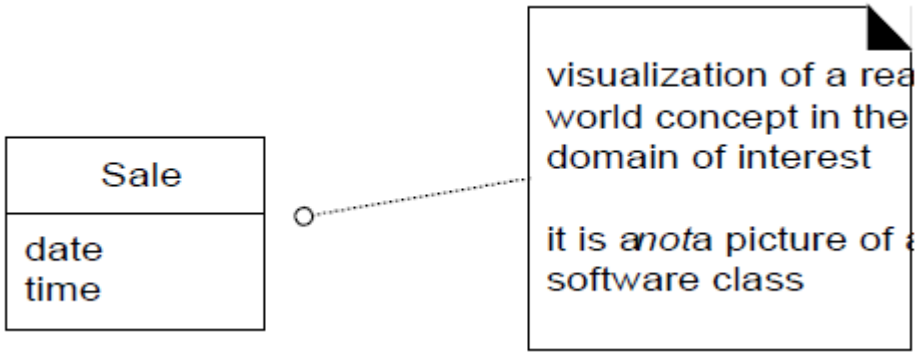
Partial Domain Model

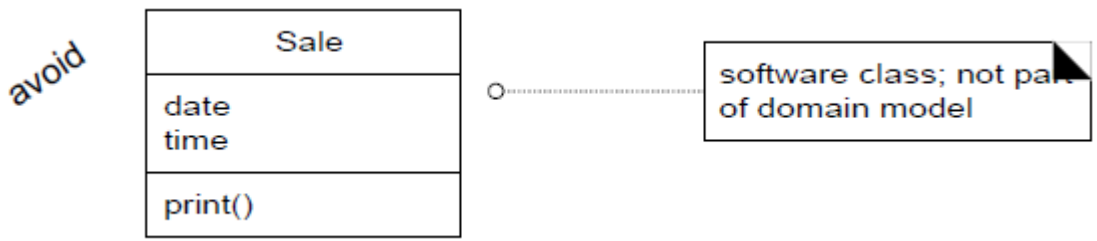


Domain Model

- In the UP, the term "**Domain Model**" means a representation of real-situation **conceptual classes**, **not of software objects**.
- Applying UML notation, a domain model is illustrated with a set of **class diagrams**.

Domain Model

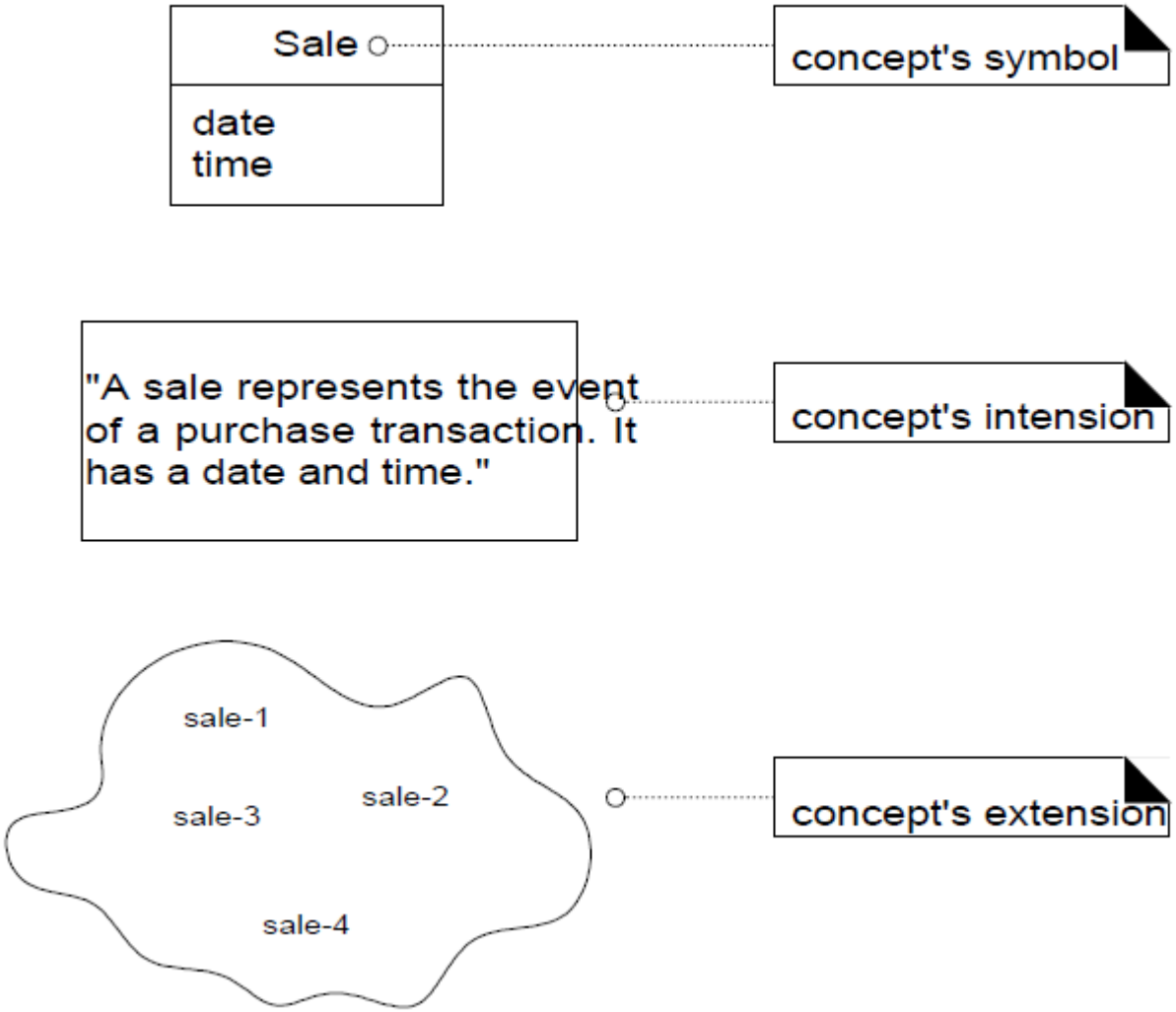




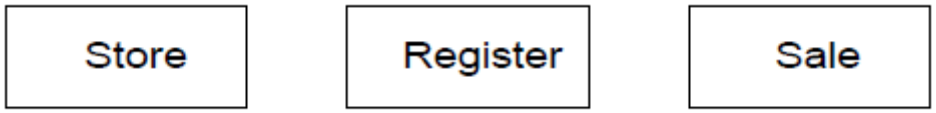
Conceptual Classes

- **Symbol** words or images representing a conceptual class.
- **Intension** the definition of a conceptual class.
- **Extension** the set of examples to which the conceptual class applies.

Conceptual Classes



Conceptual Classes in Sale Domain



Conceptual Class Identification

Identify noun phrases.

- Identify noun phrases in textual description (e.g, use cases) and choose important ones

Example of Identifying Noun Phrases

Main Success Scenario (or Basic Flow):

1. **Customer** arrives at a **POS checkout** with **goods** and/or **services** to purchase.
2. **Cashier** starts a new **sale.**
3. **Cashier** enters **item identifier.**
4. System records **sale line item** and presents **item description, price,** and running **total.** Price calculated from a set of price rules.

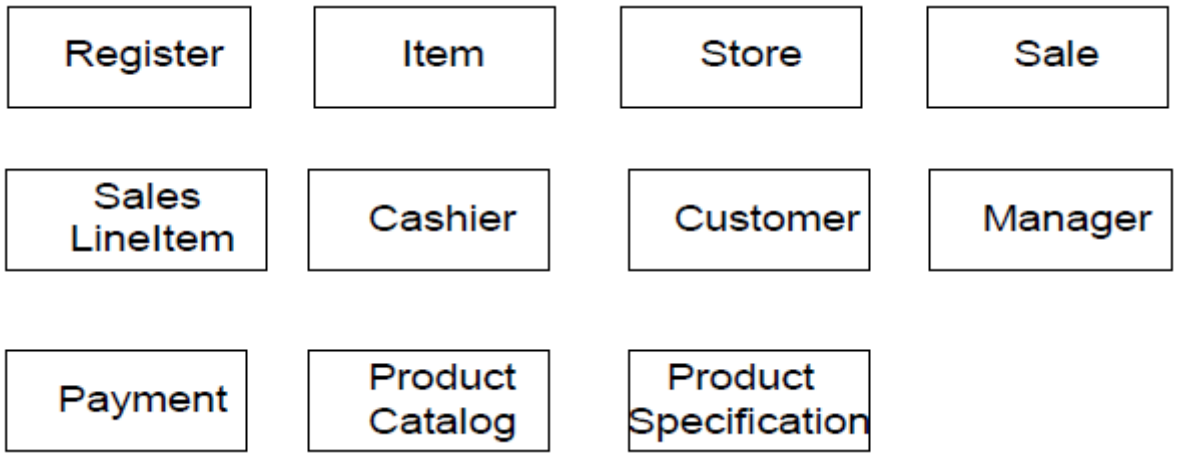
Cashier repeats steps 2-3 until indicates done.

5. System presents total with **taxes** calculated.

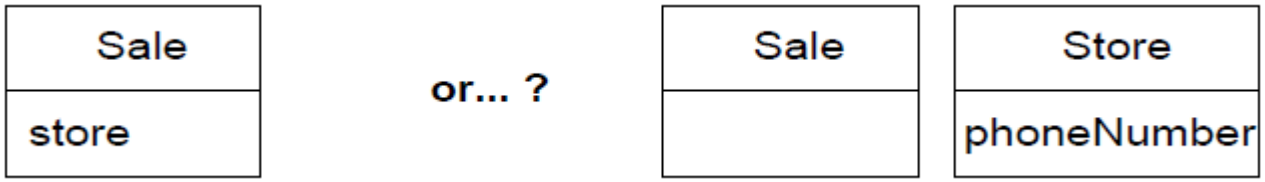
Example of Identifying Noun Phrases

6. Cashier tells Customer the total, and asks for **payment.**
7. Customer pays and System handles payment.
8. System logs the completed **sale** and sends sale and payment information to the external **Accounting** (for accounting and **commissions)** and **Inventory** systems (to update inventory).
9. System presents **receipt.**
10. Customer leaves with receipt and goods (if any).

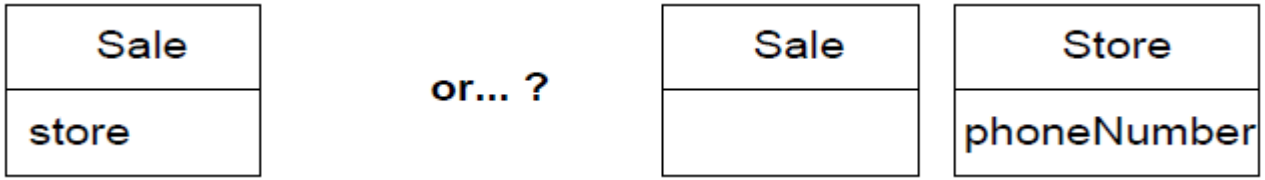
NextGen POS – Initial Domain Model



Common Mistakes

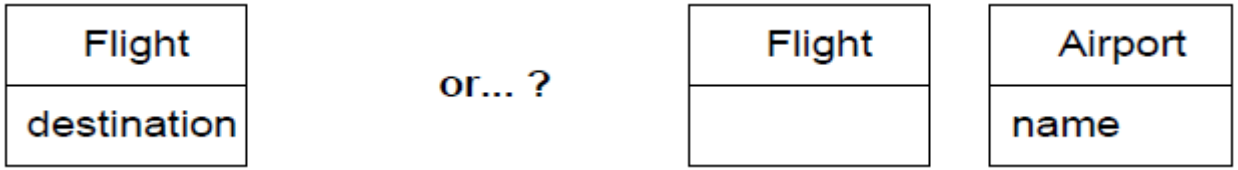


Common Mistakes

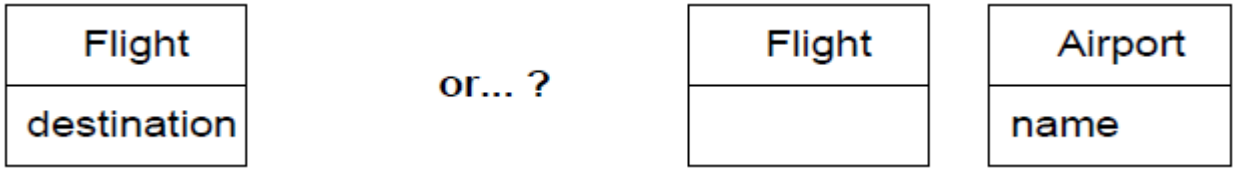


- In the real world, a store is not considered a number or text.
- The term suggests a legal entity, an organization.
- Thus, *Store* should be a conceptual class.

Common Mistakes



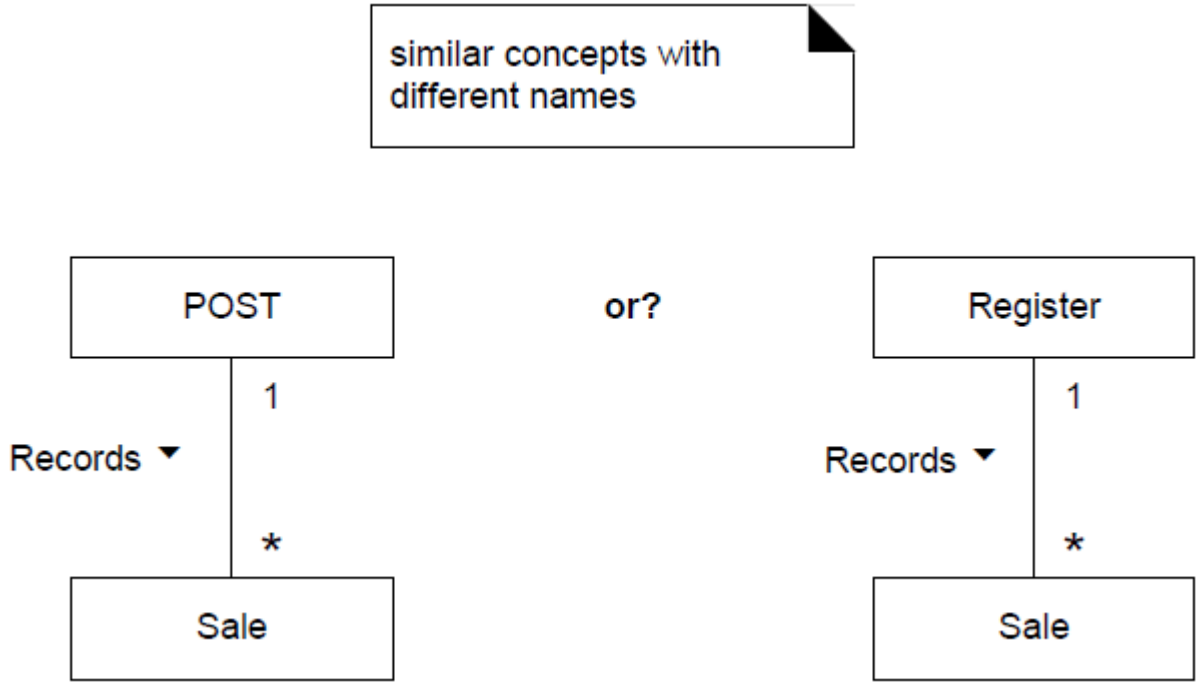
Common Mistakes



- Similarly, Airport should be a conceptual class.

Similar Conceptual Classes

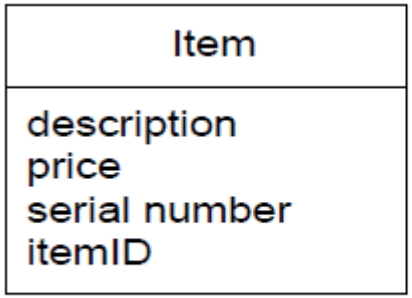
- Stick to One for similar concepts.



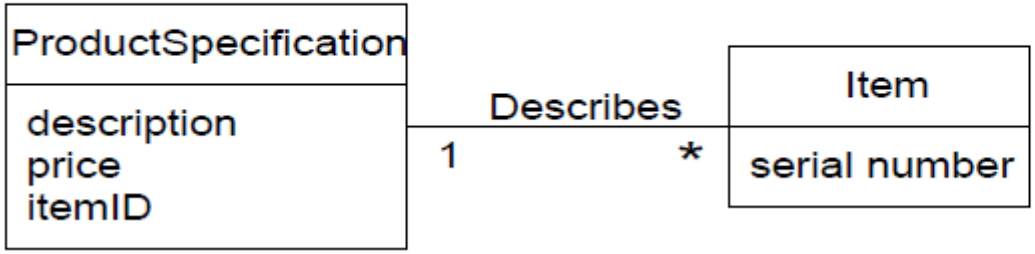
Specifications Guidelines

- Reduce duplication when possible!

Specifications



Worse

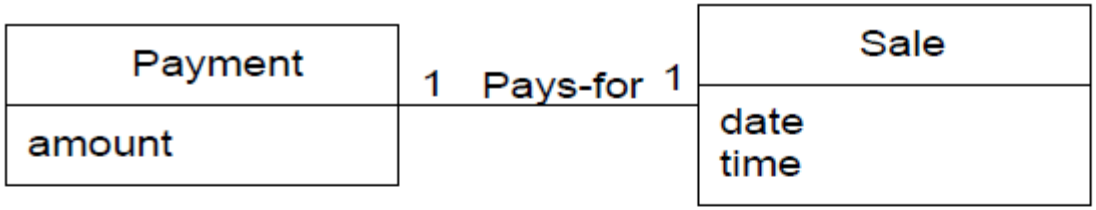


Better

Why create a Domain Model?

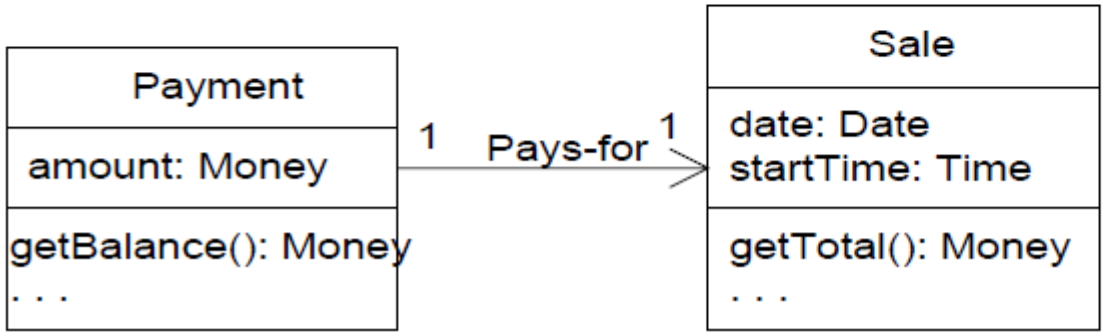
- Use software class names inspired from names in the domain model,
  - with objects having domain-familiar information and responsibilities.

Domain Model vs Design Model



UP Domain Model

Raw UML class diagra notation used in an essential model visualizing real-world concepts.



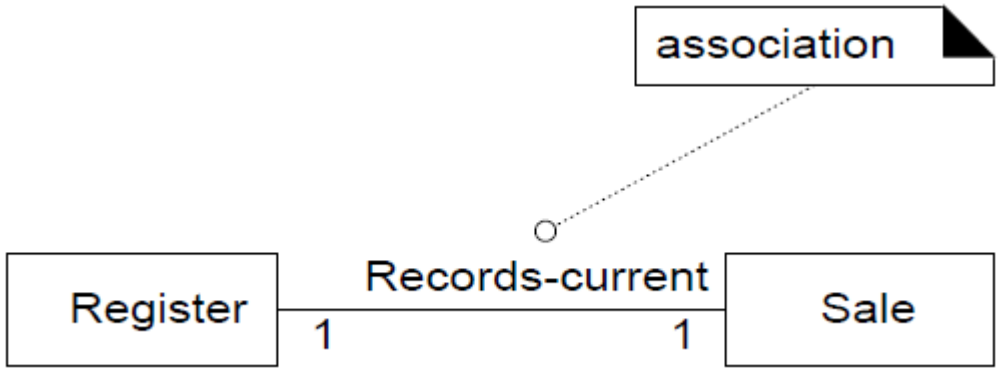
UP Design Model

Raw UML class diagra notation used in a specification model visualizing software components.

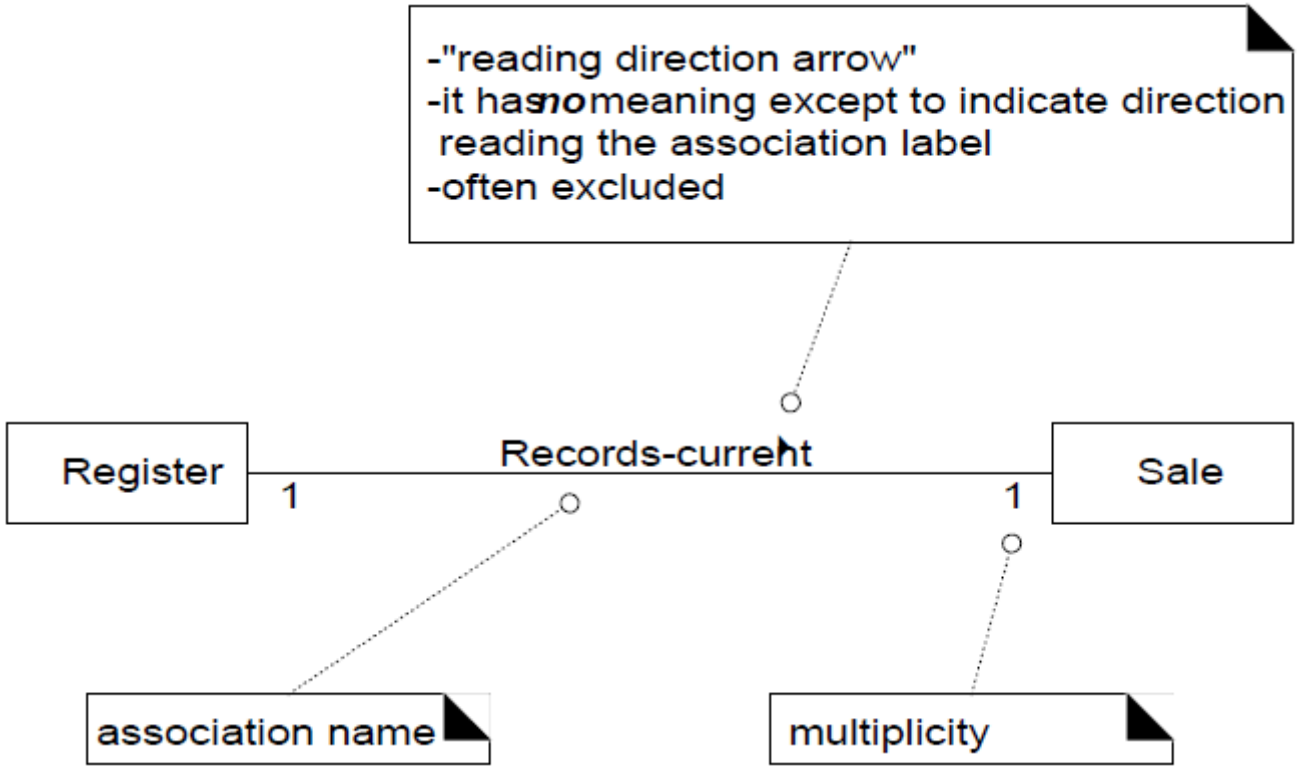
Association

- An **association** is a relationship between classes that indicates some interesting connection.
  - In the UML, associations are defined as the relationship between two or more classes.

Adding Associations

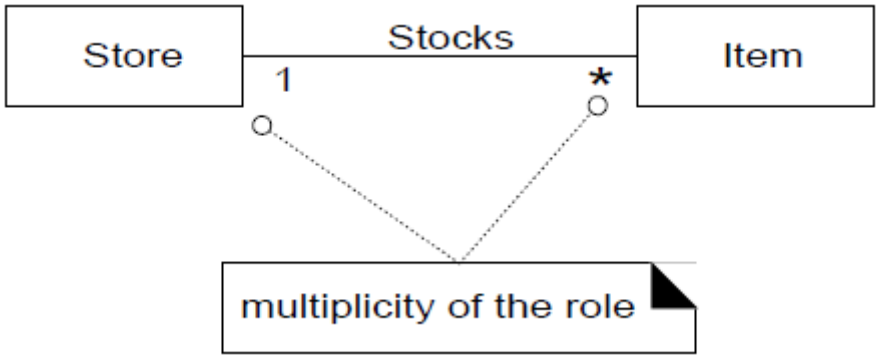


UML Notation for Associations

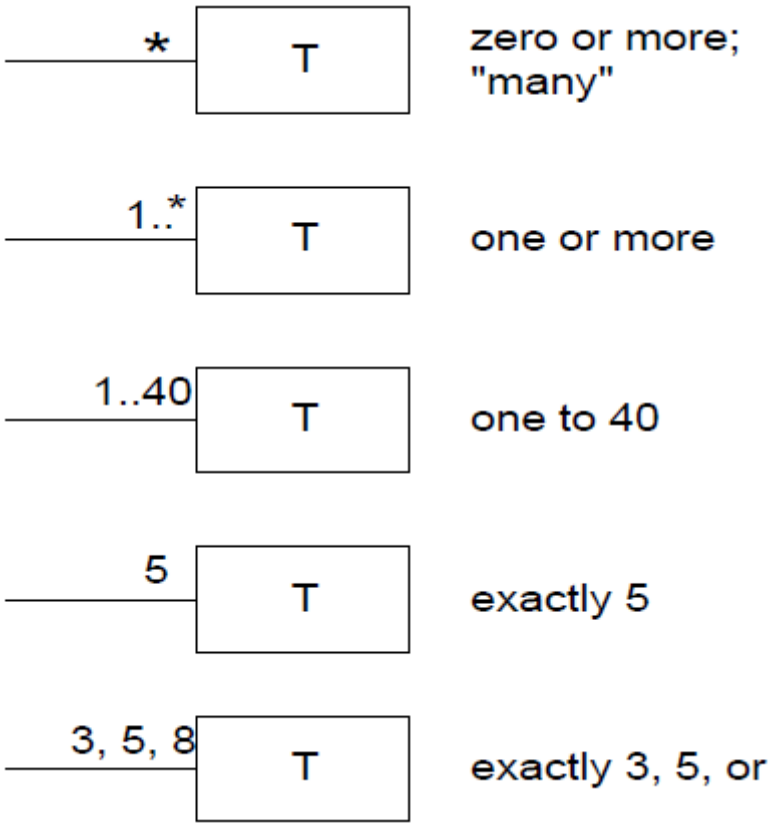


Multiplicity

- **Multiplicity** defines how many instances of a class *A* can be associated with one instance of a class *B*.

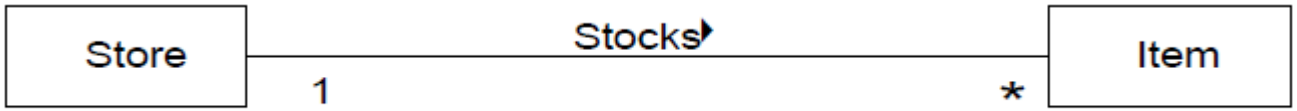


Multiplicity Values

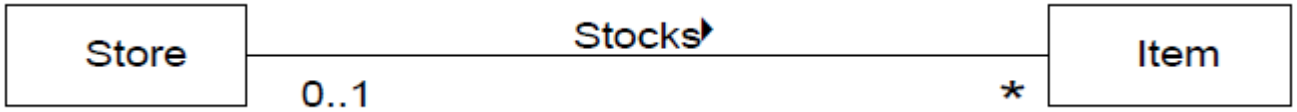


1 VS 0..1

- If we want to ensure that item has to be in the stock of the store.

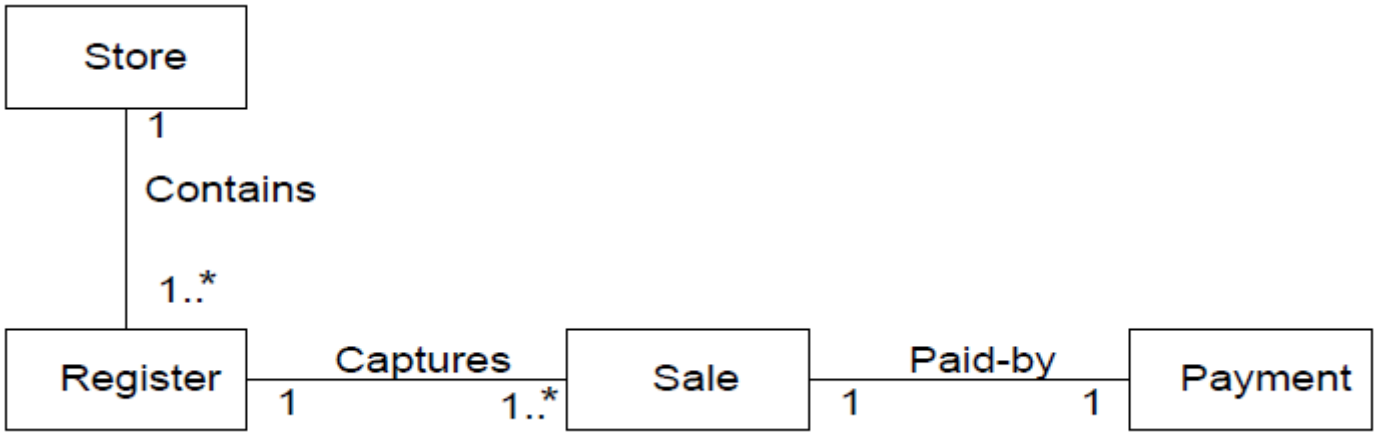


- For example, a particular item may become sold out or discarded.



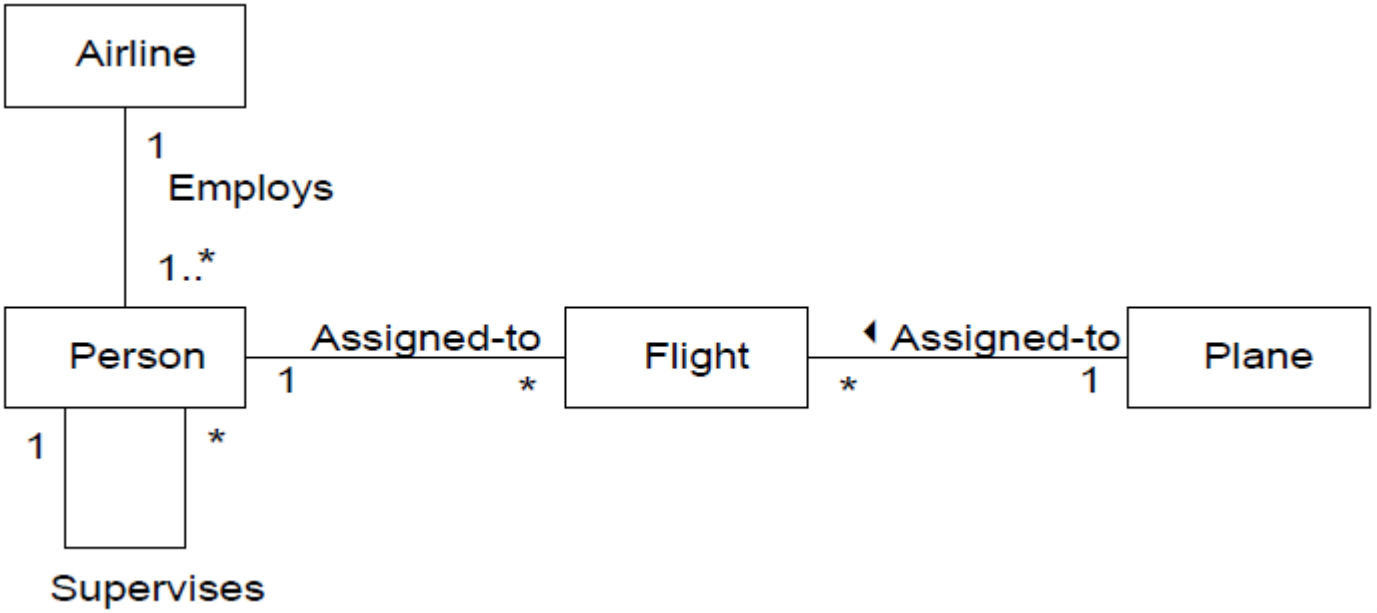
Reading an Association Name

- Typically left to right, top to down



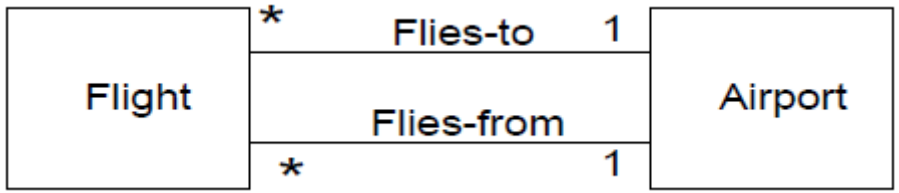
Reading an Association Name

- Unless indicated by an arrow.

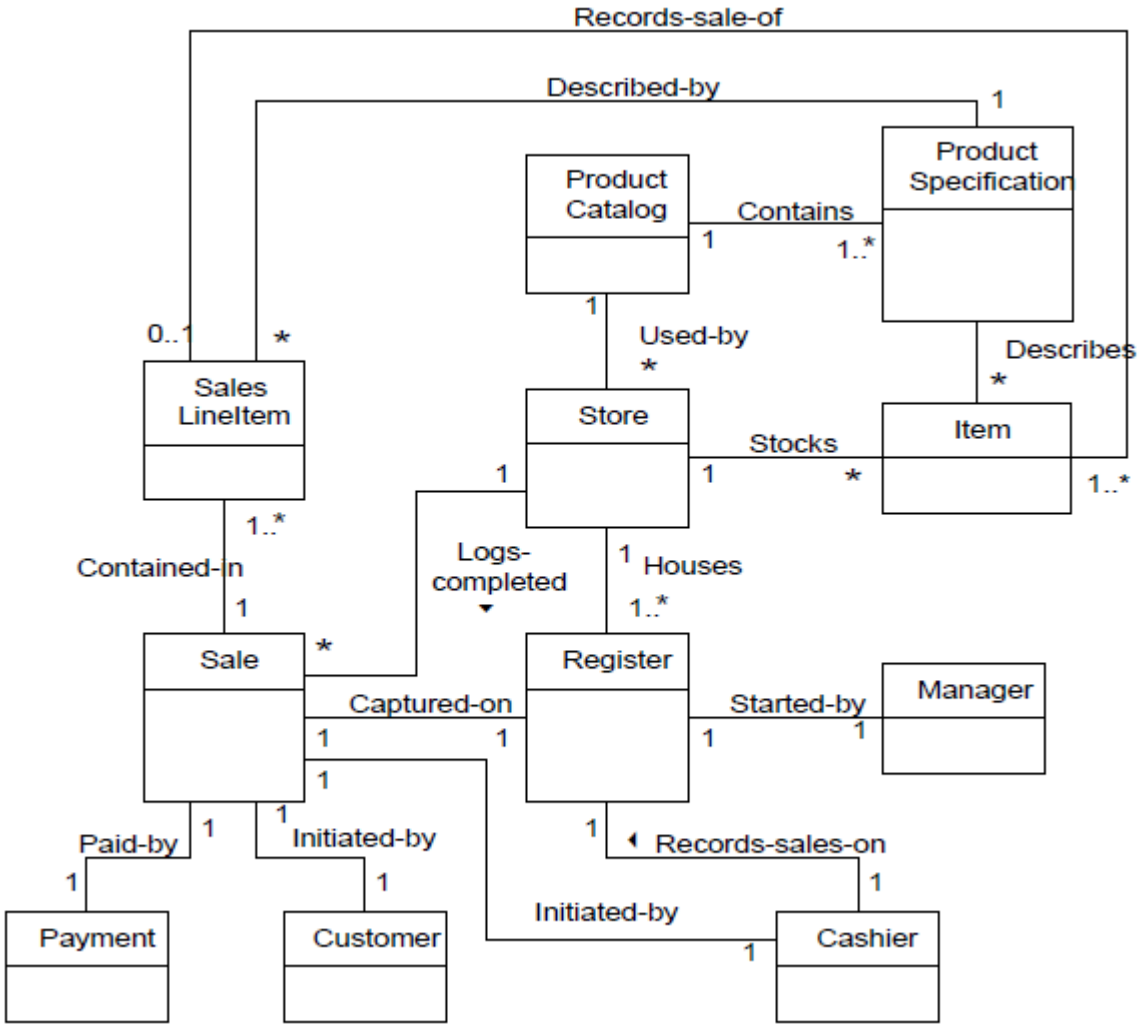


Multiple Associations

- Two classes may have multiple associations between them in a UML class diagram.
- E.g, the flying-to and flying-from associations are distinctly different relationships, which should be shown separately.

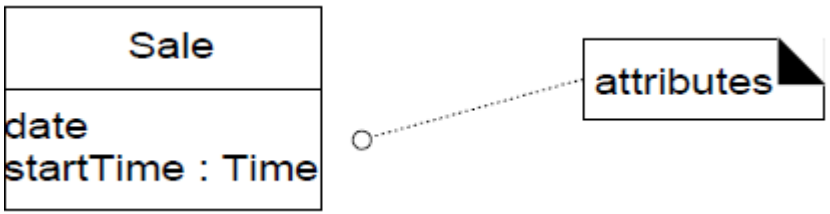


Partial Domain for POS

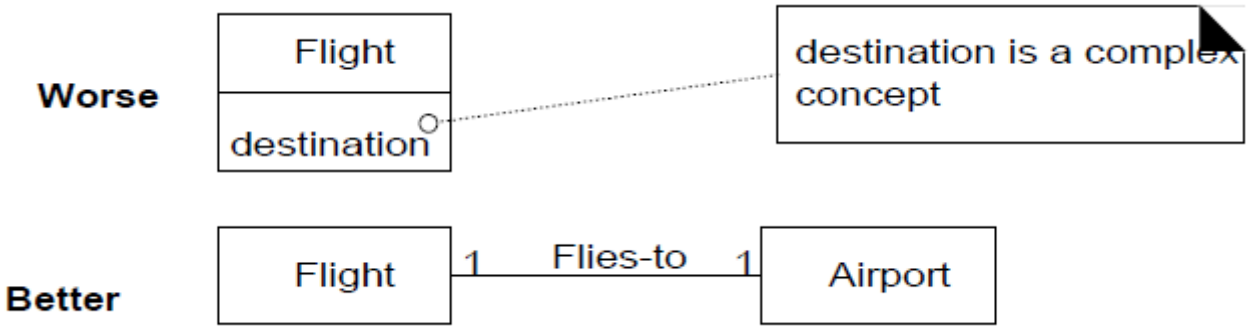


UML Attribute Notation

- It is useful to identify those attributes of conceptual classes that are needed to satisfy the information requirements of the current scenarios under development.
- An **attribute** is a logical data value of an object.



UML Attribute Notation

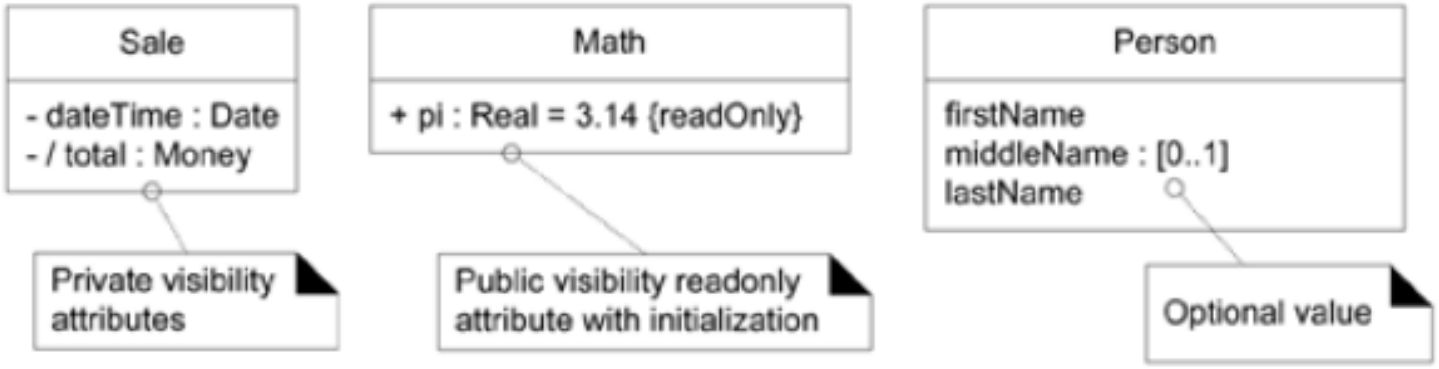


Data Types

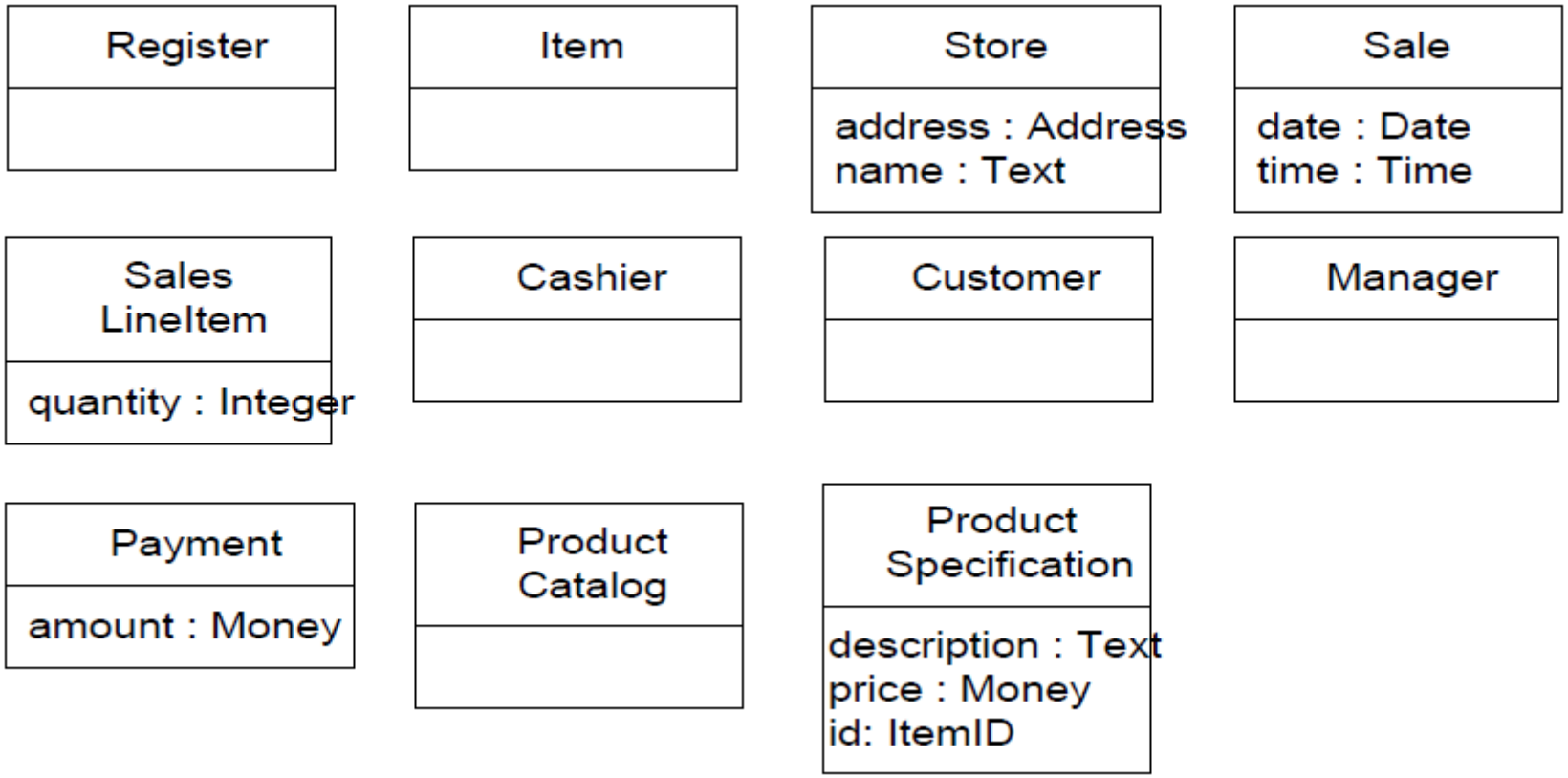

OK
Product Description
1
1
ItemID
id
manufacturerCode
countryCode
Store
1
1
Address
street1
street2
cityName
...
OK
Product Description
itemId : ItemID
Store
address : Address

Extended Attribute Notation in UML

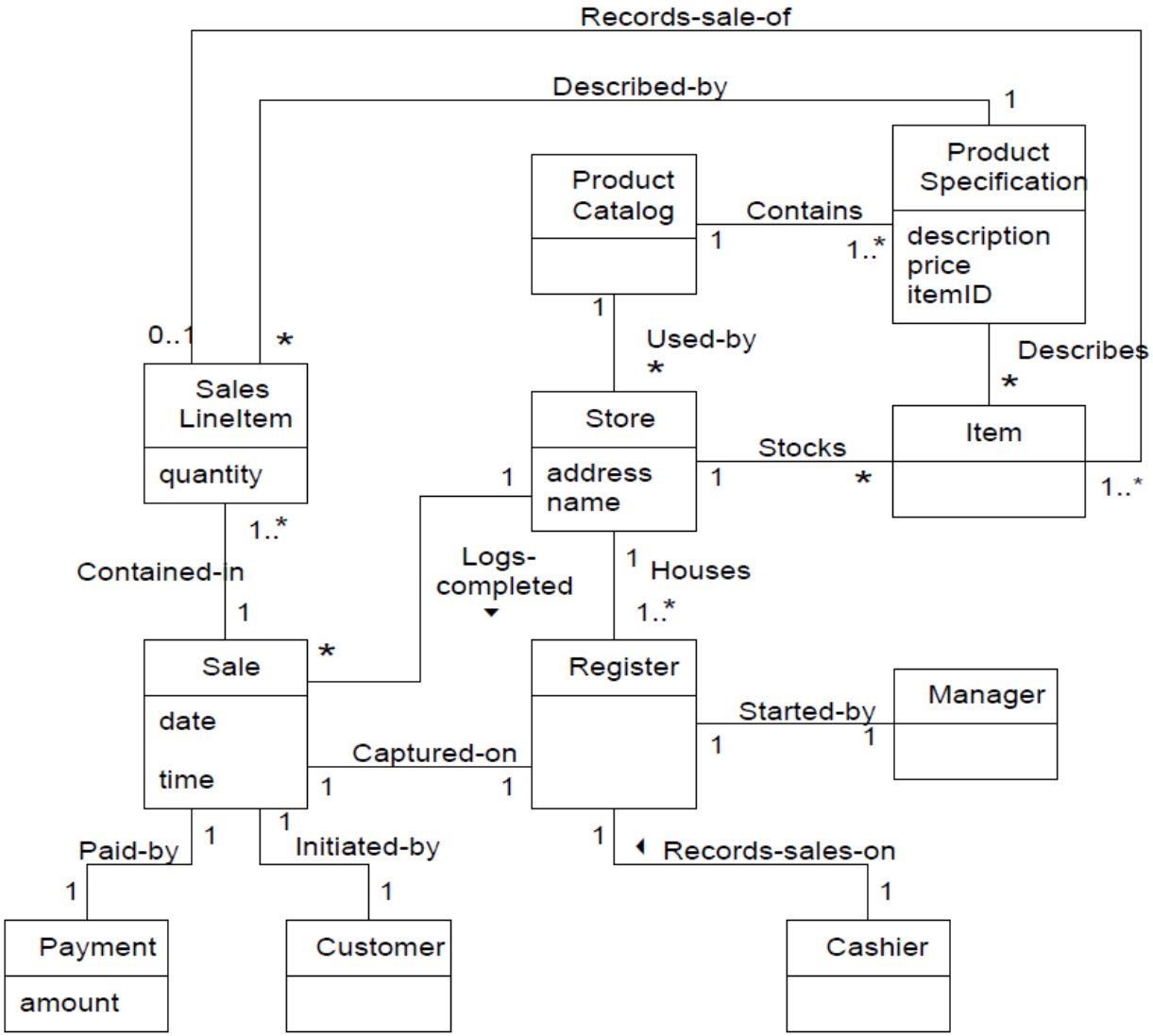
Optionally:



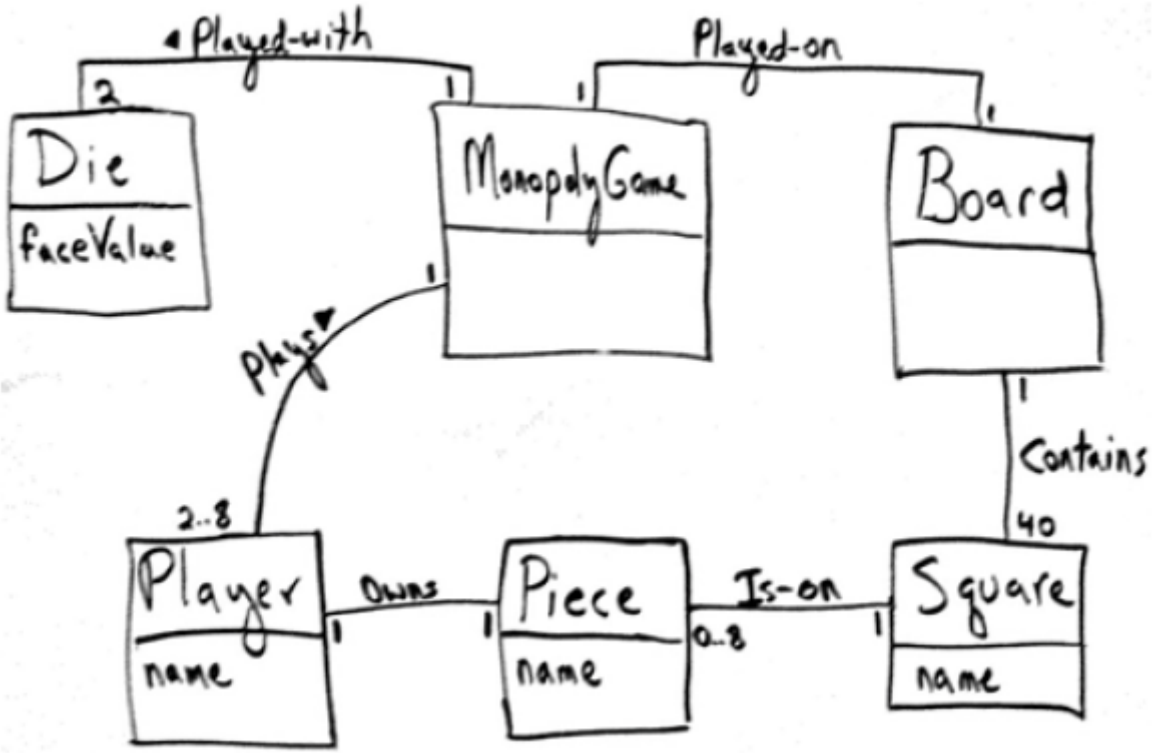
Attributes in NextGen POS Domain Model



POS Partial Domain with Attributes



Agile Diagram Model! Monopoly..



Quiz

- How to identify conceptual classes?
- What is the difference between domain model and design model? Does domain model represent software classes?
- Is domain model required in agile UP?
- Provide an example of conceptual classes connected by multiple associations.

Actions

- Review Slides.
- Read Chapter 9 (Domain Models)
  - *Applying UML and Patterns*, Craig Larman