Step 1: Identify Conceptual Classes

- Reuse or modify existing partial models created by experts
    - "recipes" for well-known problems and domains (e.g., accounting, stock market, …)
- Consider common categories
- Identify nouns and noun phrases from the fully dressed use case

Common Categories

Category	**Examples**
Physical objects	Register, Airplane
Places	Store, Airport
Transactions	Sale, Payment, Reservation
Roles of people	Cashier, Manager
Scheduled Events	Meeting, Flight
Records	Receipt, Ledger
Specifications and descriptions	FlightDescription, ProductSpecification
Catalogs of descriptions	ProductCatalog

Example: Simplified "Process Sale"

No credit cards, no taxes, no external accounting system, no external inventory system, …

- Customer arrives with goods
- Cashier starts a new sale

Possible conceptual classes: Customer, Cashier, Item (i.e., goods), Sale

Simplified “Process Sale”, cont.

- Cashier enters item ID
- System records sale line item and presents item description, price, and running total
- In the end, cashier tells customer the total and asks for payment

Possible conceptual classes: SalesLineItem, ProductSpecification (description + price + item ID), Payment

Simplified "Process Sale", cont.

- Cashier enters amount tendered (cash)
- System presents change due, and releases cash drawer
- Cashier deposits cash and returns change
- System presents receipt

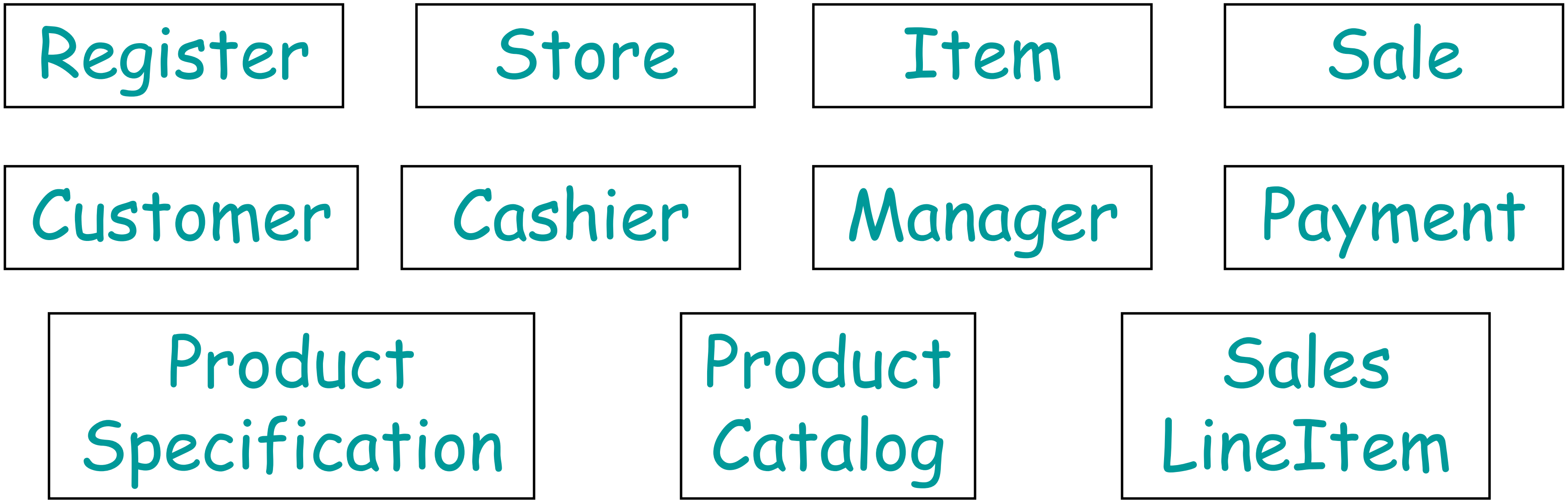
Possible conceptual classes:
Register (implied by cash drawer), Receipt

Simplified “Process Sale“, cont.

- Want a completely integrated system
  - Store: has the items and the registers
  - ProductCatalog: stores the product specifications for all items
  - Manager: starts all the registers in the morning
    - Need this for the initial implementation: to be able to start up the system
- There is no “correct solution”
  - Somewhat arbitrary collection of concepts

Possible Initial Domain Model

- Just the conceptual classes
- May evolve as more scenarios are explored



Register

Store

Item

Sale

Customer

Cashier

Manager

Payment

Product Specification

Product Catalog

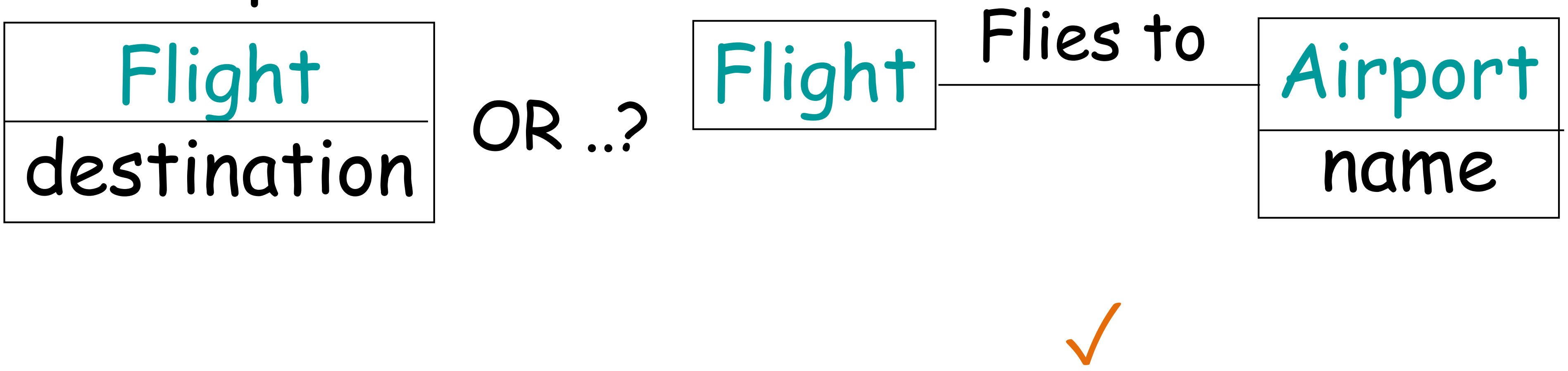
Sales LineItem

Step 2: Decide Attributes

- Properties of the conceptual classes relevant to the problem domain
  - Nouns and noun phrases that the requirements suggest or imply a need to remember
  - E.g., description, price, item ID relevant to ProductSpecification
  - E.g., change, amount relevant to Receipt

A Common Mistake

- Example



Flight

destination

OR ..?

Flight

Flies to

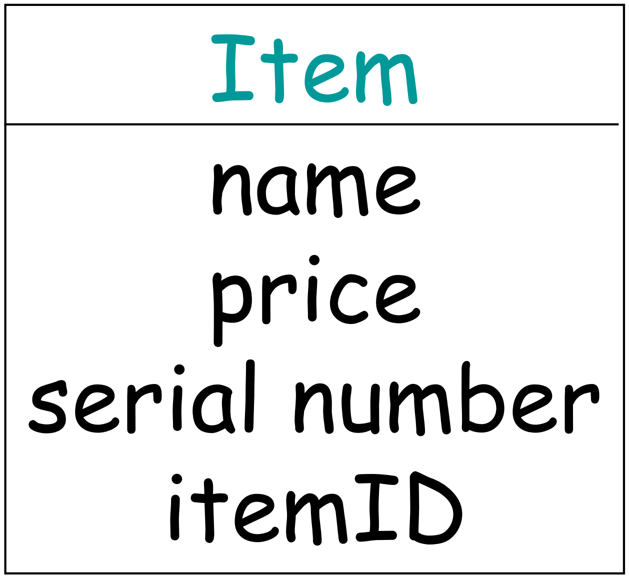
Airport

name

✓

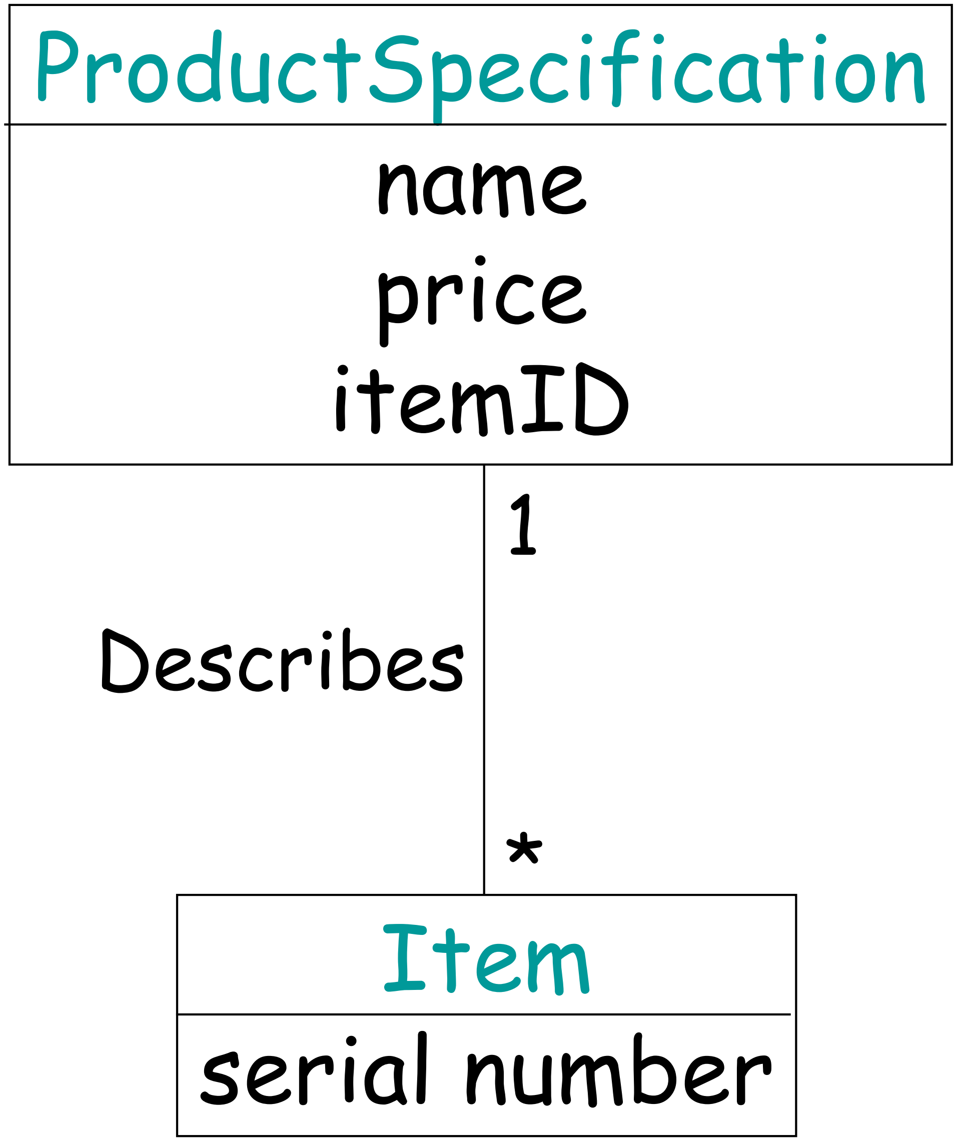
"If we do not think of some conceptual class X as a number or text in the real world, X is probably a conceptual class, not an attribute." [Larman p. 146]

Which Alternative Is Better?



Item

name
price
serial number
itemID



ProductSpecification

name
price
itemID

1

Describes

*

Item

serial number

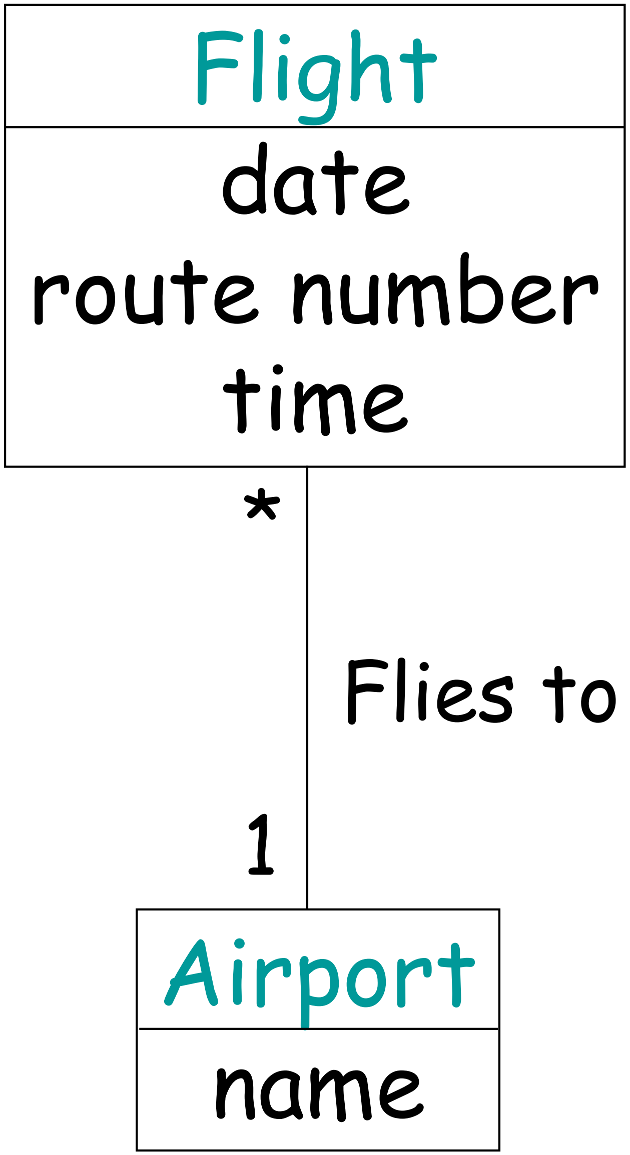
Description Class

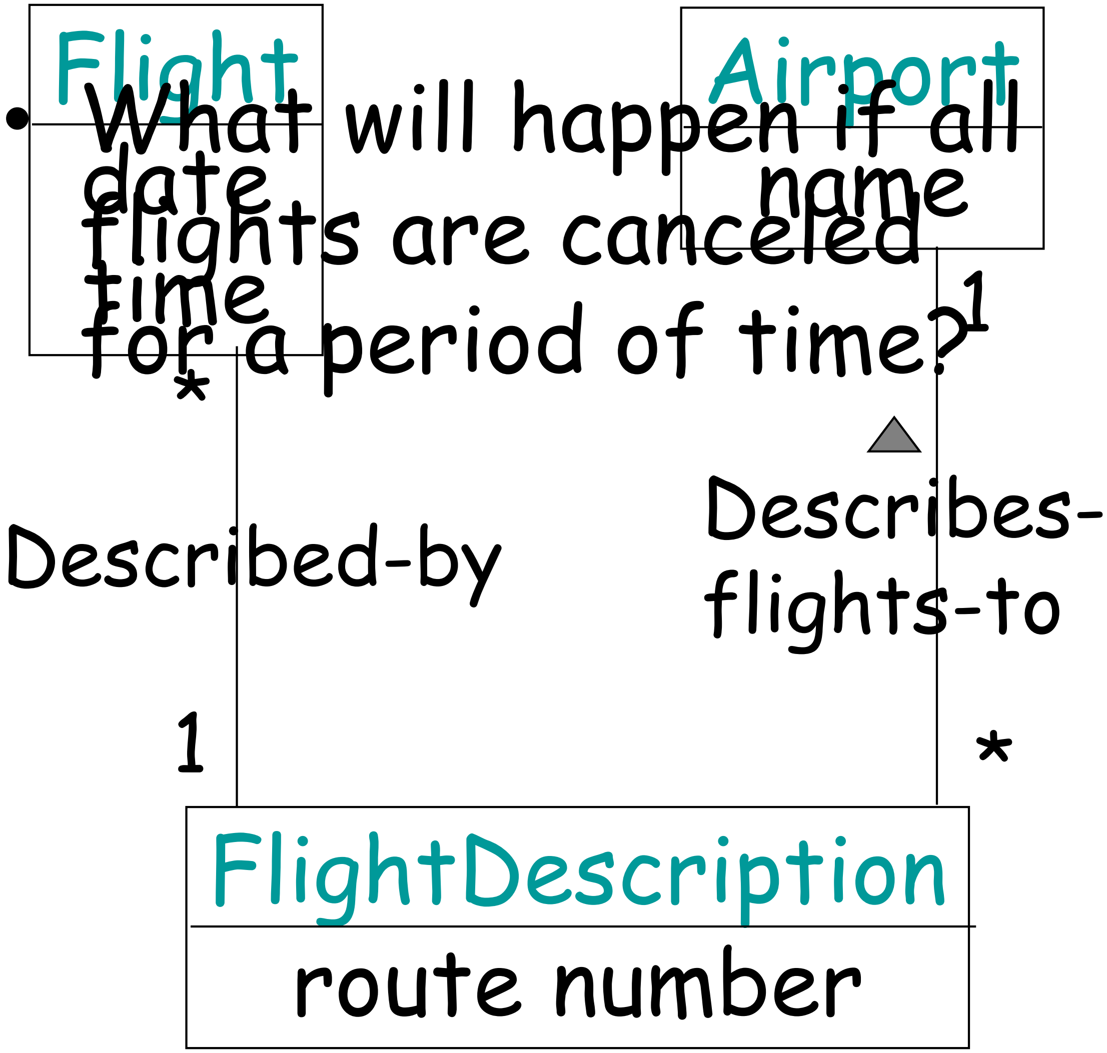
- Definition
  - It contains information that describes something else.
  - ProductDecription records the price, picture, and text description of an Item
- When do we need it?

We need a **Description Class** instead of **attributes** for a thing when

- The description exists independently of the current existence of the thing
  - Deleting things will not cause description loss
  - Adding things will not cause description redundancy

Another Example





- What will happen if all flights are canceled for a period of time?

Step 3: Identify Associations

- Relationship between instances of conceptual classes
- Think of it as a mathematical relation
  - Typically a binary relation: $R \subseteq S1 \times S2$
  - S1 = set of instances of the first class
  - S2 = set of instances of the second class

Typical Associations

- A is a physical/logical part of B
  - Wing-Airplane, SalesLineItem-Sale, FlightLeg-FlightRoute, Finger-Hand
- A is physically/logically contained in B
  - Item-Shelf, Passenger-Airplane, Flight-FlightSchedule
- A is recorded/reported/captured in B
  - Sale-Register, Reservation-FlightManifest
- A is a description of B
  - ProductSpecification-Item

Typical Associations

- A uses or manages B
  - Cashier-Register, Pilot-Airplane
- A is related to a transaction B
  - Customer-Payment, Payment-Sale, Reservation-Cancellation
- A is owned by B
  - Airplane-Airline

Finding Associations

- Consider the typical categories
  - Larman, Ch 9 p 155
- Focus on associations that are relevant with respect to the use cases
  - Don't create too many associations - common problem

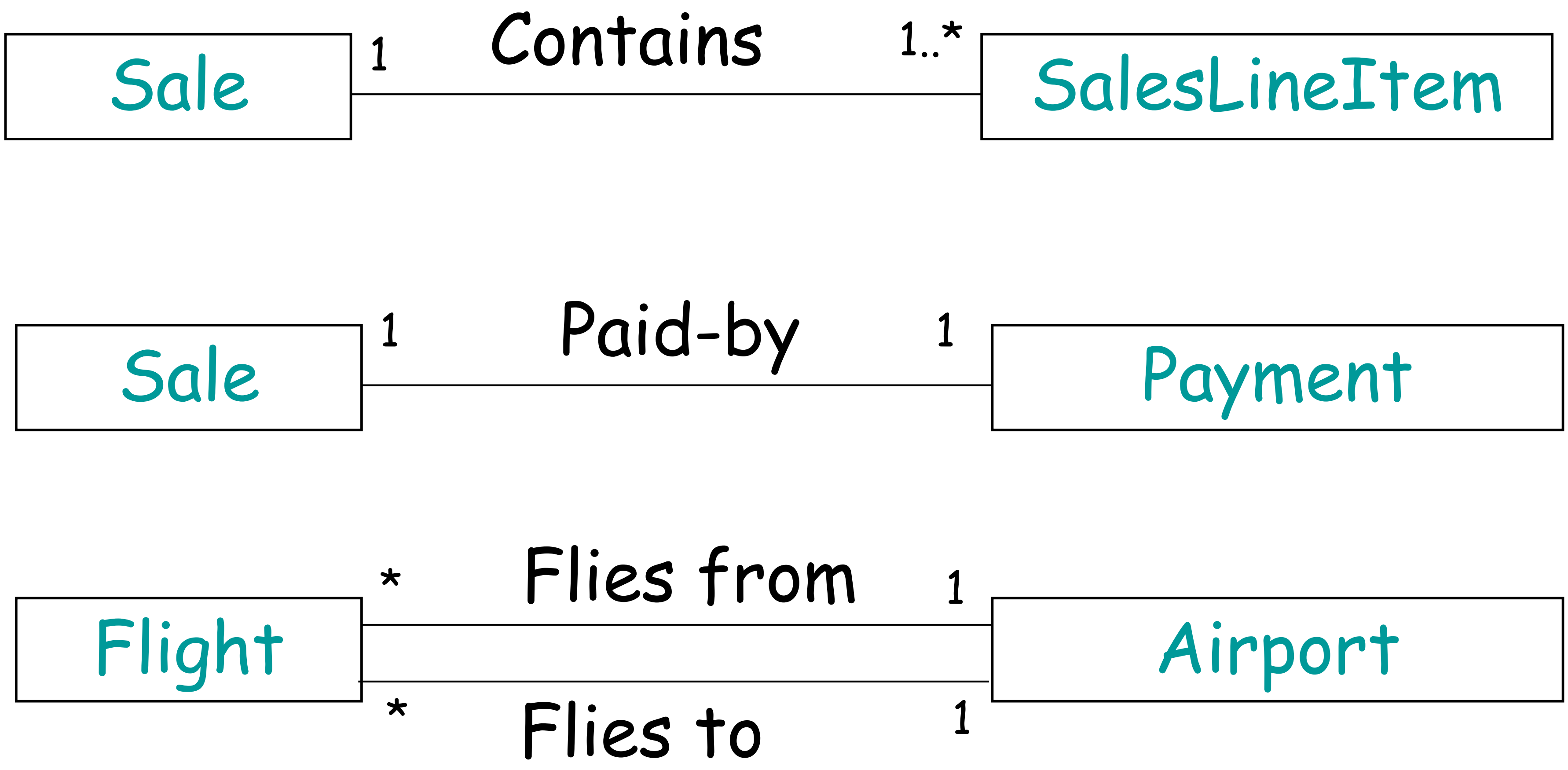
Multiplicity

- Range: x..y
- Common notation for ranges
  - x..x -> x
  - x..infinity -> x..*
  - 0..infinity -> *
- Combination of ranges
  - x..y, z..w
  - e.g. "2,4" -> number of doors in a car
- Most common multiplicities: *, 1..*, 0..1, 1

Association Examples

- SalesLineItem-Sale
  - A sale contains lines of sale items
- Payment-Sale
  - A payment is always related to a sale
- Flight-Airport
  - A flight flies from an airport and to another airport

Domain Models



Sale 1 Contains 1..* SalesLineItem

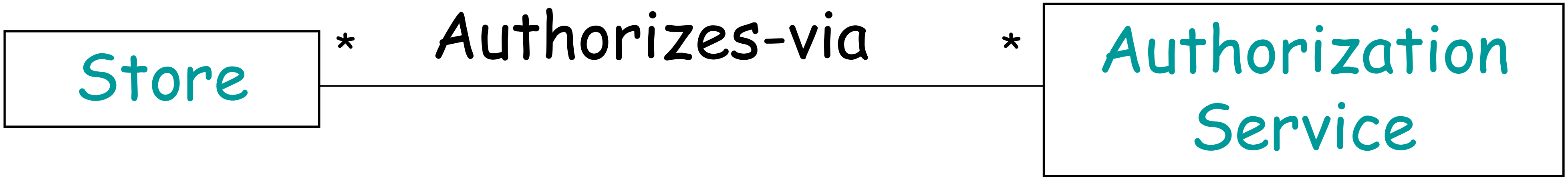
Sale 1 Paid-by 1 Payment

Flight * Flies from 1 Airport

Flight * Flies to 1 Airport

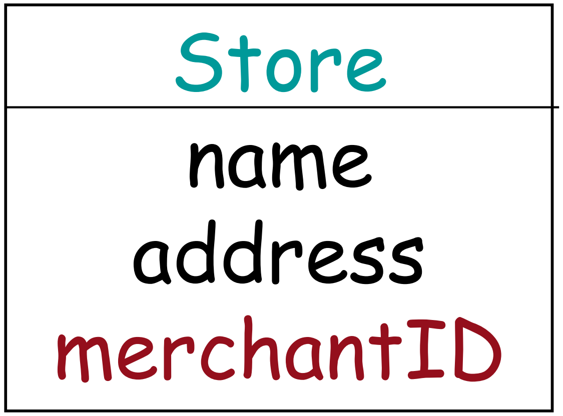
A Complicated Example

- A store uses a set of external authorization services for payments



- Each service associates a merchant ID with the store
  - For each service, different stores have different mechant IDs
  - Each store has different mechant IDs for different services

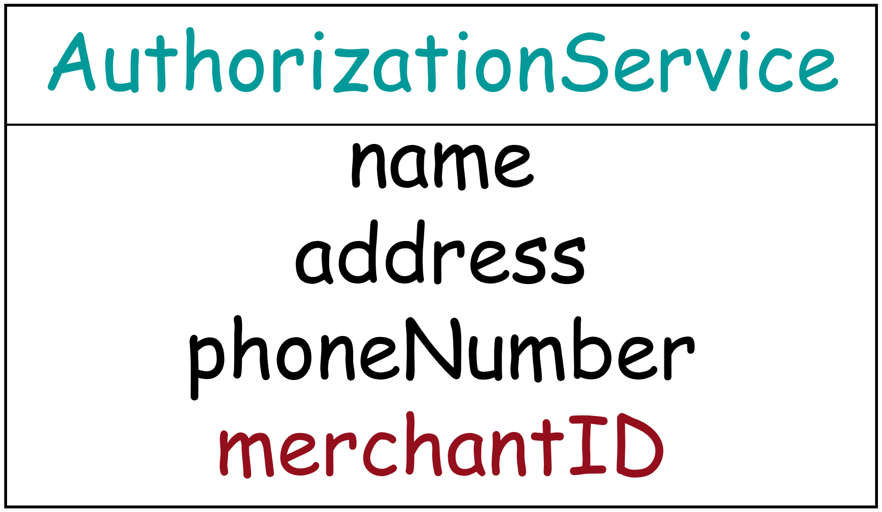
Where Should the merchantID Be Located?



Store

name
address
merchantID

Option 1



AuthorizationService

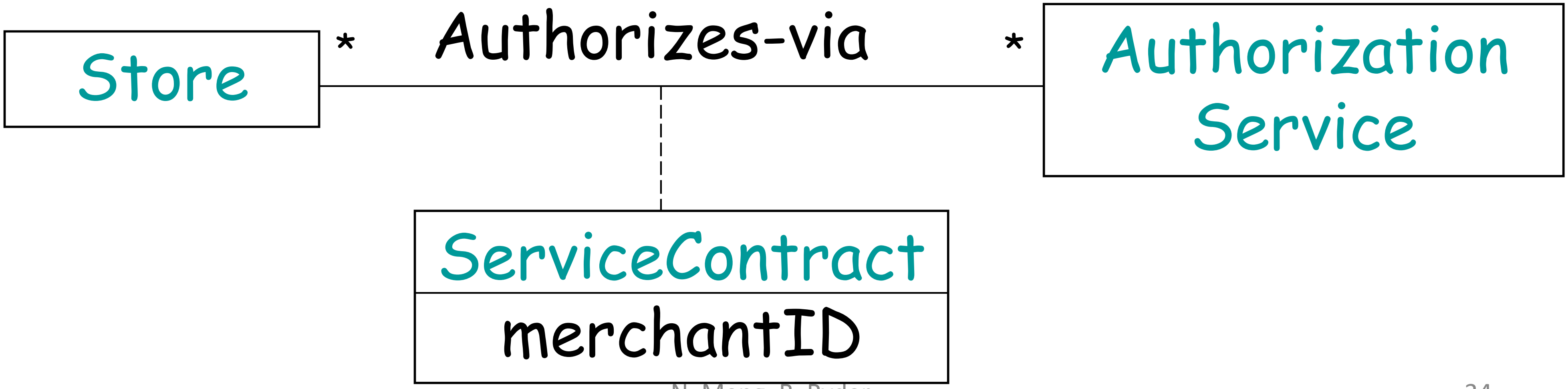
name
address
phoneNumber
merchantID

Option 2

Neither

Association Class

- merchantID is conceptually related to the association, not to either Store or Service
- Solution: association class to hold attributes of the association

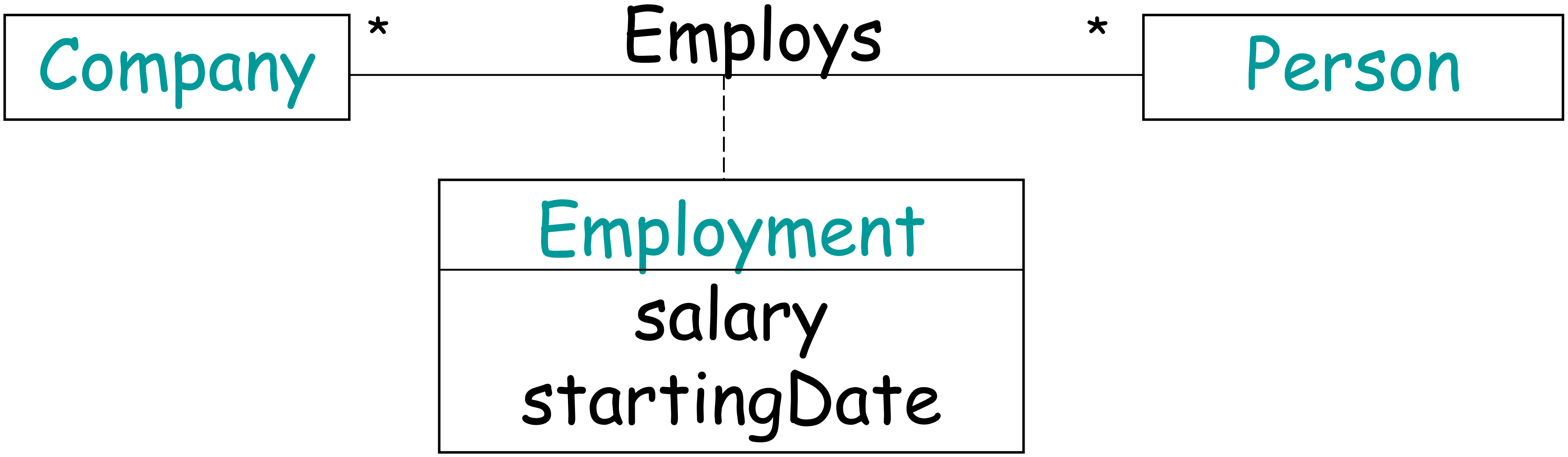


When to Use Association Classes?

- When an attribute “doesn't fit” in the classes participating in an association
- When the lifetime of the attribute depends on the lifetime of the association
- Often used with many-to-many associations

Many-to-Many Association

- A company may employ several persons
- A person may be employed by several companies
- Attributes: salary, starting date, …


Company
*
Employs
*
Person
Employment
salary
startingDate