Larman
Chapter 9

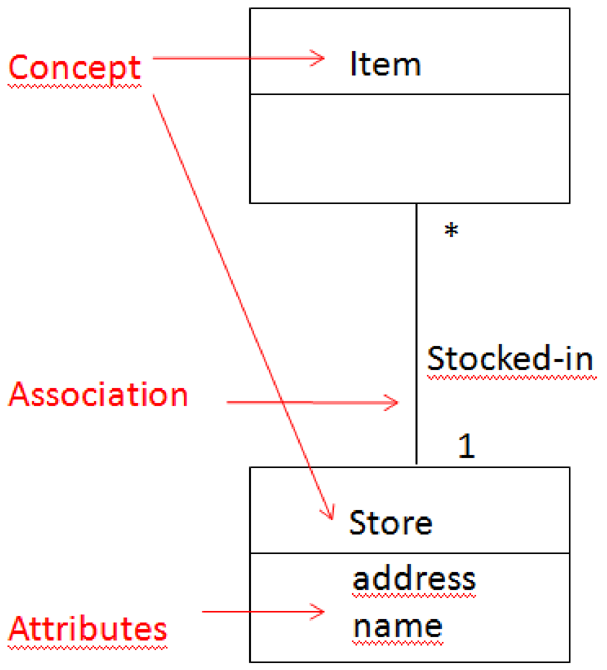
Elaboration-Iteration 1 basic

Domain Model

Domain Models

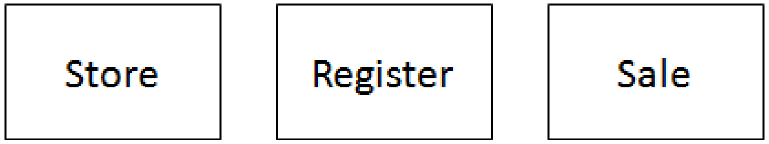
- A Domain Model illustrates meaningful concepts in a problem domain.
- A Domain Model has conceptual classes
- It is a representation of real-world things, not software components.
- It is a set of static structure diagrams; no operations are defined.
- It may show:
  - **concepts**
  - **associations between concepts**
  - **attributes of concepts**

Domain Models



- A Domain Model is a description of things in the real world.
- A Domain Model is not a description of the software design.
- A concept is an idea, thing, or object.

Conceptual Classes in the Sale Domain



Partial Domain Model.

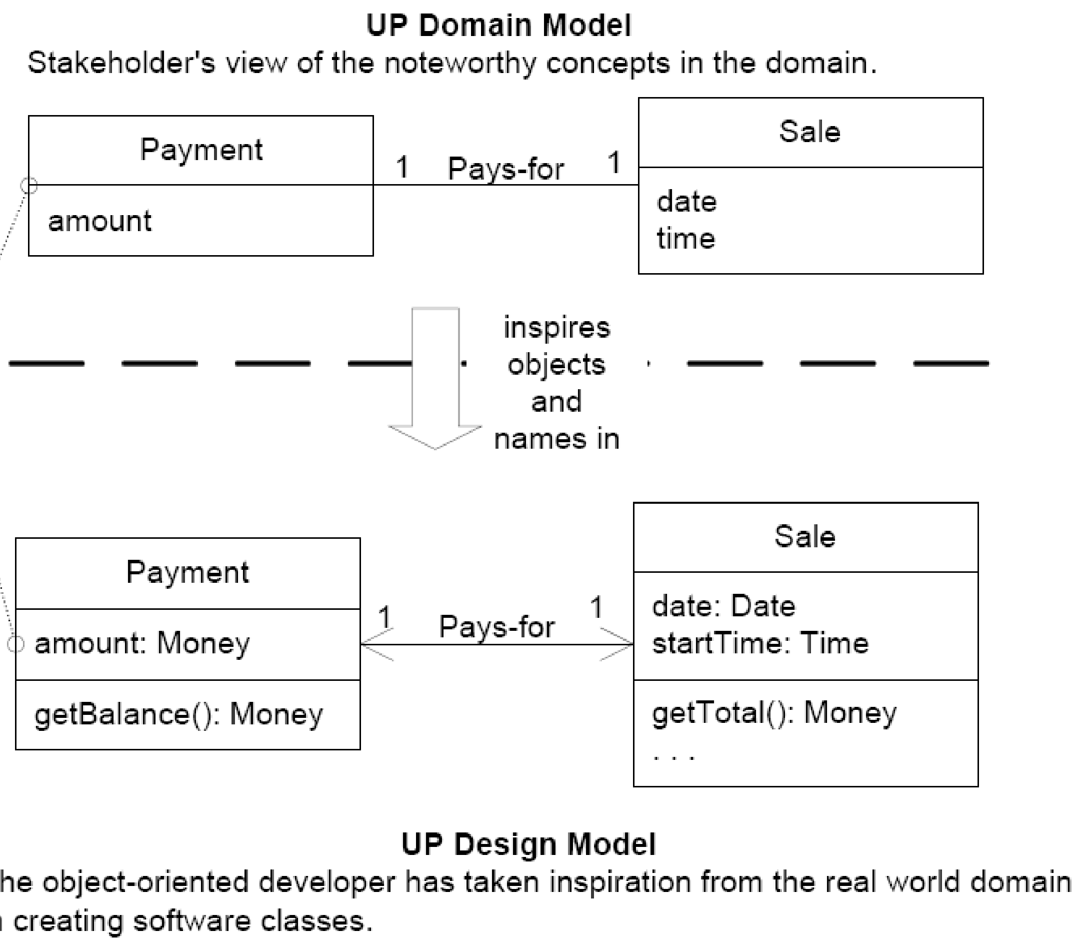
- A central distinction
- between object-oriented
- and structured analysis:
- division by concepts
- (objects) rather than
- division by functions.

Domain model versus Design model

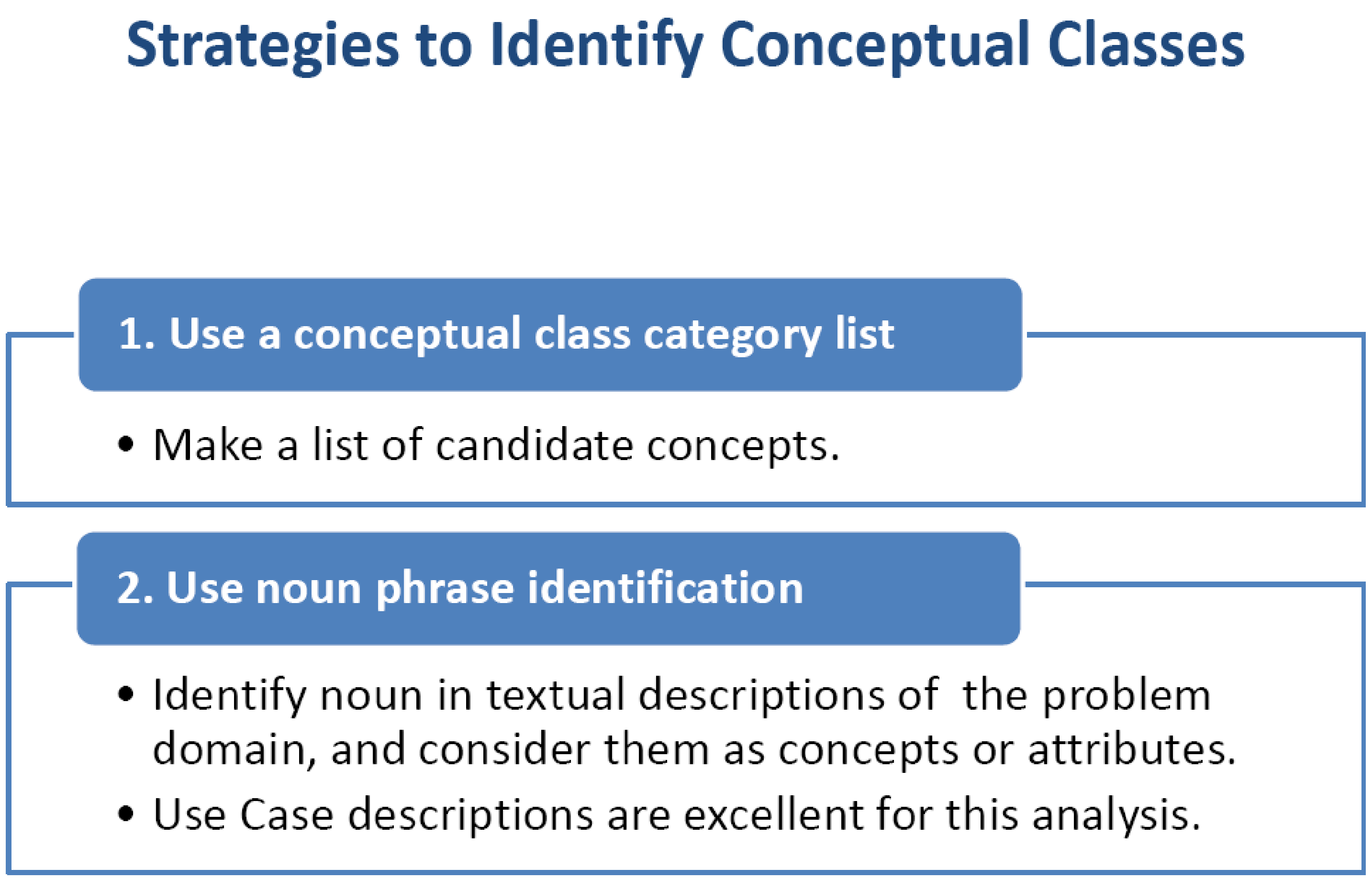
A Payment in the Domain Model is a concept, but a Payment in the Design Model is a software class. They are not the same thing, but the former *inspired* the naming and definition of the latter.

This reduces the representational gap.

This is one of the big ideas in object technology.


UP Domain Model
Stakeholder's view of the noteworthy concepts in the domain.
Payment
amount
1
Pays-for
1
Sale
date
time
inspires objects and names in
Payment
amount: Money
getBalance(): Money
1
Pays-for
1
Sale
date: Date
startTime: Time
getTotal(): Money
. . .
UP Design Model
The object-oriented developer has taken inspiration from the real world domain in creating software classes.

noun phrase - navneord



Strategies to Identify Conceptual Classes

1. Use a conceptual class category list

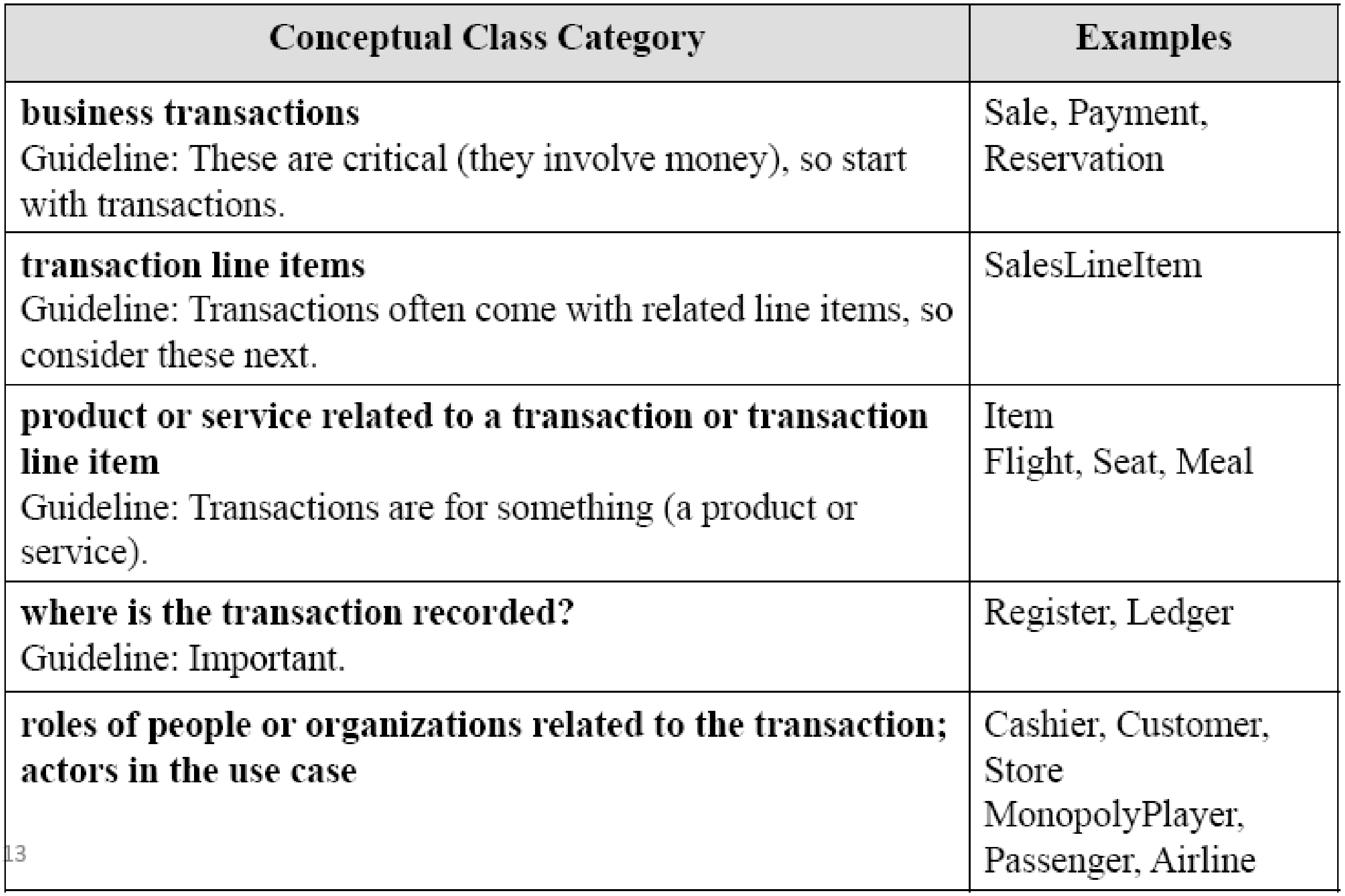
- Make a list of candidate concepts.

2. Use noun phrase identification

- Identify noun in textual descriptions of the problem domain, and consider them as concepts or attributes.
- Use Case descriptions are excellent for this analysis.

Using a Category List

• Use a list of categories and see if they apply within your problem domain :



Conceptual Class Category	Examples
business transactions Guideline: These are critical (they involve money), so start with transactions.	Sale, Payment, Reservation
transaction line items Guideline: Transactions often come with related line items, so consider these next.	SalesLineItem
product or service related to a transaction or transaction line item Guideline: Transactions are for something (a product or service).	Item Flight, Seat, Meal
where is the transaction recorded? Guideline: Important.	Register, Ledger
roles of people or organizations related to the transaction; actors in the use case 13	Cashier, Customer, Store MonopolyPlayer, Passenger, Airline

Continued …

Conceptual Class Category	Examples
place of transaction; place of service	Store,Airport, Plane, Seat
noteworthy events, often with a time or place we need to remember	Sale, Payment ,MonopolyGame ,Flight
physical objects Guideline: This is especially relevant when creating device-control software, or simulations.	Item, Register Board, Piece, Die Airplane
descriptions of things	ProductDescription FlightDescription
catalogs Guideline: Descriptions are often in a catalog.	ProductCatalog FlightCatalog
containers of things (physical or information)	Store, Board, Airplane
things in a container	Item Square (in a Board) Passenger
other collaborating systems	CreditAuthorizationSystem AirTrafficControl

Continued …

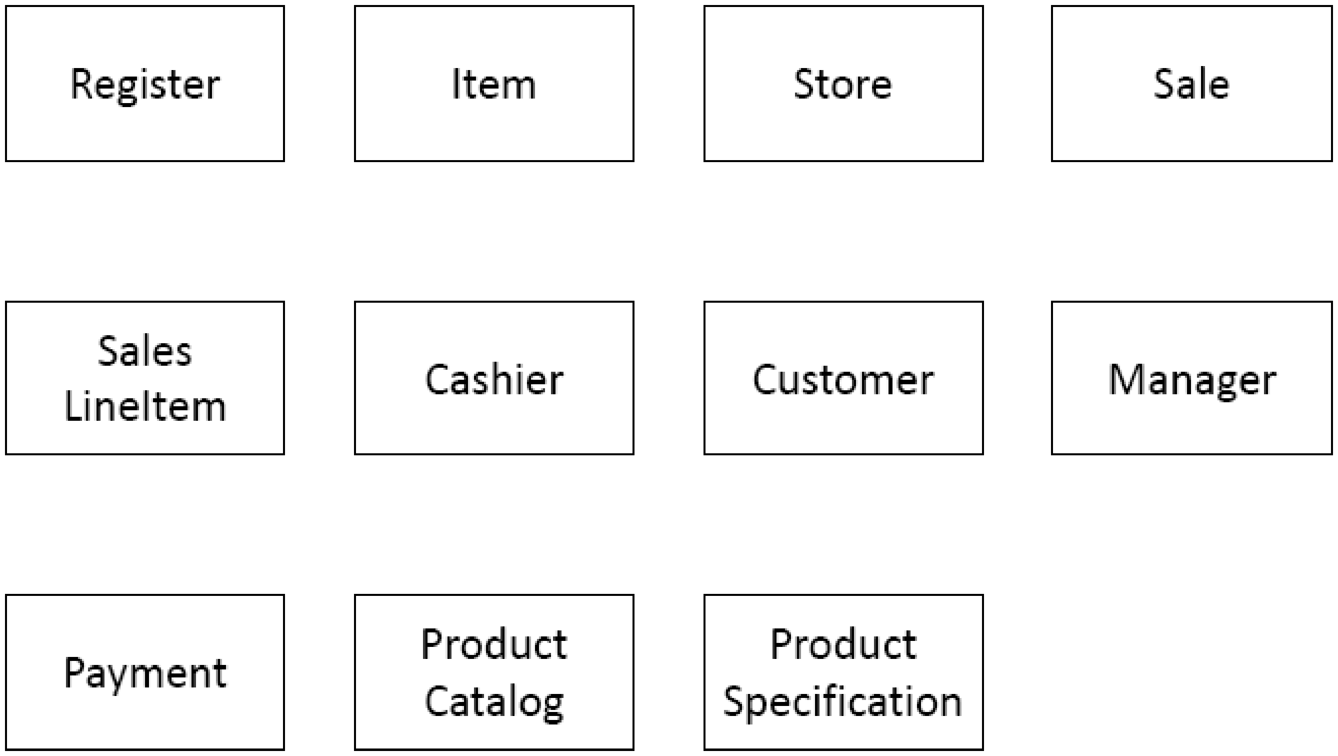
Conceptual Class Category	**Examples**
records of finance, work, contracts, legal matters	Receipt, Ledger MaintenanceLog
financial instruments	Cash, Check, LineOfCredit TicketCredit
schedules, manuals, documents that are regularly referred to in order to perform work	DailyPriceChangeList, RepairSchedule

Finding Conceptual Classes with Noun Phrase Identification

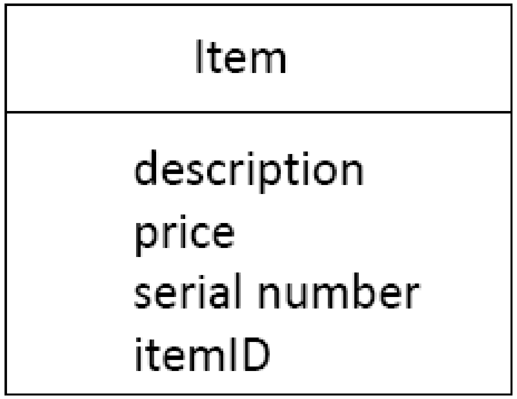
- 1. This use case begins when a **Customer** arrives at a **Register checkout** with items to purchase.
- 2. The **Cashier** starts a new sale.
- 3. **Cashier** enters **item ID**.
- …

- The fully dressed Use Cases are useful for this analysis.
- Some of these noun phrases are candidate concepts; some may be attributes of concepts.
- A mechanical noun-to-concept mapping is not possible, as words in a natural language are (sometimes) ambiguous.

The NextGen POS (partial) Domain Model

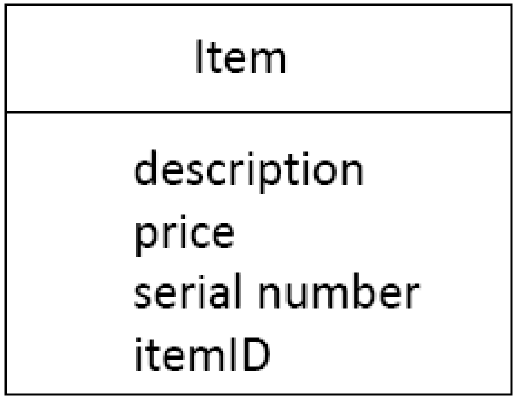


The Need for Specification or Description Conceptual Classes



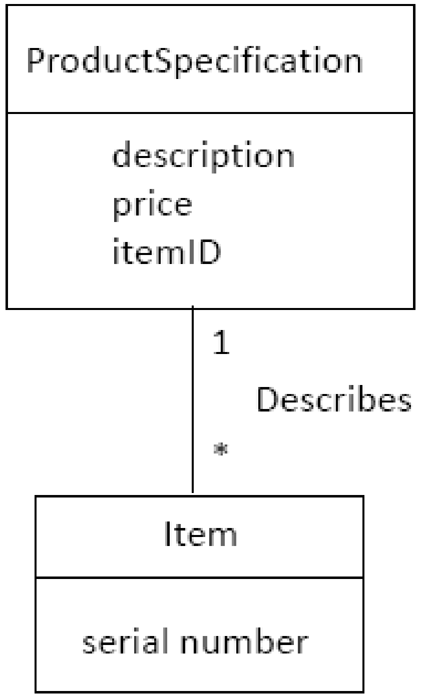
- What is wrong with this picture?
- Consider the case where all items are sold, and thus deleted from the computer memory.
- How much does an item cost?

The Need for Specification or Description Conceptual Classes



- The memory of the item's price was attached to inventoried instances, which were deleted
- Notice also that in this model there is duplicated data (description, price, itemID).

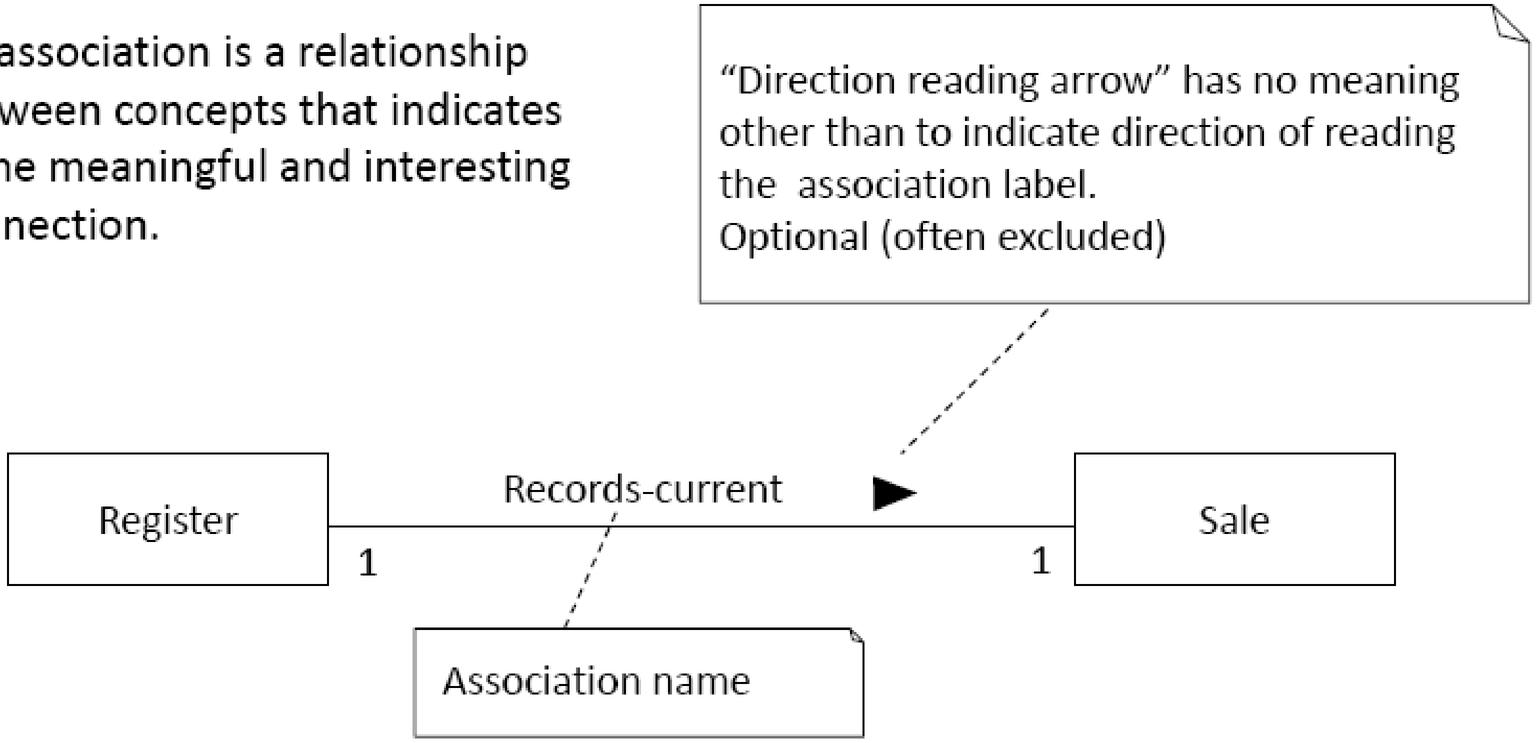
The Need for Specification or Description Conceptual Classes



- Add a specification or description concept when:
  - Deleting instances of things they describe results in a loss of information
  - It reduces redundant or duplicated information.

Adding Associations

An association is a relationship between concepts that indicates some meaningful and interesting connection.

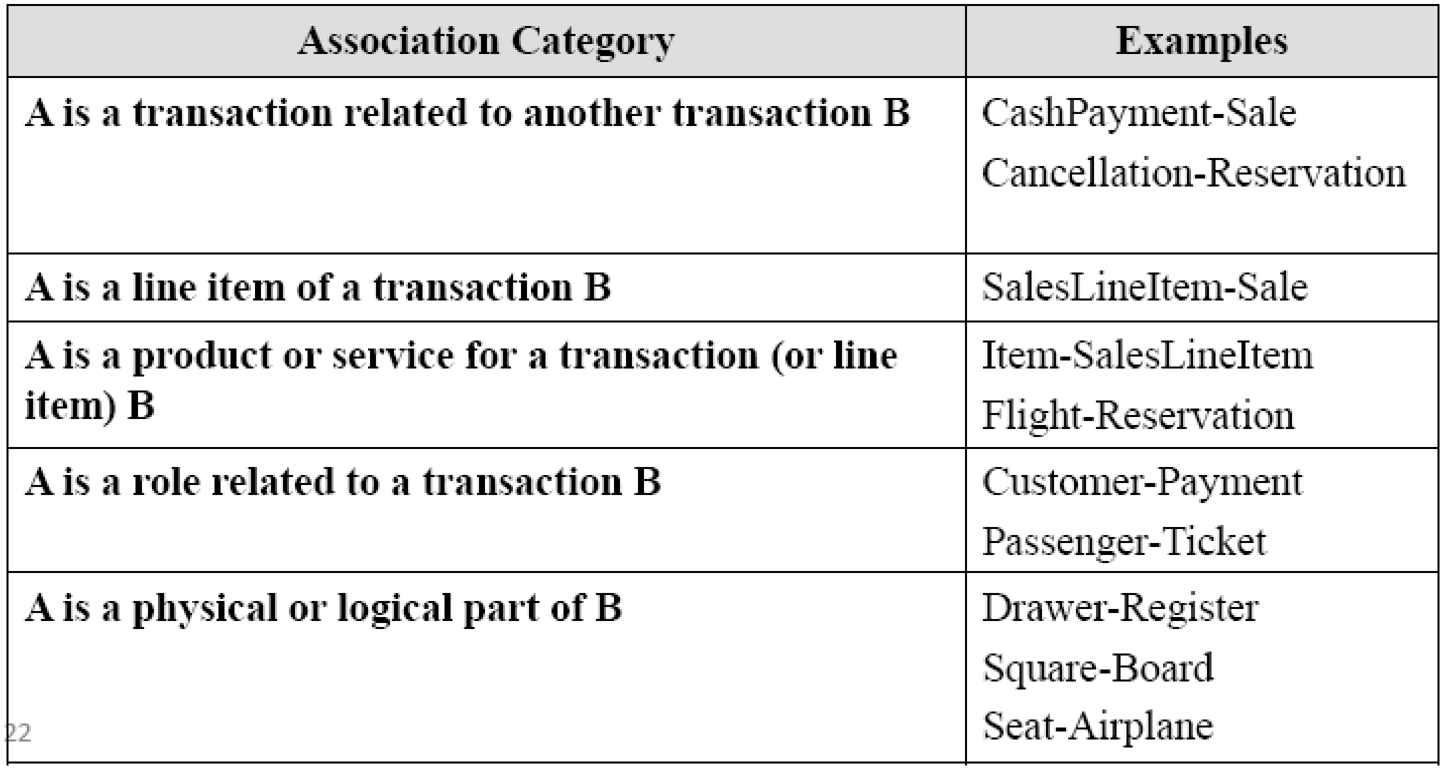

"Direction reading arrow" has no meaning other than to indicate direction of reading the association label.
Optional (often excluded)
Register
Records-current ▶
Sale
1
1
Association name

How to Find Associations?

Two main ways:

1. By reading the current, relevant, requirements and asking ourselves what information is needed to fulfil these requirements: what need to know associations are necessary given our current list of candidate concepts?

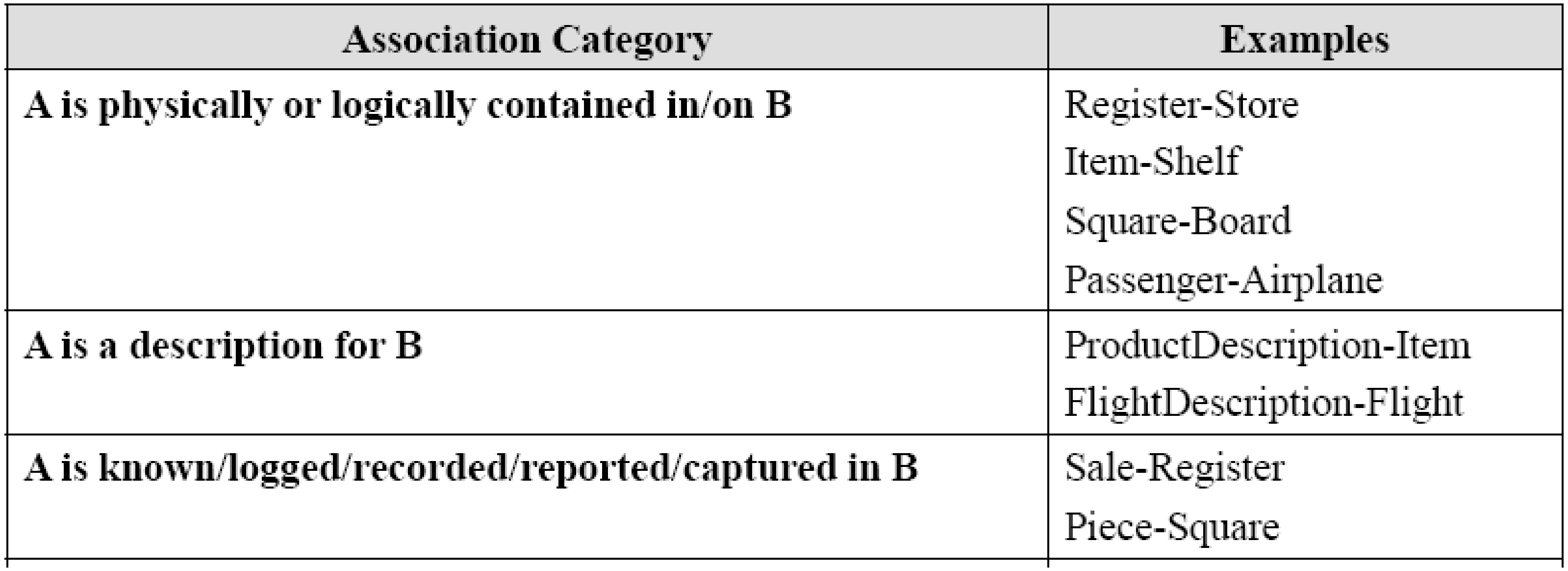
2. Using a list of association categories.



Association Category	Examples
A is a transaction related to another transaction B	CashPayment-Sale Cancellation-Reservation
A is a line item of a transaction B	SalesLineItem-Sale
A is a product or service for a transaction (or line item) B	Item-SalesLineItem Flight-Reservation
A is a role related to a transaction B	Customer-Payment Passenger-Ticket
A is a physical or logical part of B	Drawer-Register Square-Board Seat-Airplane

22

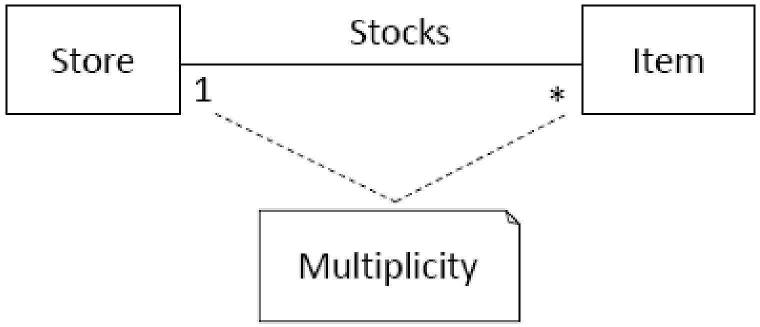
Continued …



Association Category	Examples
A is physically or logically contained in/on B	Register-Store Item-Shelf Square-Board Passenger-Airplane
A is a description for B	ProductDescription-Item FlightDescription-Flight
A is known/logged/recorded/reported/captured in B	Sale-Register Piece-Square

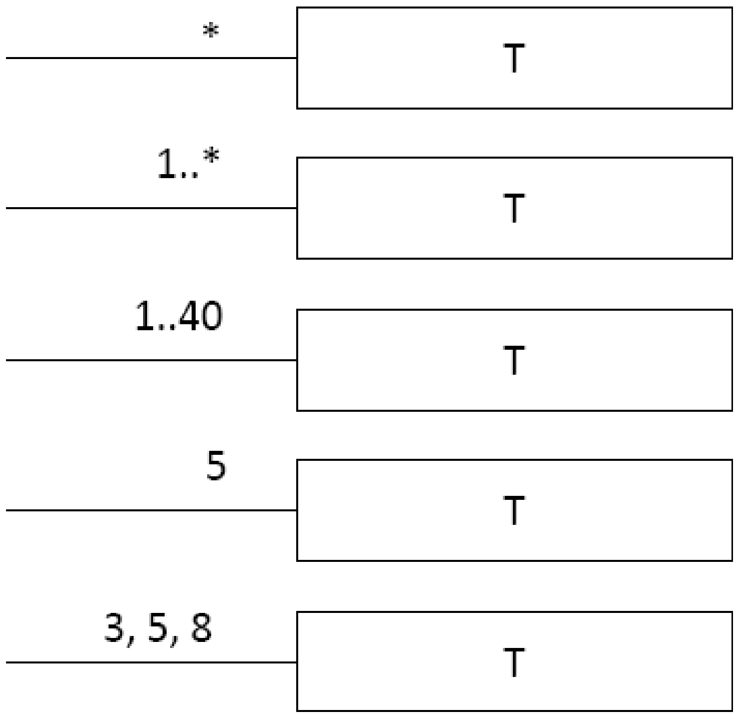
See complete list in Larman 3rd. ed., pp. 155-156

Multiplicity



- Multiplicity defines how many instances of a type A can be associated with one instance of a type B, at a particular moment in time.
- For example, a single instance of a Store can be associated with "many" (zero or more) Item instances.

Multiplicity



Zero or more; "many"

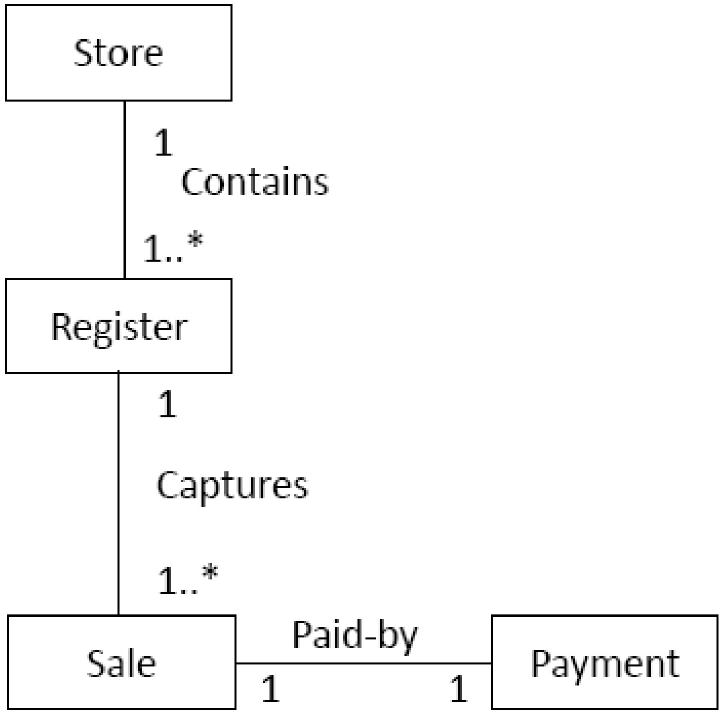
One or more

One to forty

Exactly five

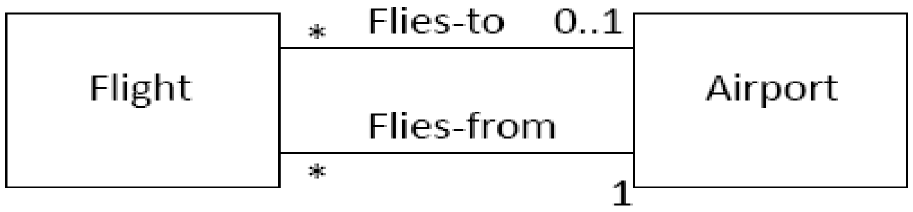
Exactly three, five or eight.

Naming Associations



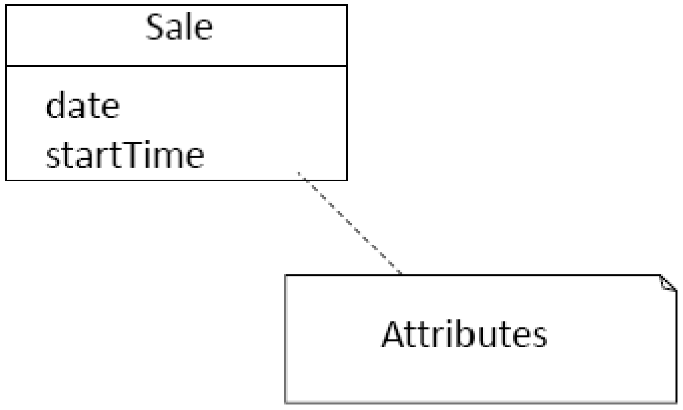
- Name an association based on a TypeName-VerbPhrase-TypeName format.
- Association names should start with a capital letter
- A verb phrase should be constructed with hyphens
- The default direction to read an association name is left to right, or top to bottom.

Multiple Associations Between Two Types



- It is not uncommon to have multiple associations between two types.
- In the example, not every flight is guaranteed to land at an airport.

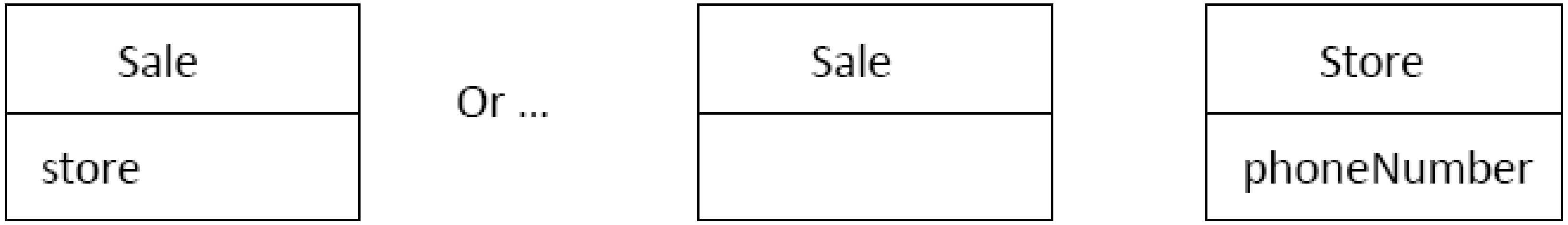
Adding Attributes



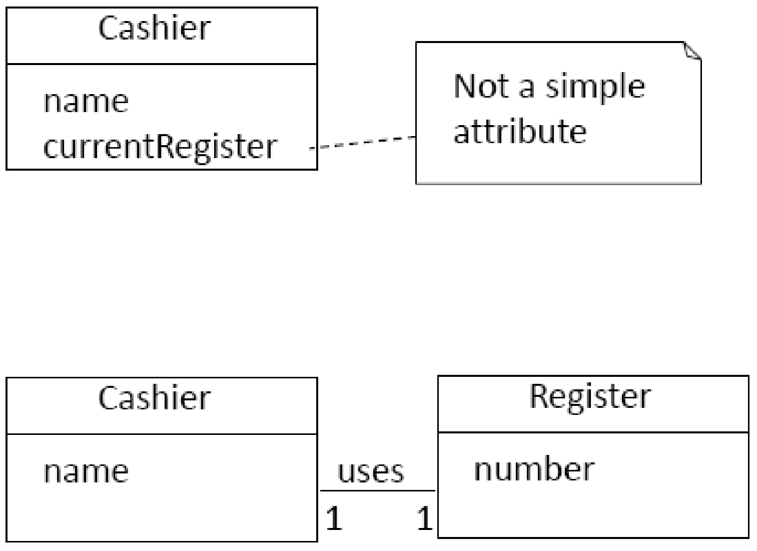
- An attribute is a logical data value of an object.
- Include the following attributes: those for which the requirements suggest or imply a need to remember information.
  - For example, a Sales receipt normally includes a date and time.
  - The Sale concept would need a date and time attribute.

Common mistake in a domain model:

- Representing something as an attribute when it should be a concept
- Guideline: if something is not a number or a string then it is probably a conceptual class, not an attribute.
- Here, since a store can have many interesting attributes (it is not a simple string) it should be made a separate concept.

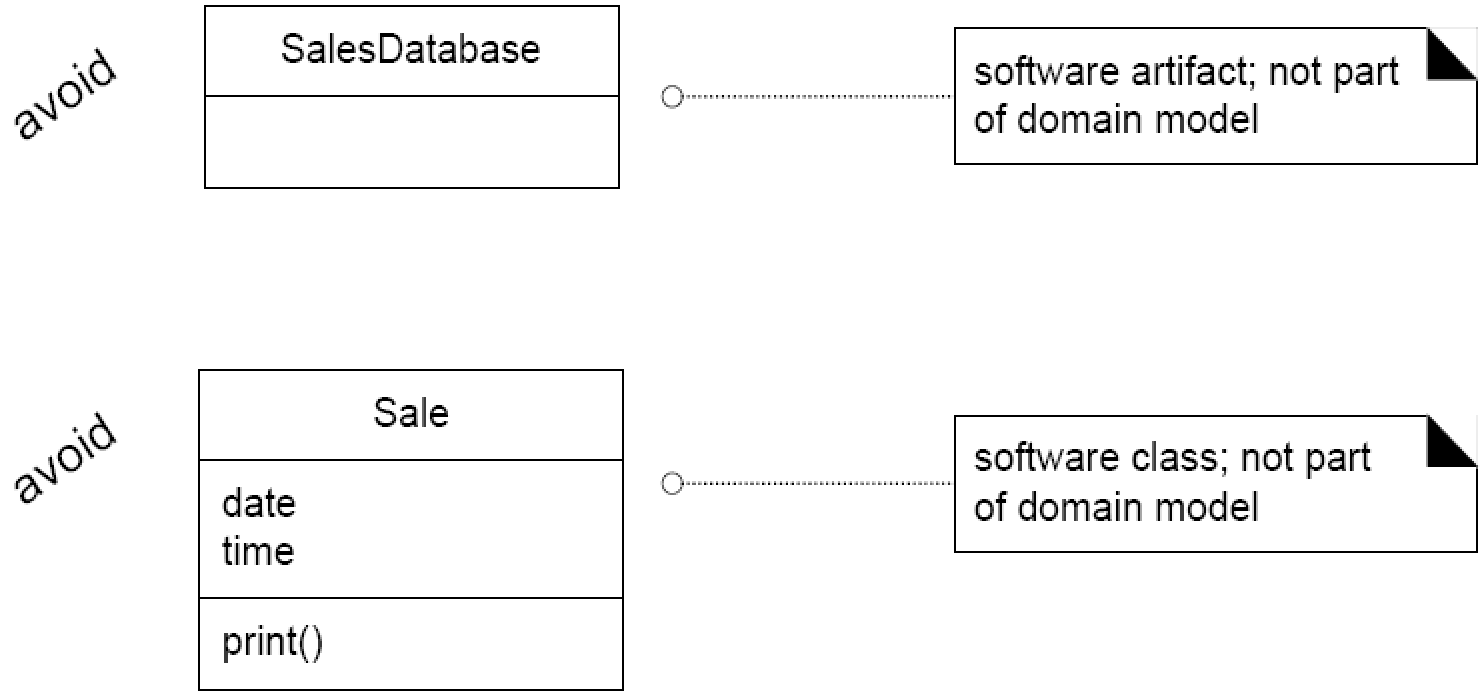


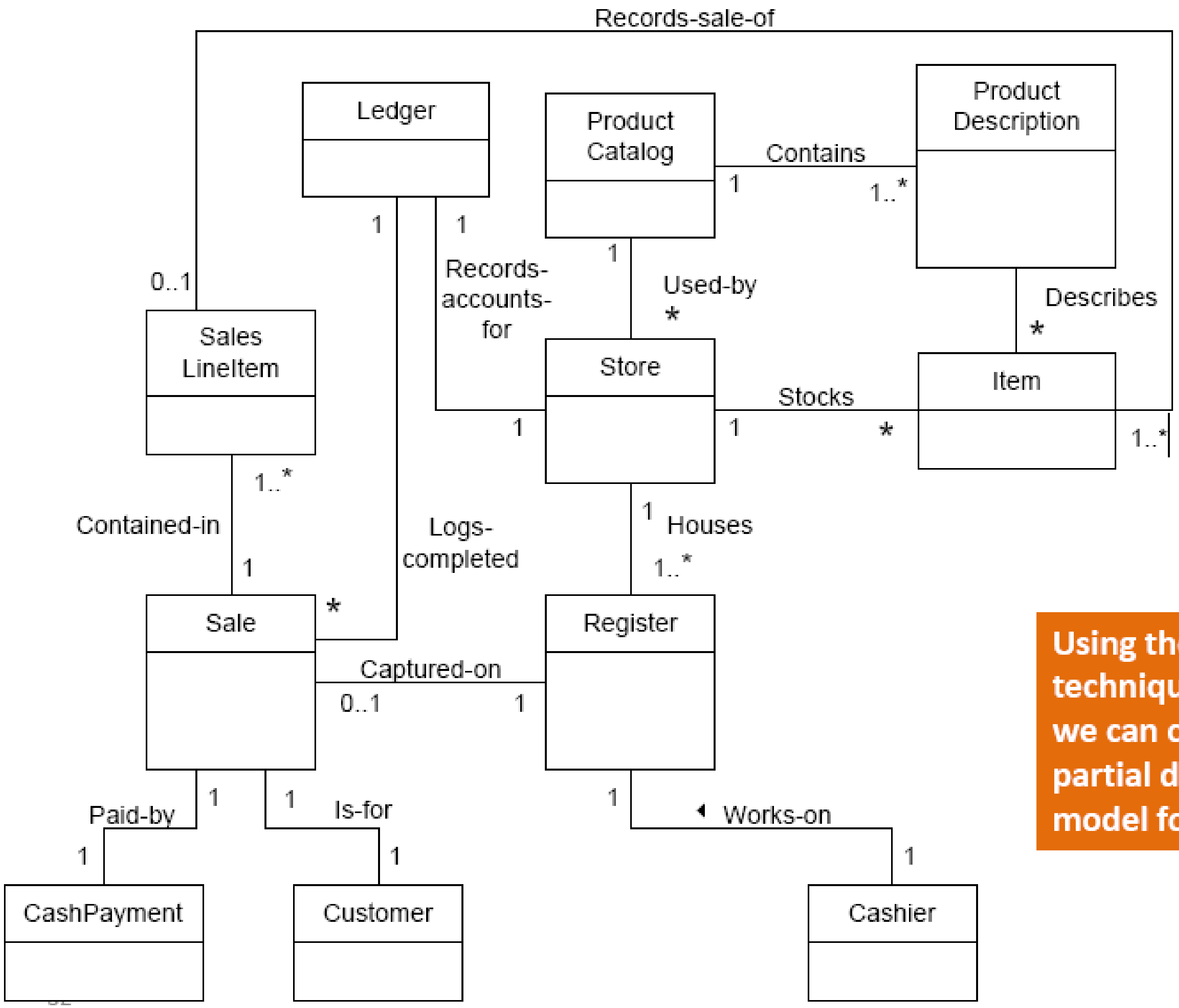
Valid Attribute Types



- Keep attributes simple.
- The type of an attribute should not normally be a complex domain concept, such as Sale or Airport.
- Attributes in a Domain Model should preferably be
  - Pure data values: Boolean, Date, Number, String, …
  - Simple attributes: color, phone number, zip code …

- Another common mistake is to include a *database* concept: whether some of the information will be held in a database is a design decision. It is wrong to include it in a domain model.


avoid
SalesDatabase
software artifact; not part of domain model
avoid
Sale
date
time
print()
software class; not part of domain model



Using the techniques seen we can create a partial domain model for the POS.