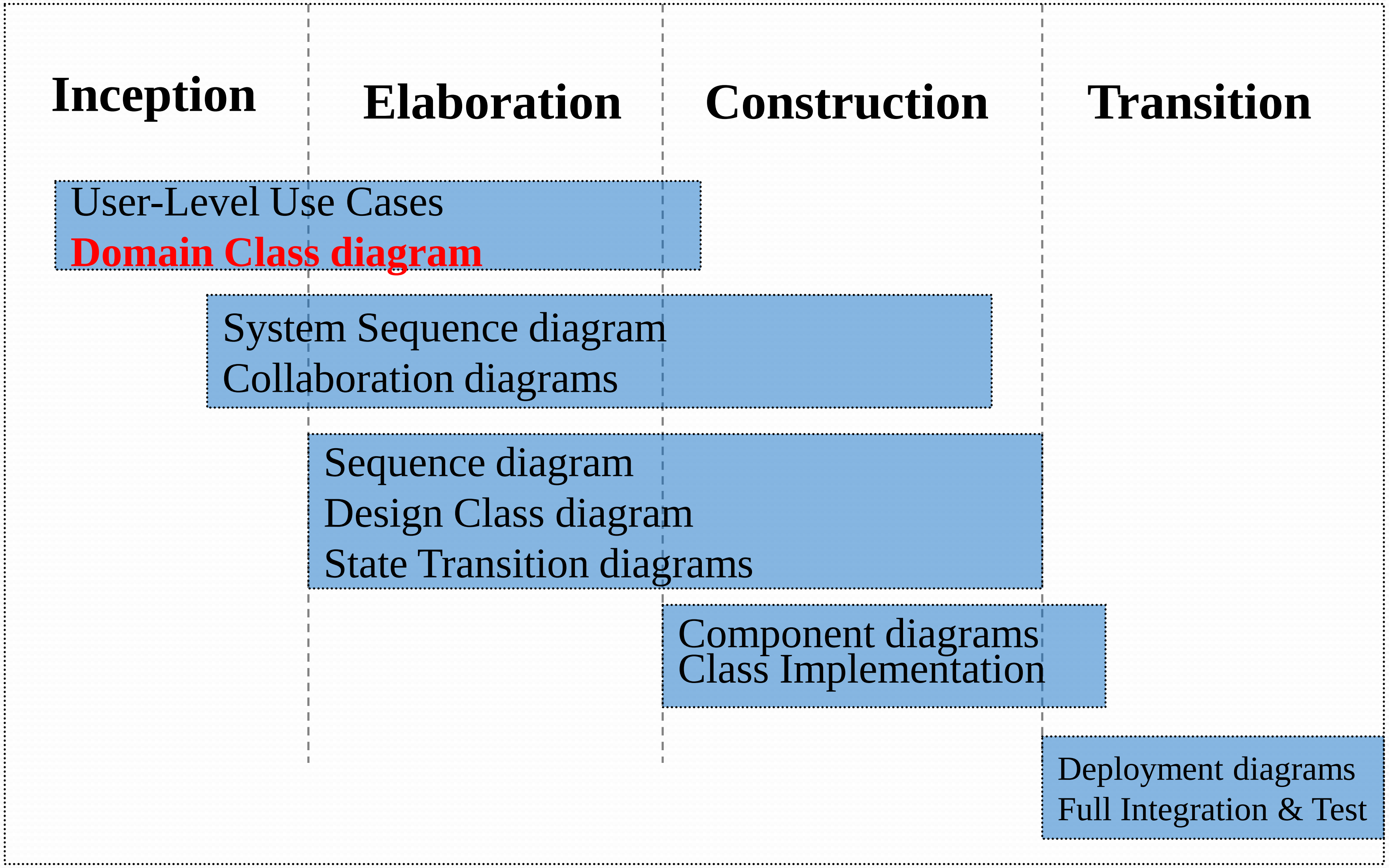
COMP 3711

(OOA and OOD)

Domain Model

Larman Chapter 9

UML And UP


Inception
Elaboration
Construction
Transition
User-Level Use Cases
Domain Class diagram
System Sequence diagram
Collaboration diagrams
Sequence diagram
Design Class diagram
State Transition diagrams
Component diagrams
Class Implementation
Deployment diagrams
Full Integration & Test

Domain Model

What is a Domain Model?

- A visual representation of conceptual classes or real-world objects in a domain of interest.
- A domain model is a representation of real world conceptual classes, not of software components.

Domain and Data Models

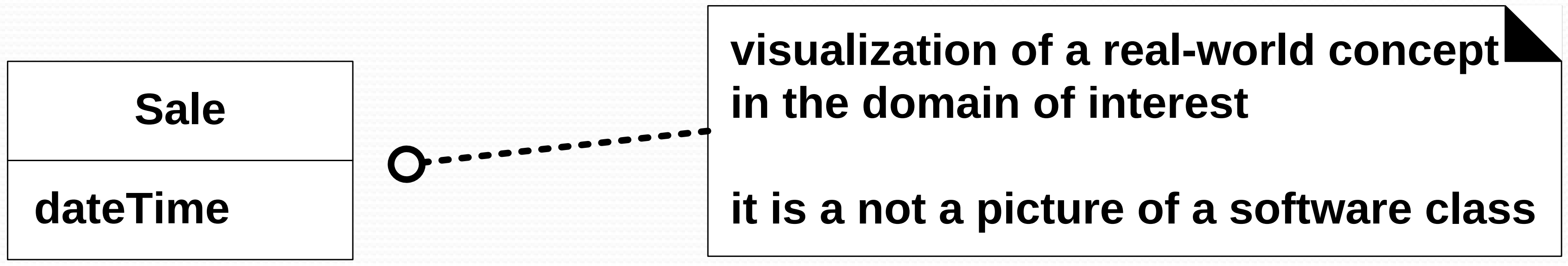
- A domain model is not a data model
  - Data model : persistent data to be stored somewhere
  - Conceptual class information doesn't necessarily need to be remembered
  - Conceptual class doesn't require attributes
  - Conceptual classes can have a purely behavioural role in the domain instead of an information role

UML - Domain Model

- Using UML notation, a Domain Model is illustrated using a set of class diagrams with:
  - Domain objects or Conceptual classes
  - Associations between conceptual classes
  - Attributes of conceptual classes
  - No operations (methods)

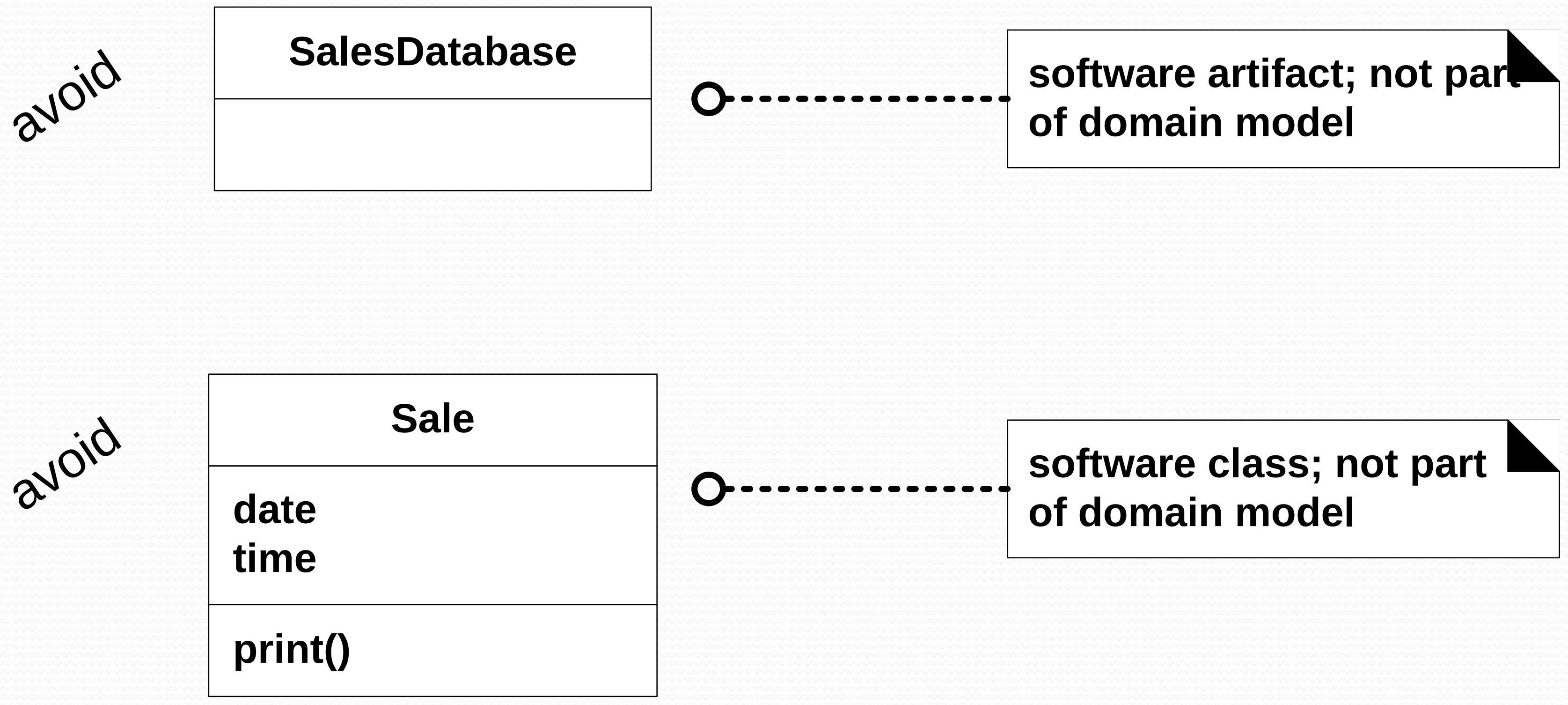
Real-World Perspective

Domain model is a visualization of things in the real world domain.


Sale
dateTime
visualization of a real-world concept in the domain of interest
it is a not a picture of a software class

Not A Software Perspective

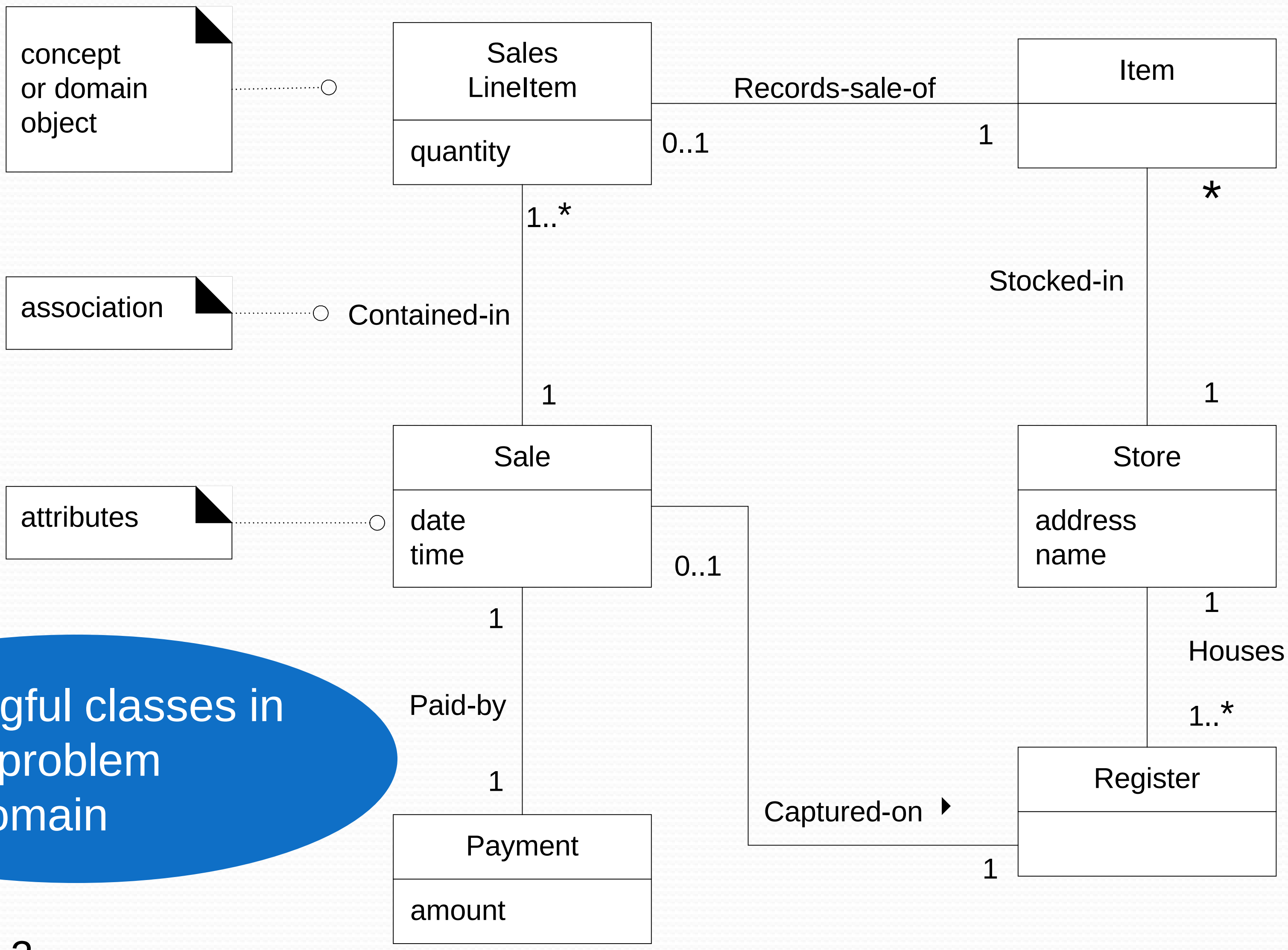
Domain model is not of software components such as Java & C++.


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

Conceptual Model

- Domain Models are also known as conceptual models, domain object models, or analysis object models.
- A visual dictionary of the noteworthy abstractions (conceptual classes), domain vocabulary, and information content of the domain.

A Conceptual Perspective



Meaningful classes in a problem domain

Larman Fig 9.2

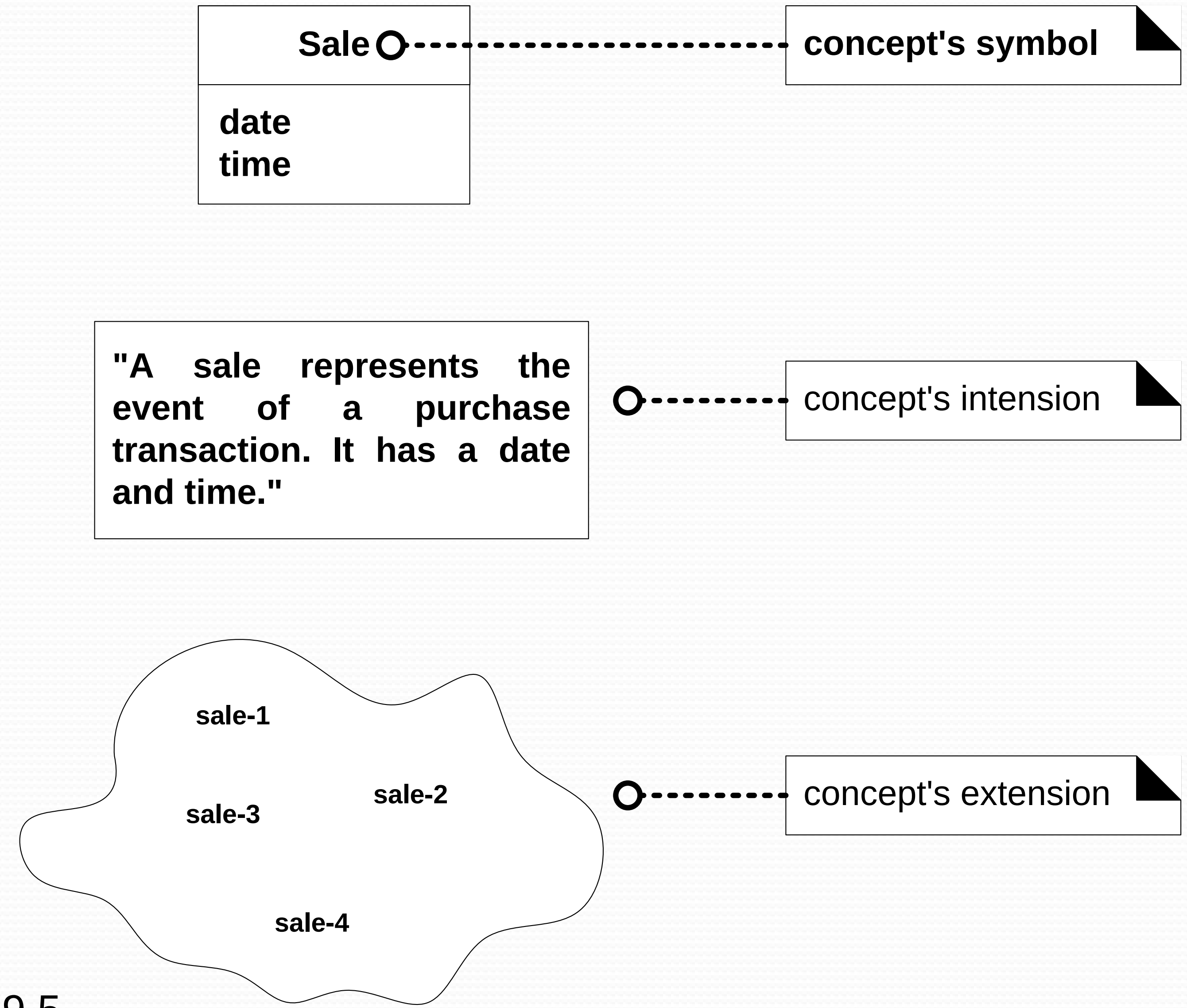
Conceptual Class

- Symbol
  - Word or images representing a conceptual class
- Intension
  - Definition of a conceptual class
- Extension
  - Examples to which the conceptual class applies

Example - Sales Transaction

- Symbol → Sales
- Intension → event of the sales transaction
- Extension → all the instances of sales

A Conceptual Perspective

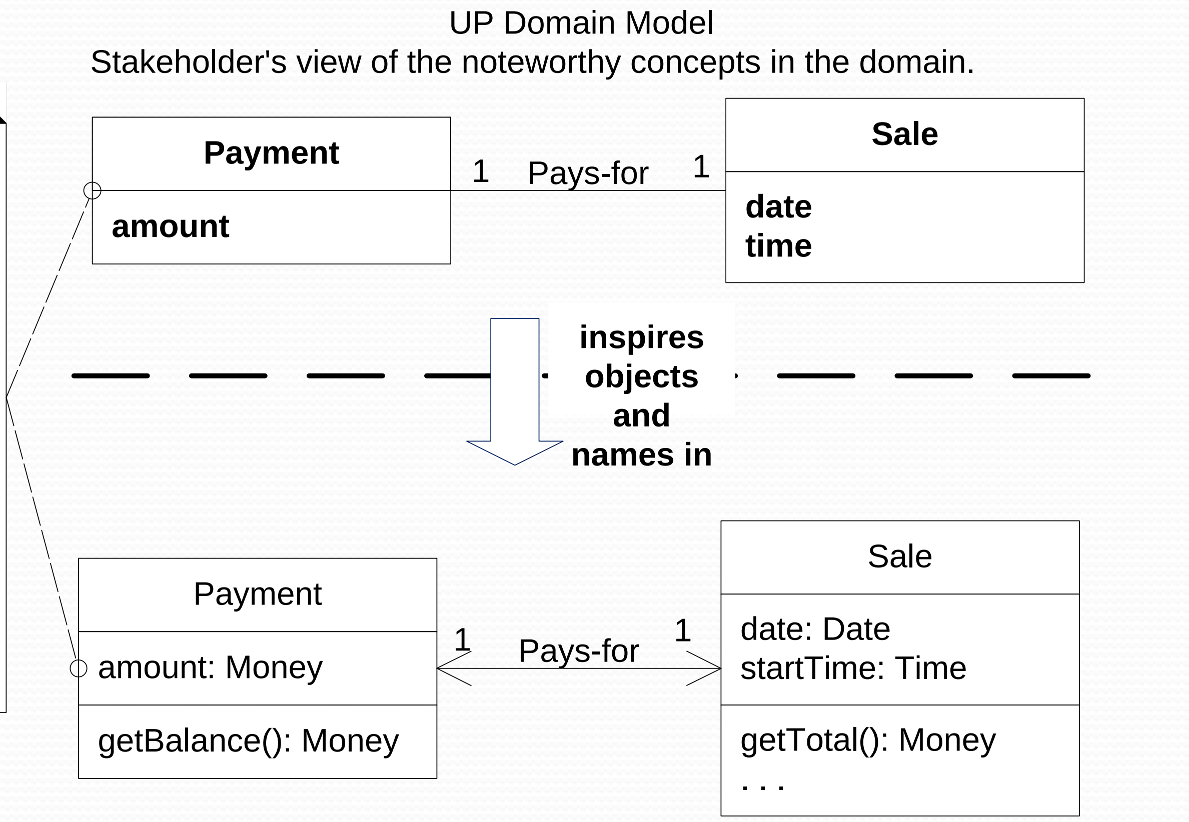


Larman Fig 9.5

Conceptual Class

- A conceptual class is an idea thing or object
- Often something concrete in the problem domain
  - Like an automobile
- May also be something abstract in the problem domain
  - Like an insurance policy

Domain Model drives 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 Design Model
The object-oriented developer has taken inspiration from the real world domain in creating software classes.

Therefore, the representational gap between how stakeholders conceive the domain, and its representation in software, has been lowered.

Larman Fig 9.6

Domain Models and decomposition

- Problem → Software problems can be complex.
- Solution → Decompose or Divide-and-conquer

The dimension of decomposition is by entities (objects) in the domain.

Conceptual Class Identification

- Incrementally build a domain model over several iterations in the elaboration phase
- In each phase, the domain model is limited to the prior and current scenarios under consideration
- Central task is to identify conceptual classes related to the scenario under consideration
- It is better to over-specify a domain model with lots of fine-grained conceptual classes than to under-specify it.
- It is valid to have conceptual classes without attributes which have purely behavioral role

Strategies to identify conceptual classes

1. Use conceptual class category list
   - See next slide ….
2. Identify noun phrases in textual descriptions
   - Fully dressed use cases are an excellent description to draw from

Conceptual Class Category	Examples
Physical or tangible objects	Register, Airplane
Specifications, designs, or descriptions	ProductSpecification, FlightDescription
Places	Store, Airport
Transactions	Sale, Payment, Reservation
Transaction line items	SaleLineItem
Roles of people	Cashier, Pilot
Containers of other things	Store, Bin, Airplane
Things in a container	Item, Passenger
Other external systems	CCPaymentSystem, AirTrafficControl
Abstract noun concepts	Hunger, Acrophobianger
Organizations	SalesDepartment, SuperAirline
Events	Sale, Payment, Meeting, Flight, Landing
Processes	SellingAProduct, BookinhASeat
Rules and policies	RefundPolicy, CancellationPolicy
Catalogs	ProductCatalog, PartsCatalog
Records of finance, work, contracts, legal	Receipt, Ledger, EmploymentContract
Financial instruments and services	LineOfCredit, Stock
Manuals, Documents, Reference Papers	DailyPriceChangeList, RepairManual

Conceptual class category list

Conceptual Class Category	Examples
Physical or tangible objects	Register, Airplane
Specifications, designs, or descriptions	ProductSpecification, FlightDescription
Places	Store, Airport
Transactions	Sale, Payment, Reservation
Transaction line items	SaleLineItem
Roles of people	Cashier, Pilot
Containers of other things	Store, Bin, Airplane
Things in a container	Item, Passenger

Conceptual class category list

Conceptual Class Category	Examples
Other external systems	CCPaymentSystem, AirTrafficControl
Abstract noun concepts	Hunger
Organizations	SalesDepartment, SuperAirline
Events	Sale, Payment, Meeting, Flight, Landing
Processes	SellingAProduct, BookingASeat

Conceptual class category list

Conceptual Class Category	Examples
Rules and policies	Rules and policies
Catalogs	ProductCatalog, PartsCatalog
Records of finance, work, contracts, legal	Receipt, Ledger, EmploymentContract
Financial instruments and services	LineOfCredit, Stock
Manuals, Documents, Reference Papers	DailyPriceChangeList, RepairManual

Creating A Domain Model

- Identify the conceptual classes from the Use Cases in the first iteration of the Elaboration phase
- Create the Domain Model and draw the conceptual classes in UML
- Add the attributes and the associations to the classes in the Domain Model

Can you find Conceptual Classes?

Simple cash-only Process Sale scenario:

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 and quantity, if greater than one.
4. System records sale line item and presents item description, price, and running total.
5. Cashier repeats steps 2-3 until indicates done.
6. System presents total with taxes calculated.

Can you find Conceptual Classes?

Simple cash-only Process Sale scenario:

7. Cashier tells Customer the total, and asks for payment.
8. Customer pays with cash.
9. Cashier enters cash tendered.
10. System records payment and presents change due.
11. System logs the completed sale, but does not interact with external systems.
12. System presents receipt.
13. Customer leaves with receipt and goods.

Conceptual classes from nouns

Simple cash-only Process Sale scenario:

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 and quantity, if greater than one.
4. System records sale line item and presents item description, price, and running total.
5. Cashier repeats steps 2-3 until indicates done.
6. System presents total with taxes calculated.
7. Cashier tells Customer the total, and asks for payment.
8. Customer pays with cash.
9. Cashier enters cash tendered.
10. System records payment and presents change due.
11. System logs the completed sale, but does not interact with external systems.
12. System presents receipt.
13. Customer leaves with receipt and goods.

Candidate conceptual classes for the Sales domain.

Register

Item

Store

Sale

Sales LineItem

Cashier

Customer

Manager

Payment

Product Catalog

Product Specification

- This is, somewhat, an arbitrary list of abstractions that the modelers consider noteworthy

What about …. receipt?

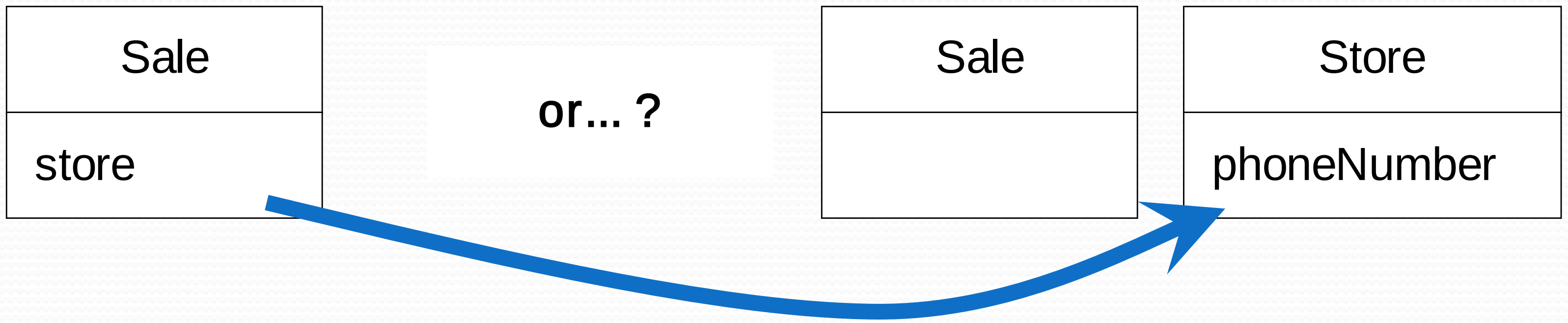
- In general don’t show report in domain model
  - is not useful since information is derived or duplicated from other sources
  - So exclude receipt ?
- However, if it has a special role in terms of the business rules should include it.
  - Receipt gives the bearer the right to return bought items … so include receipt ?
- But item returns are not being considered in this iteration, so receipt will be excluded

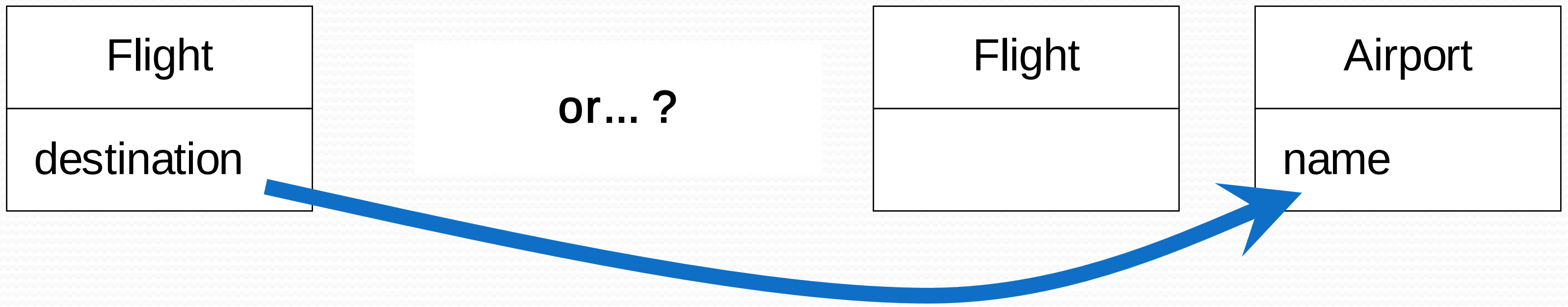
Guidelines

- Use names relevant in the domain
  - If developing a model for a library, name the customer a "Borrower" or "Patron“
  - The terms used by the library staff
- Exclude irrelevant or out-of-scope features.
- Do not add things that are not there.

Common mistake in identifying classes

- Representing something as an attribute when it should be a conceptual class





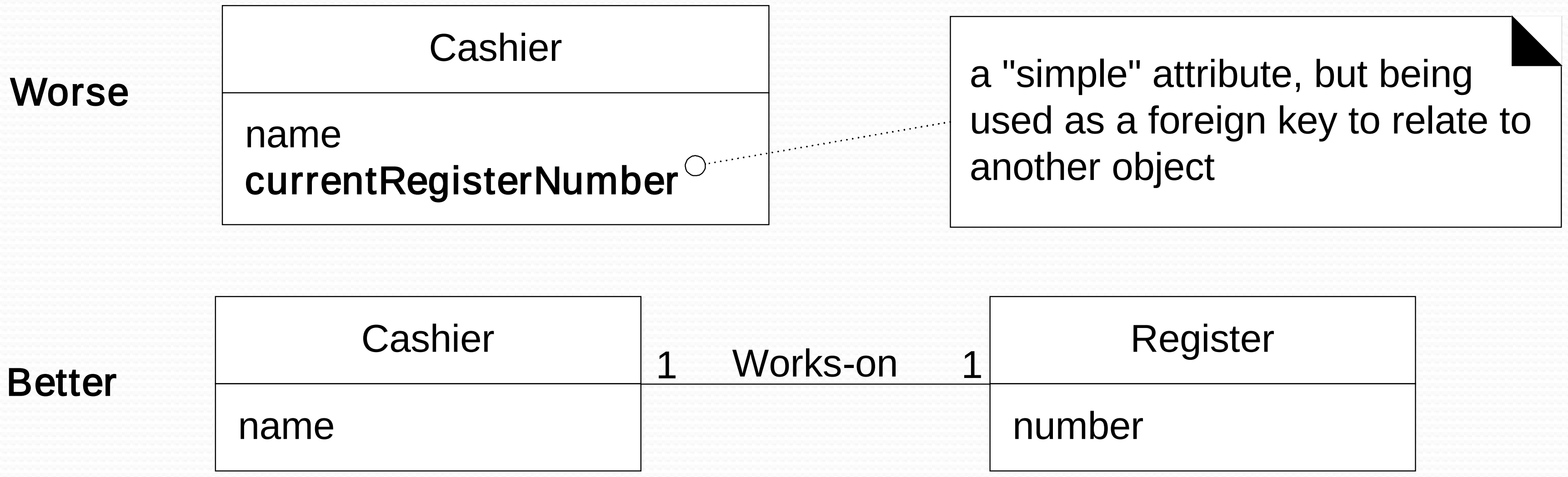
Resolve similar conceptual classes

- Sometimes, two classes represent the same thing in a particular domain:
  - Register & POS
  - Item & Product
  - Customer & Client
  - Outlet & Shop
- Decide upon which class identifier is to be used and stick to it.

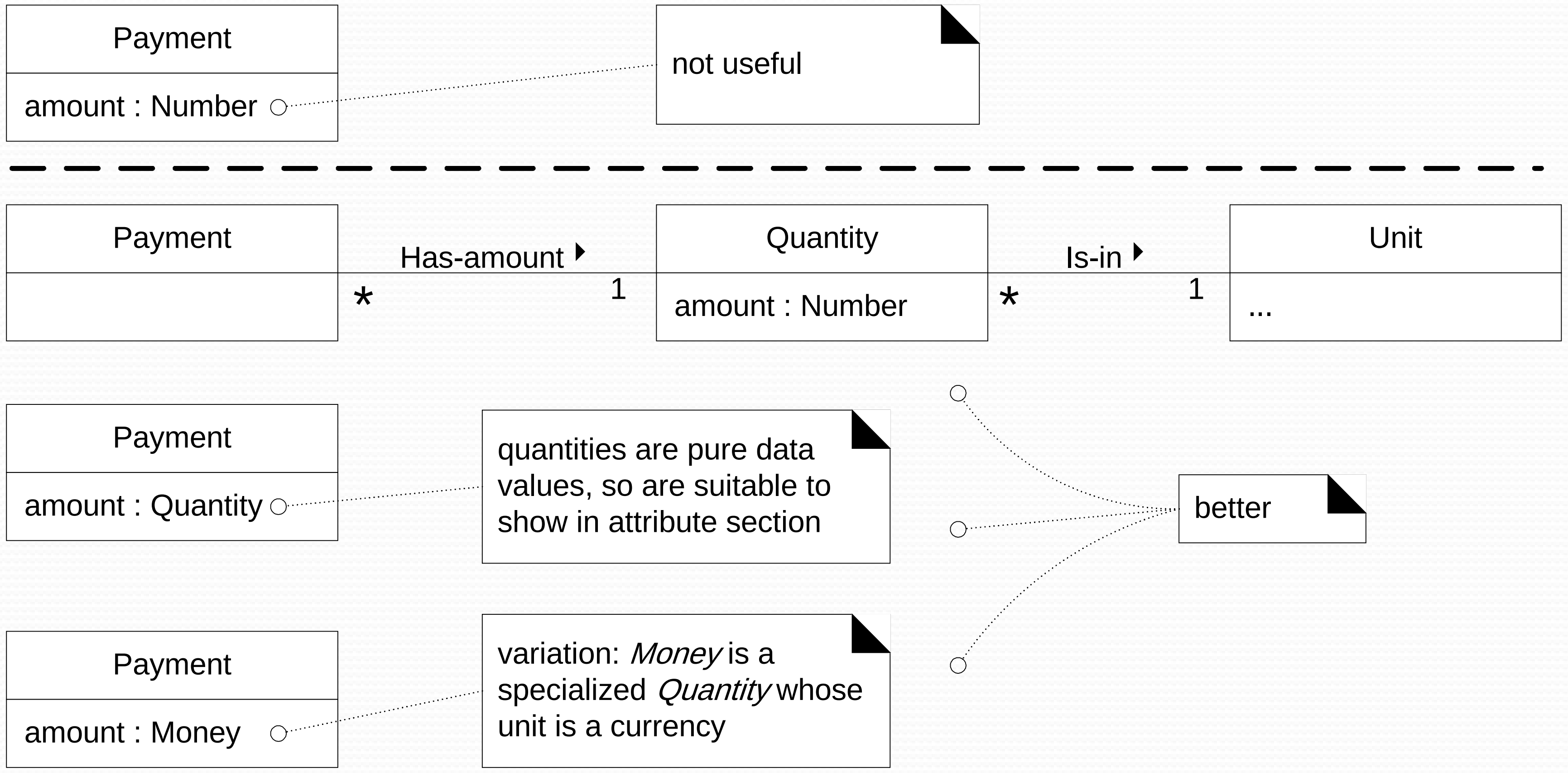
Domain Modeling Guidelines

- List the candidate conceptual classes using following techniques in a domain class model
  - Conceptual Class Category List
  - and/or Noun Phrase Identification
- Draw them in the Domain Model.
- Add associations necessary to record relationships.
- Add the attributes necessary to fulfill information requirements.

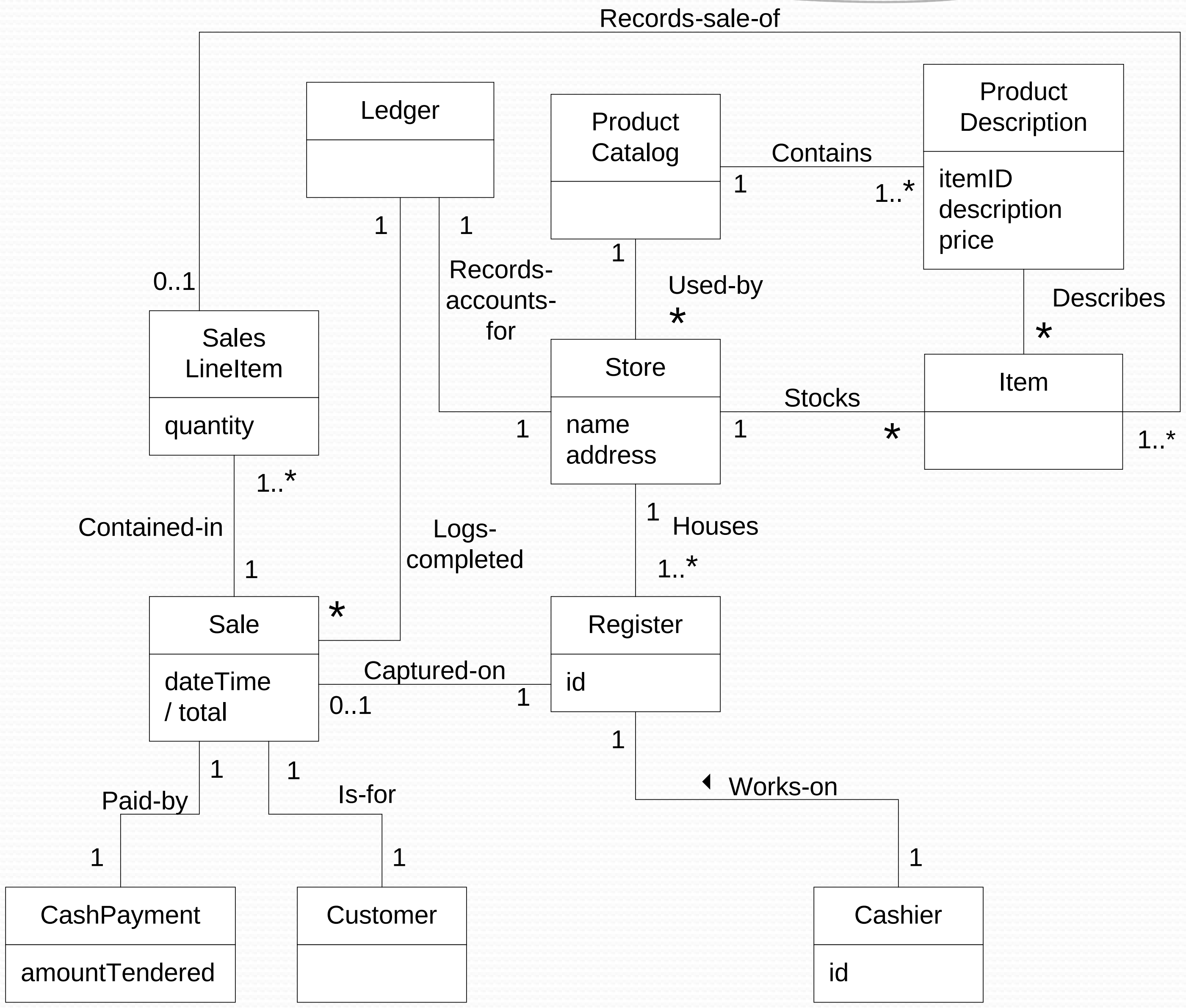
Better Specification Example


Worse
Cashier
name
currentRegisterNumber
a "simple" attribute, but being used as a foreign key to relate to another object
Better
Cashier
name
1
Works-on
1
Register
number

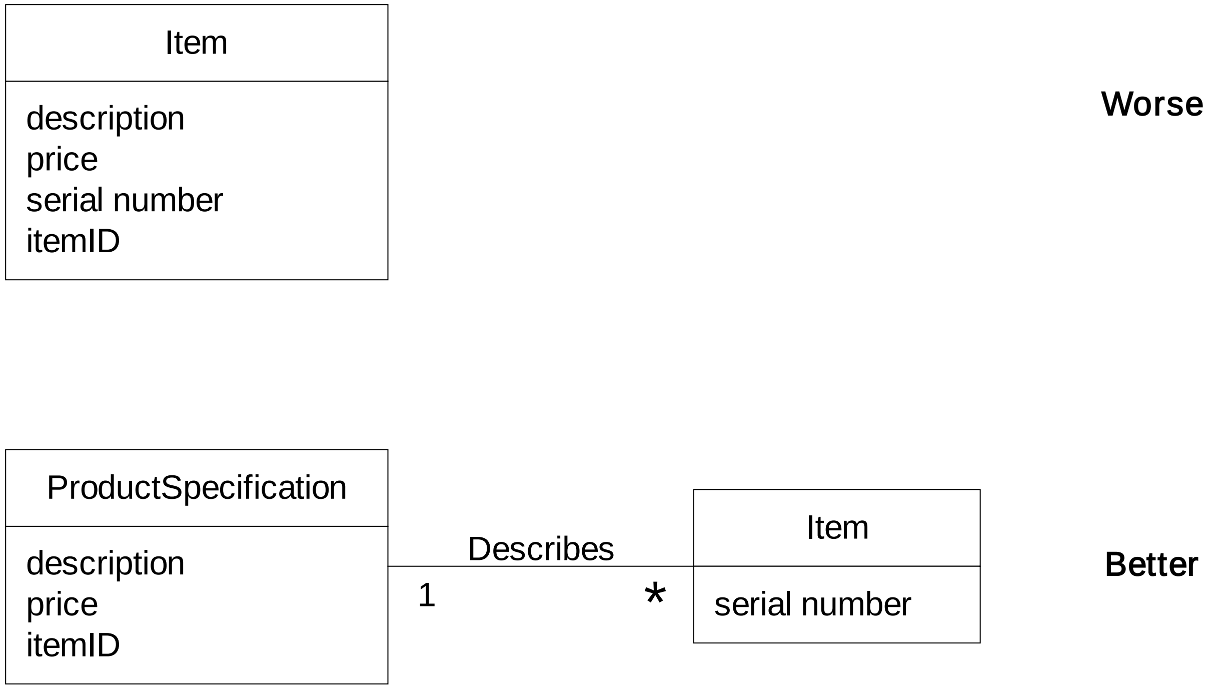
Better Specification Example


Payment
amount : Number
not useful
Payment
Has-amount
*
1
Quantity
amount : Number
Is-in
*
1
Unit
...
Payment
amount : Quantity
quantities are pure data values, so are suitable to show in attribute section
better
Payment
amount : Money
variation: *Money* is a specialized *Quantity* whose unit is a currency

Conceptual Classes Specification



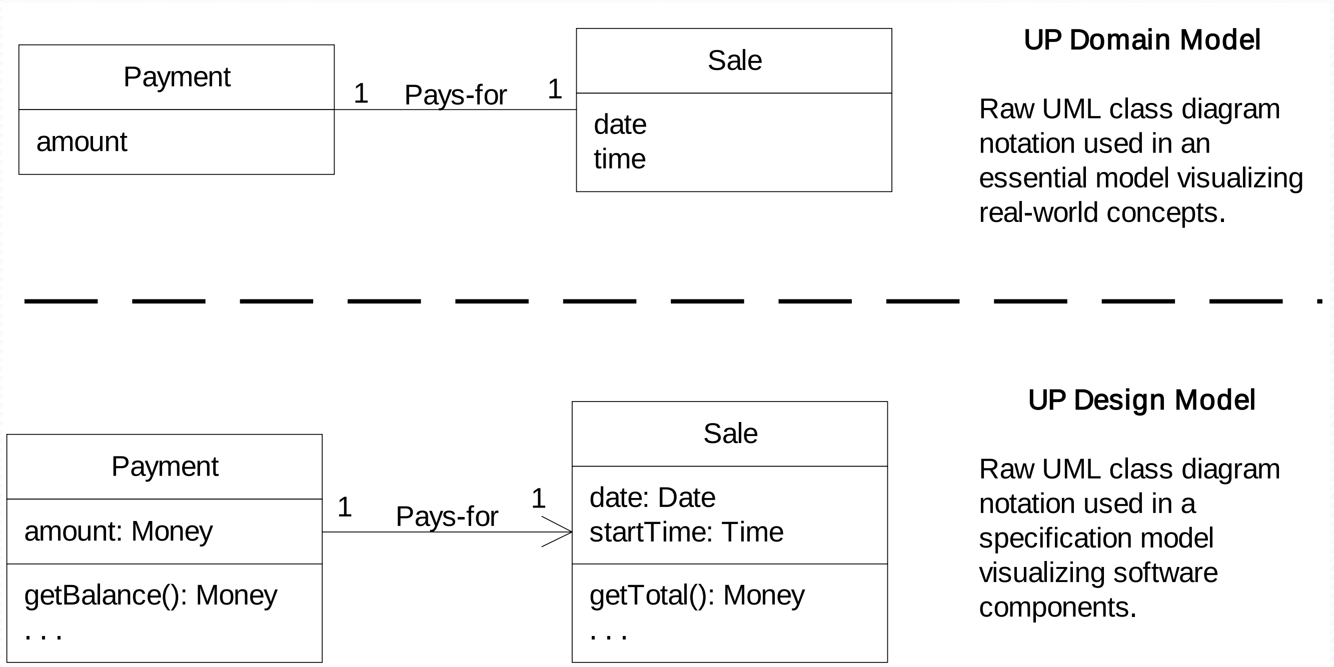
Specification Conceptual Classes



Add specification conceptual class when:

1. There needs to be a description about an item or service, independent of the existence of those items or services
2. Deleting instances of things they describes results in a loss of information
3. Reduced duplicated information

Domain Model versus Class Diagram


Payment
amount
1
Pays-for
1
Sale
date
time
UP Domain Model
Raw UML class diagram notation used in an essential model visualizing real-world concepts.
Payment
amount: Money
getBalance(): Money
. . .
1
Pays-for
1
Sale
date: Date
startTime: Time
getTotal(): Money
. . .
UP Design Model
Raw UML class diagram notation used in a specification model visualizing software components.

- When UML boxes are drawn in the Domain Model, they are called conceptual classes (or domain concepts) – no methods are captured.
- When UML boxes are drawn in the Design Model, they are called design classes.

Class related terms

Conceptual Class	Real-world concept or thing
Software Class	A class representing a specification or implementation perspective of a software component
Design Class	A class in the design model
Implementation Class	A class implemented in an OO language such as Java
Class	The general term representing either a real-world or software thing

UP & Domain Models

Discipline	Artifact	Inception	Elaboration	Construction	Transition
Business Modeling	Domain Model		**start**		
Requirements	Use-Case Model	start	**refine**		
	Vision	start	**refine**		
	Supplementary Specification	start	**refine**		
	Glossary	start	**refine**		
Design	Design Model		**start**	refine	
	SW Architecture Document		**start**	refine	
	Data Model		**start**	refine	
Implementation	Implementation Model		**start**	refine	refine
Project Management	SW Development Plan	start	**refine**	refine	refine
Testing	Test Model		**start**	refine	
Environment	Development Case	start	**refine**		

Domain models normally started and completed in elaboration

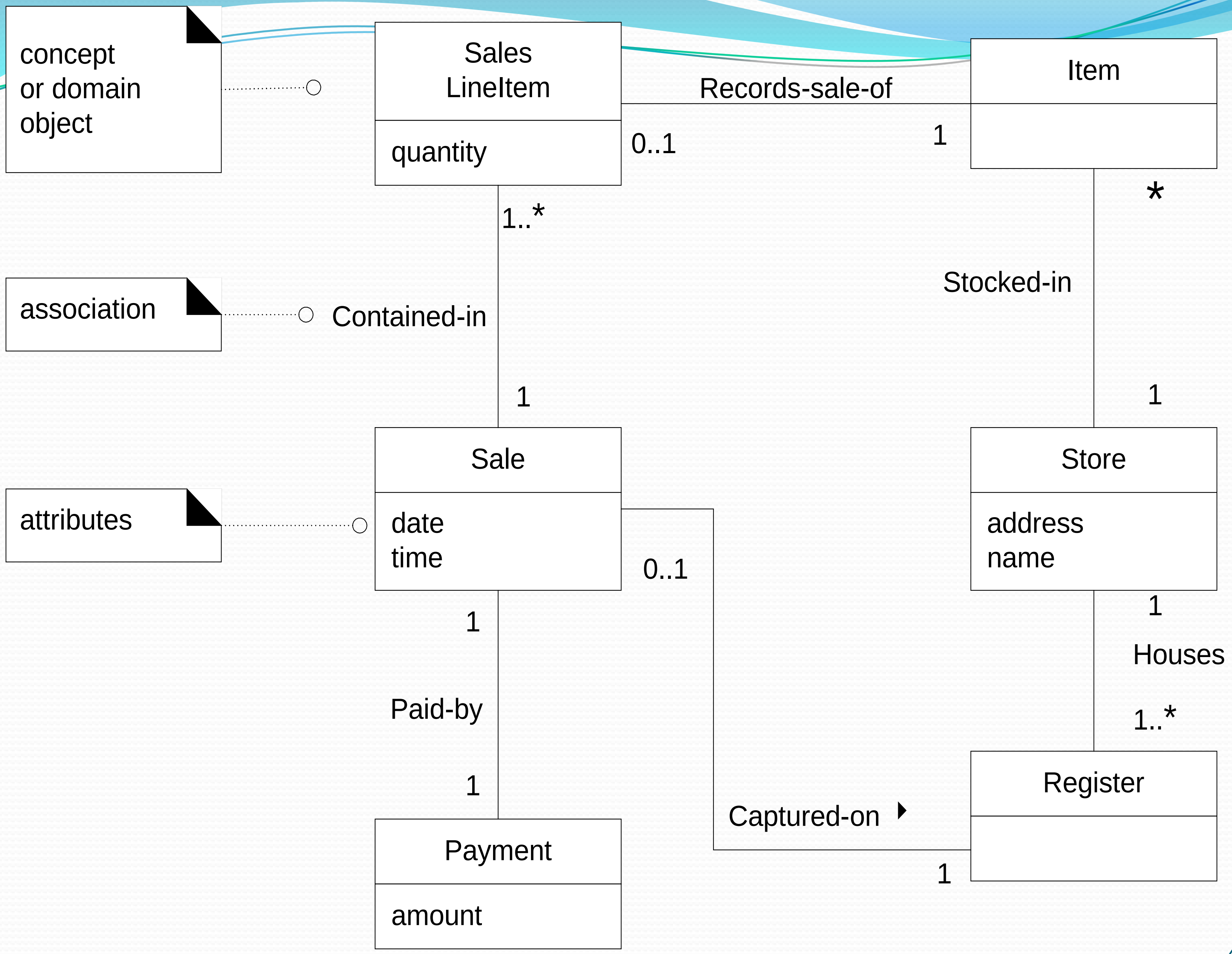
Summary

- *A domain model is represented using a set of UML class diagrams with:*

  - Conceptual classes
  - Associations between conceptual classes
  - Attributes of conceptual classes

  - No operations / methods – these are software concepts

  - Not a *Design Class Diagram*
  - *Not a data model*
  - But a ***Conceptual Class Diagram***


concept or domain object
association
attributes
Sales LineItem
quantity
Records-sale-of
0..1
1
Item
*
1..*
Contained-in
Stocked-in
1
1
Sale
date
time
Store
address
name
0..1
1
1
Houses
Paid-by
1..*
1
Register
Captured-on
Payment
amount
1

Questions