Domain Models

COMP 3831

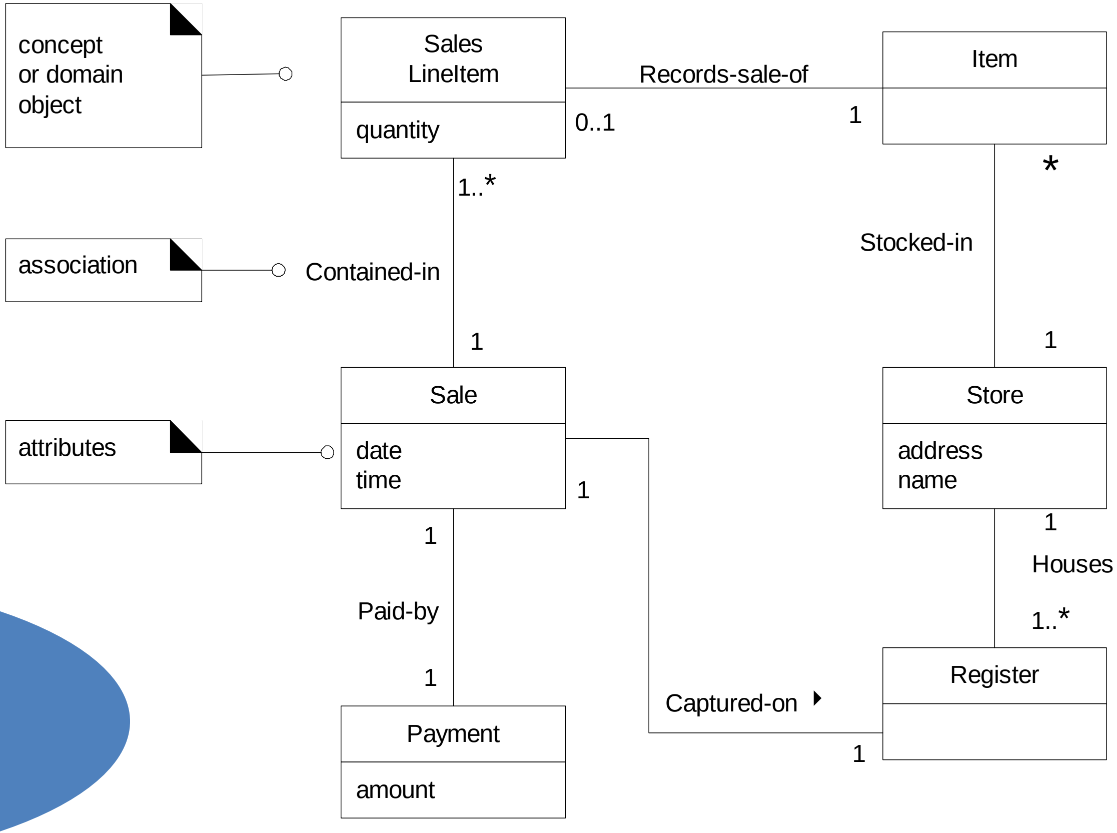
Larman: Chapter 9

Objectives

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

What is a domain model?

- A domain model is a representation of real world conceptual classes, not of software components.



Meaningful classes in a problem domain

Domain Models

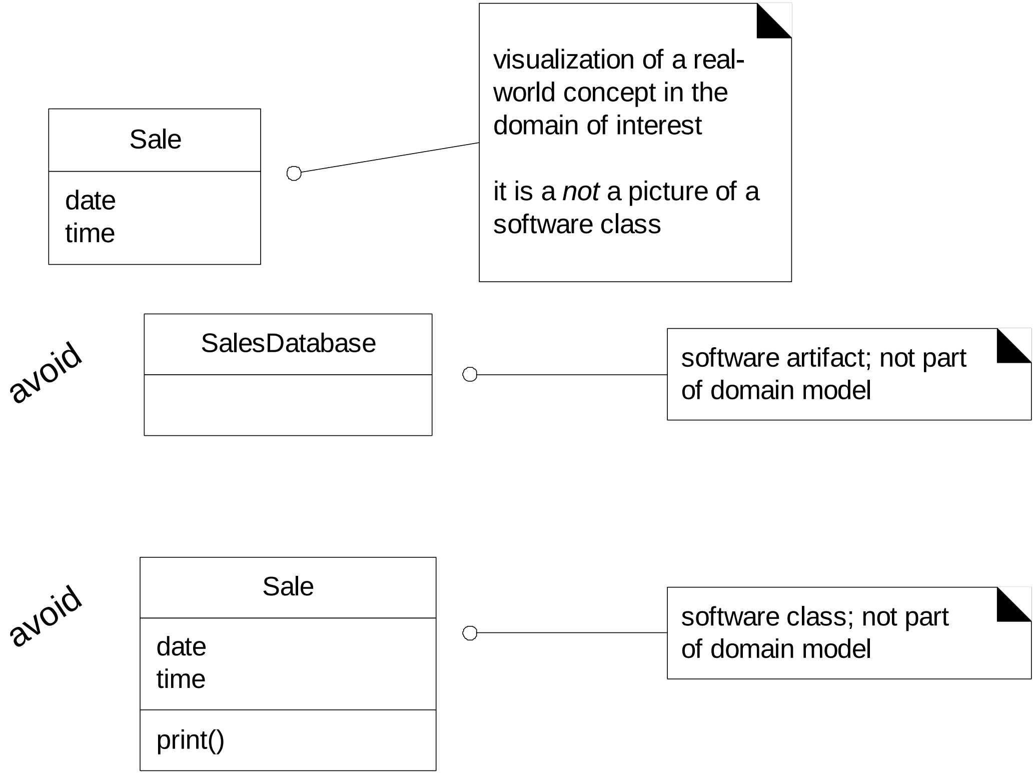
- Also known as conceptual models, domain object models, or analysis object models
- A visual representation of conceptual classes or real-world objects in a domain of interest
- A visual dictionary of the noteworthy abstractions (conceptual classes), domain vocabulary, and information content of the domain

UML representation of Domain Model

- Using UML notation, a domain model is illustrated using a set of class diagrams with:
  - Conceptual classes
  - Attributes of conceptual classes
  - No operations
  - Associations between conceptual classes

Domain Models and software

Domain model is a visualization of things in the real world domain, not of software components such as Java & C++.



visualization of a real-world concept in the domain of interest

it is a *not* a picture of a software class

software artifact; not part of domain model

software class; not part of domain model

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 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

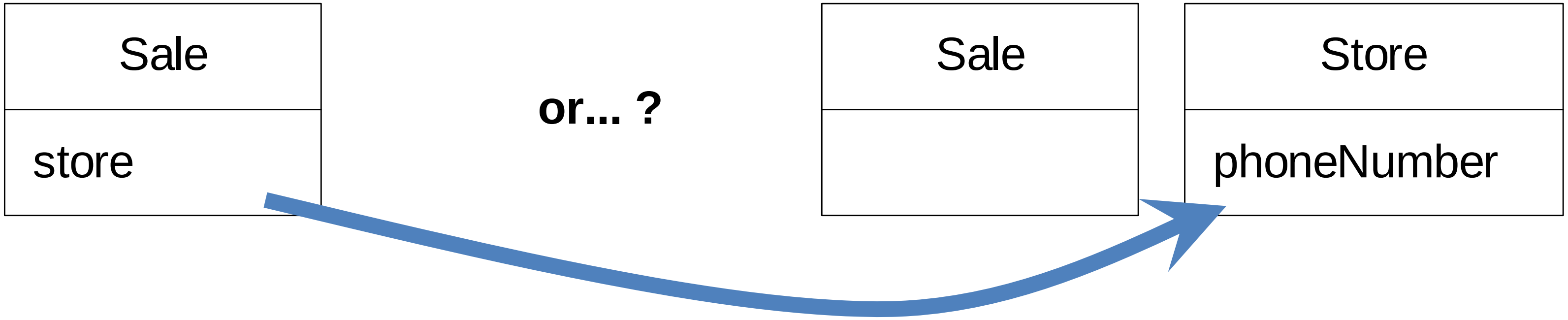
Product Catalog

Product Specification

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

Common mistake in identifying classes

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



Sale

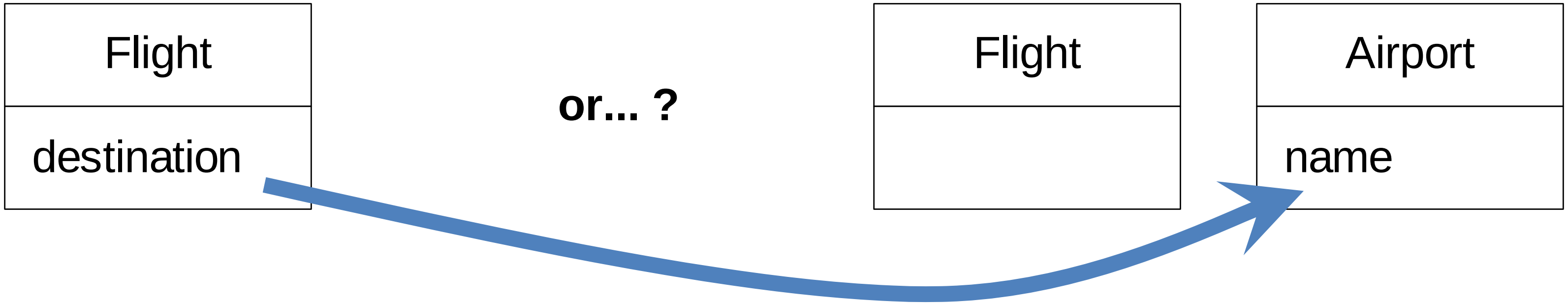
store

or… ?

Sale

Store

phoneNumber



Flight

destination

or… ?

Flight

Airport

name

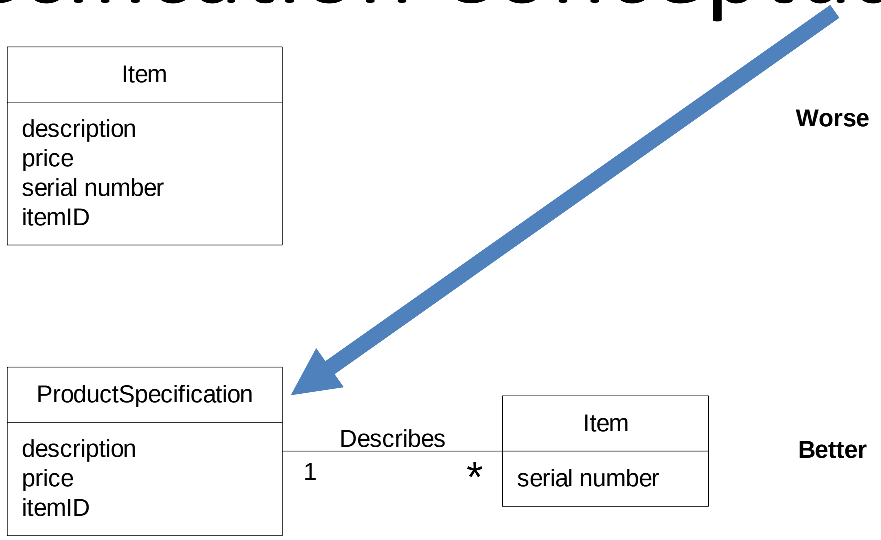
Resolve similar conceptual classes

- Sometimes, two classes represent the same thing in a particular domain:
  - Register & (P)oint (O)f (S)ale (T)erminal
  - Item & Product
  - Customer & Client
  - Outlet & Shop
- Decide upon which class identifier is to be used and stick to it.

Domain Modeling Guidelines

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

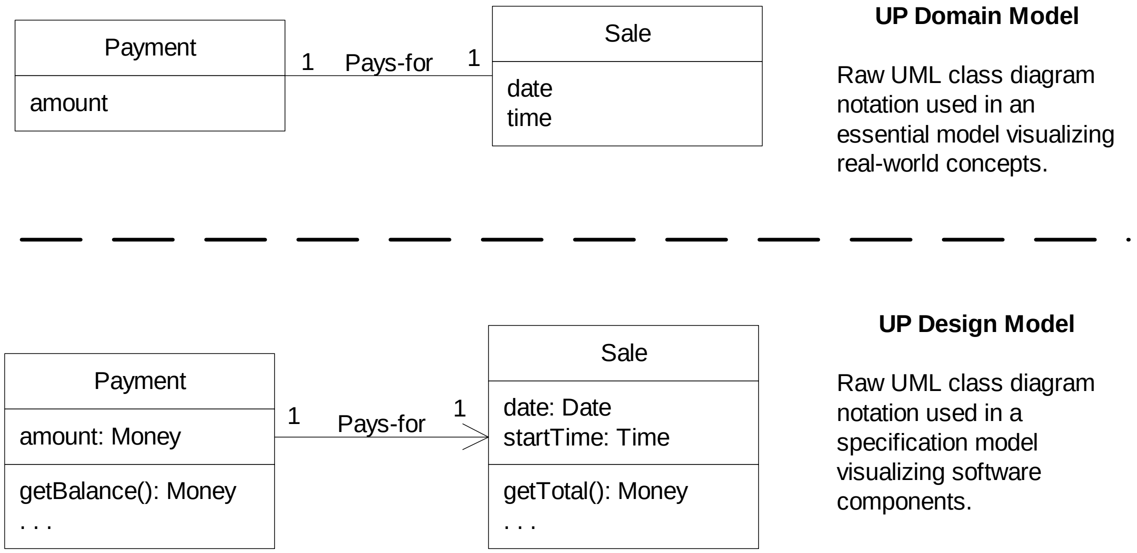
Specification Conceptual Classes



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)

- 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

Questions and Conclusions