FARABI UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Lecture 4.
UML Class Diagrams

Classes, responsibilities, relationships and constraints

Dr. Bazargul Matkerim, Professor-Researcher

Lecture goals and plan

Learning outcomes

Distinguish classes from objects.

Read attributes and operations.

Interpret relationships and multiplicities.

Derive a class model from a scenario.

Lecture plan

1. Analysis and class discovery
2. Compartments and visibility
3. Relationships and ownership
4. Worked class models
5. Constraints and review

Class diagrams in analysis and design

Analysis model

Build a shared domain vocabulary.

Show concepts such as Customer, Order and Product.

Describe relevant information and relationships.

Design model

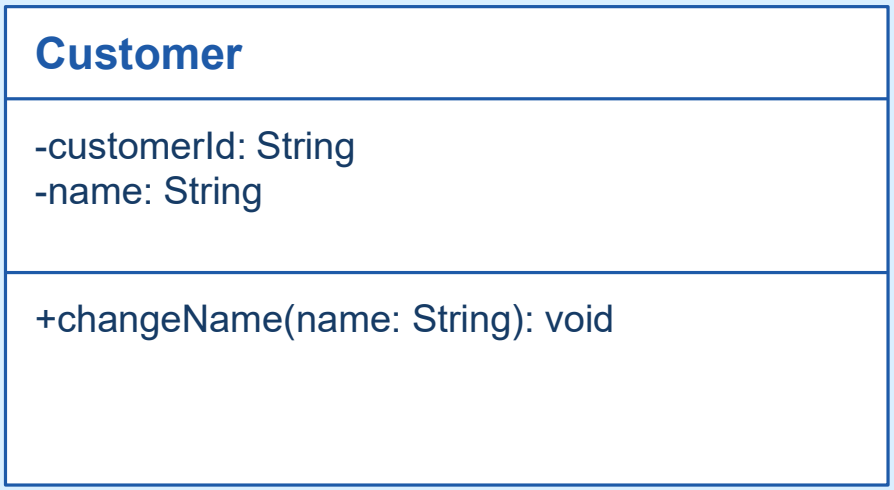
Assign software responsibilities.

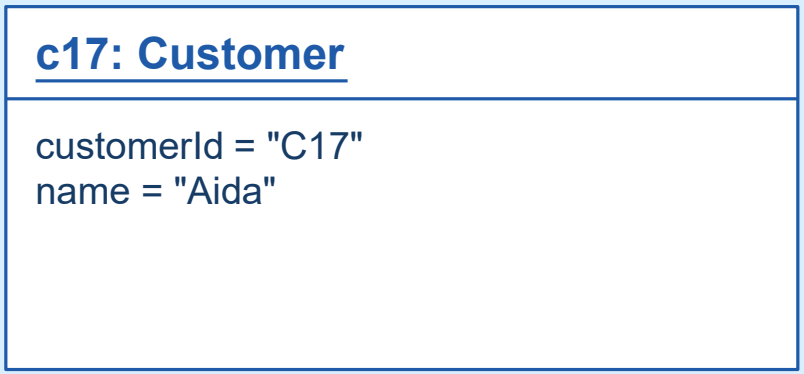
Specify types, visibility and operations.

Add services, interfaces and implementation decisions as needed.

Class diagrams show static structure; interaction order belongs in behavioral models.

A class describes a family of objects





Class: features shared by its instances

Object: identity and current values

Discover classes with scenarios and CRC

Use language as a clue

Candidate nouns: Customer, Order, Product.

Candidate responsibilities: price an order, reserve stock.

Validate identity, purpose and behavior before adding a class.

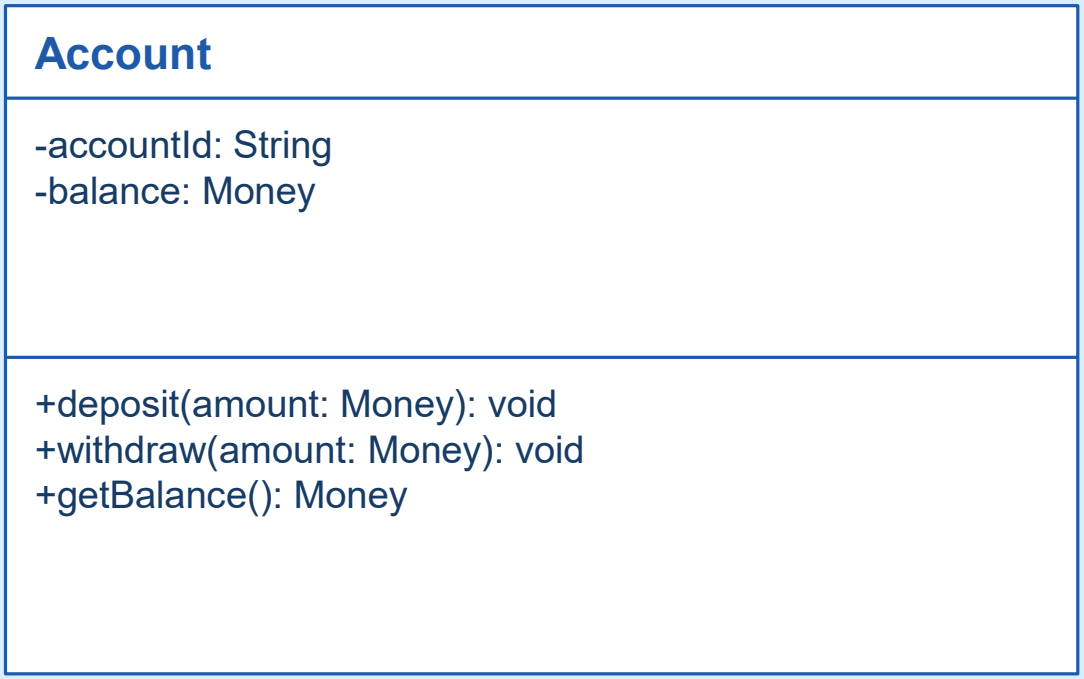
CRC card: Order

Class
Order

Responsibilities
Track items; calculate total; close

Collaborators
OrderLine; Product

Read the three class compartments



What each part means

Name: the classifier

Attributes: named, typed properties

Operations: callable behavior, parameters and results

Attributes, types and derived values

Notation	Interpretation
-balance: Money	Private property with a domain value type
-openedOn: Date	A date-valued property
-/availableFunds: Money	Derived property; define the calculation
-tags: String [0..*]	Zero or more string values
-accountId: String {readOnly}	A value that is not freely reassigned

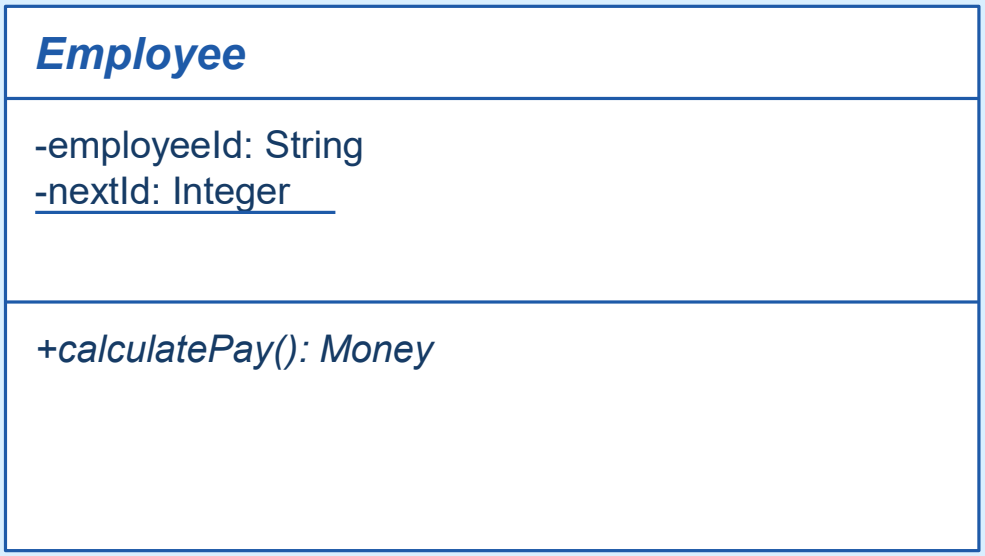
Use an association when the relationship itself matters to the model.

Operations and visibility

Symbol	Visibility	Example
+	Public	+close(): void
–	Private	-recalculate(): Money
#	Protected	#validate(): Boolean
~	Package	~importEntry(): void

Queries return information; commands update state.
An operation signature is a contract, not its algorithm.

Abstract classifiers and static features



Read the typography

Italic class name: abstract class.

Italic operation: abstract operation.

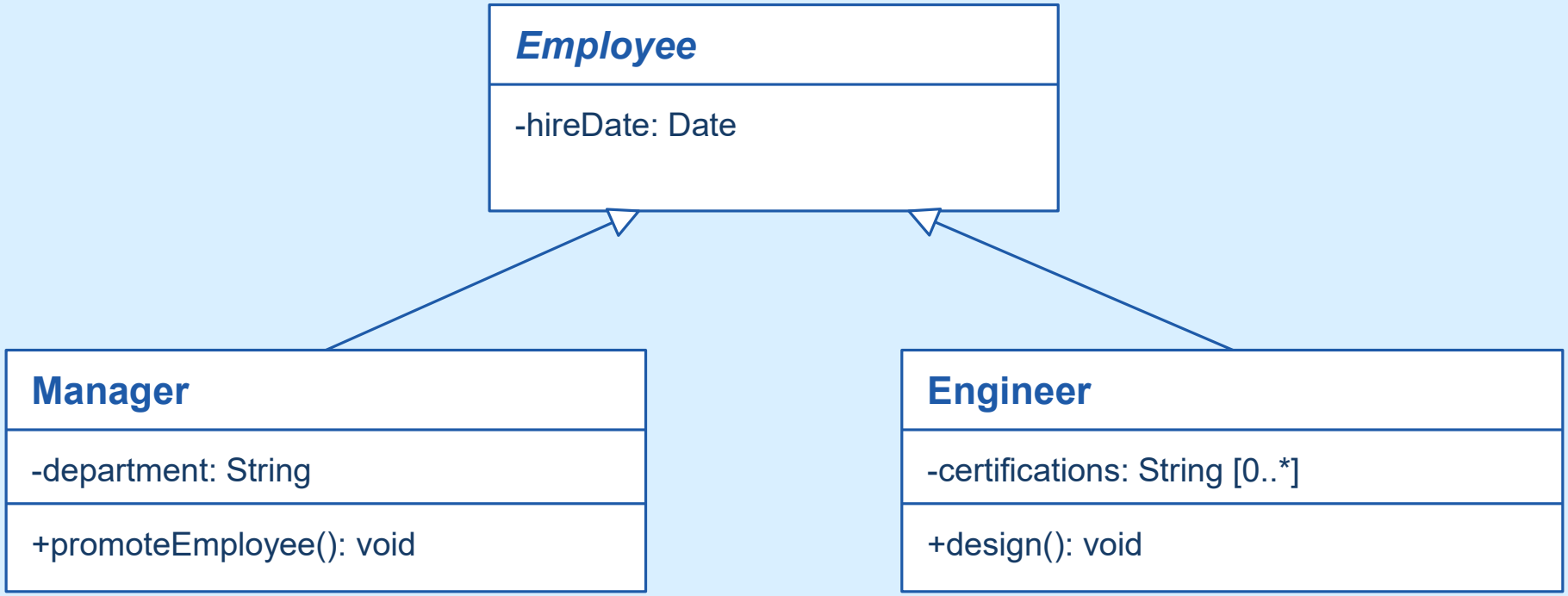
Underlined feature: classifier scope (static).

An abstract class cannot be instantiated directly; a concrete subclass supplies required behavior.

Choose the relationship by its meaning

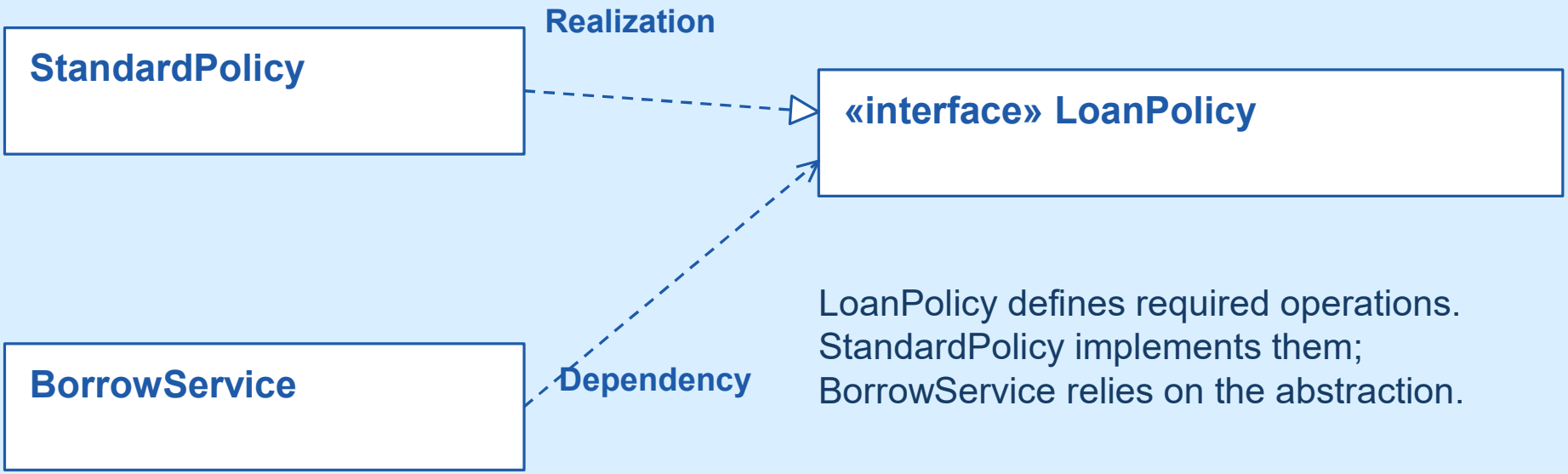
Relationship	Meaning	Main notation
Association	Structural links between instances	Solid line
Generalization	A more specific classifier is a kind of a general one	Solid line; hollow triangle
Interface realization	A classifier fulfills an interface contract	Dashed line; hollow triangle
Dependency	A client depends on a supplier	Dashed line; open arrow
Composition	Exclusive composite ownership of parts	Filled diamond at the whole

Generalization: a kind-of relationship



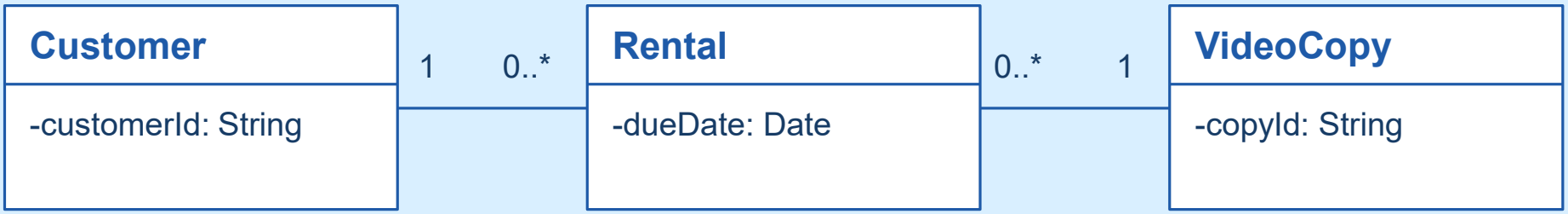
The hollow triangle points to the general classifier, whether it is abstract or concrete.

Realization and dependency are different



LoanPolicy defines required operations.
StandardPolicy implements them;
BorrowService relies on the abstraction.

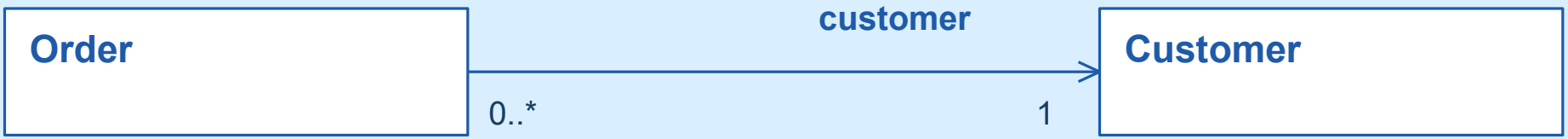
Associations and multiplicity



Read each association in both directions

Each Rental has one Customer and one VideoCopy.
A Customer and a VideoCopy can each have zero or more historical Rentals.

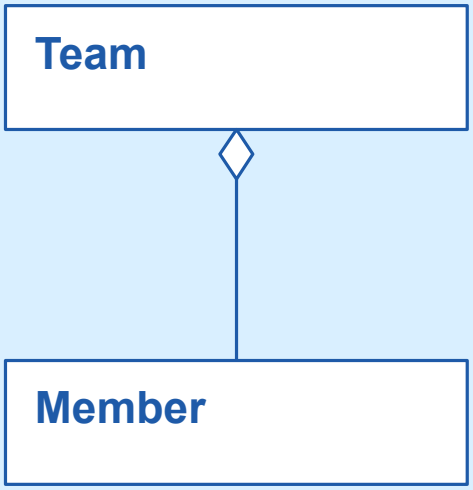
Roles, multiplicities and navigability



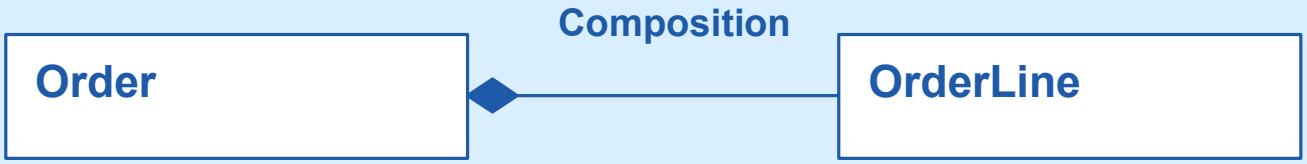
0..1	Optional: at most one
1	Exactly one
*** or 0..***	Zero or more
2..4	Between two and four

Navigability shows access in the model; it is not a message-order arrow.

Aggregation and composition

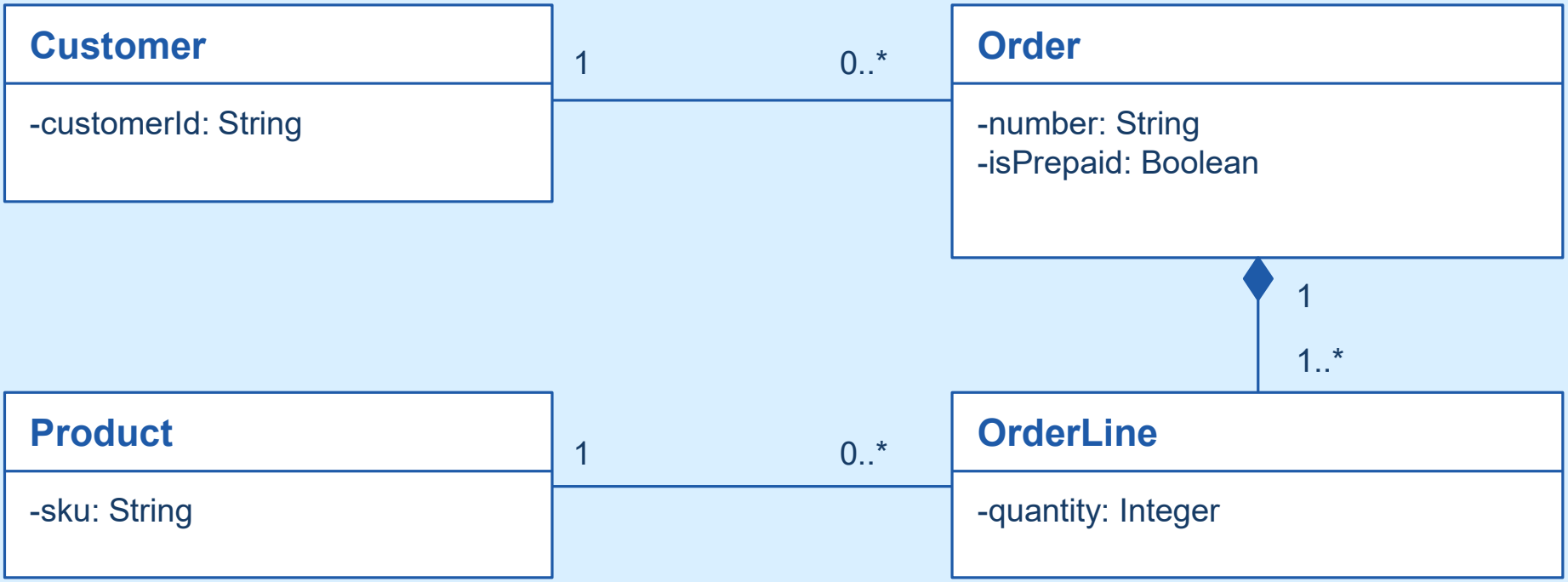


Shared aggregation
Hollow diamond at the whole.
Precise meaning is model-dependent.



A part belongs to at most one composite at a time.
Deleting the whole deletes its still-owned object parts.

Worked model: customers and orders



Assumption: a confirmed order contains at least one line; draft orders may need different bounds.

Constraints add rules beyond the lines

Structural constraints

{quantity > 0}

{order total = sum of line subtotals}

{a confirmed order has at least one line}

Business constraints

{poor credit rating implies prepayment}

{one active rental per video copy}

State when each rule is evaluated and what happens if it fails.

Braces denote a constraint; natural language or a formal constraint language may be used.

Derive a model from a use-case scenario

Scenario: rent a video copy

1. Customer chooses an available copy.
2. Staff records a rental and due date.
3. System links the customer and copy.
4. Return closes the active rental.

Modeling decisions

Customer and VideoCopy have identity.

Rental records the transaction.

dueDate is an attribute of Rental.

close(on: Date) is a candidate operation.

Review and classroom practice

Check your diagram

Does each class serve a requirement?

Are attributes and operations owned by the right class?

Can every multiplicity be explained?

Are ownership and inheritance justified?

Practice: library lending

Model Reader, Loan and BookCopy.

Add a due date and close operation.

Show historical multiplicities.

Add the rule: one active loan per copy.

Explain where a LoanPolicy fits.

Reading and references

Core reading

Dennis, A., Wixom, B. H., & Tegarden, D. (2021).
Systems Analysis and Design: An Object-Oriented Approach
with UML, 6th ed. Wiley. ISBN 9781119559917.

UML notation reference

Object Management Group (2017). Unified Modeling Language,
Version 2.5.1. Classifiers, properties and structured classifiers.

Adapted from the supplied slides credited to Dr. Suzanne Smith; full credits in notes.