FARABI UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE



Lecture 11.
UML Package Diagrams

Namespaces, visibility and architectural dependencies

Dr. Bazargul Matkerim, Professor-Researcher

Lecture goals and plan

Learning outcomes

Organize elements into packages.

Read nesting and qualified names.

Distinguish dependency, import and access.

Review architectural boundaries.

Lecture plan

1. Package notation and ownership
2. Names and visibility
3. Imports and dependencies
4. Layers and order-system example
5. Review and practice

Purpose of package diagrams

Model organization

Group related packageable elements.

Hide detail in large class models.

Represent nested namespaces.

Show dependencies between groups.

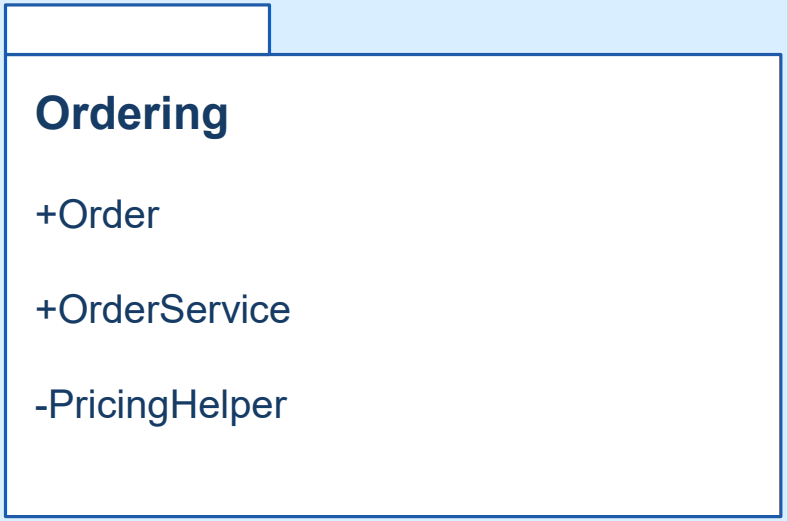
Appropriate scope

A package can own classes, interfaces and other packages.

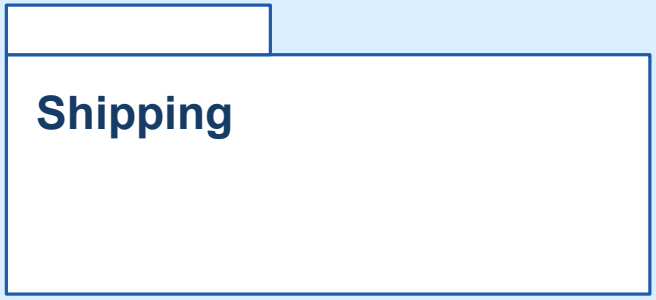
A diagram presents a selected view of the model.

A package is not inherently a runtime service or deployable file.

Package notation and contents

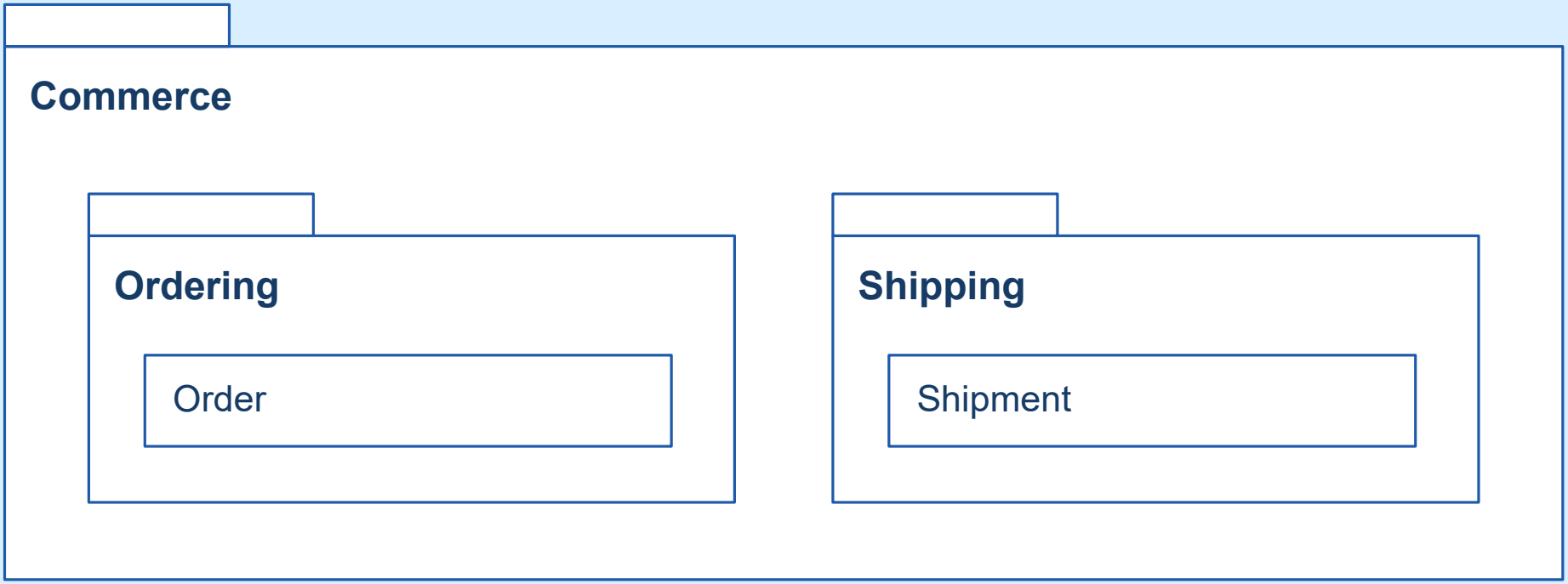


Expanded contents



Contents hidden

Nested packages and ownership



Qualified names and name collisions

Qualified names

Commerce::Ordering::Order

Commerce::Shipping::Shipment

The :: separator identifies the ownership path.

Repeated simple names

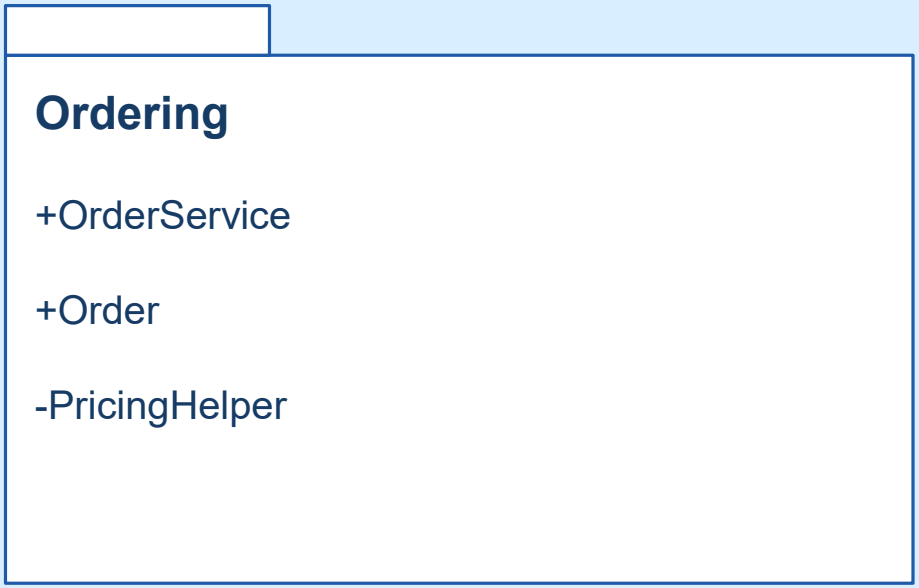
Sales::Customer

Support::Customer

Different namespaces can contain elements with the same simple name.

When a short name is ambiguous, qualify it or define an explicit element-import alias.

Public and private package members



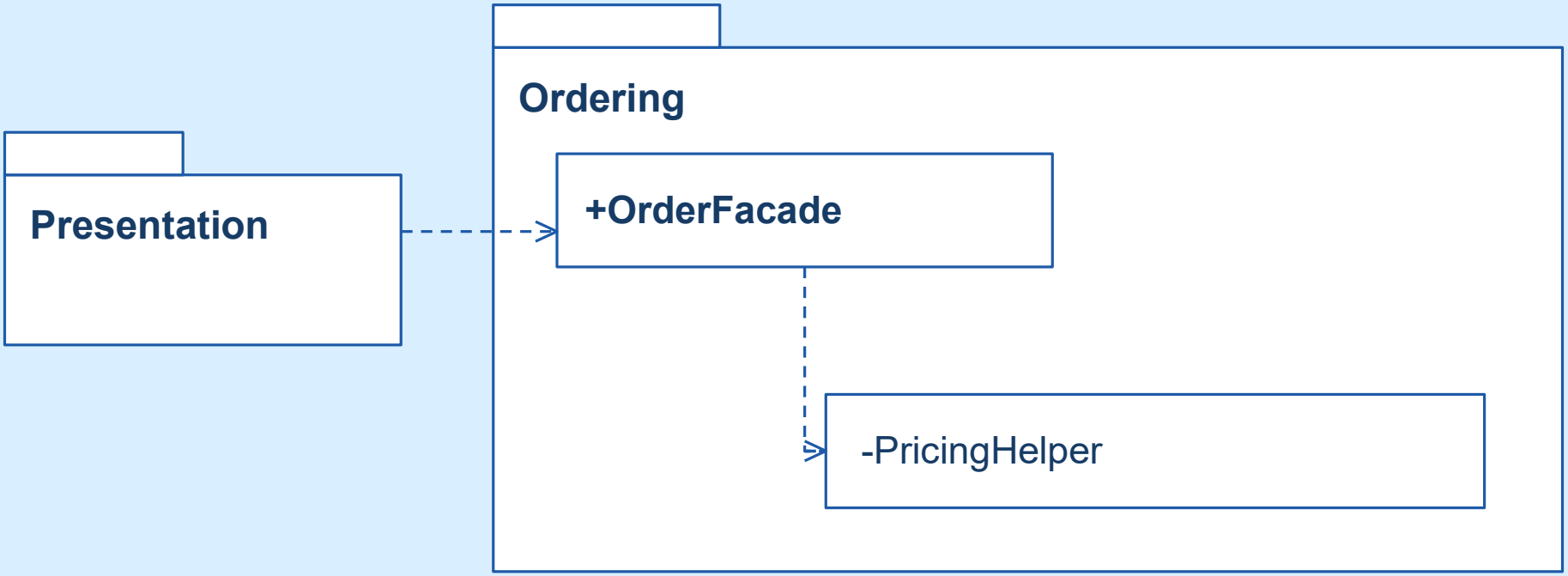
Public (+)

Visible outside the owning package, subject to namespace rules.

Private (−)

An internal member that other packages cannot import.

A public façade for package access



Clients depend on a small public entry point.

Package dependencies

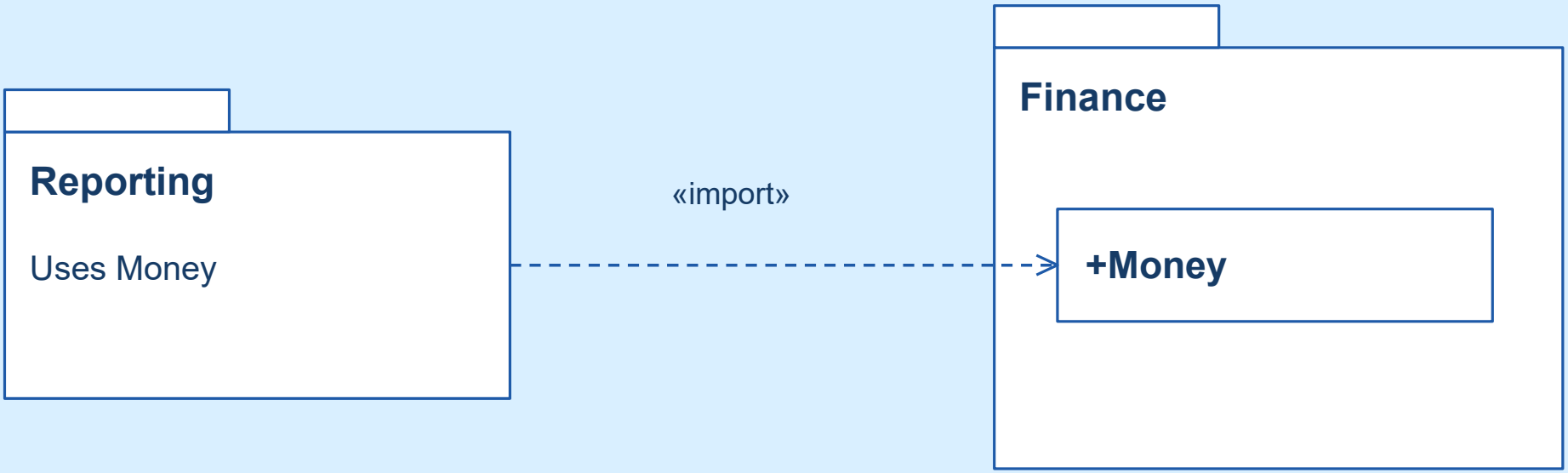


Dependent package

Supplier package

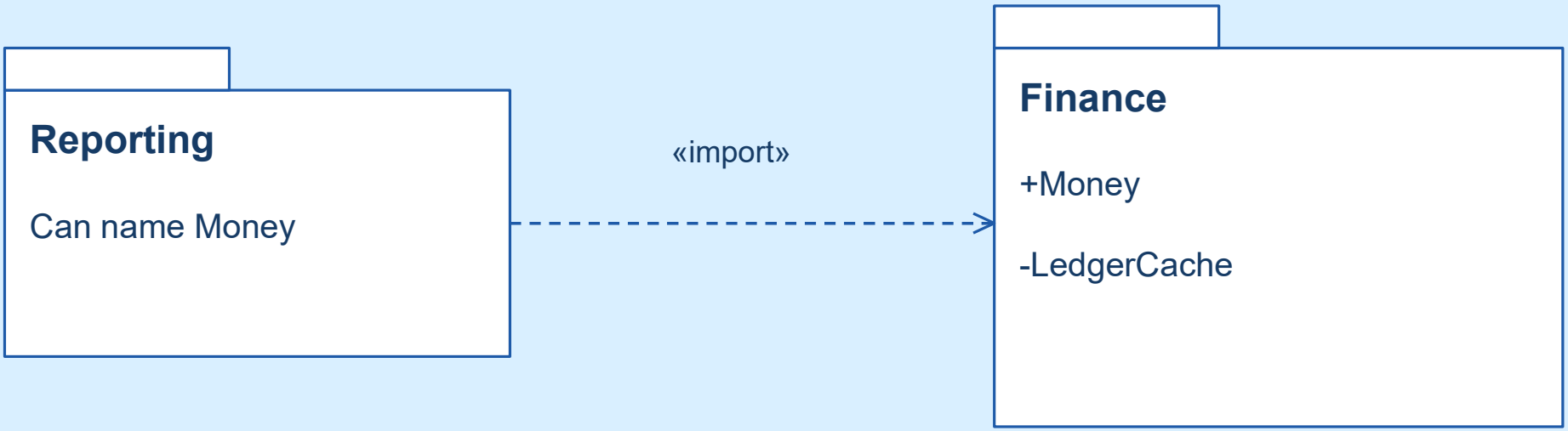
The arrow points from the dependent client to the supplier.

Element import: one named element



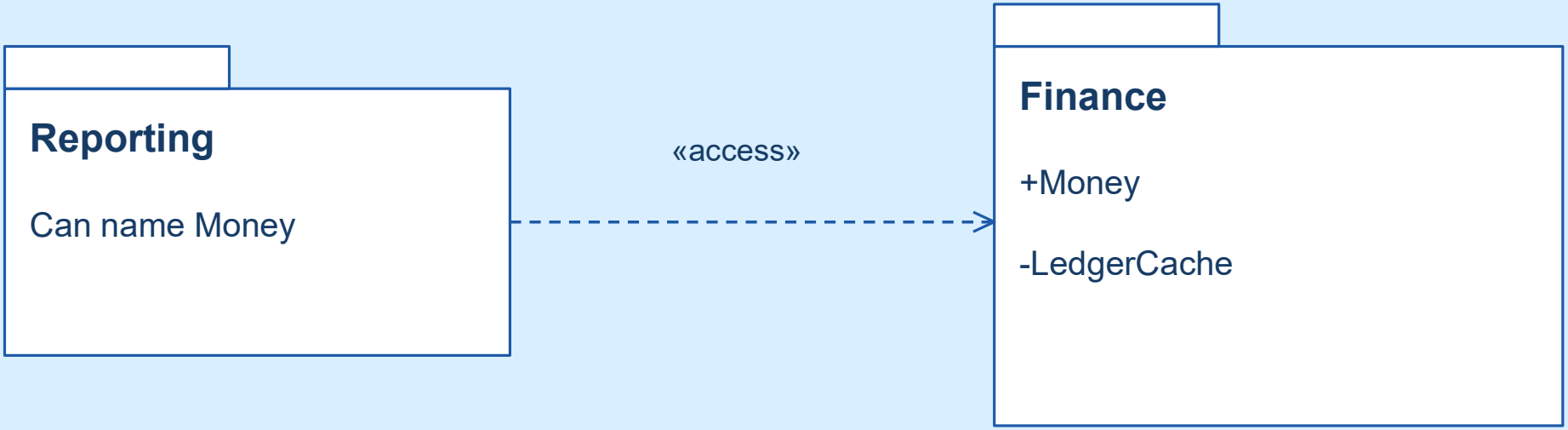
The import targets Money, so the client can refer to that element by its imported name.

Package import: public visibility



Public imported members can become visible through the importing package.

Package access: private import



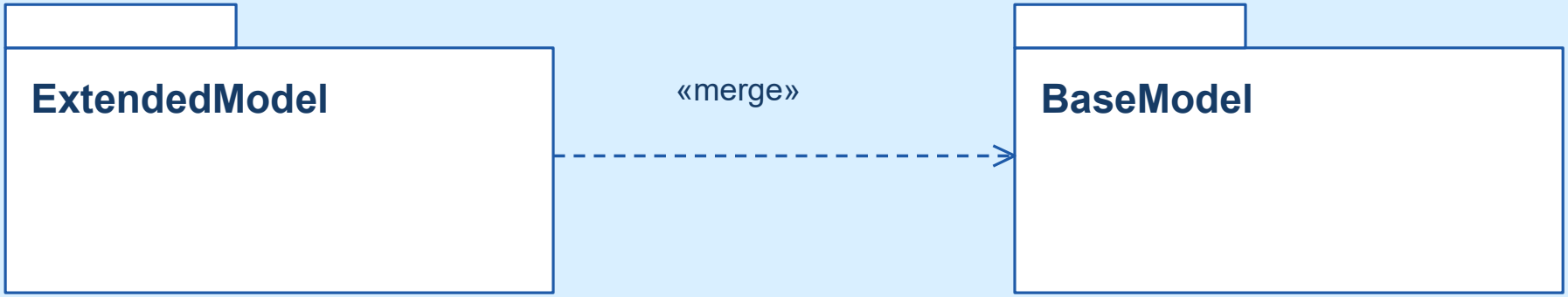
Reporting can use imported names internally without re-exporting them through this import.

Dependency, import and access

Relationship	Effect on names	Ownership
Dependency	No automatic name import	Unchanged
Element import	Adds one eligible element name	Unchanged
«import»	Public import, may re-export	Unchanged
«access»	Private import, no re-export	Unchanged

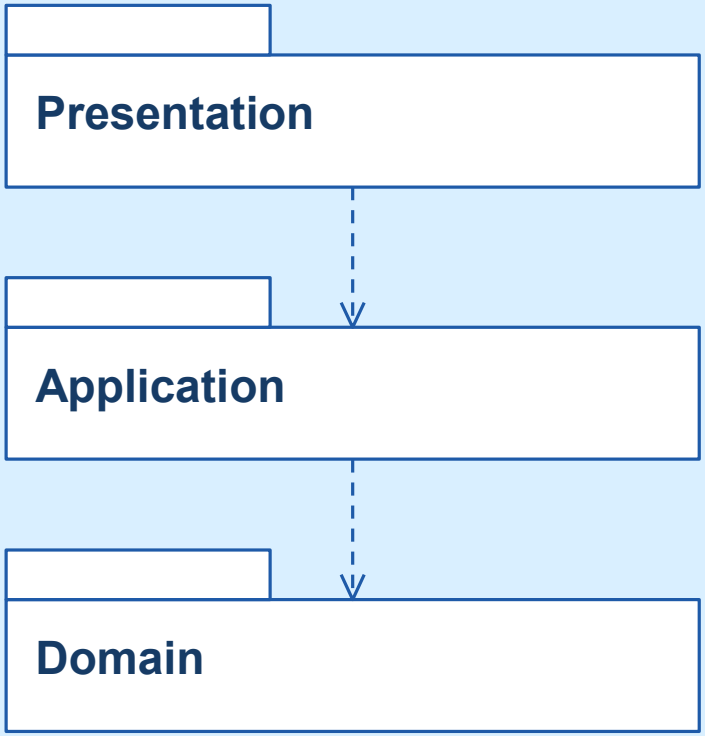
Public and private imports can target a package or an individual element.

Package merge: definition extension



The receiving package combines compatible definitions from the merged package.

Layered package structure

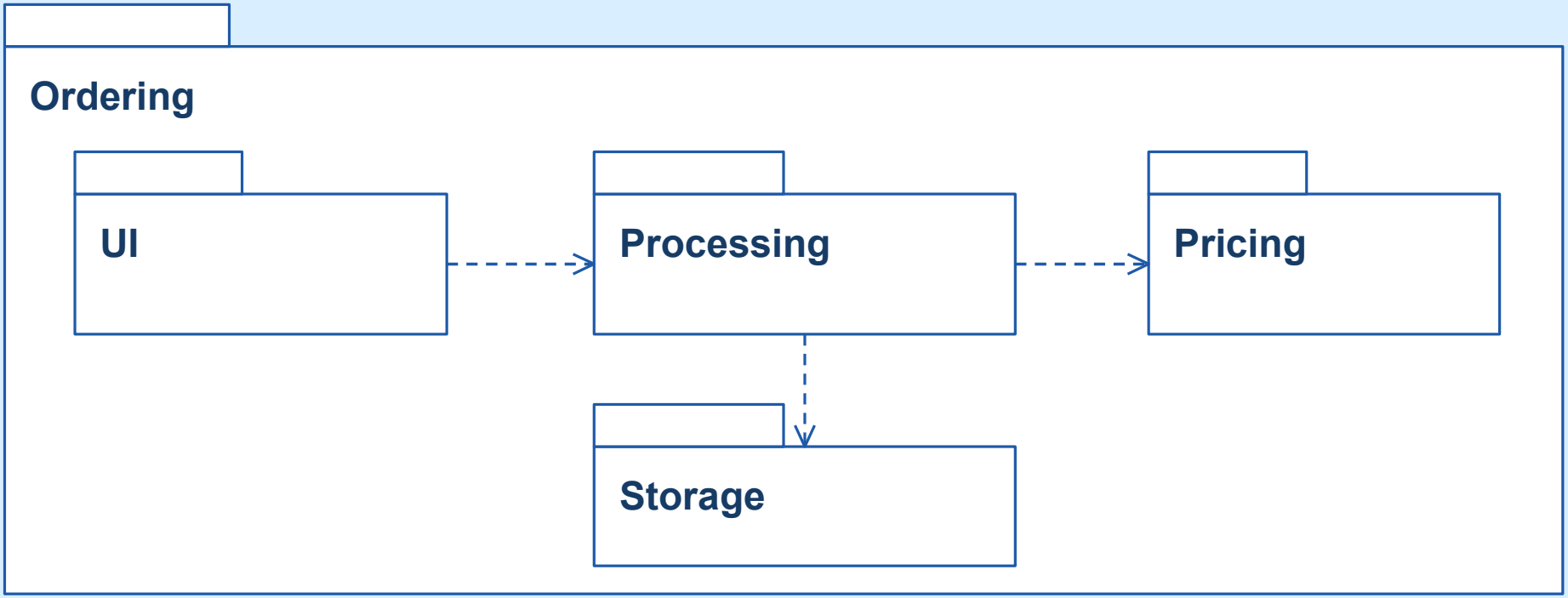


User-facing interaction

Use-case coordination

Business rules and concepts

Order system: package responsibilities



Cyclic dependencies and boundary review



Both packages depend on each other.

Review responsibility boundaries or move a shared contract into a separate package.

Workflow for building a package model

Step	Review question
1. Choose scope	Which part of the system does this view describe?
2. Group elements	Which elements share a responsibility?
3. Name packages	Are names clear within their namespaces?
4. Set boundaries	Which members should be public?
5. Add relationships	Is each relation dependency, import or merge?
6. Review coupling	Are cycles and cross-layer references justified?

Review and classroom practice

Questions for review

What does package nesting mean?

Does an import change ownership?

Can «access» import private members?

Does a dependency allow short names?

Practice: order reporting

Add a Reporting package.

Import one public OrderSummary type.

Keep the import private to Reporting.

Explain the arrow direction and ownership.

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. Chapter 7: Common Structure
and Chapter 12: Packages.

Examples adapted from the supplied package-diagram lecture.