Vietnam National University of HCMC
International University
School of Computer Science and Engineering

Object – Oriented Analysis and Design

Package Diagram

Instructor: Le Thi Ngoc Hanh, Ph.D

ltnhanh@hcmiu.edu.vn

Content

- Why package diagram.
- Approaches of breaking down large system to small system.
- Basics in package diagram.
- Benefits of package diagram.
- Reading: [R3] Chapter 5, Section 5.2

@Credit images:

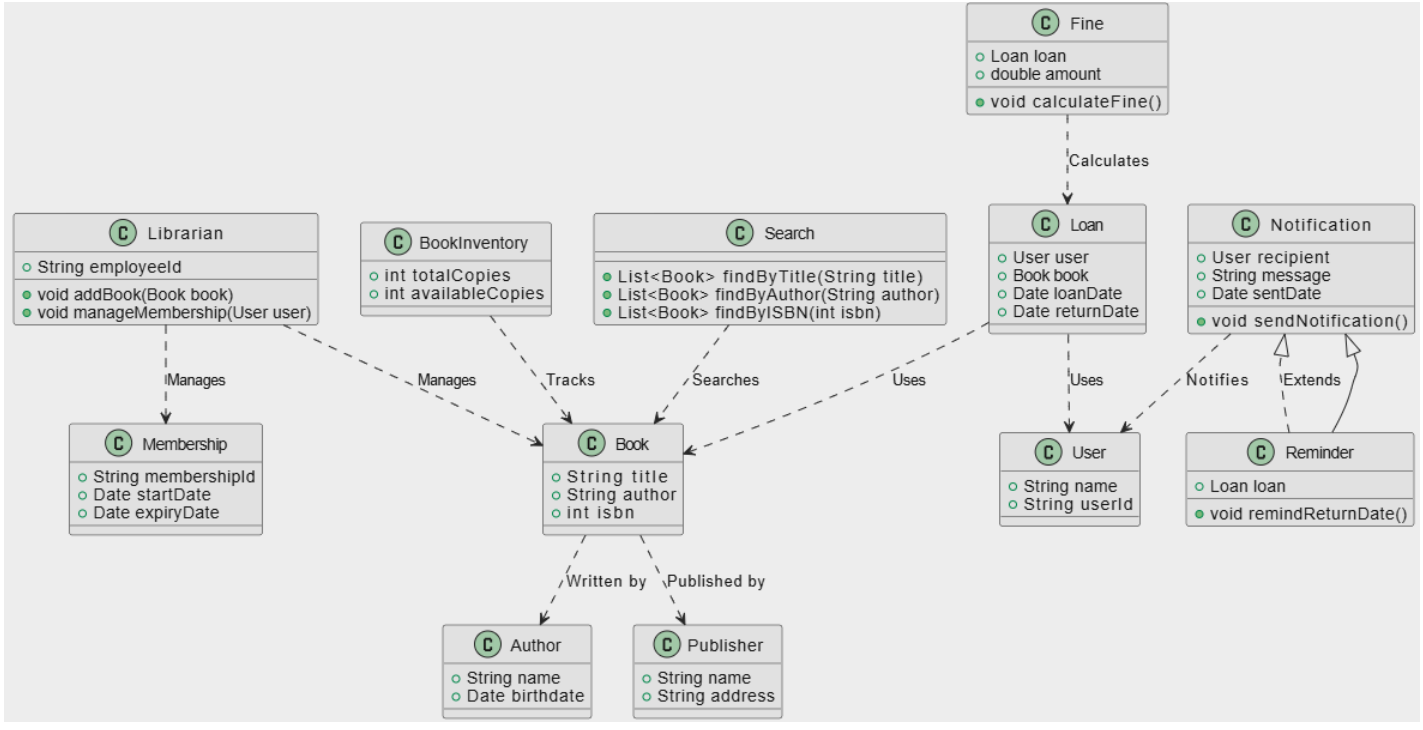
- uml-diagrams.org
- CSE's lecture

Why Package Diagram?

Why Package Diagram?

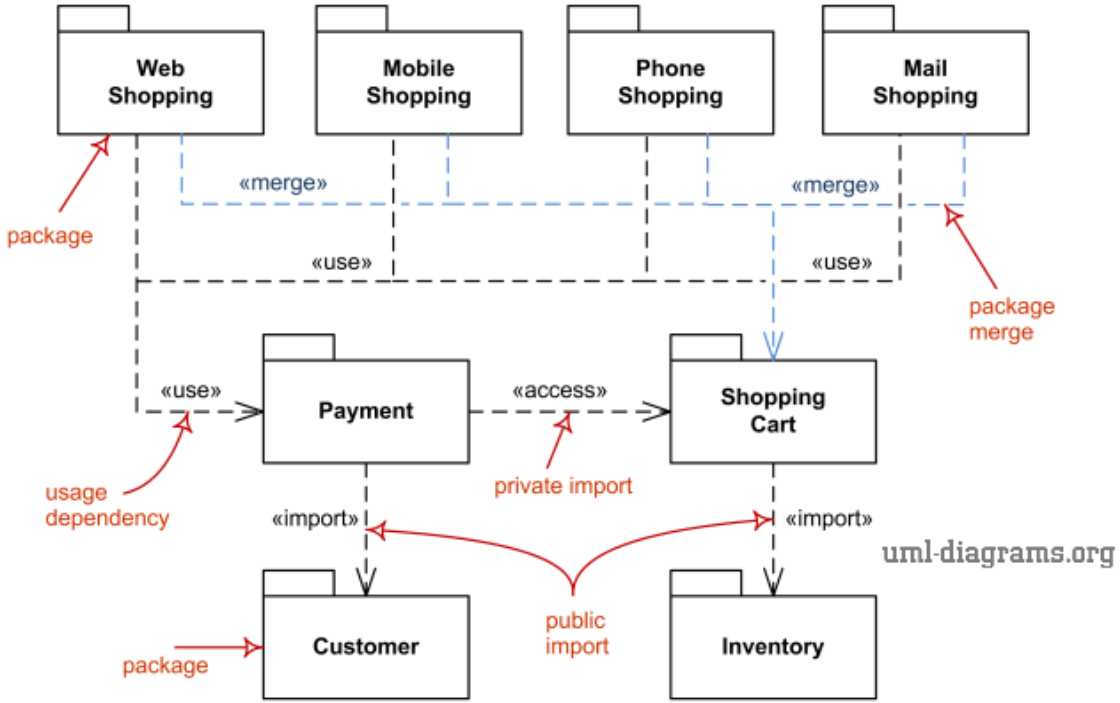
- How can we identify and design reusable components or modules within a system?
- Which parts of the system can be reused in other systems or projects?
- How do we organize a large number of classes and components into logical groups for better understanding and maintenance?
- How can we make a complex system more manageable and modular?
- How can we separate different responsibilities or functionalities in our system to ensure clear boundaries?
- How can we design a system so that changes in one part have minimal impact on others?
- If the system needs to scale, how can we identify which packages to modify or extend?
- How do we design the system to accommodate future changes with minimal disruption
- ...

Flat structure



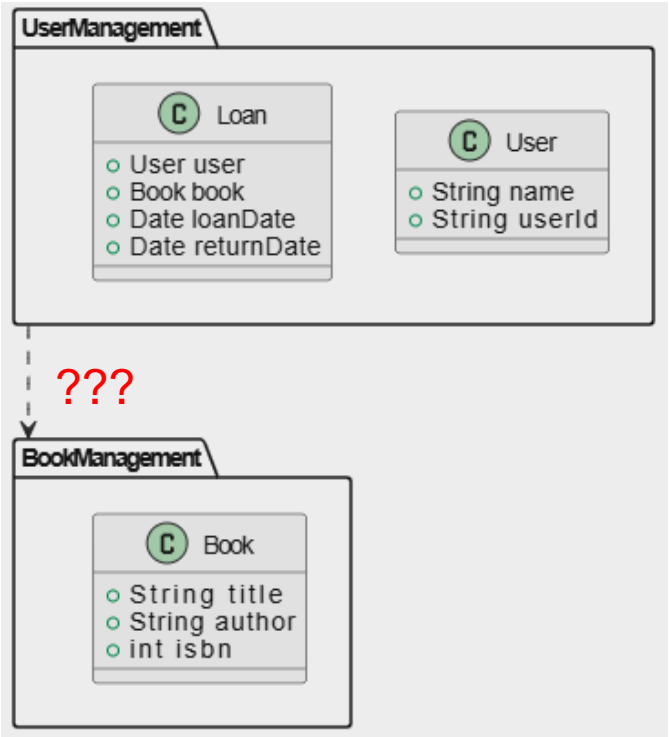
How do you break down a large system into smaller system?

Package Diagram



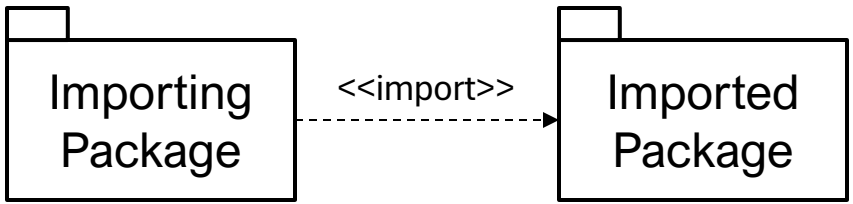
Dependency relationship

- Dependency relationship is used to show how one package relies on another.
- Indicating the changes or updates in one package might affect the dependent package.

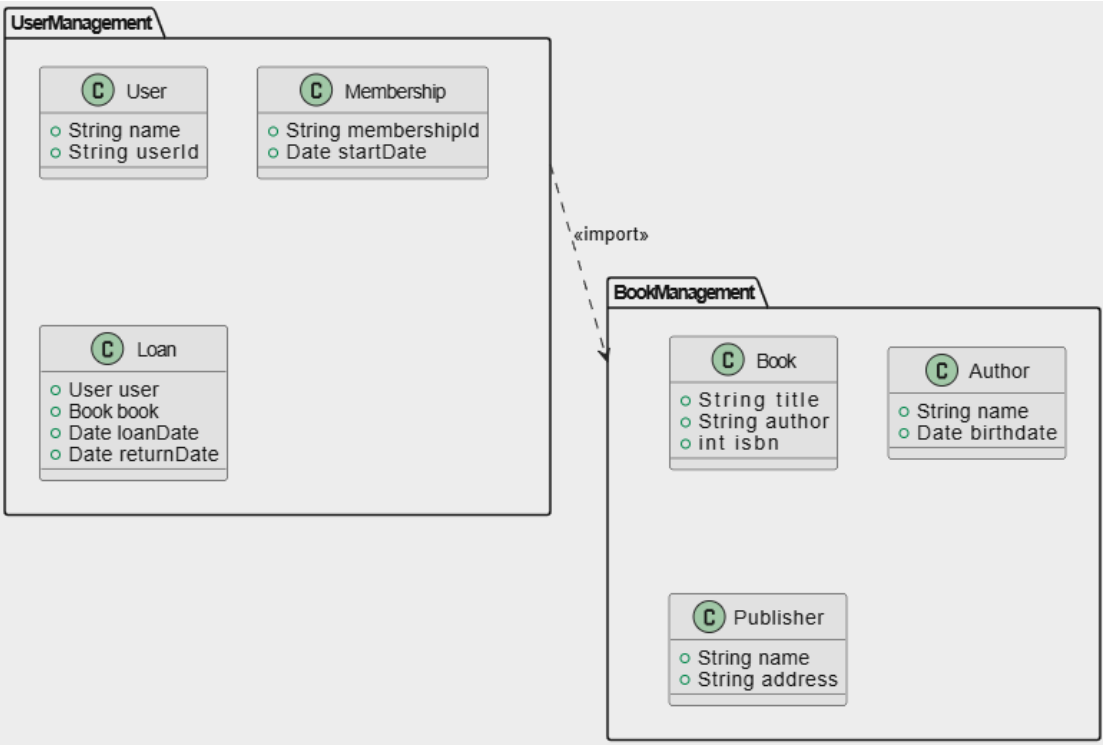


Import Packaging

- When one package *imports* another, it means the importing package has visibility to the elements (e.g., classes) of the imported package.
- The importing package can use the imported package's public elements as if they were part of its own package.



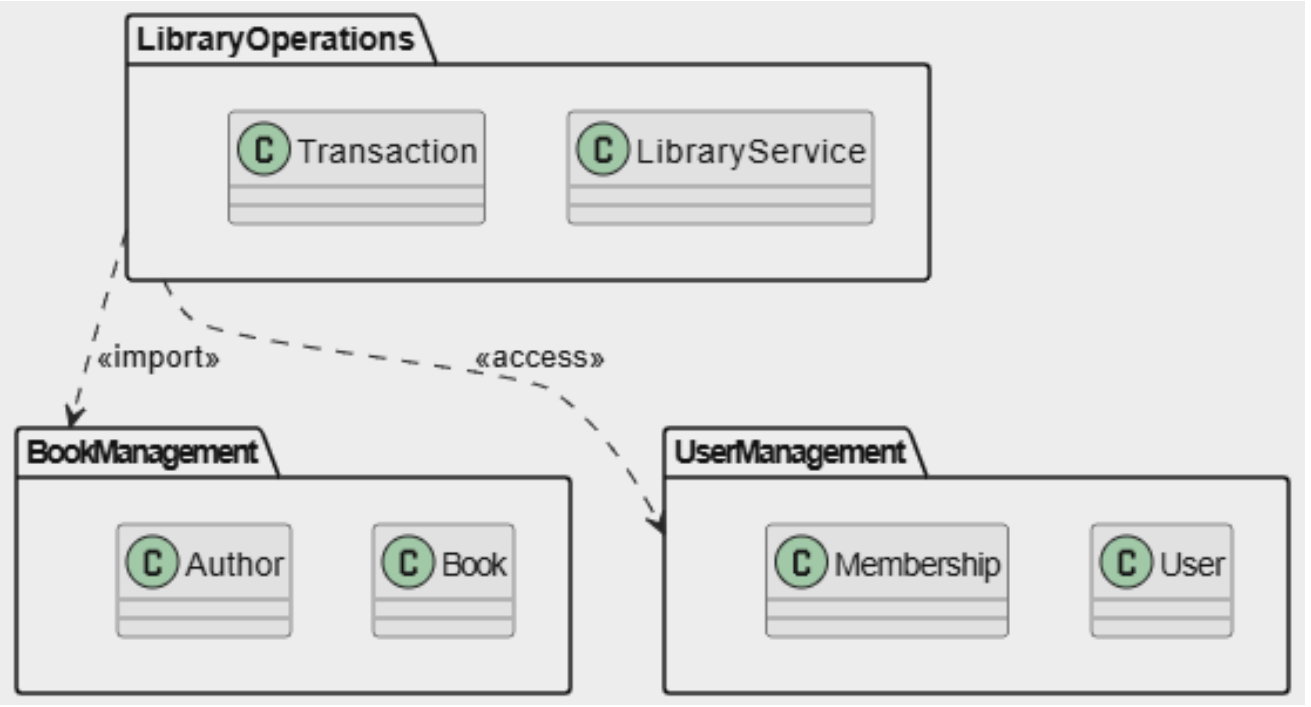
Import Packaging



Access Packaging

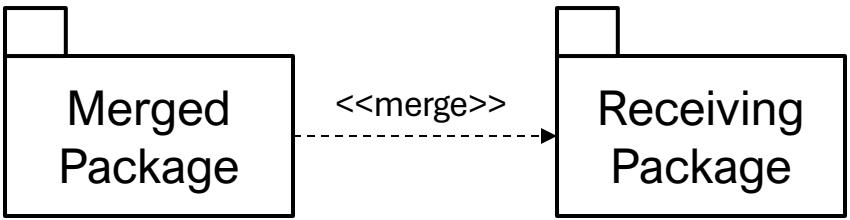
- When one package *accesses* another, it means the accessing package references specific elements of the accessed package without importing all its elements.

Access Packaging



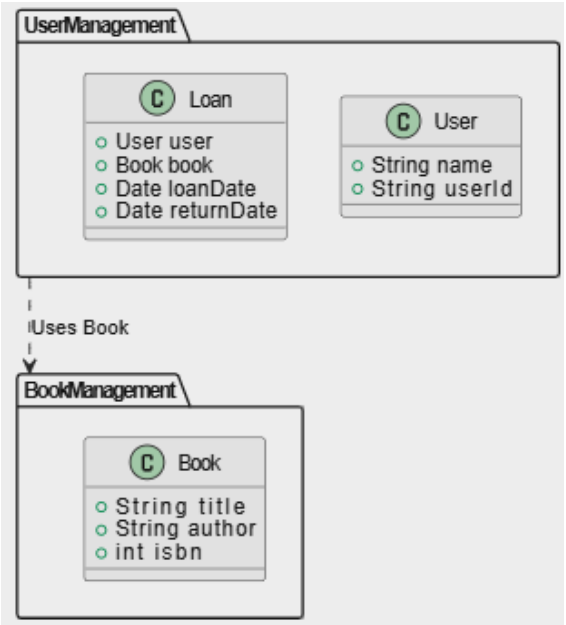
Merge Packaging

A PackageMerge is a directed relationship between two Packages that indicates that the contents of the target mergedPackage are combined into the source receivingPackage according to a set of rules.

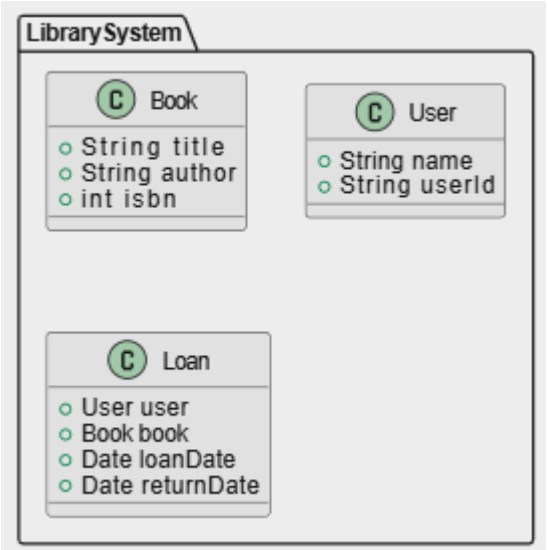


Merging Package

Before merging



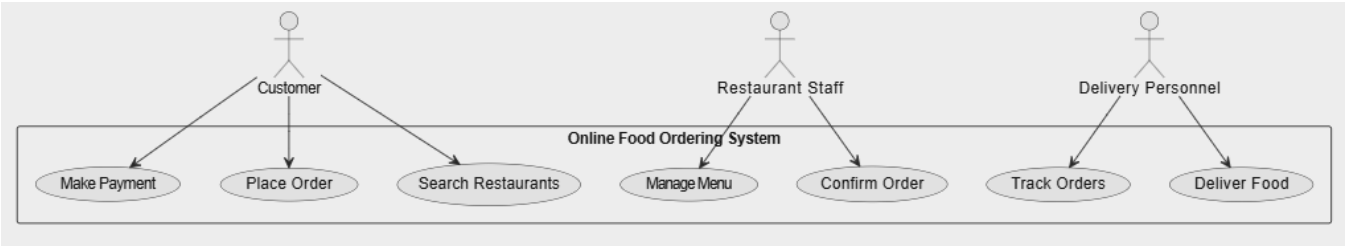
After merging



Benefits of Package Diagrams

- Simplifies large systems
- Enhances maintainability
- Promotes reusability
- Supports scalability
- Supports better communication
- What else???

Extend to Use case diagram



Extend to Use case diagram