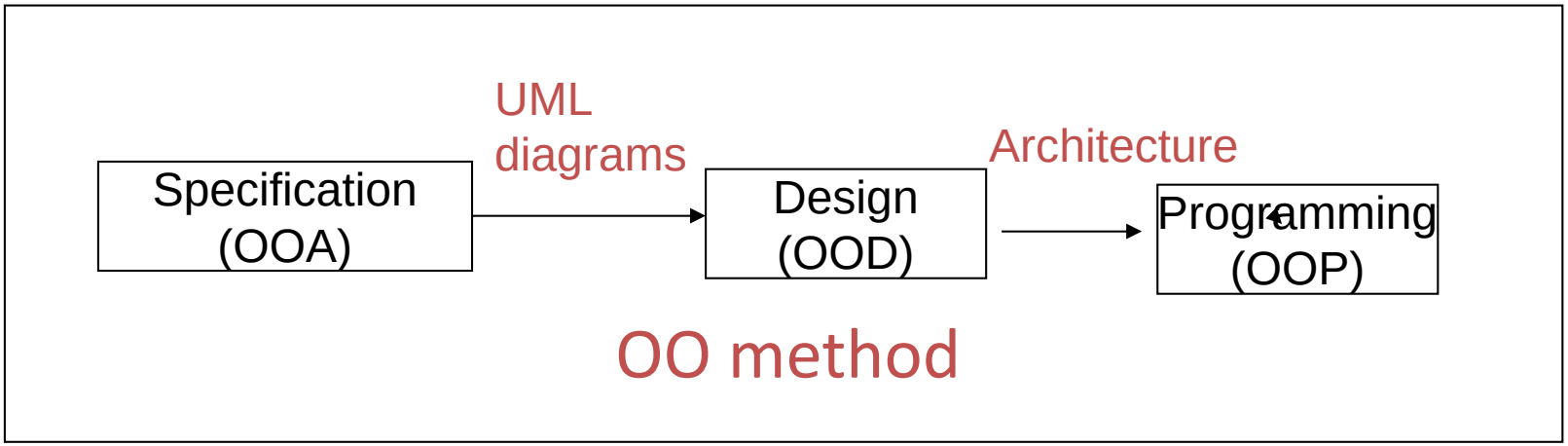
Object-Oriented Design

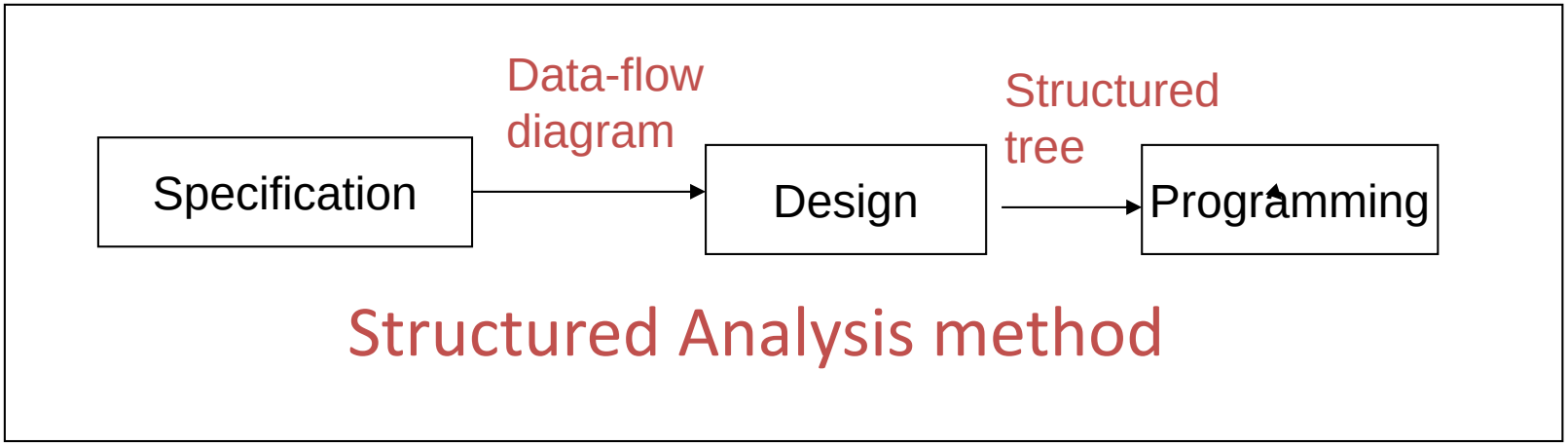
Xin Feng

Outline

- OOD
- OOD models

Object-Oriented Design

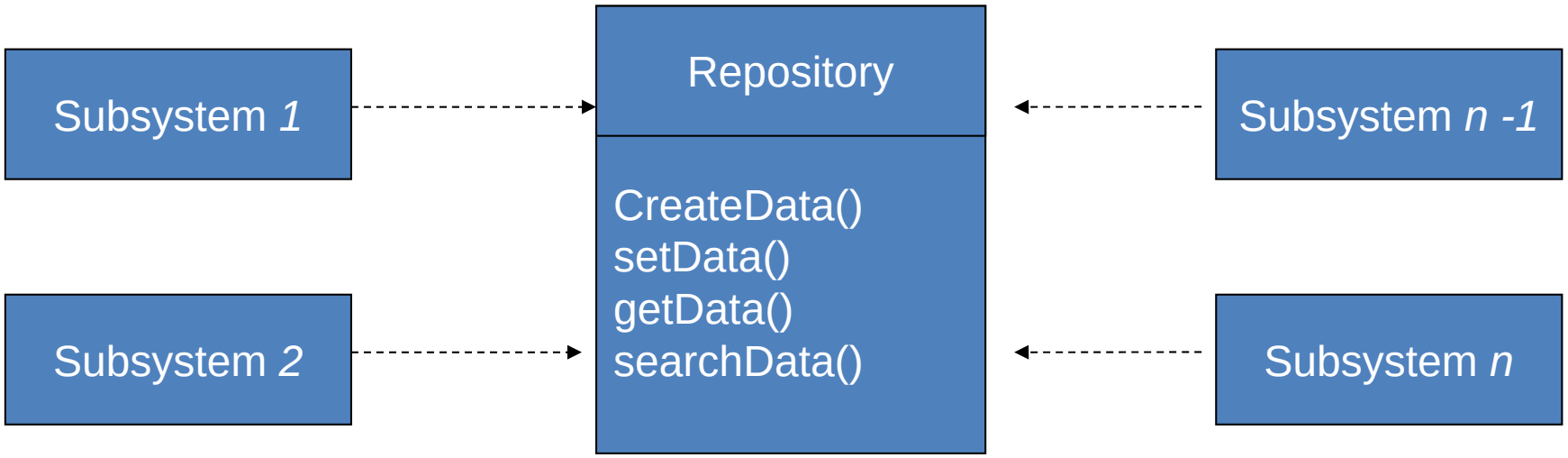




System Decomposition

- System organization models
  - The repository (存储库) model
  - The client-server （客户-服务）model
  - The layered （分层次的）model
  - MVC model ……

The Repository Model



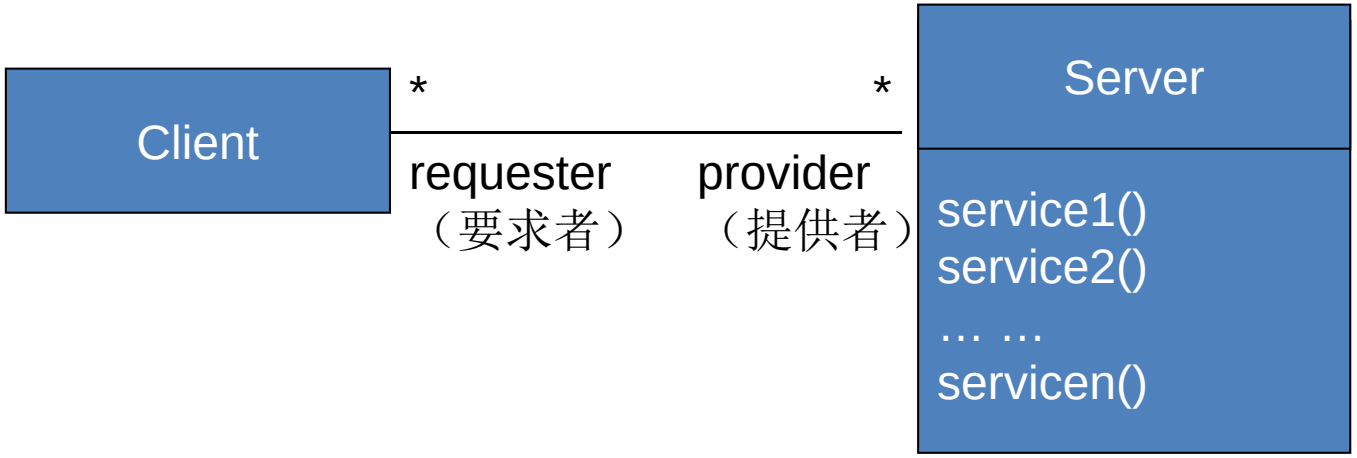
The Repository Model

- Subsystems access and modify the repository
- Typically used in
  - Bank system
  - Compiler
  - Software development environment
- Repository should be well defined
  - New services can be easily added

The Repository Model

- Disadvantages
  - Performance (bottleneck（瓶颈）)
  - Coupling （耦合）is high
  - Hard to modify the data models

The Client-Server Model



The Client-Server Model

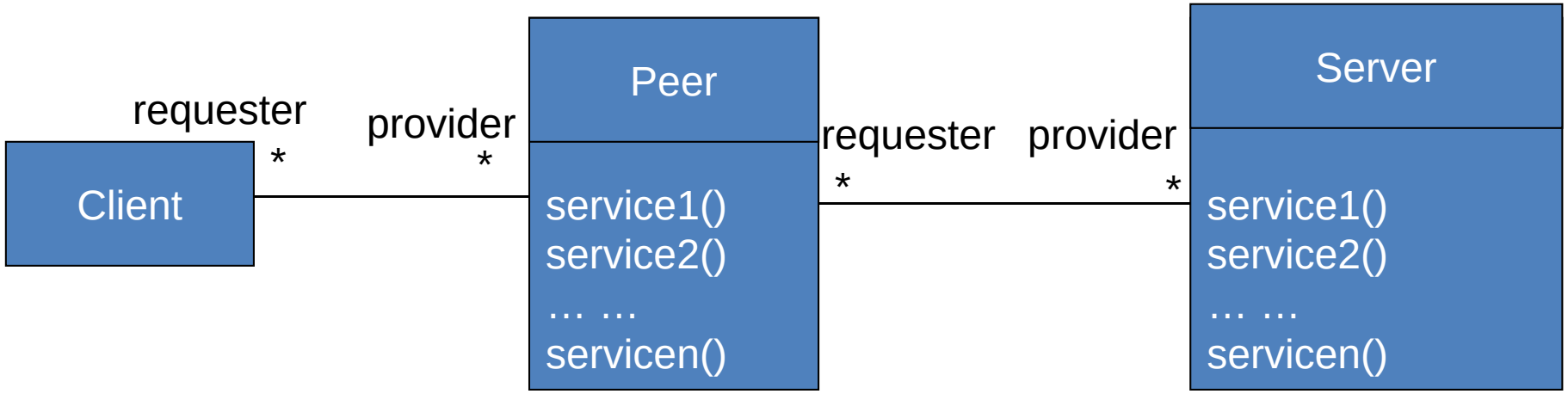
- A server (subsystem) provides services to instances （实例） of clients (subsystems)
- Usually go through internet
- Many to many relationships

The Client-Server Model

- E.g. Booking and flight
  - Client
    - Booking
  - Server
    - Flight

The Peer-To-Peer Model

- A special case of the client server style
- A subsystem can be both as a client and as a server



The Peer-To-Peer Model

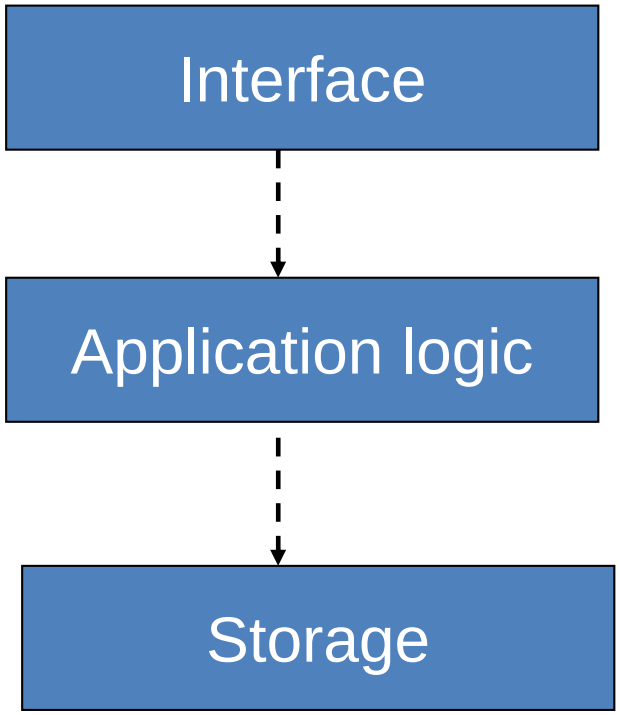
- E.g. Database system
  - The data in the database is updated according to the request from an application
  - It can notify （通知）another application of this change.

The Layered Model – Three Tier

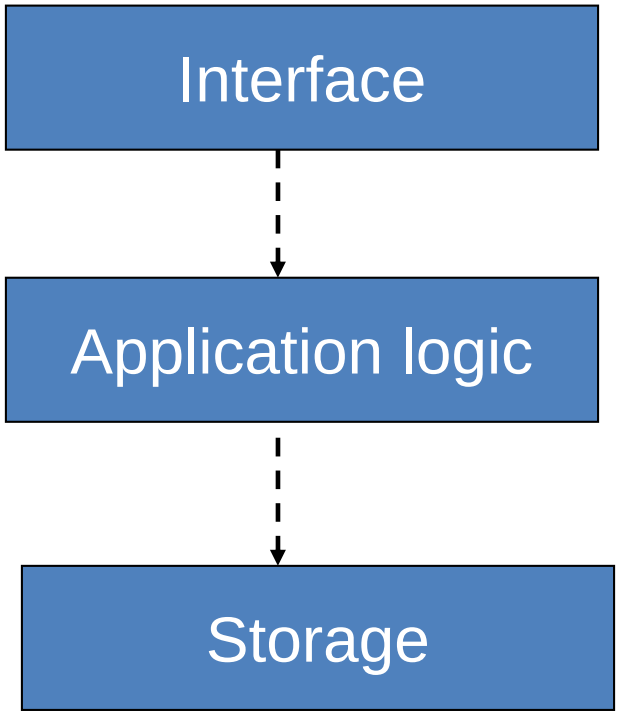
Include all boundary objects that deal with （处理）interactions with users (forms, Windows)

Include all control and entity objects

Realizes （实现） the storage, retrieval, and query （查询）of persistent （永久） objects



The Layered Model – Three Tier



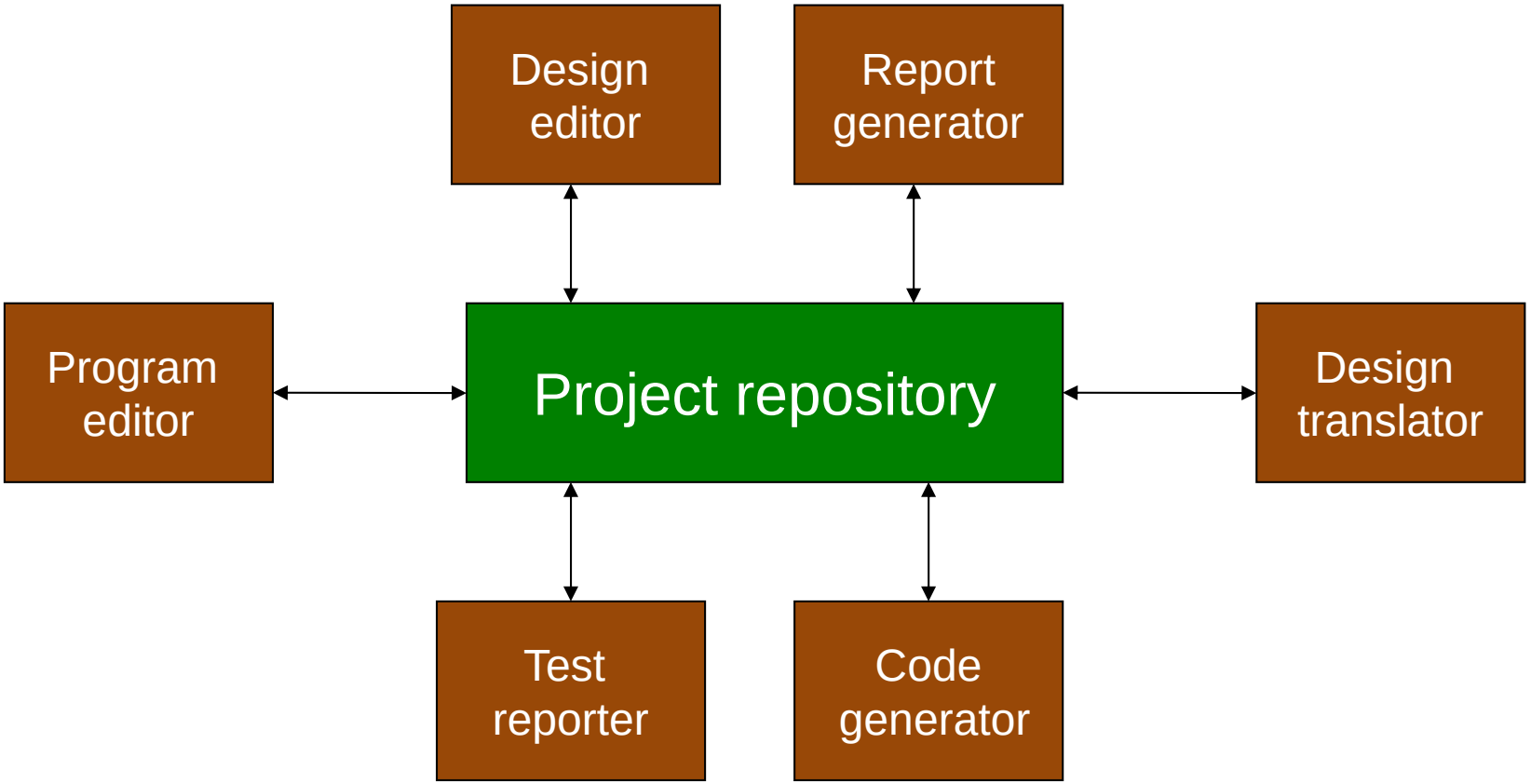
Can be shared by different subsystems with same application logic and storage (存储) but with different interfaces （界面，接口）

Can be shared by different subsystems

Combined use of models

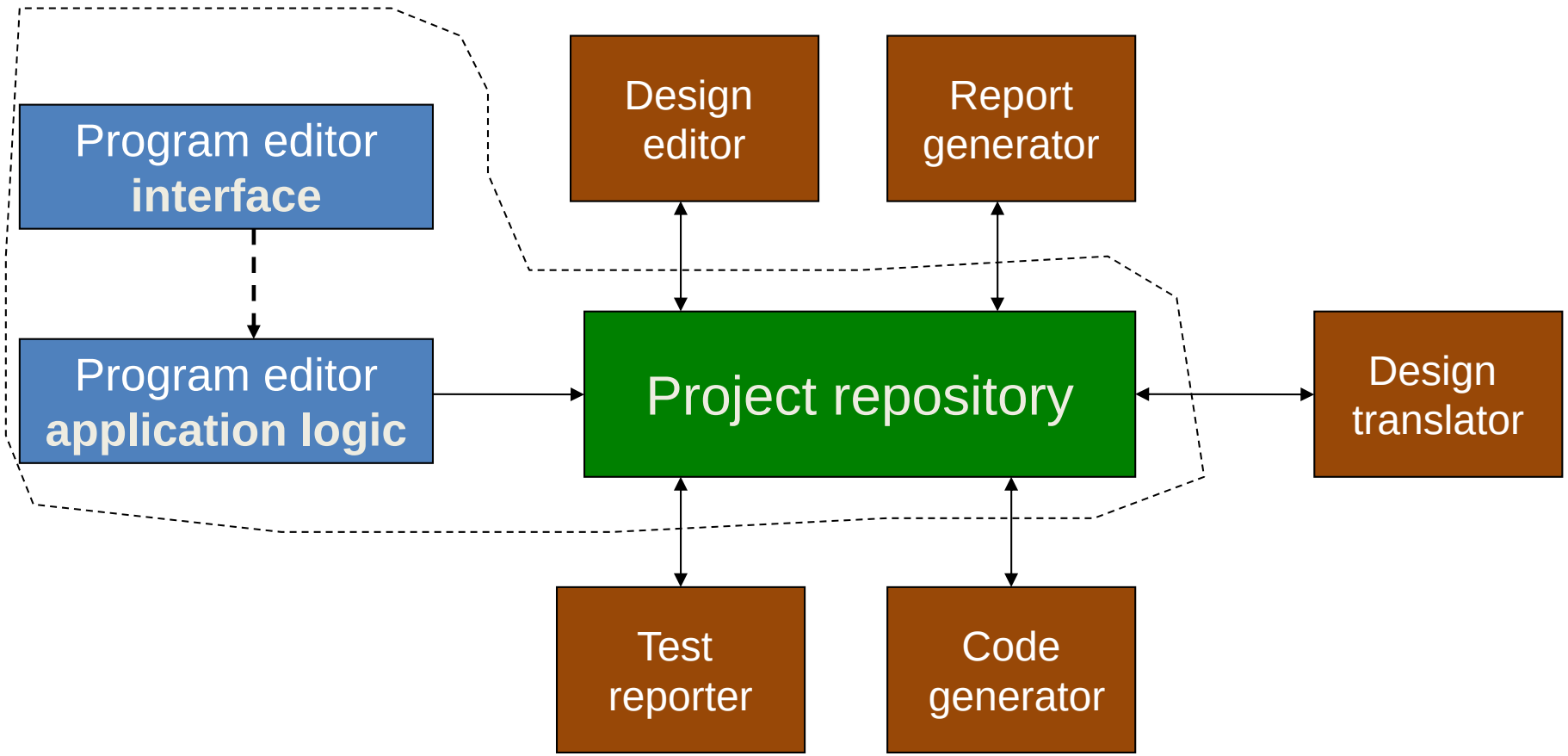
Several models can be used together
in one system

The Repository Model – An Example



CASE Toolset Architecture

Connected with Repository Style



CASE Toolset Architecture

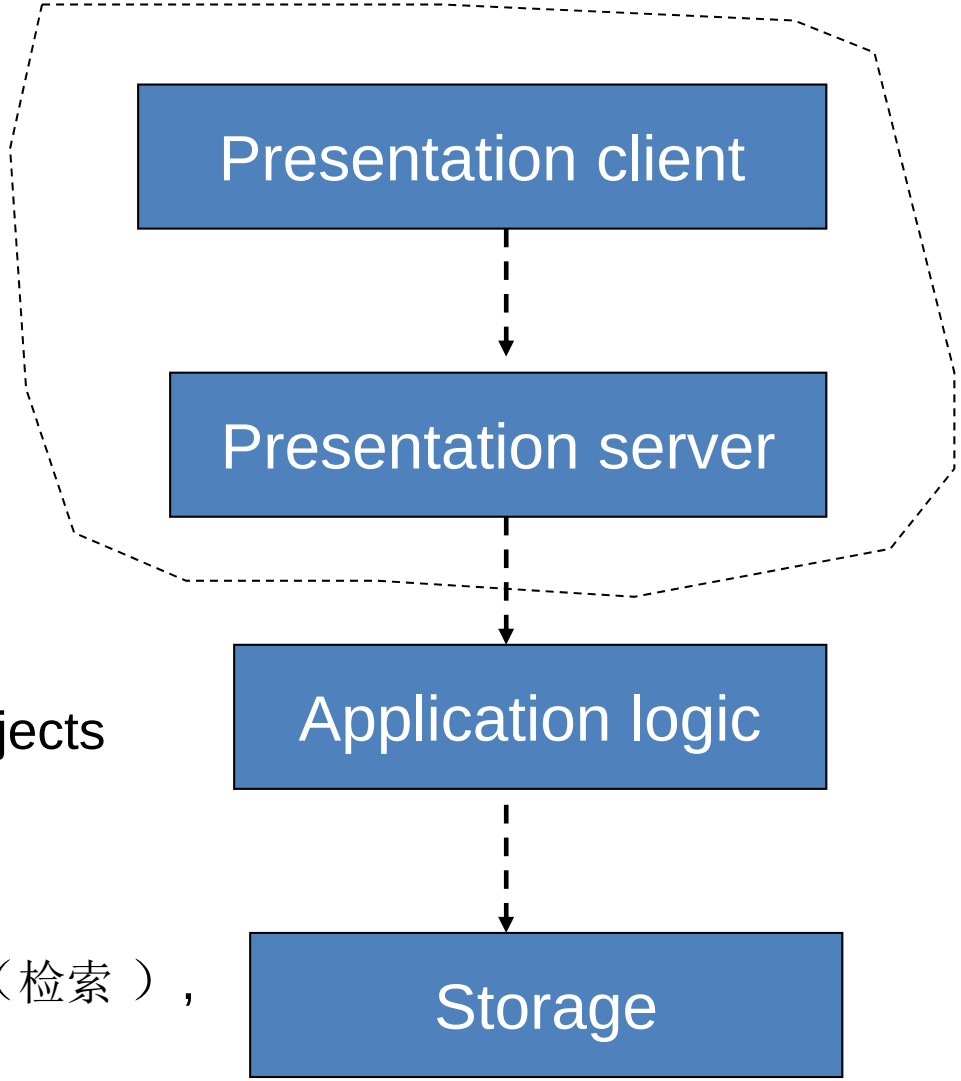
The Layered Model – Four Tier

Located on the user machine

Located on one or more servers

Include all control and entity objects

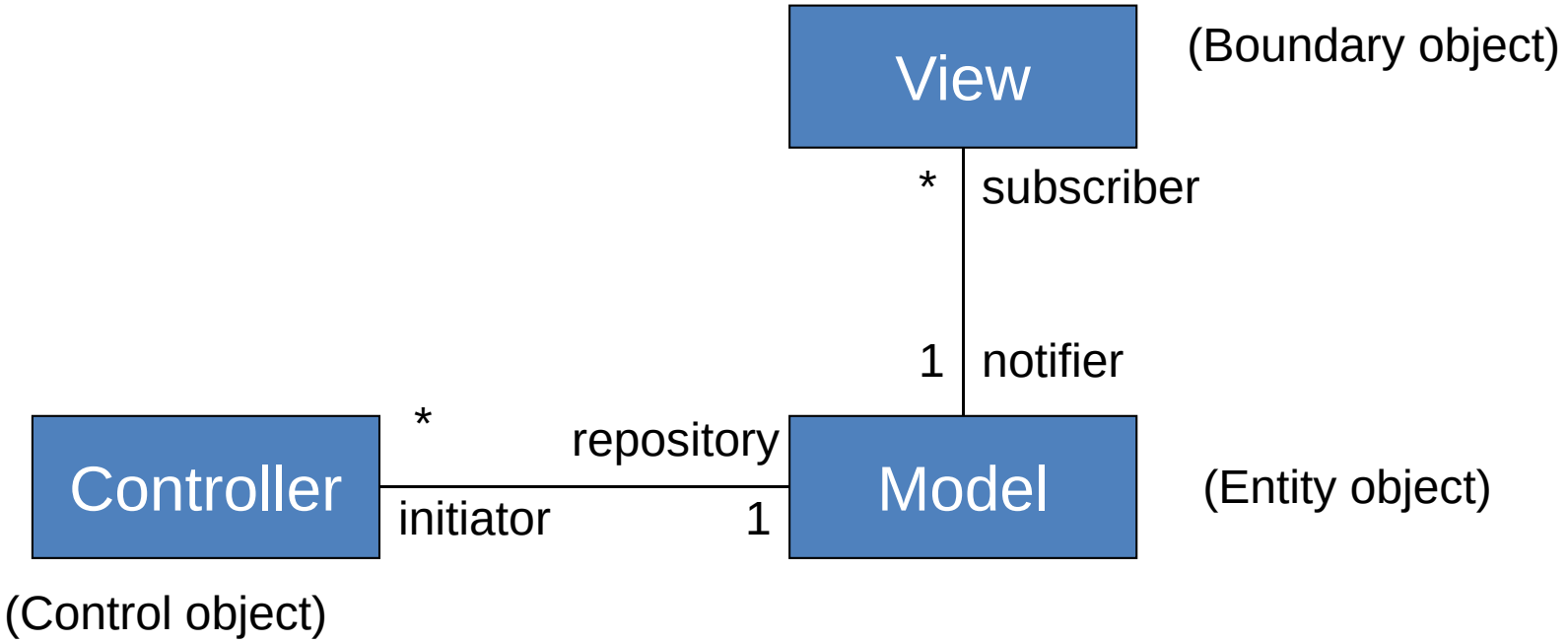
Realizes the storage, retrieval （检索）, and query of persistent objects



MVC Model

- Three subsystems
  - Model subsystems: maintain the real world objects
  - View subsystems: display the data in model subsystems to users
  - Controller subsystems: manage the sequence of interactions with the users
- Model does not depend on any view or controller subsystems
- Changes in the models are detected （探测）and displayed by view subsystems

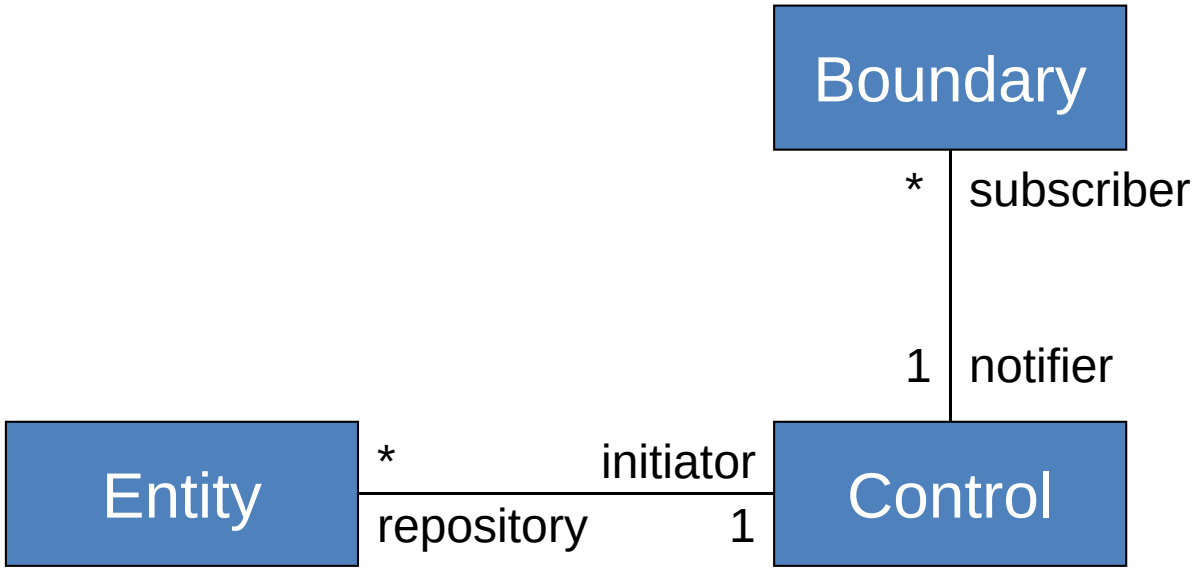
MVC Model


View
(Boundary object)
*
subscriber
1
notifier
*
repository
Controller
Model
(Entity object)
initiator
1
(Control object)

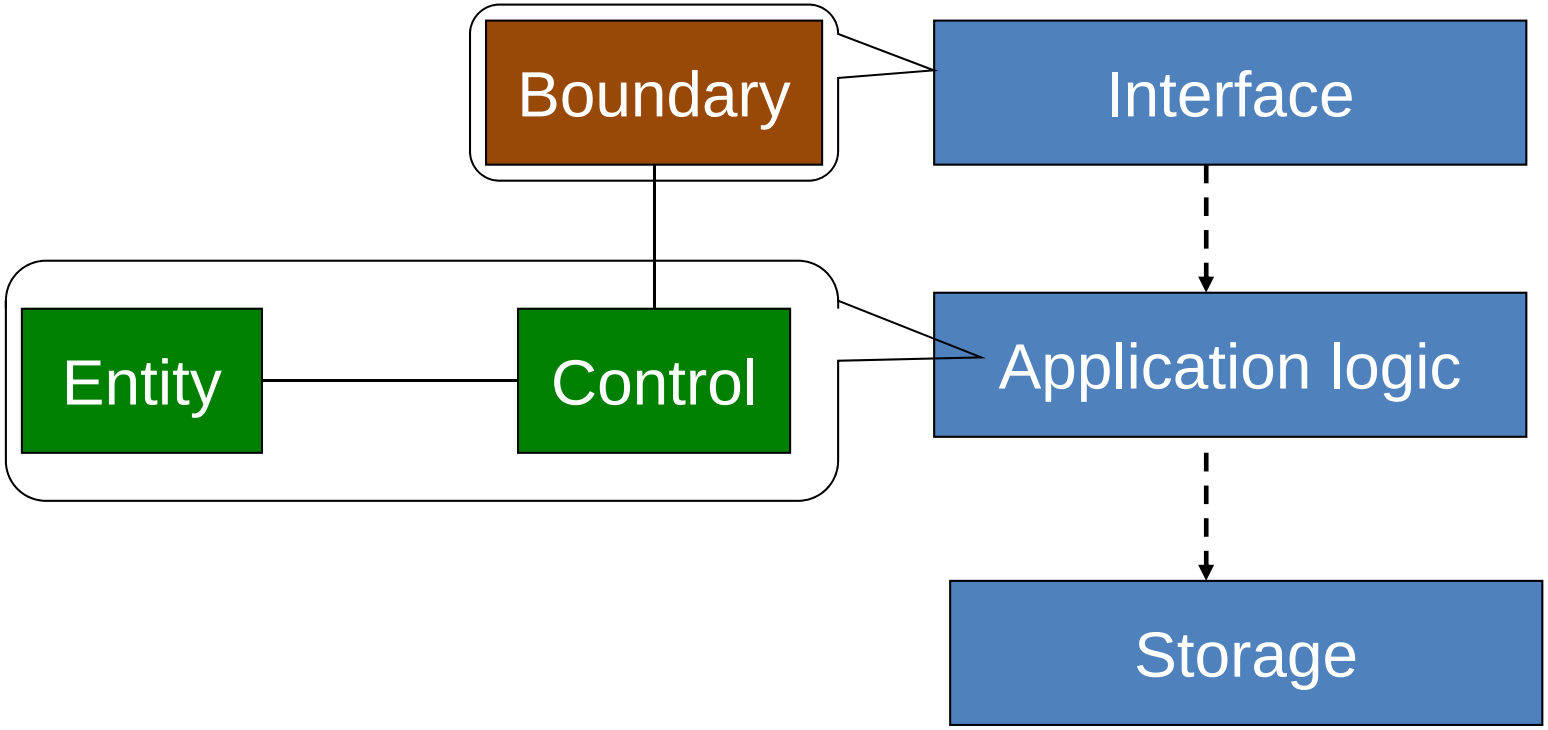
MVC Model

- Controller subsystem gets input from users and send messages to model subsystems
- View displays the information in model and are notified of the changes in the model subsystems
- This style is suitable for interactive system
- Bottleneck （瓶颈）problems

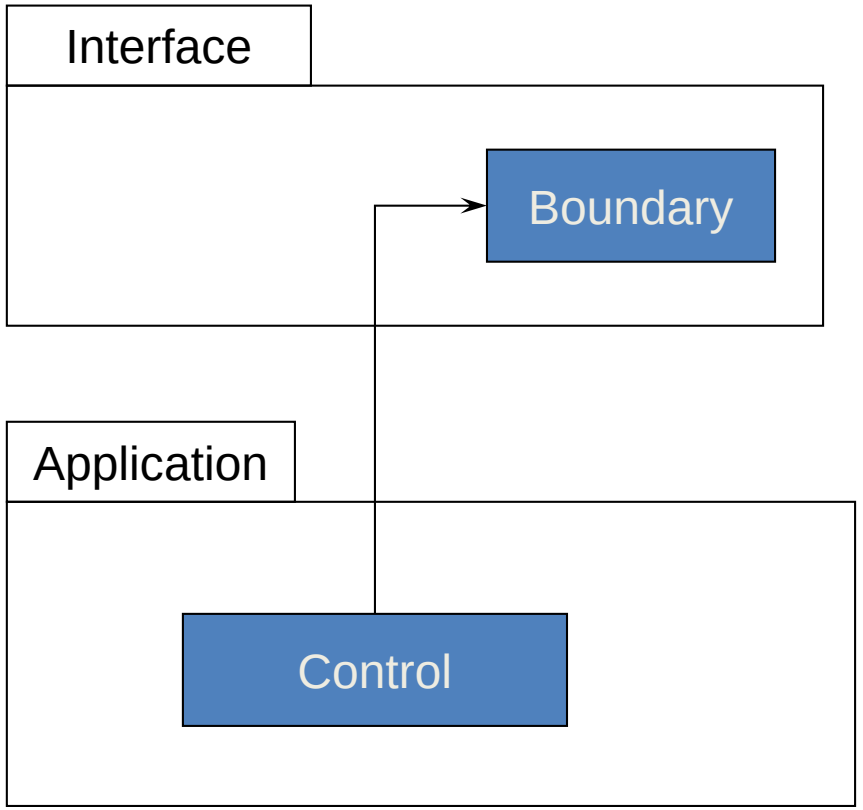
MVC UML Model



The Layered Model – Three Tier



Observer Design Pattern - MVC

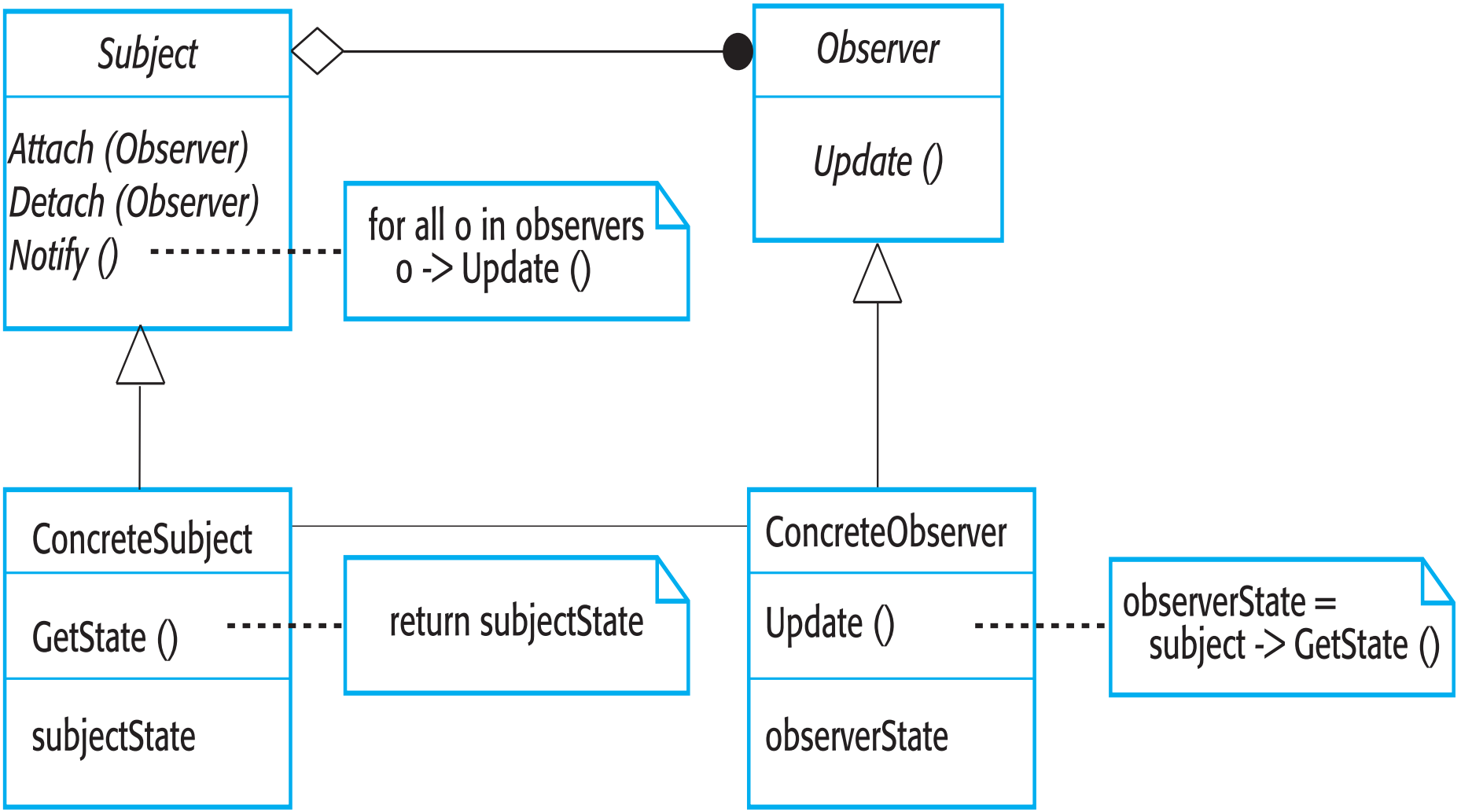


?

Design Patterns

- A pattern （模板）is a description of the problem and the essence （基本）of its solution.
- Be sufficiently （充分）abstract -> reused in different settings （设置）
- Pattern elements
  - Name: pattern identifier
  - Problem : describe the problem to solve
  - Solution: describe the solution to the problem
  - Consequences （后果）: results and trade-off in applying this pattern

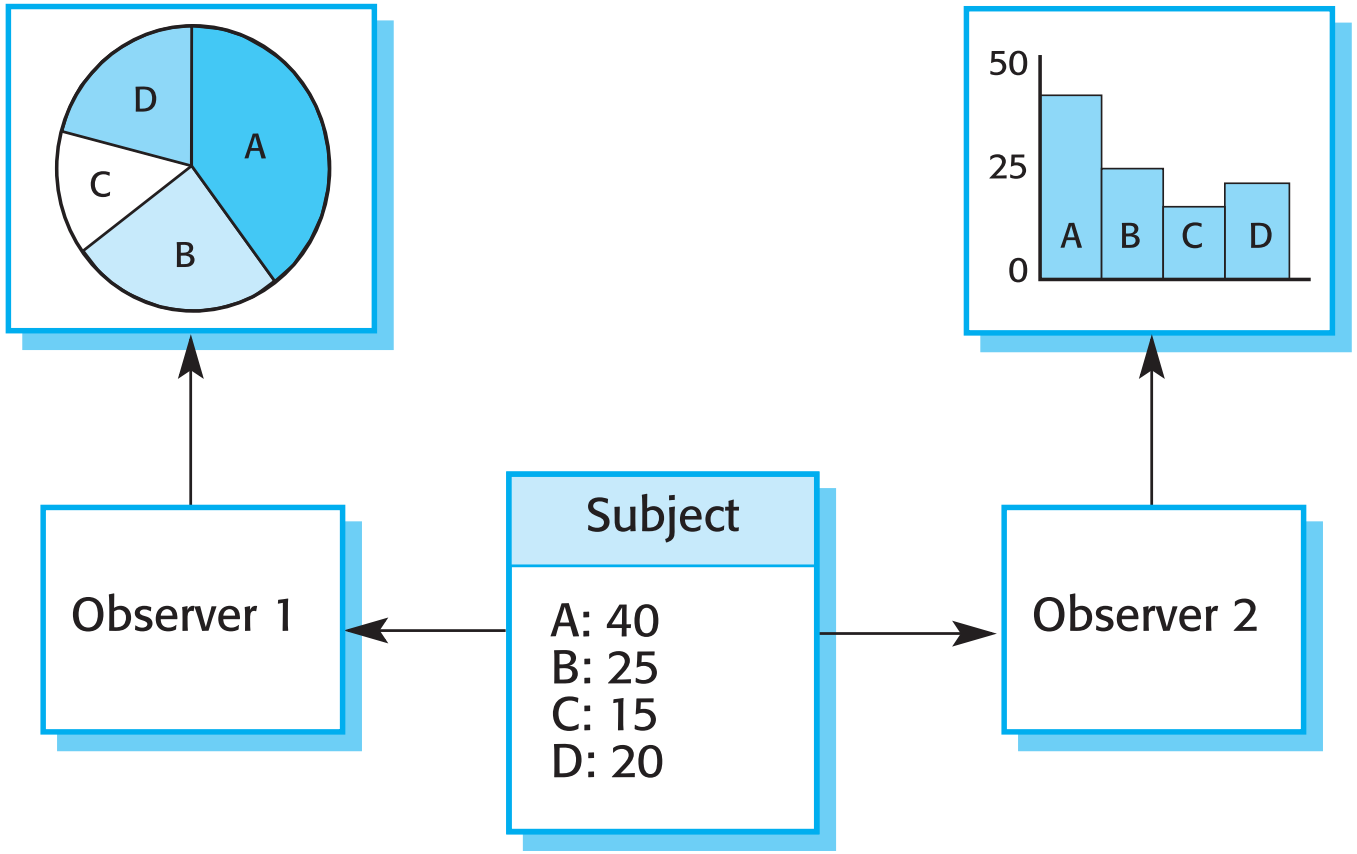
Observer Design Patterns



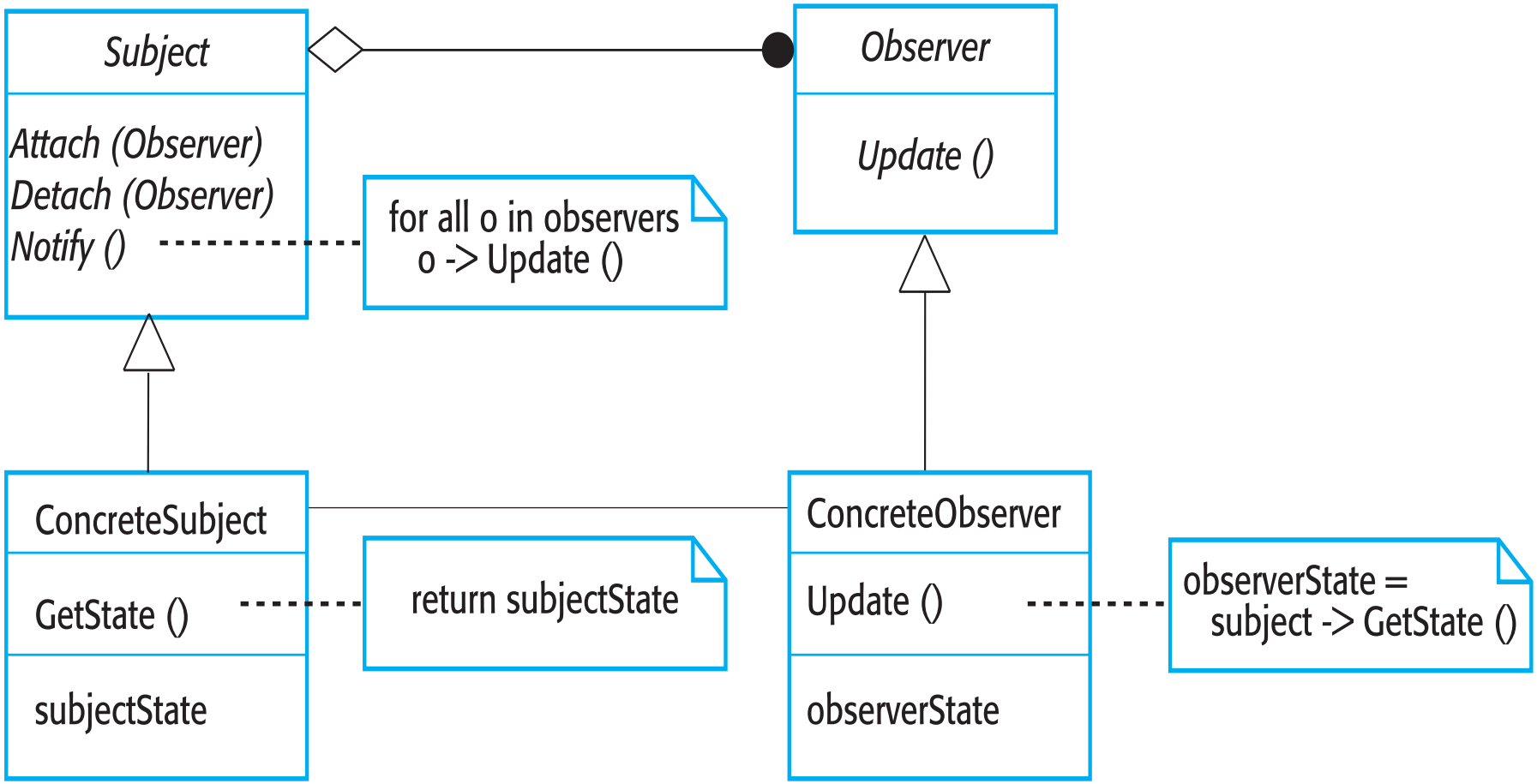
Observer Design Pattern

- Name
  - Observer
- Problem description
  - Used when multiple displays of state are needed
- Solution
  - A subject is an object whose primary function is to maintain some state
  - One or more observers use the state maintained by the subject
- Consequences
  - Decouple a subject from the observers
  - Result in many false broadcasts when the state of a subject changes

Observer Design Pattern



Observer Design Patterns



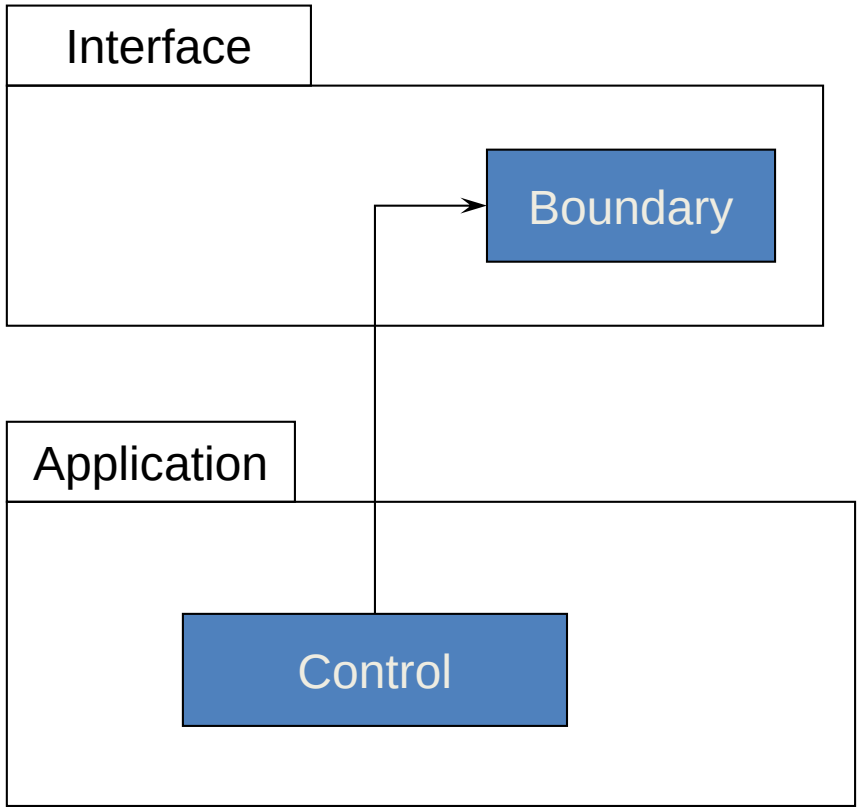
Observer Design Pattern

- Attach() (增加)
  - Add a new observer to the list of observers observing the subjects
- Detach() （移走）
  - Removes an existing observer from the list of observers observing the subject
- Notify()
  - Notifies each observer by calling the update function in the observer

Observer Design Pattern

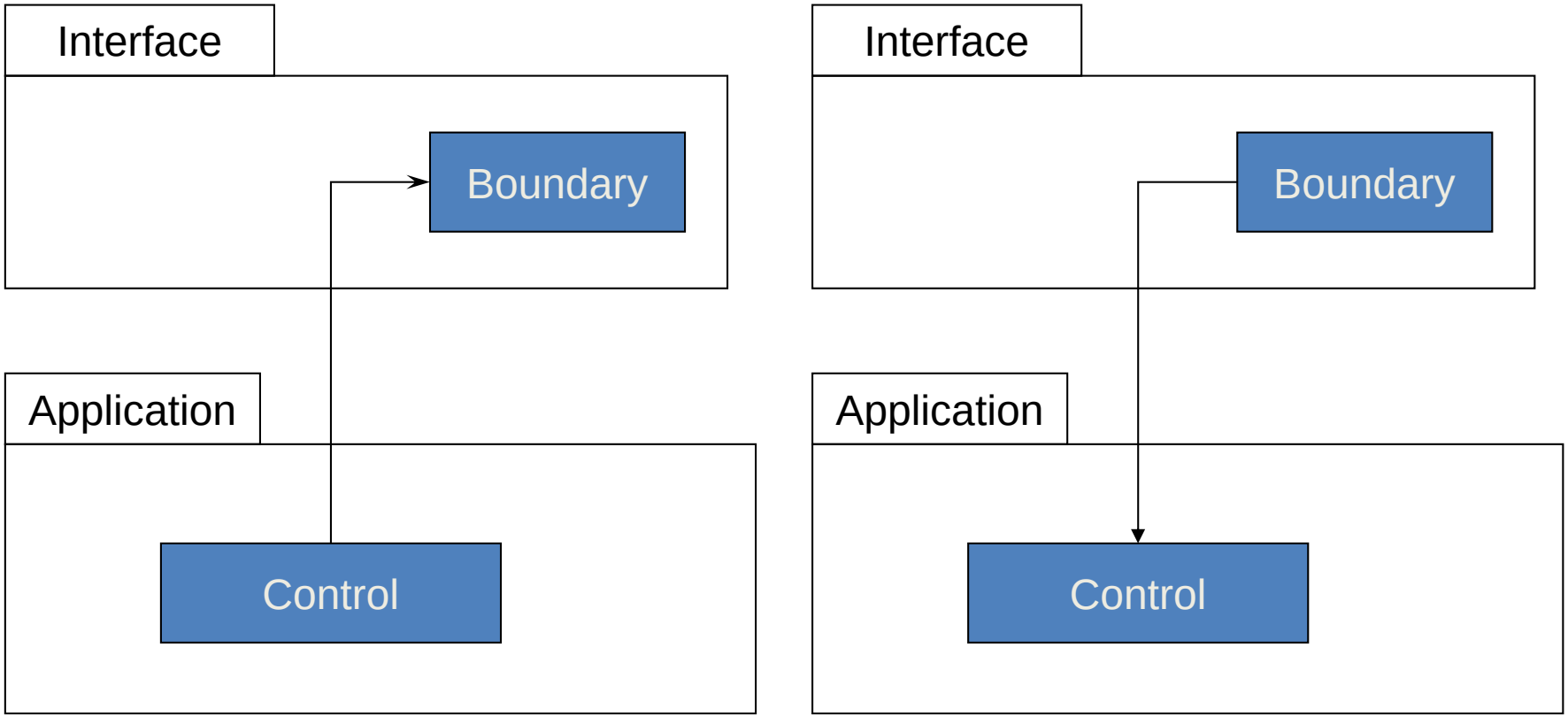
- Examples
  - MVC Model
  - Java observer interface and observable class

Observer Design Pattern - MVC

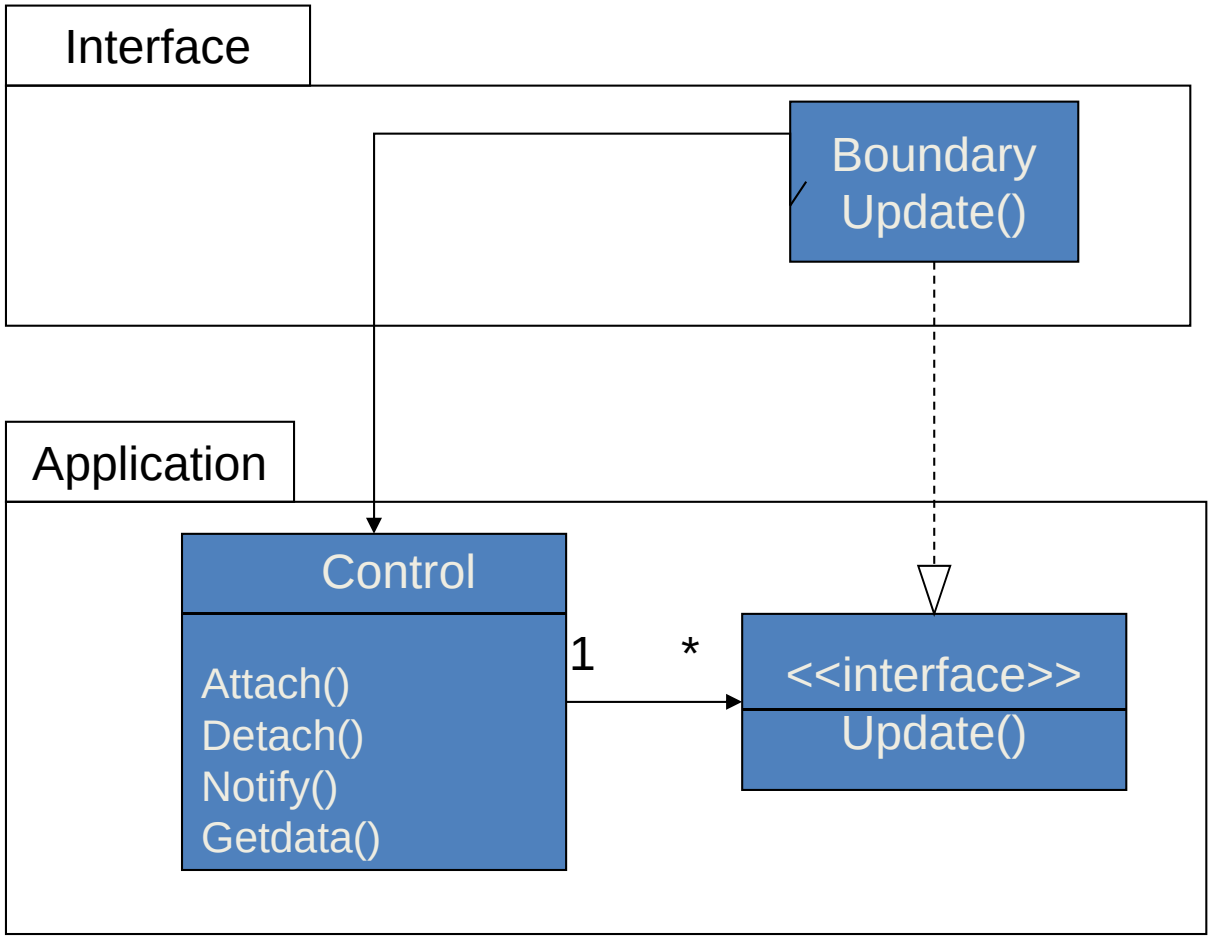


?

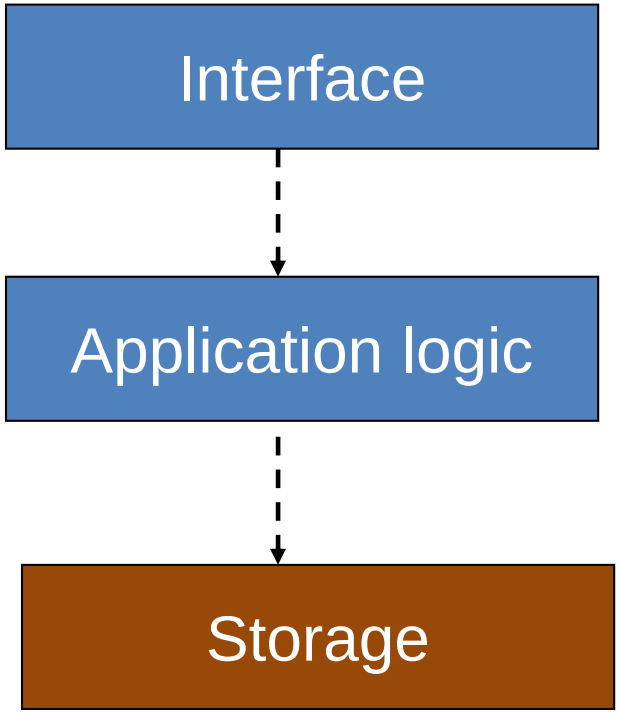
Observer Design Pattern - MVC



Observer Design Pattern - MVC



The Layered Model – Three Tier



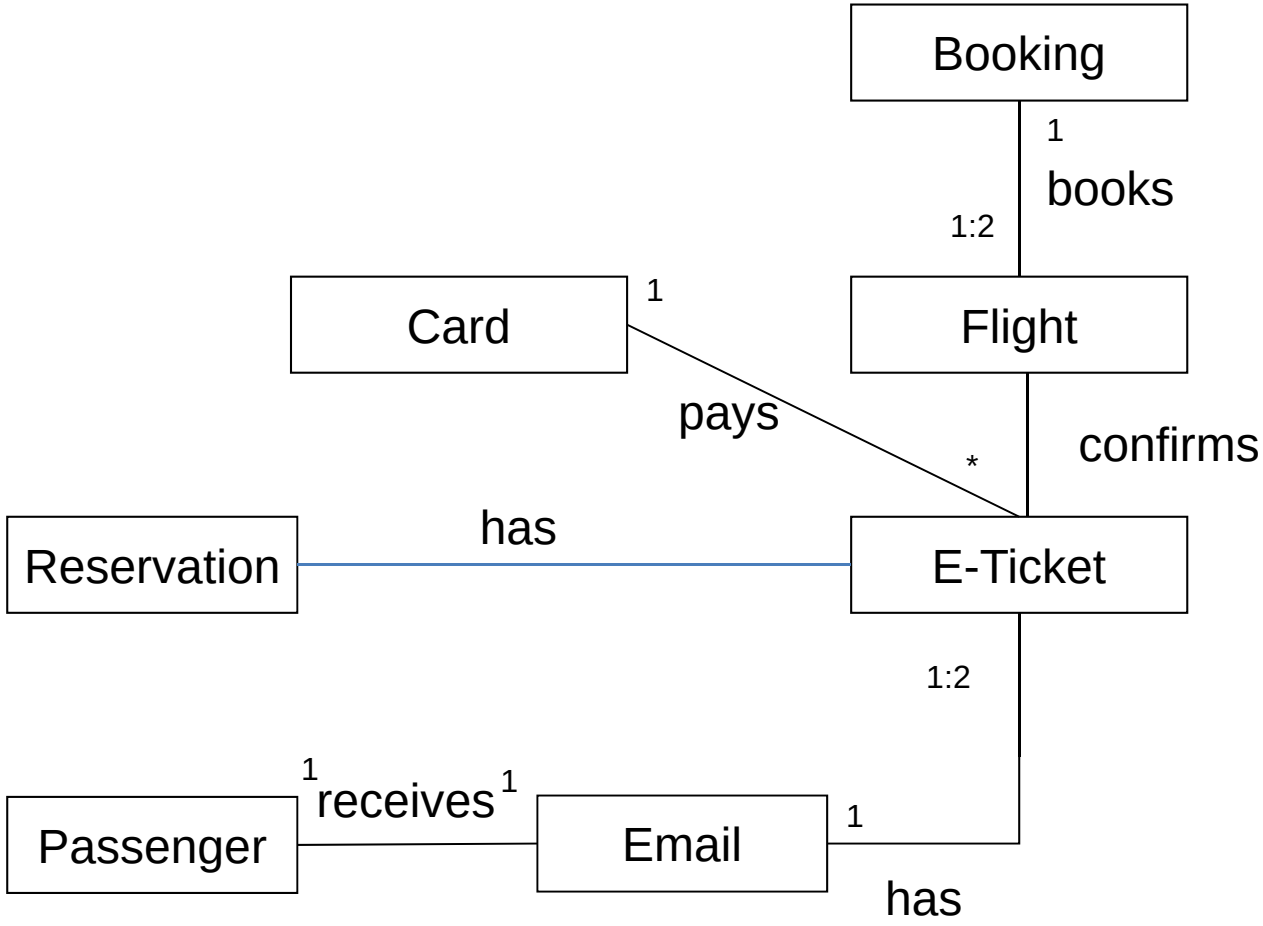
Identifying Persistent Data

- Persistent （永久）data
  - The data that is stored permanently （永久） in some way so the data will not be lost when the system is closed
  - The data can be required later.

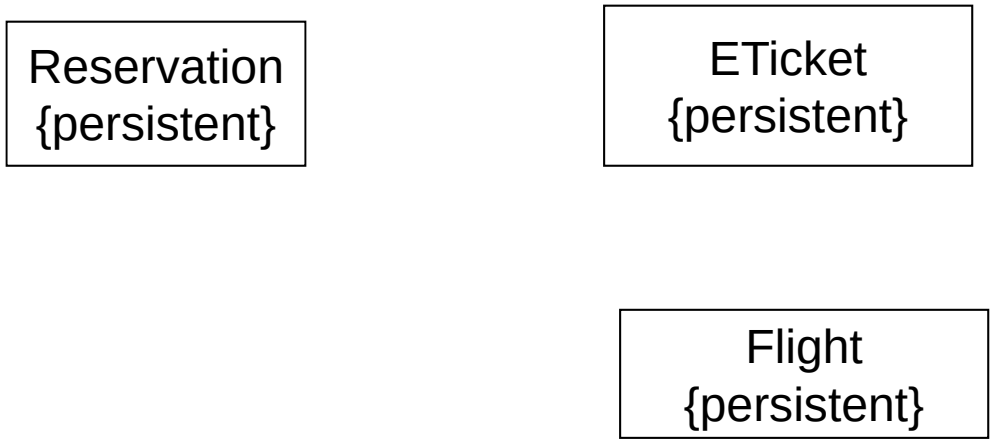
Identifying Persistent Data

- Storage strategies
  - Write data in a file
  - A database management system
    - A rational model （关系模型）can be used
- How to design a database in OO?
  - Design a database schema （模式）storing and retrieving （检索） system’s objects
  - the code to access （存取） the database

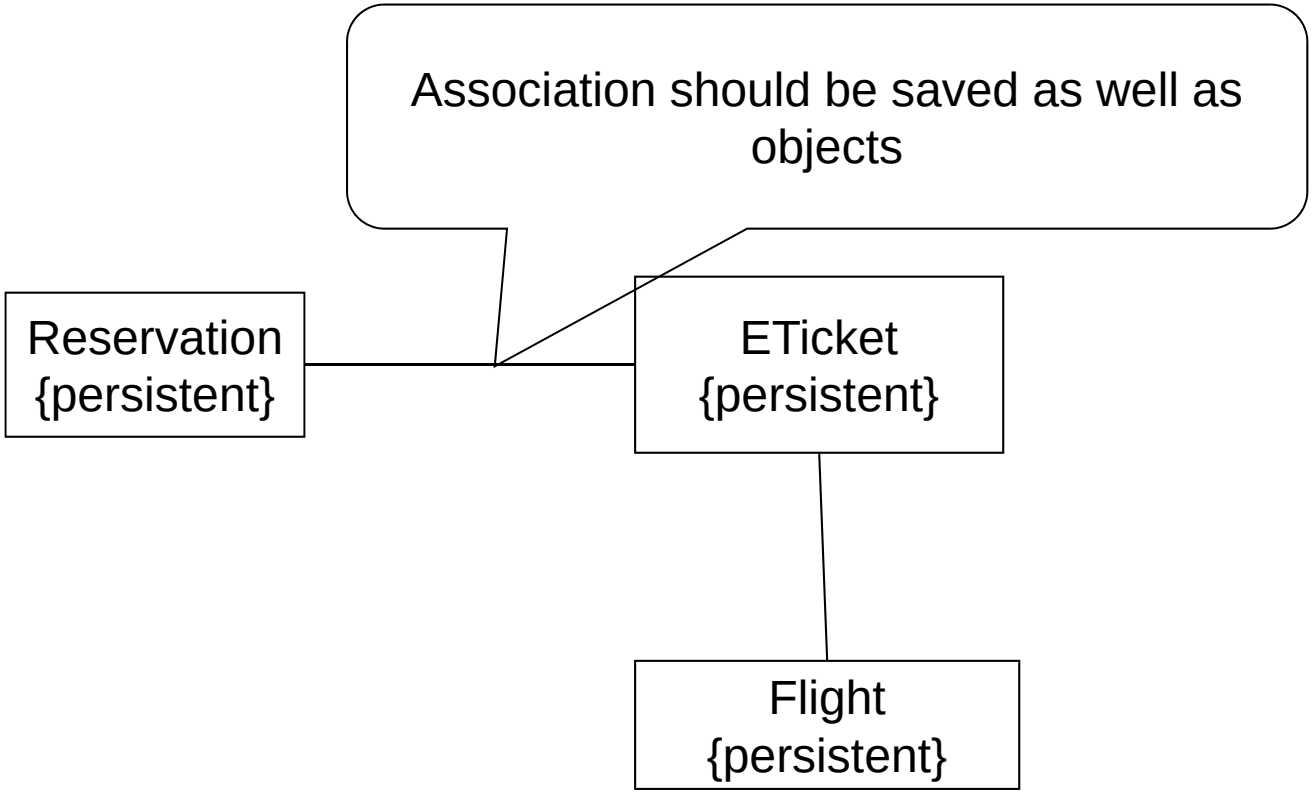
Identifying Persistent Data



Identifying Persistent Data



What should Be Stored?



How to Store?

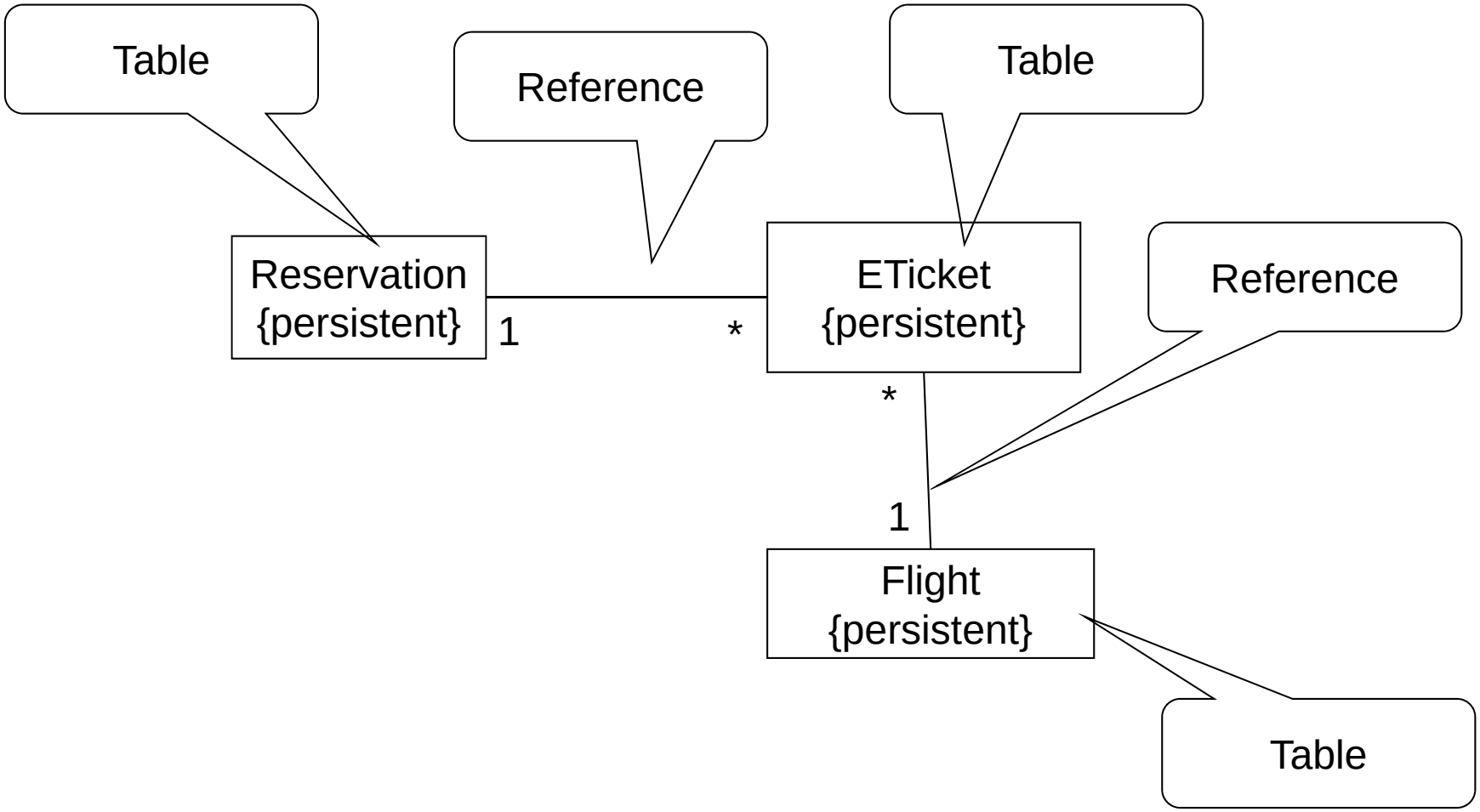


Table
Reference
Table
Reservation
{persistent}
1
*
ETicket
{persistent}
Reference
*
1
Flight
{persistent}
Table

How to Store Data?

eTicket			
Ticket no.	Issue Date	Issue Time	Fare

Tables

Flight				
Flight no	Departure	Destination	Depart-Time	Arrival time

How to Store Association (one-to one, one-to-many)?

eTicket				
Ticket no.	Issue Date	Issue Time	Fare	Flight no.

Reference key

Flight				
Flight no	Departure	Destination	Depart-Time	Arrival time

Keys

How to Store Association (one-to one, one-to-many)?

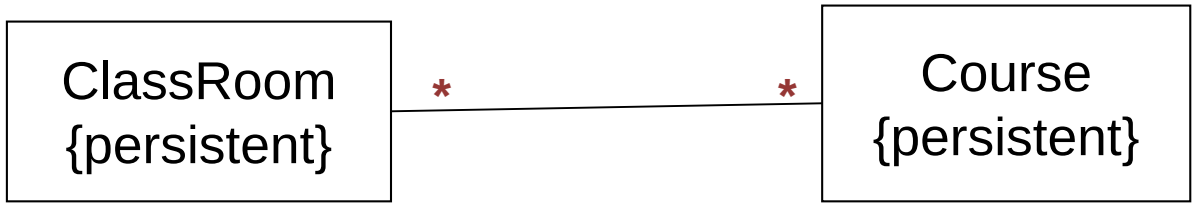
eTicket			
Ticket no.	Issue Date	Issue Time	Fare

Keys

Flight					
Ticket no.	Flight no	Departure	Destination	Depart-Time	Arrival time

Reference key

How to Store Association (many-to-many)?



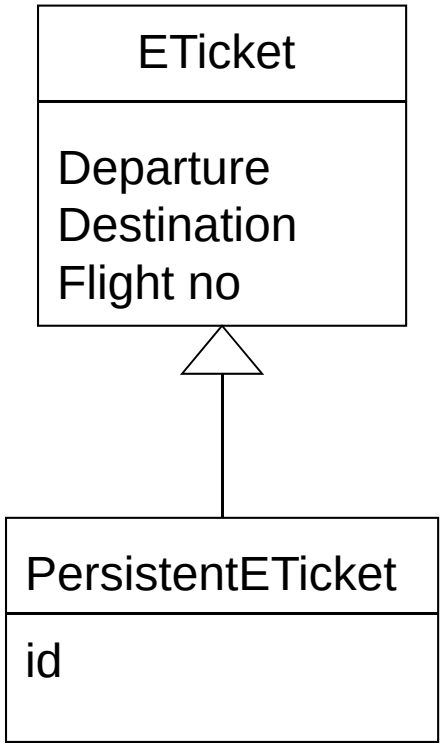
ClassRoom		
Classroom no.	Seats	Type
C402	50	Computer
D407	100	Normal
……		

Match Table	
Code	Classroom
COMP3053	D407
COMP 3063	C402
COMP3053	D405
COMP 2023	C402

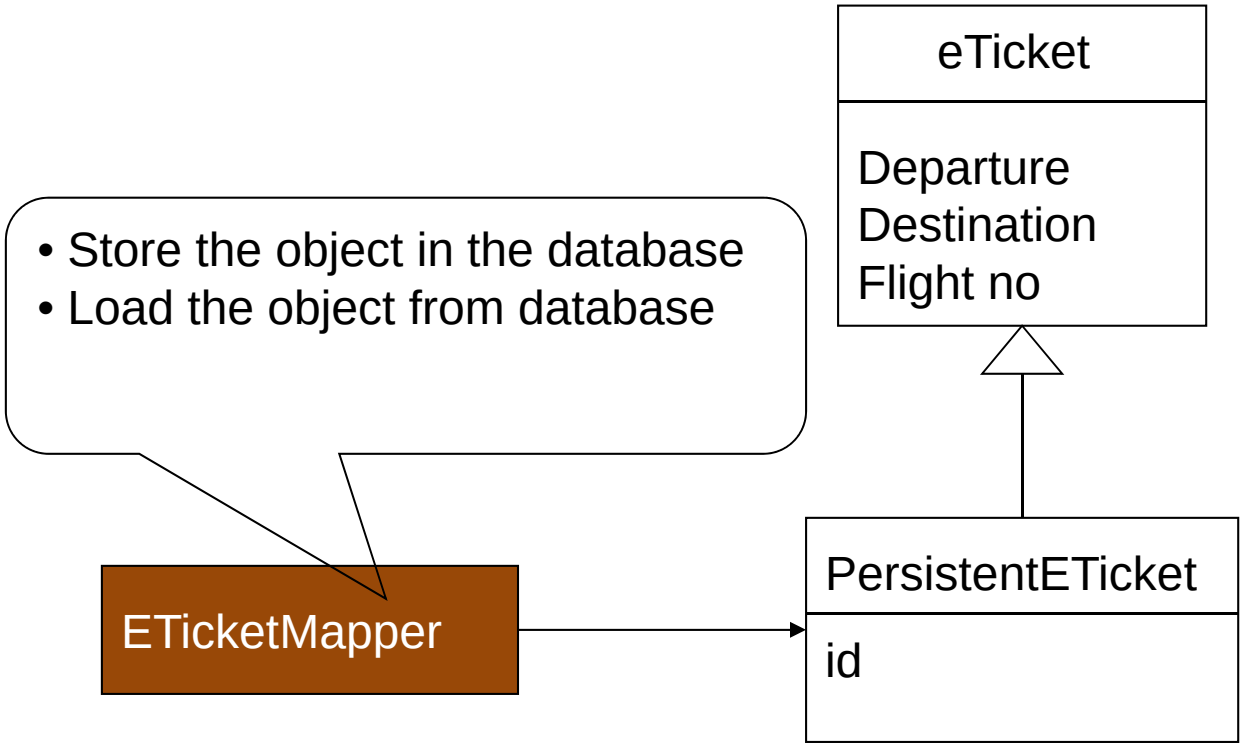
Course	
Code	Title
COMP3053	SE
COMP 3063	SDW
……	

Separate table

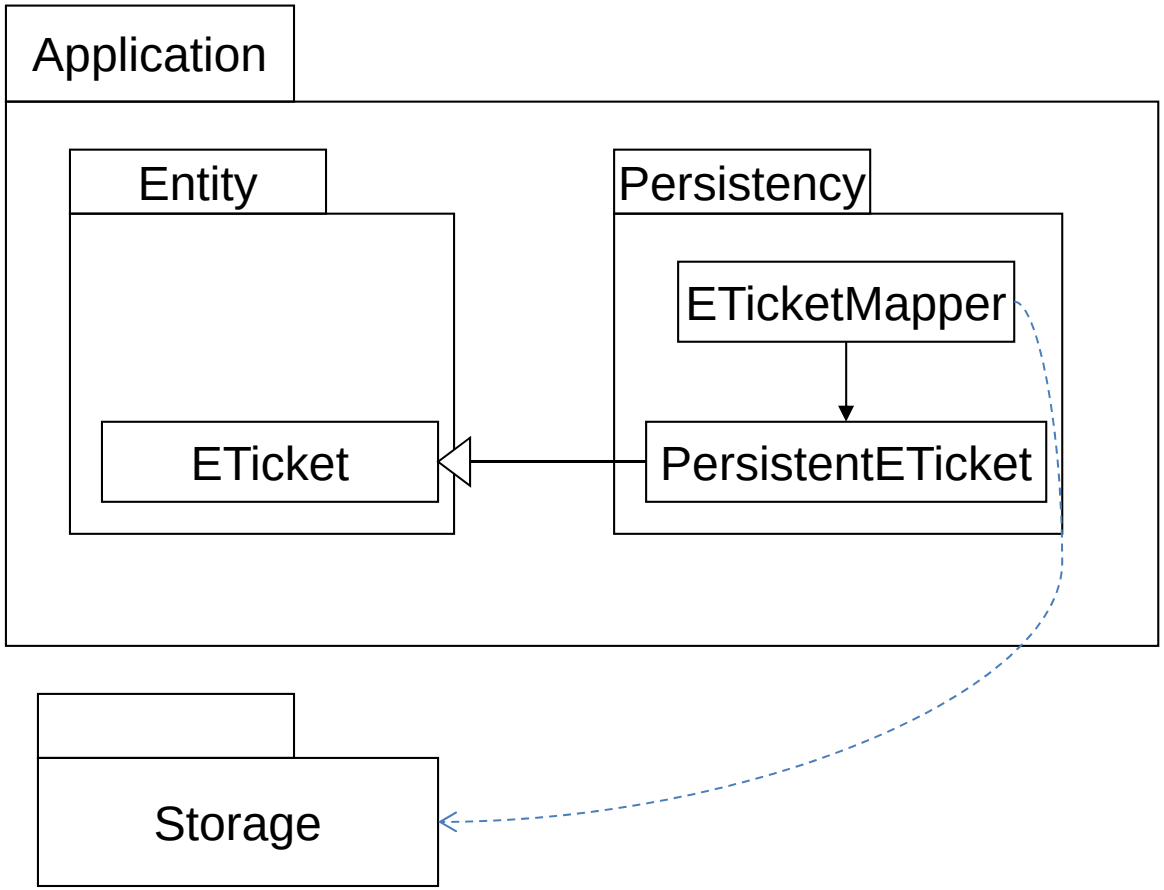
How to Store?


ETicket
Departure
Destination
Flight no
PersistentETicket
id

How to Load?


eTicket
Departure
Destination
Flight no
• Store the object in the database
• Load the object from database
ETicketMapper
PersistentETicket
id

Architecture for Persistency



Summary

- The repository models, client server models and layer models can be used together in one system
- MVC model include controllers (control objects), view objects (boundary objects) and model objects (entity objects).
- Design pattern Observer can be used so that the control layer will not directly access the boundary layer.
- Database design includes the decision of schema and the access of data.

Summary

- Steps in designing a database
  - Identify the persistent objects
  - Decide the storage strategies
  - Map objects to tables if using relational data model
  - Decide how to load the data.