COMP2511

WEEK 10

Did you get much sleep?

ADMIN STUFF

- If you did assignment-iii, please make sure you submit it following the instructions in the spec.
  - Include video that is public. We will not be chasing this up since marking will happen during week 11/12

AGENDA

- myExperience

- Visitor Pattern

- Adaptor Pattern

- Special Surprise

MYEXPERIENCE

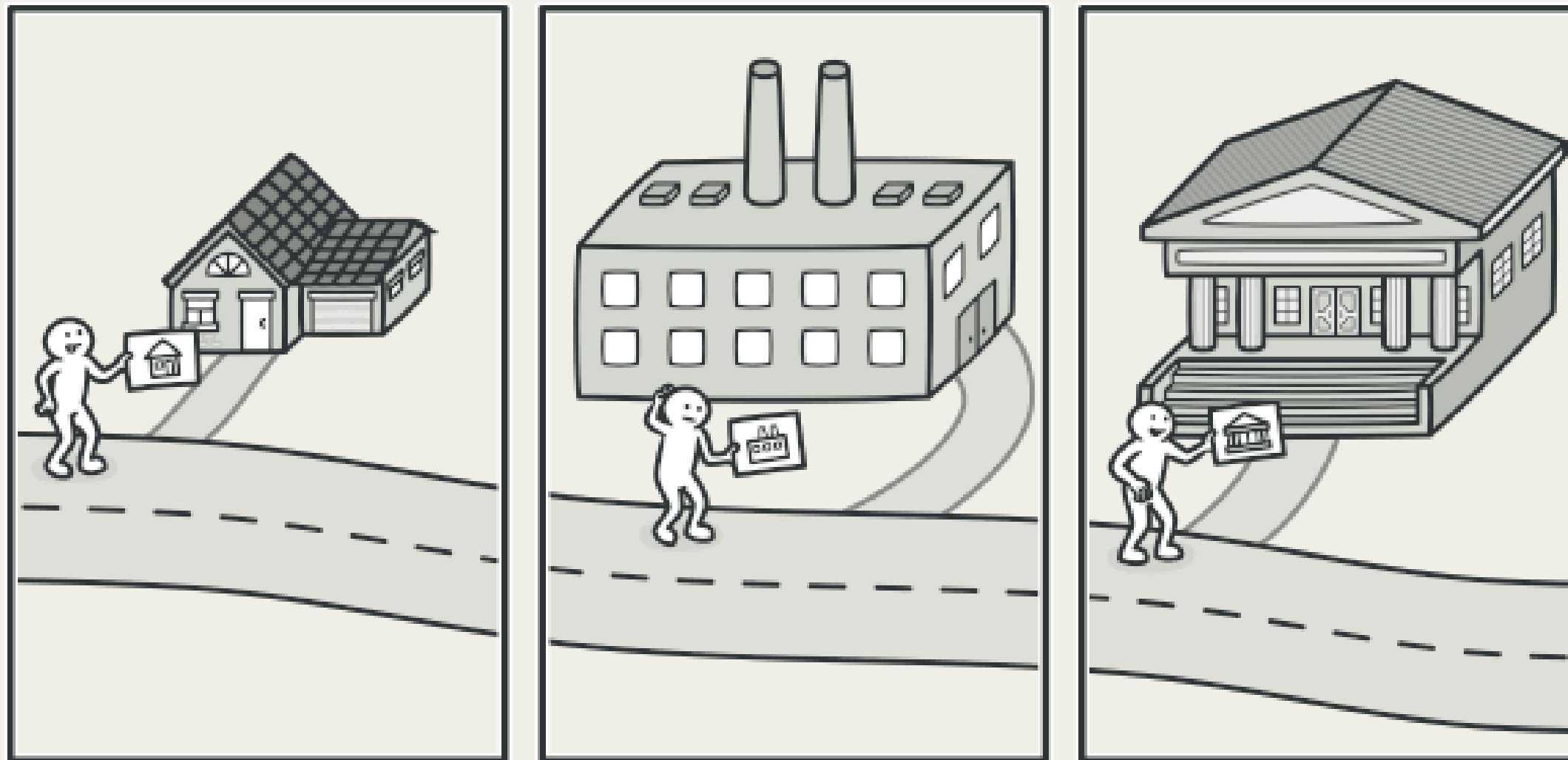Please complete your myExperience survey of your experience of me!

It will greatly help me improve for whatever I am teaching in the future but also help the course improve for future iterations.

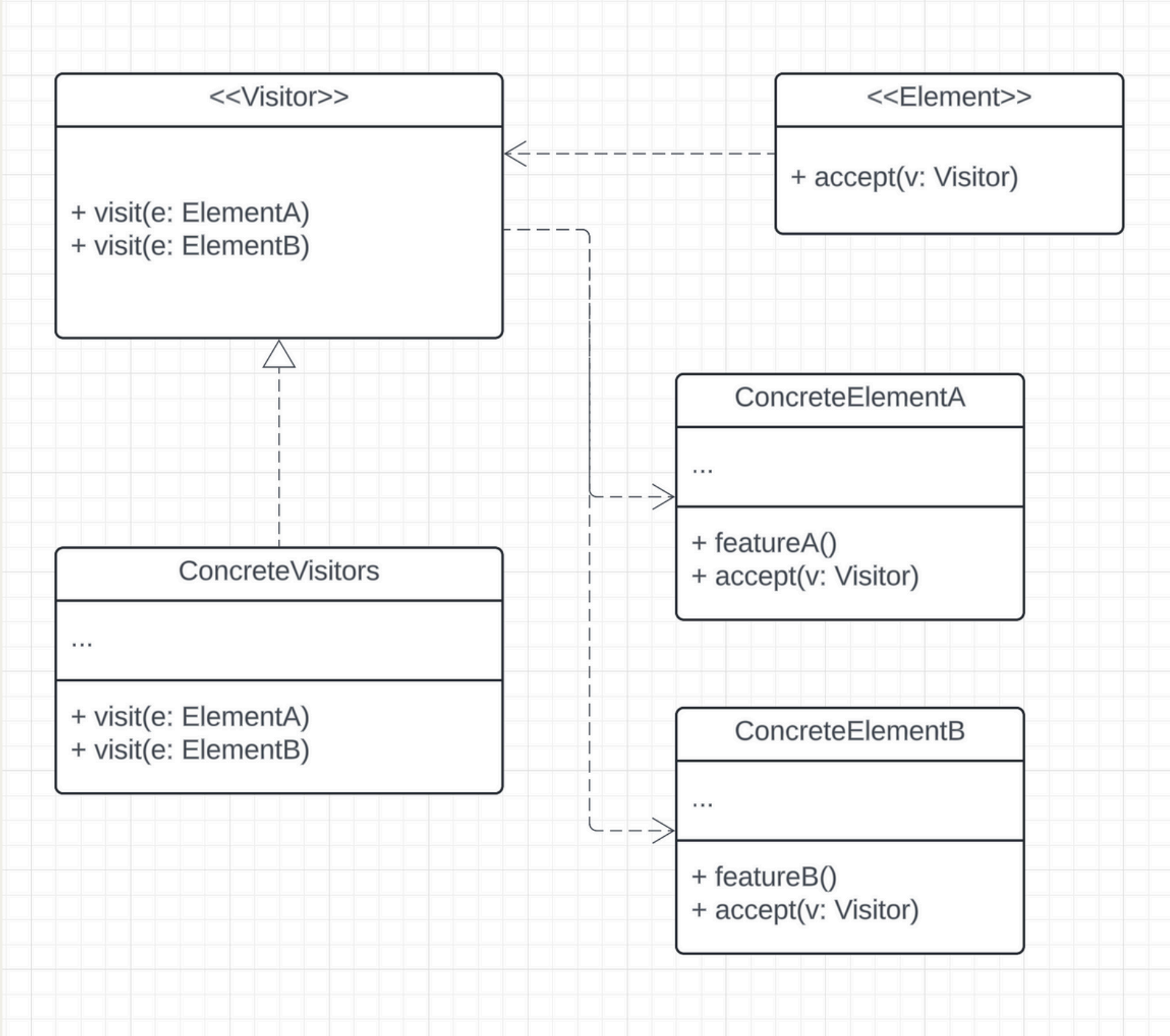All constructive feedback is welcome. I would like to become a better tutor for future students!
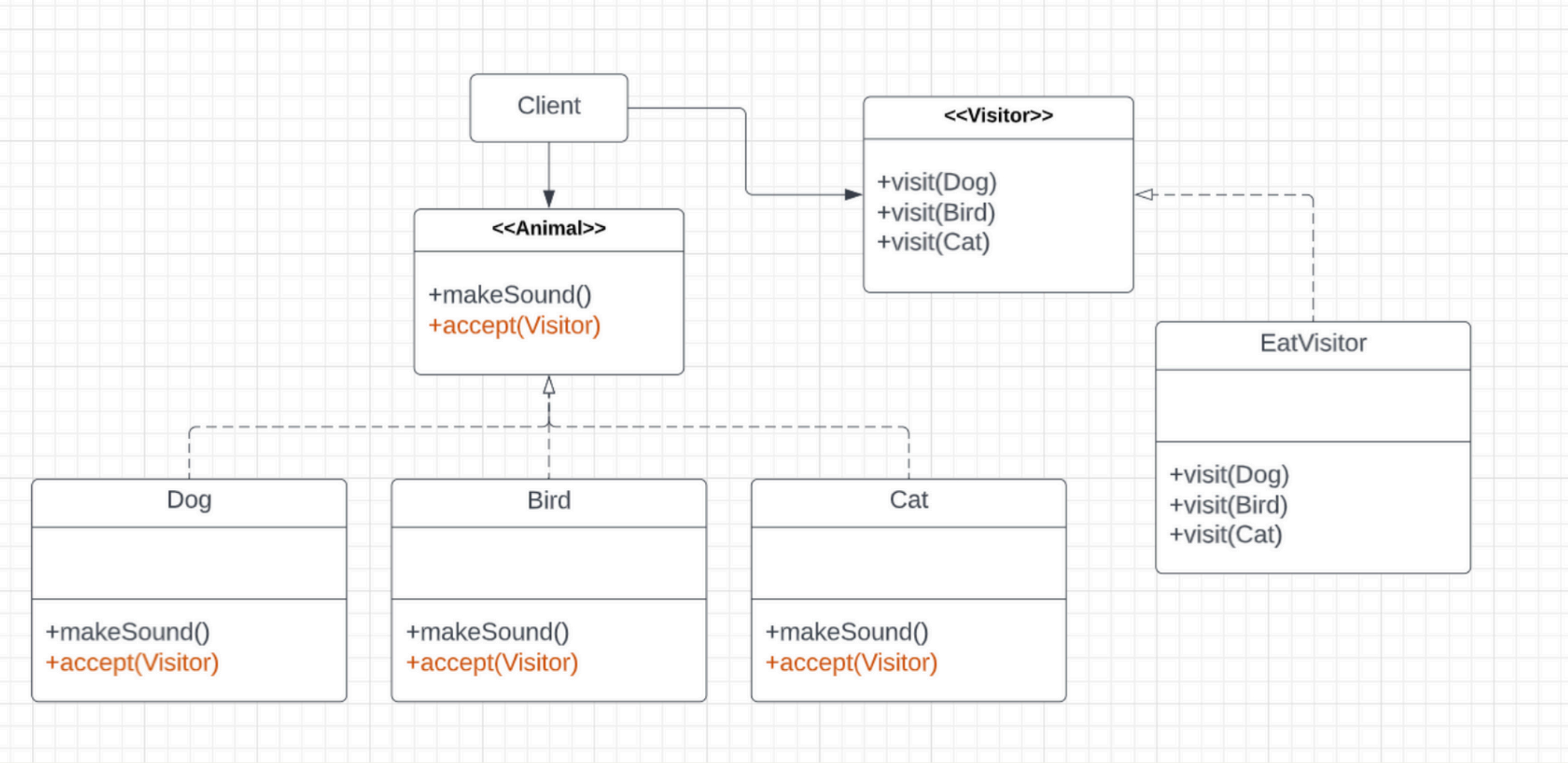
Visitor Pattern

VISITOR PATTERN

Visitor is a behavioural design pattern that lets you separate algorithms from the objects on which they operate.

VISITOR PATTERN

<<Visitor>>

+ visit(e: ElementA)
+ visit(e: ElementB)

<<Element>>

+ accept(v: Visitor)

ConcreteElementA

...

+ featureA()
+ accept(v: Visitor)

ConcreteVisitors

...

+ visit(e: ElementA)
+ visit(e: ElementB)

ConcreteElementB

...

+ featureB()
+ accept(v: Visitor)

VISITOR PATTERN

VISITOR PATTERN

In this scenario we have Computers, Keyboards and Mouses which all are of type ComputerComponent. We want to be able to 'visit' different types of Computer components by logging the following messages:

- Looking at computer Corelli with memory 500 GB.
- Looking at keyboard Mechanical keyboard which has 36 keys.
- Looking at mouse Bluetooth mouse.

In particular though, anyone which is visiting a Computer must be validated prior to being able to visit.

Extend/modify the starter code to use the Visitor Pattern to allow different computer components to be visited.

Adapter Pattern

ADAPTER PATTERN

Adapter is a structural design pattern that allows objects with incompatible interfaces to collaborate.
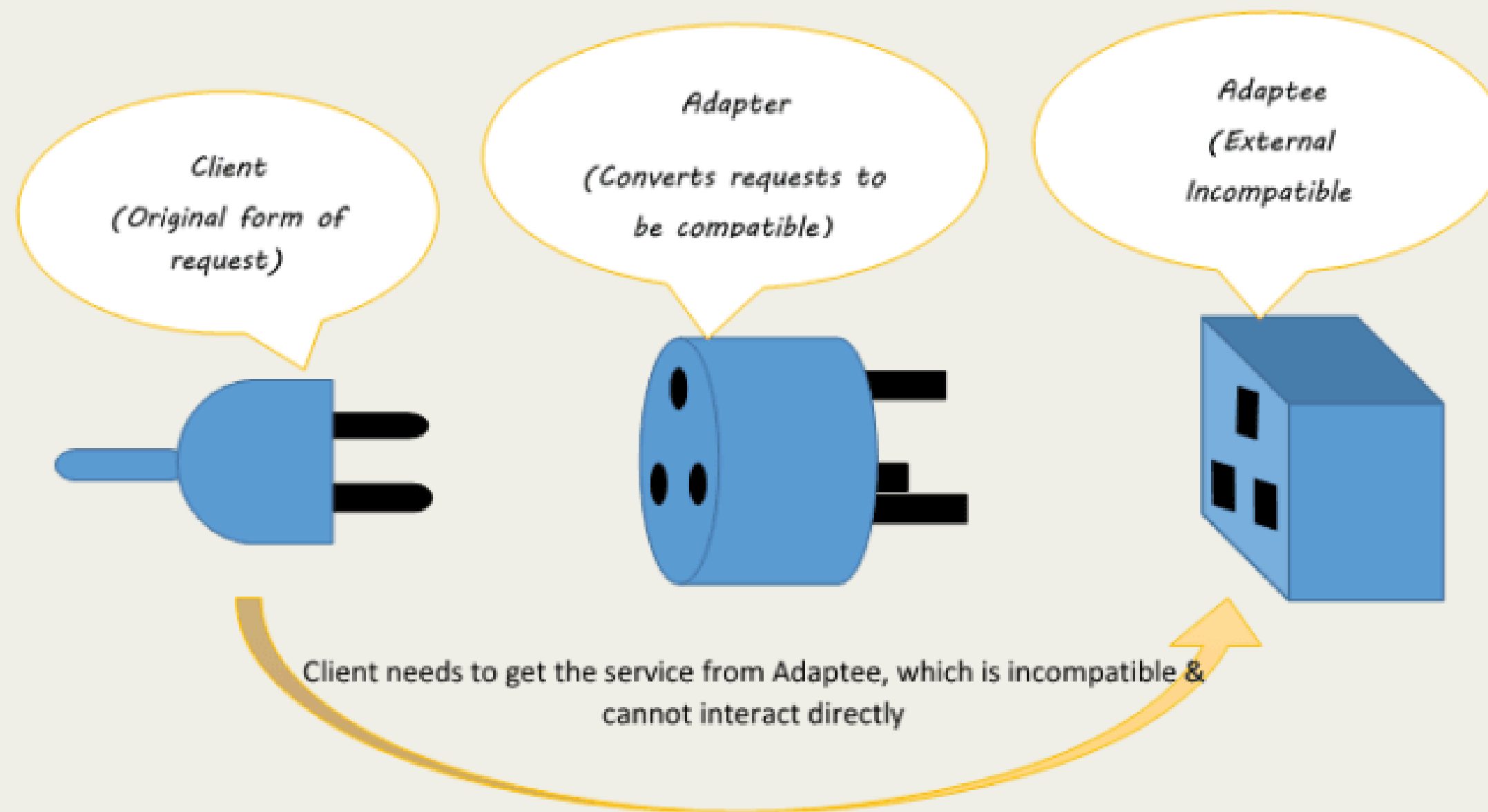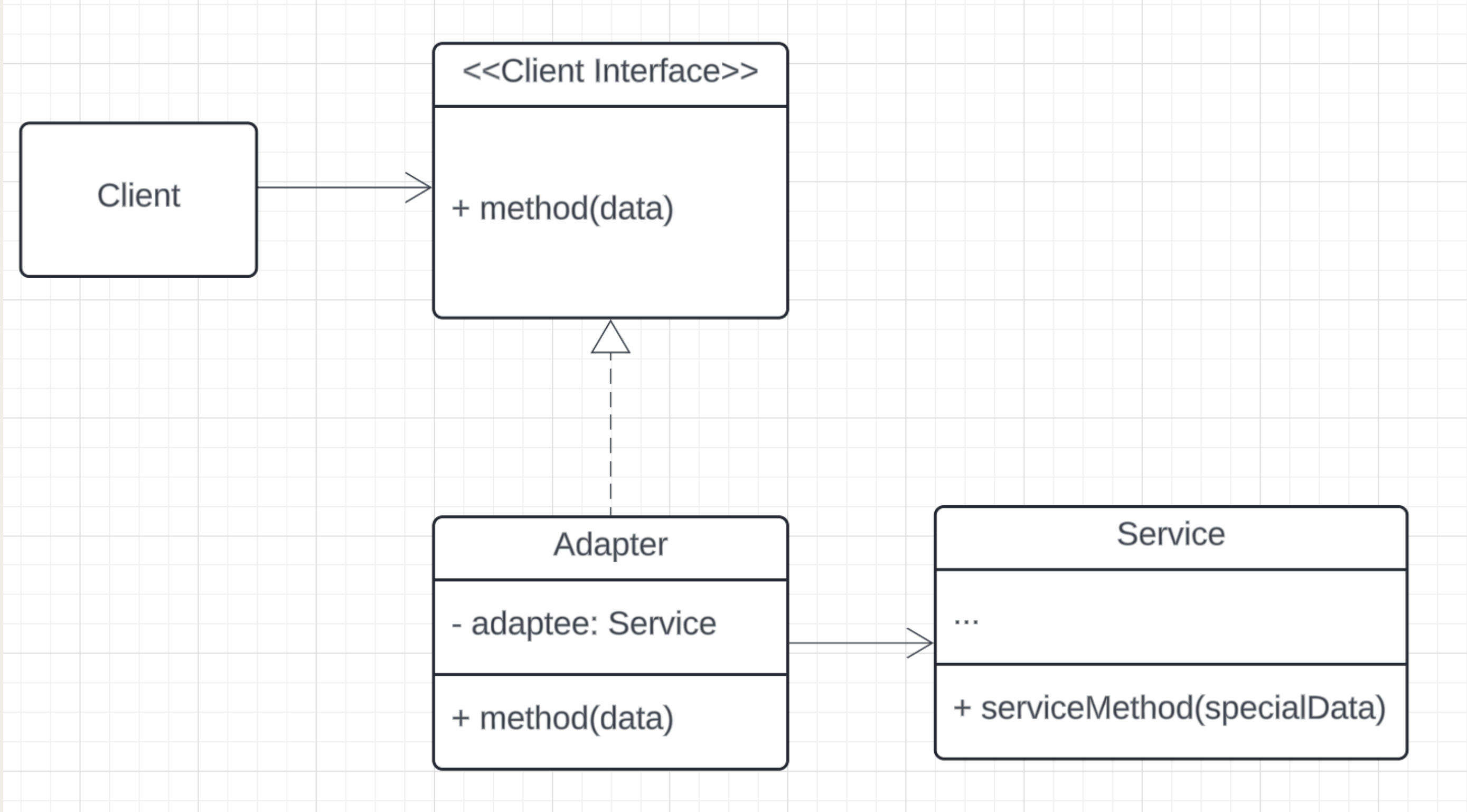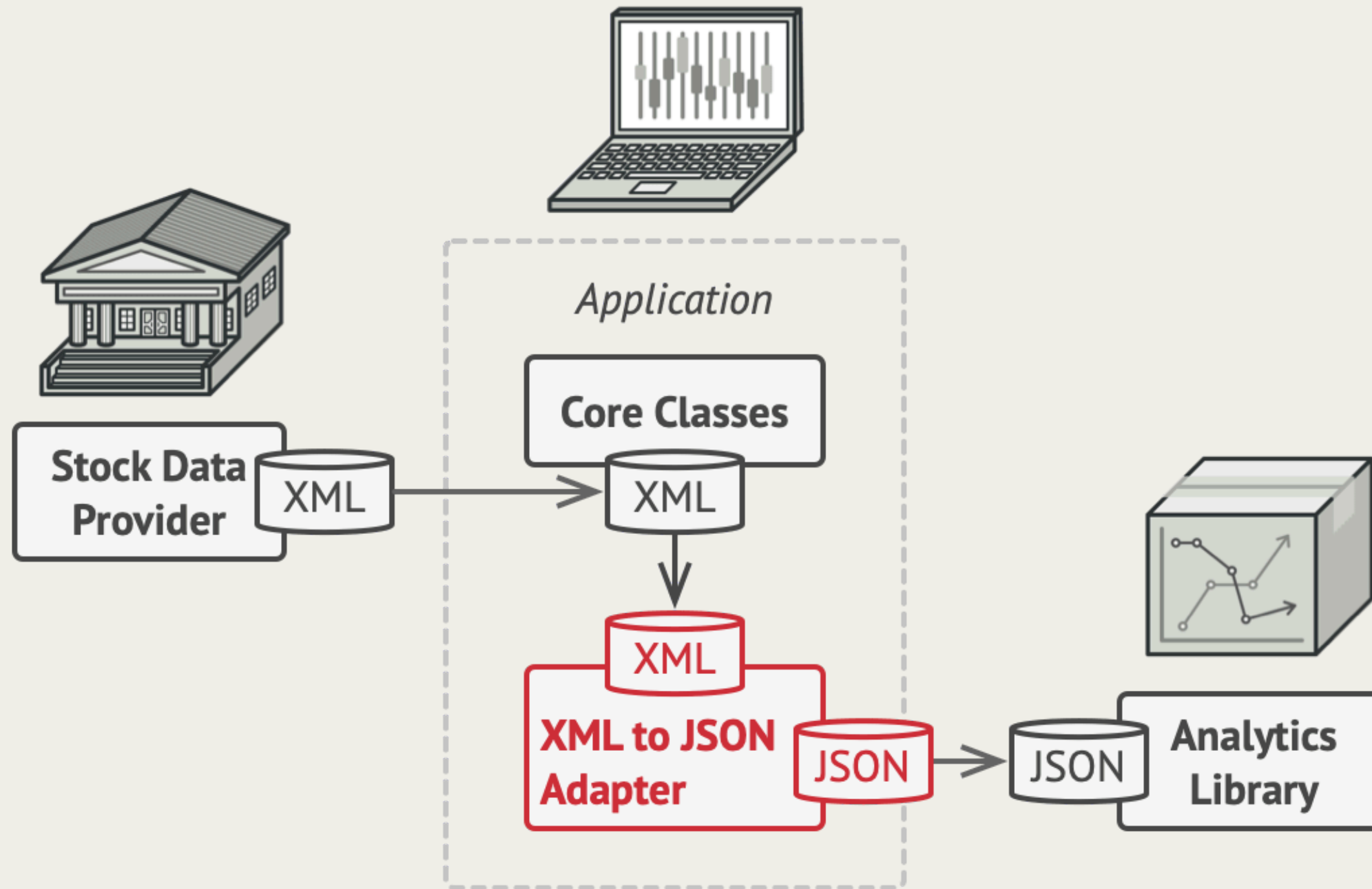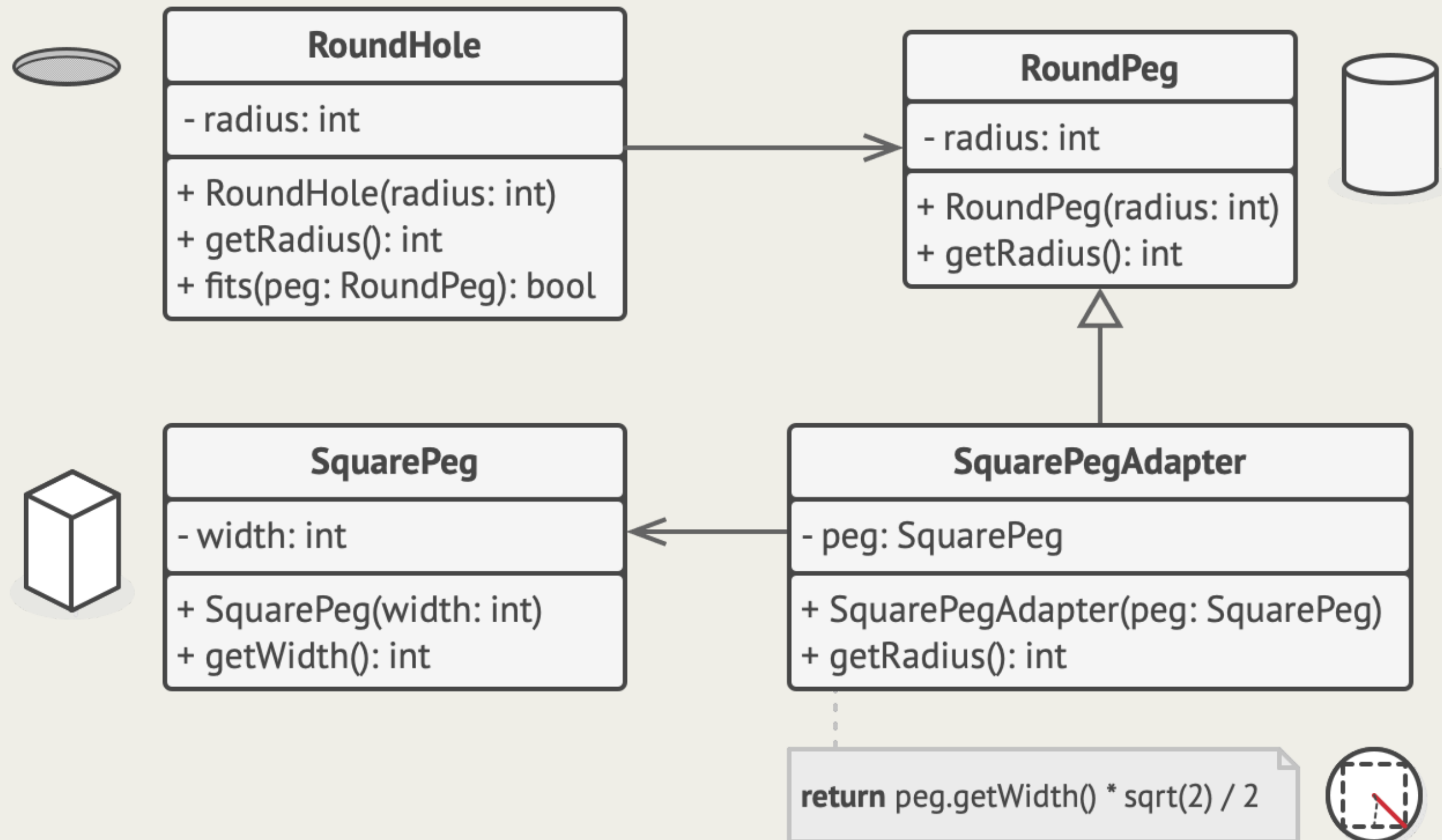

Figure 1-Adapter Pattern Concept

ADAPTER PATTERN

Client → <<Client Interface>>
+ method(data)

Adapter
- adaptee: Service
+ method(data)

Service
...
+ serviceMethod(specialData)

ADAPTER PATTERN

ADAPTER PATTERN

RoundHole

- radius: int

+ RoundHole(radius: int)
+ getRadius(): int
+ fits(peg: RoundPeg): bool

RoundPeg

- radius: int

+ RoundPeg(radius: int)
+ getRadius(): int

SquarePeg

- width: int

+ SquarePeg(width: int)
+ getWidth(): int

SquarePegAdapter

- peg: SquarePeg

+ SquarePegAdapter(peg: SquarePeg)
+ getRadius(): int

return peg.getWidth() * sqrt(2) / 2

KAHOOOTTTTTTTTT!!!!!!

LABBBBBBB

PRACTICE EXAM