CS 525 - ASD

Advanced Software Development

MS.CS Program

Department of Computer Science

Rene de Jong, MsC.

Maharishi University
OF MANAGEMENT

CS 525 - ASD

Advanced Software Development

Maharishi University
OF MANAGEMENT

Lesson 8 Adapter pattern

L1: ASD Introduction
L2: Strategy, Template method
L3: Observer pattern
L4: Composite pattern, iterator pattern
L5: Command pattern
L6: State pattern
L7: Chain Of Responsibility pattern

Midterm

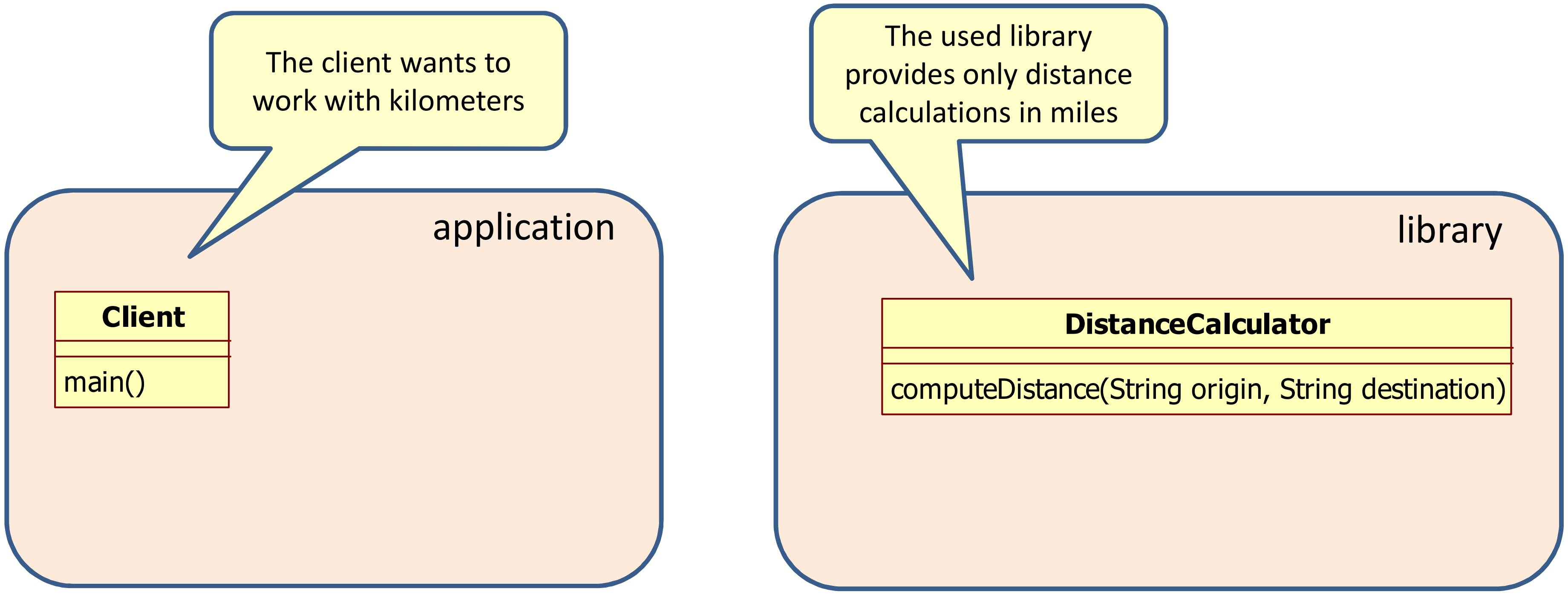
L8: Proxy, Adapter, Mediator
L9: Factory, Builder, Decorator, Singleton
L10: Framework design
L11: Framework implementation
L12: Framework example: Spring framework
L13: Framework example: Spring framework

Final

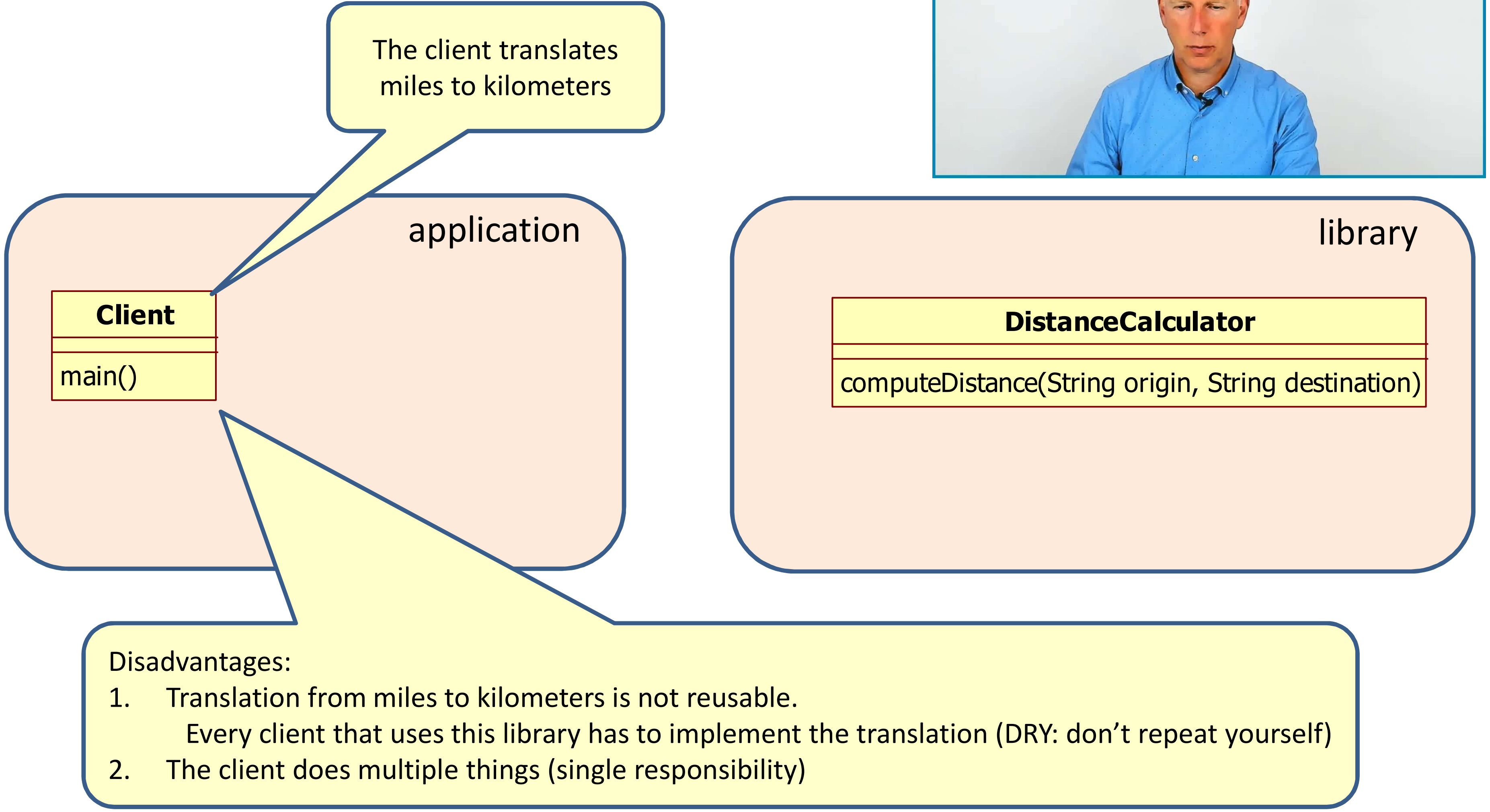
Adapter pattern

- Translates the existing interface of a class into an interface that the client requires.
  - (Reusable) wrapper

Design problem

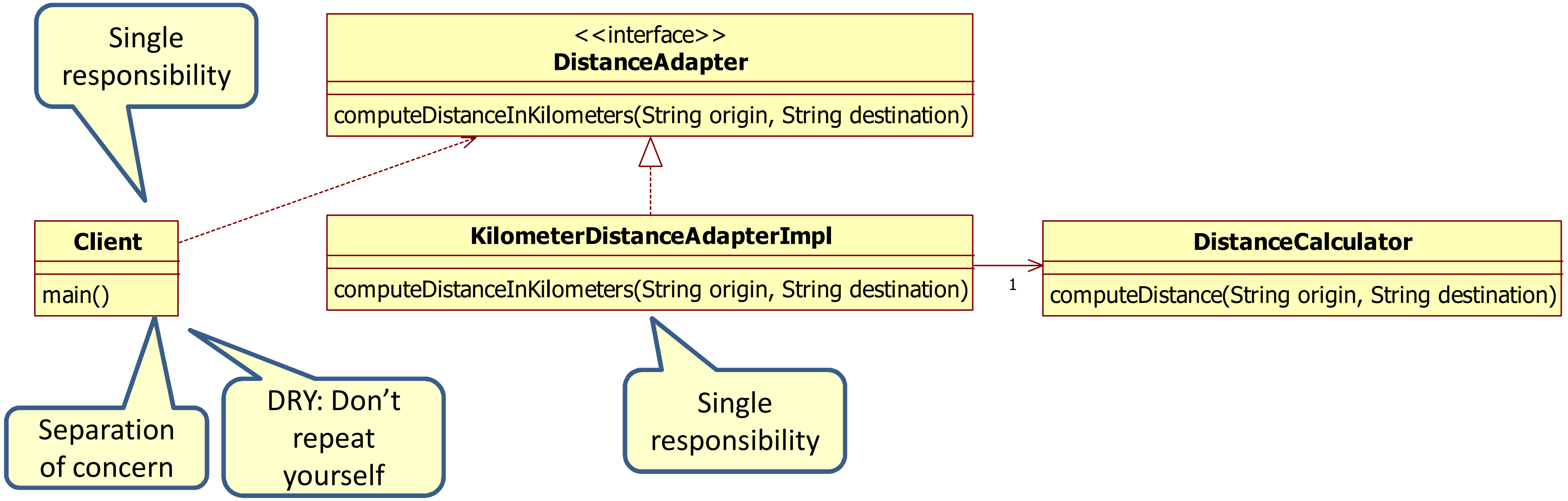

The client wants to work with kilometers
application
Client
main()
The used library provides only distance calculations in miles
library
DistanceCalculator
computeDistance(String origin, String destination)

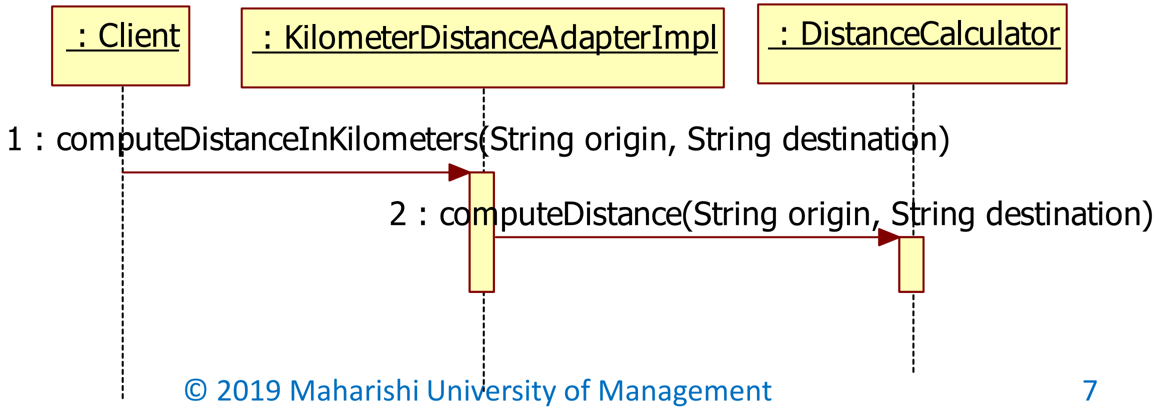
Solution


The client translates miles to kilometers
application
Client
main()
library
DistanceCalculator
computeDistance(String origin, String destination)

Disadvantages:
1. Translation from miles to kilometers is not reusable.
   Every client that uses this library has to implement the translation (DRY: don't repeat yourself)
2. The client does multiple things (single responsibility)

Better solution: Adapter


Single responsibility
<<interface>>
DistanceAdapter
computeDistanceInKilometers(String origin, String destination)
Client
main()
KilometerDistanceAdapterImpl
computeDistanceInKilometers(String origin, String destination)
DistanceCalculator
computeDistance(String origin, String destination)
1
Separation of concern
DRY: Don't repeat yourself
Single responsibility


: Client
: KilometerDistanceAdapterImpl
: DistanceCalculator
1 : computeDistanceInKilometers(String origin, String destination)
2 : computeDistance(String origin, String destination)

Adapter + Adaptee

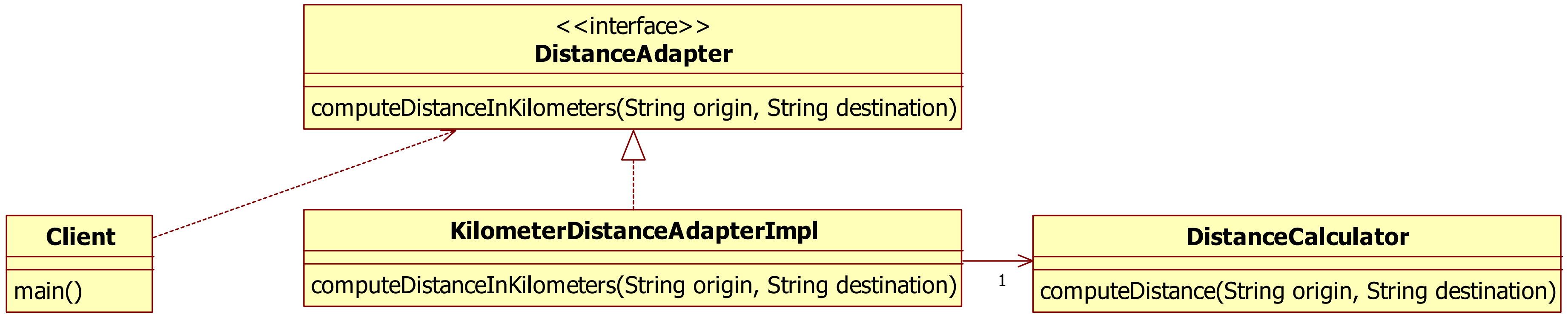
```java
public class DistanceCalculator {
  public double computeDistance(String origin, String destination) {
    return (new Random()).nextInt(100);
  }
}
```

```java
public interface DistanceAdapter {
  double computeDistanceInKilometers(String origin, String destination);
  void setDistanceCalculator(DistanceCalculator distanceCalculator);
}
```

```java
public class KilometerDistanceAdapterImpl implements DistanceAdapter {
  private DistanceCalculator distanceCalculator;

  public double computeDistanceInKilometers(String origin, String destination) {
    double distanceInMiles = distanceCalculator.computeDistance(origin, destination);
    return distanceInMiles * 1.609344;
  }

  public void setDistanceCalculator(DistanceCalculator distanceCalculator) {
    this.distanceCalculator = distanceCalculator;
  }
}
```

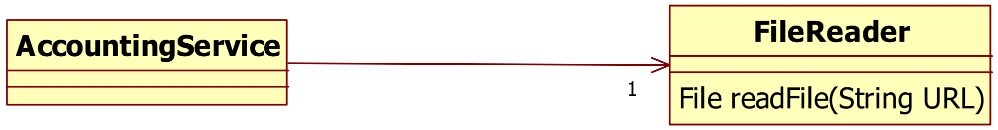
Client

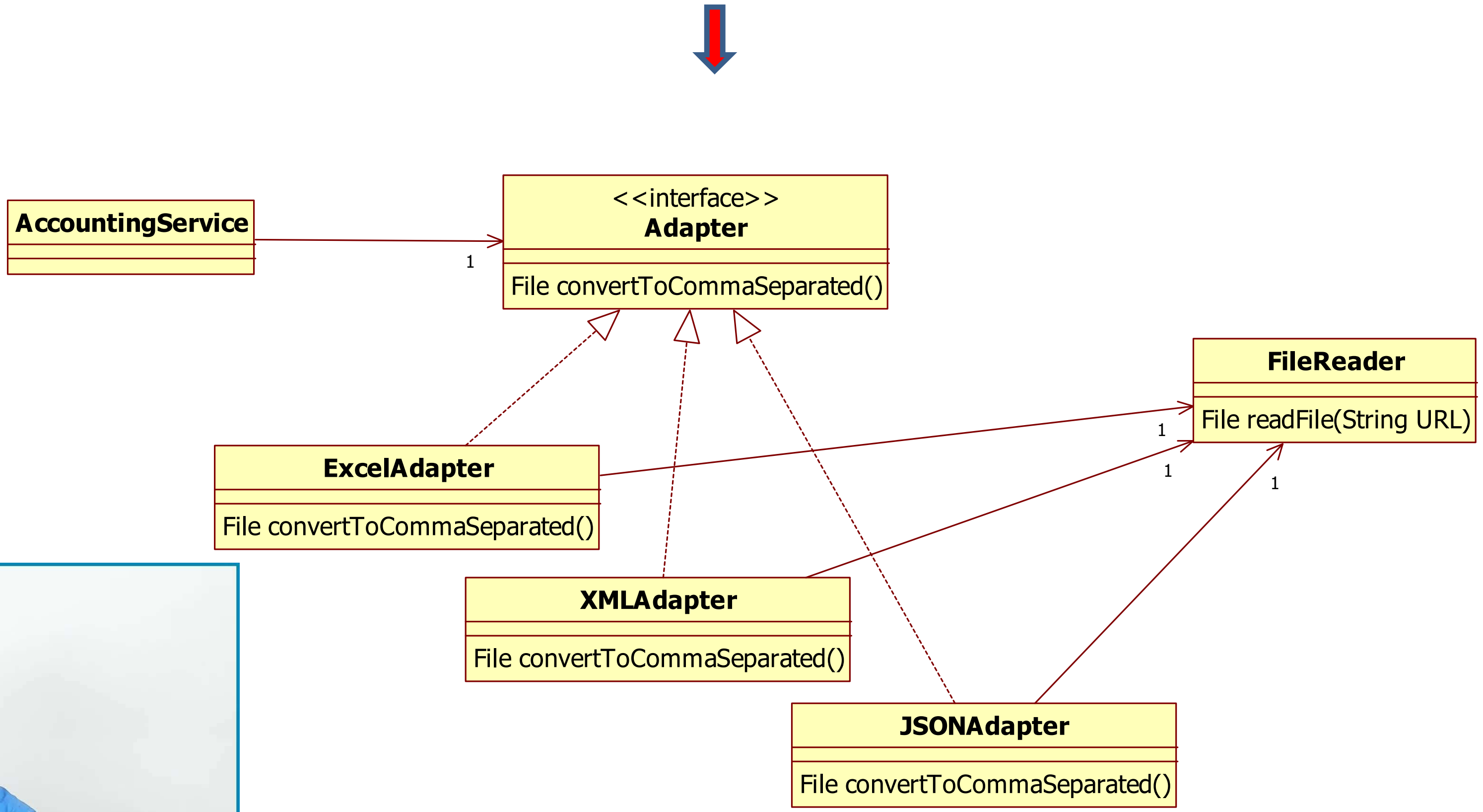
```
public class Client {

  public static void main(String[] args) {
    DistanceCalculator distanceCalculator = new DistanceCalculator();
    double distanceInMiles = distanceCalculator.computeDistance("city1", "city2");
    System.out.println("The distance between city1 and city2 ="+distanceInMiles+" miles");

    DistanceAdapter distanceAdapter = new KilometerDistanceAdapterImpl();
    distanceAdapter.setDistanceCalculator(distanceCalculator);

    double distanceInKilometers = distanceAdapter.computeDistanceInKilometers("city3", "city4");
    System.out.println("The distance between city3 and city4 ="+distanceInKilometers+" kilometers");
  }
}
```



<<interface>>
DistanceAdapter
computeDistanceInKilometers(String origin, String destination)
Client
main()
KilometerDistanceAdapterImpl
computeDistanceInKilometers(String origin, String destination)
1
DistanceCalculator
computeDistance(String origin, String destination)

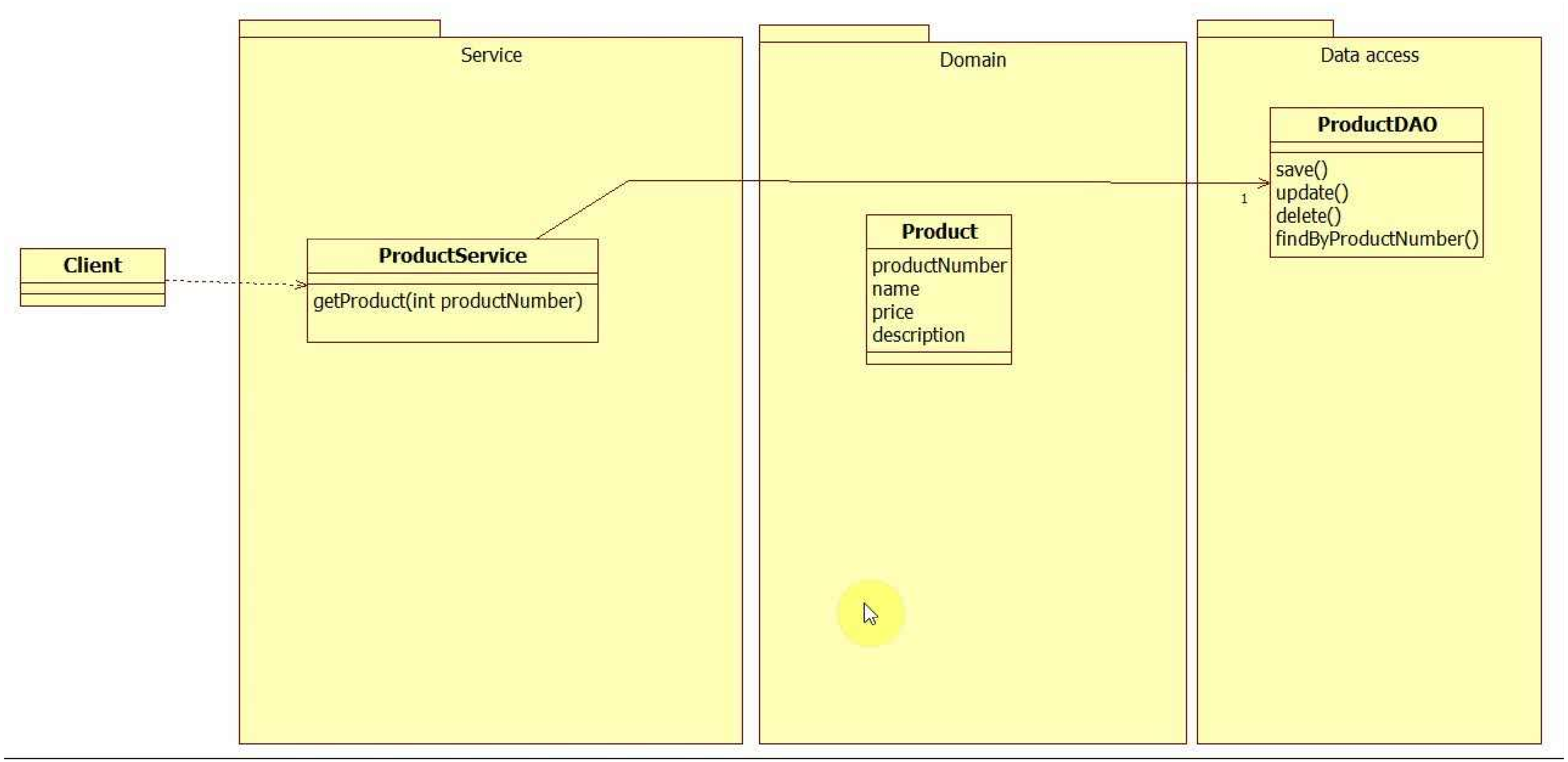
Where are adapters used



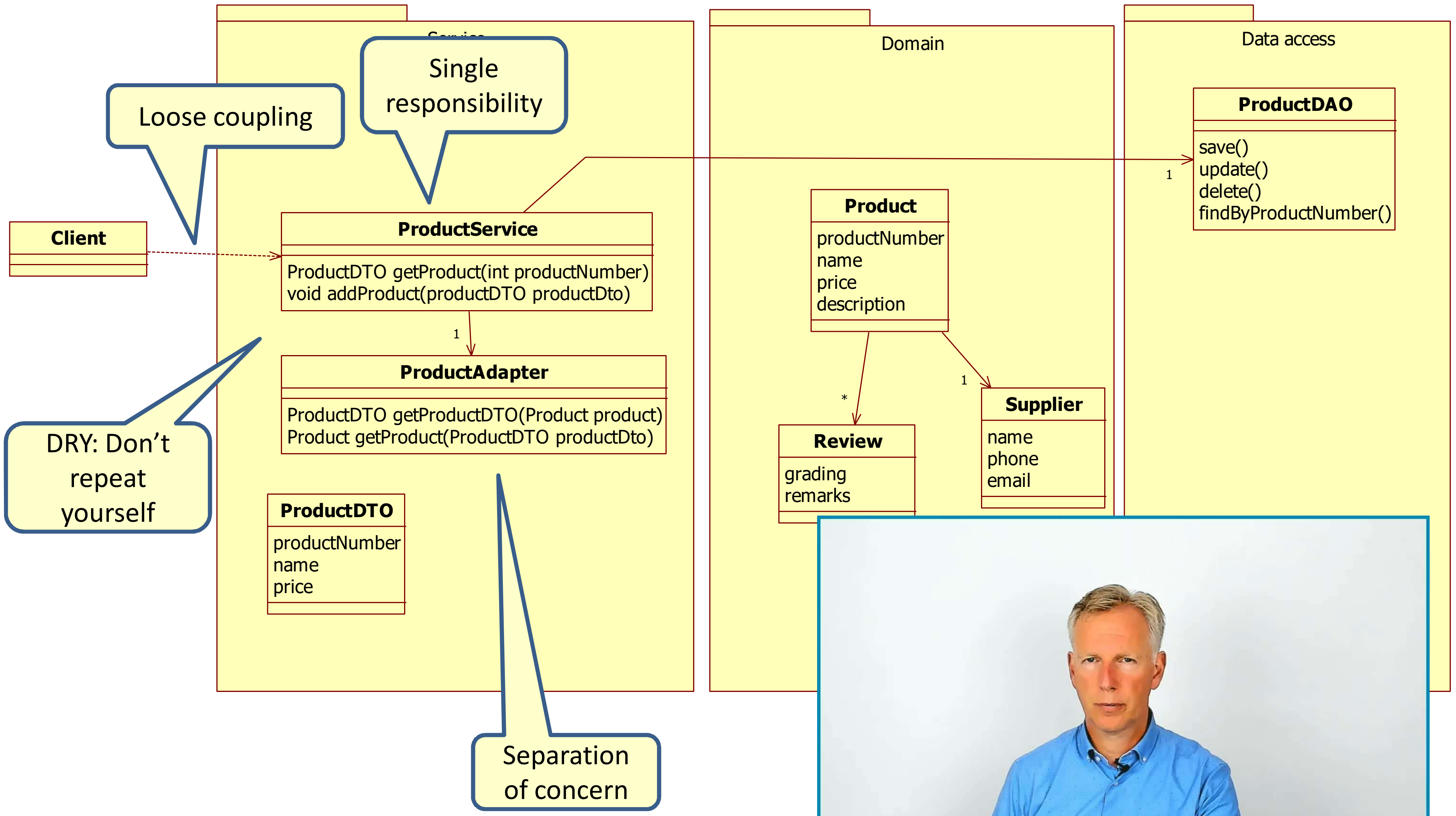


AccountingService
<<interface>>
Adapter
File convertToCommaSeparated()
1
FileReader
File readFile(String URL)
ExcelAdapter
File convertToCommaSeparated()
XMLAdapter
File convertToCommaSeparated()
JSONAdapter
File convertToCommaSeparated()
1
1
1

Where are adapters used

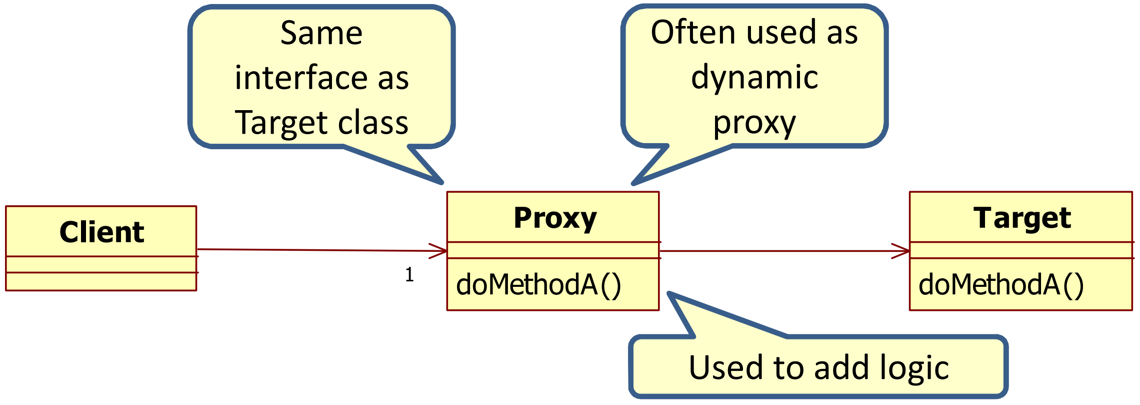


Where are adapters used

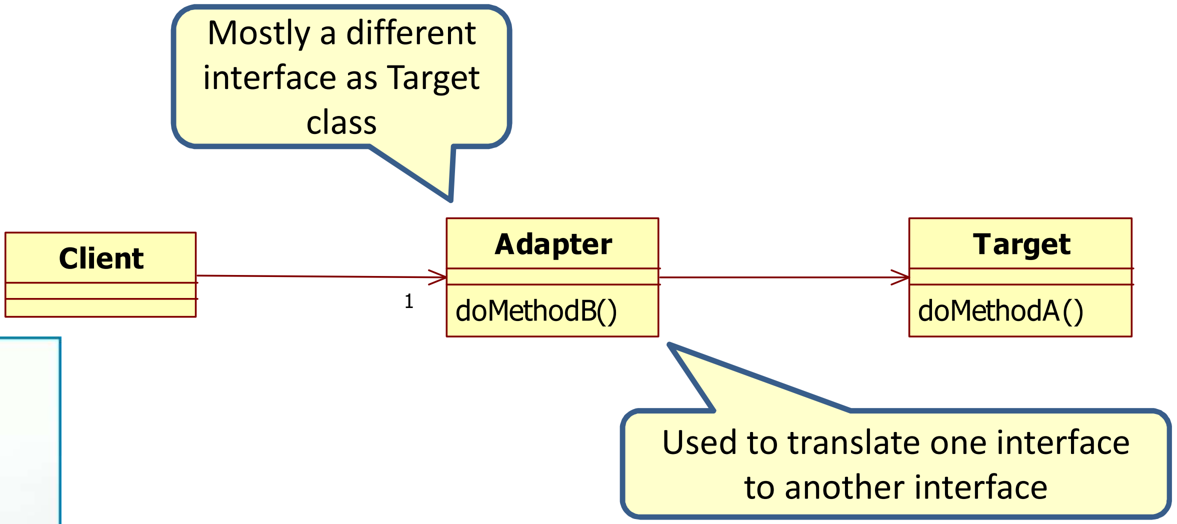


Adapter and Proxy: wrapper

- Proxy



- Adapter


Mostly a different interface as Target class
Client
Adapter
doMethodB()
1
Target
doMethodA()
Used to translate one interface to another interface

Main point

- The Adapter translates an existing interface to a required interface.

- Life is found in layers, from the most abstract transcendental layer (Unified Field) to the most concrete layer.