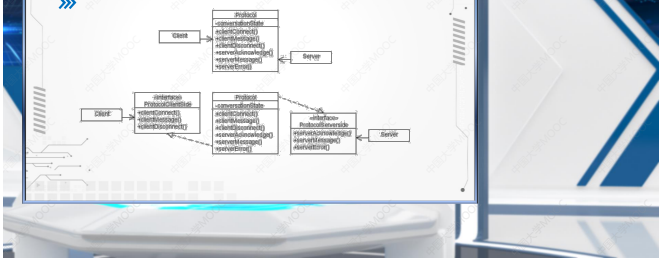
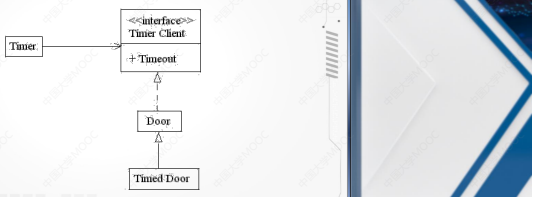
ISP -接口分离原则

- ISP原则解决“胖”接口问题。
- 接口分离原则：
  - 客户程序不应该被迫依赖其不使用的方法。
- 消除多职责的类（接口不内聚的类）
- 将“胖”接口分解成多个接口，每个接口服务特定类型的客户程序。

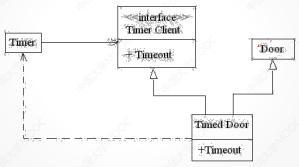
ISP – 例子1



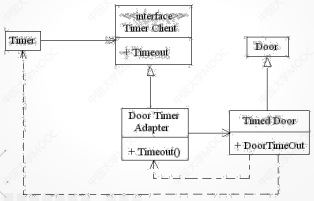
例子 II



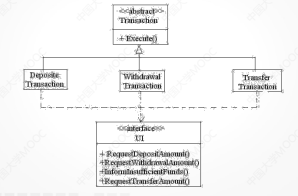
例子 II



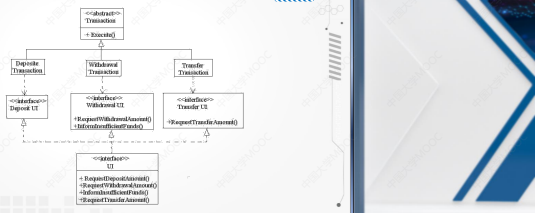
例子 II



例子 III



例子 III



例子 IV

```
// interface segregation principle - bad example
interface IWorker {
public void work();
public void eat();
}

class Worker implements IWorker{
public void work() { // ....working   }
public void eat() {
// ..... eating in launch break
}
}
```

例子 IV

```
class SuperWorker implements IWorker{
  public void work() {   //.... working much more   }
  public void eat() {    //.... eating in launch break    }
  }

  class Manager {
  IWorker worker;

  public void setWorker(IWorker w) {    worker=w;   }
  public void manage() {    worker.work();    }
  }
```

例子 IV

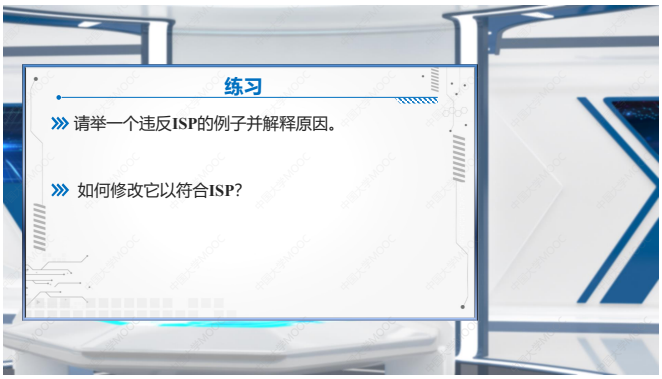
```
// interface segregation principle - good example
interface IWorker extends Feedable, Workable { }

interface IWorkable {     public void work();    }

interface IFeedable{     public void eat();     }

class Worker implements IWorkable, IFeedable{
public void work() {         // ....working      }

public void eat() {   //.... eating in launch break   }
}
```

例子 IV

```
class Robot implements IWorkable{
public void work() {     // ....working    }
}
class SuperWorker implements IWorkable, IFeedable{
public void work() {    //.... working much more   }
public void eat() {     //.... eating in launch break    }
}
class Manager {
Workable worker;
public void setWorker(IWorkable w) {    worker=w;   }
public void manage() {    worker.work();    }
}
```

结论

- ”胖“类会导致类中的多个职责耦合。
  - 当”胖“类的一个职责发生变化时，其所有客户程序都会受到影响。
- ”胖“类的接口应该分解为特定于不同客户程序的多个接口。
  - 客户程序不再依赖其不调用的方法，并且客户程序彼此独立，不再因为“胖”类耦合在一起。


练习
请举一个违反ISP的例子并解释原因。
如何修改它以符合ISP?