河北师范大学软件学院
Software College of Hebei Normal University

第11章 桥接模式

任课教师：武永亮
wuyongliang@edu2act.org

上节回顾

- 代理模式解决的问题是“提供一种代理控制对一个对象的访问”

课程内容

- 环境及问题
- 桥接模式详解
- 桥接模式实现
- 扩展练习

课程内容

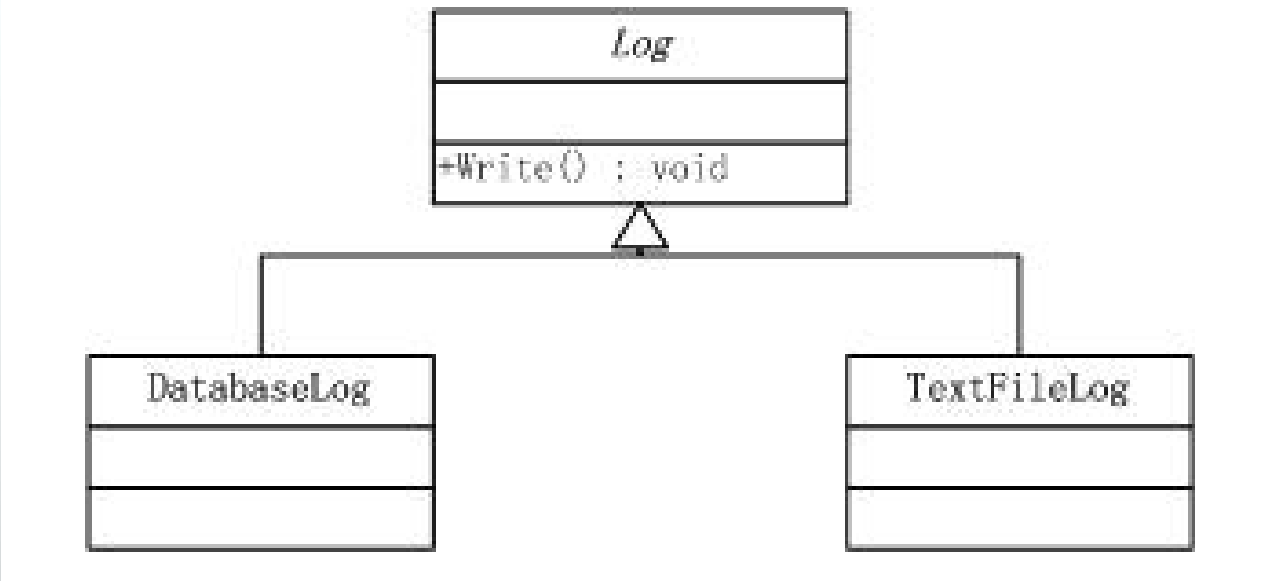
- 环境及问题
- 桥接模式详解
- 桥接模式实现
- 扩展练习

环境

- 现在我们要开发一个通用的日志记录工具
- 它既可以运行在.NET平台，也可以运行在Java平台上
- 它支持数据库记录和文本文件记录

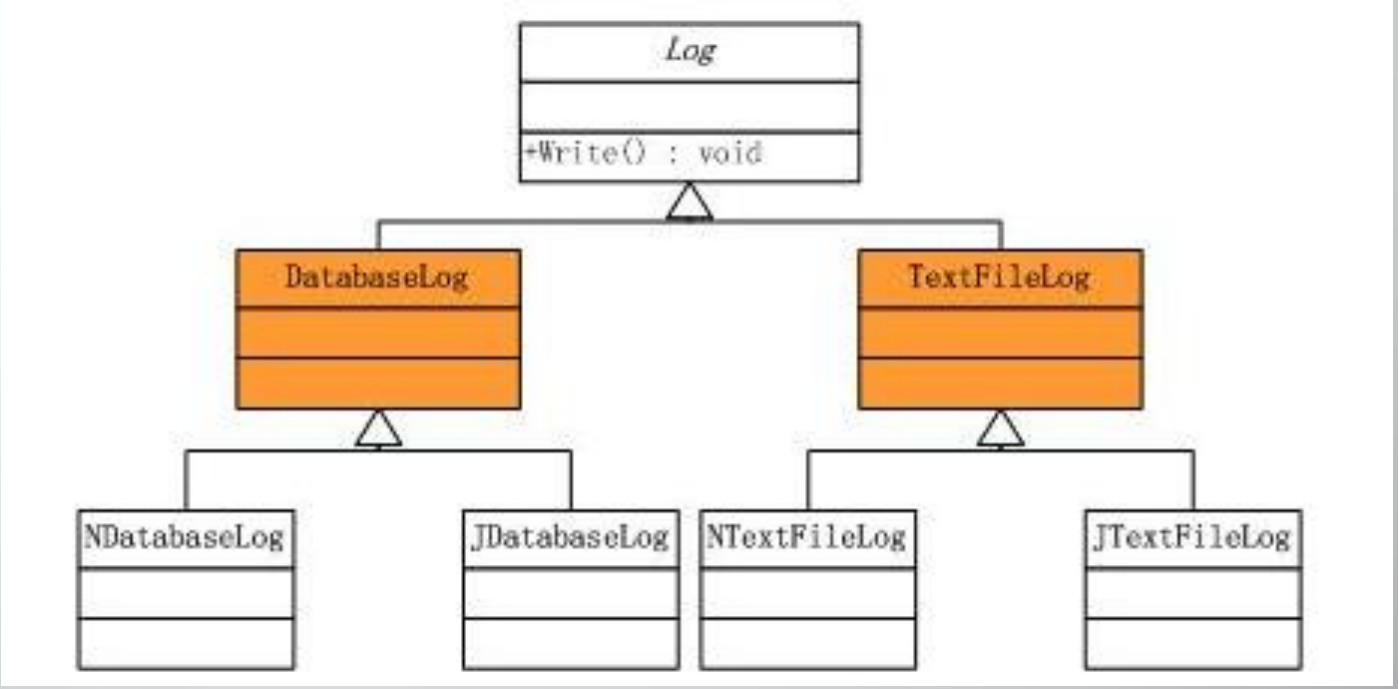
解决方案

- 根据我们的设计经验，应该把不同的日志记录方式分别作为单独的对象来对待，并为日志记录类抽象出一个基类Log出来



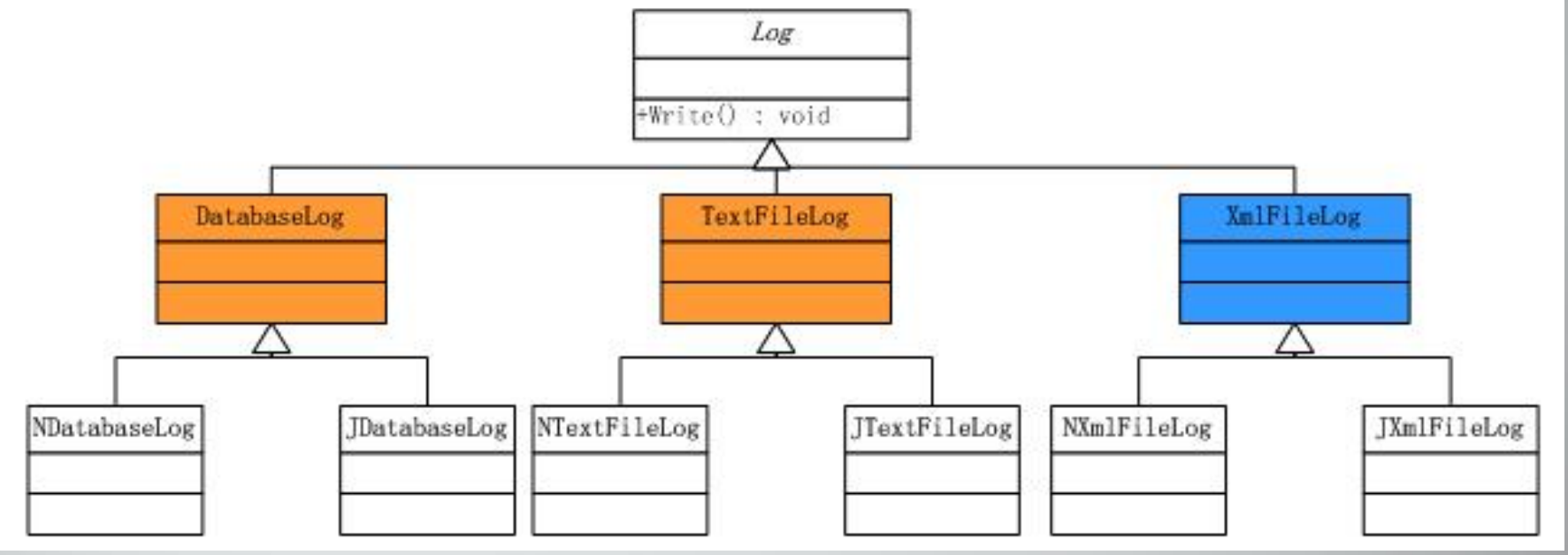
解决方案

- 不同平台的日志记录，对于操作数据库、写入文本文件所调用的方式可能是不一样的



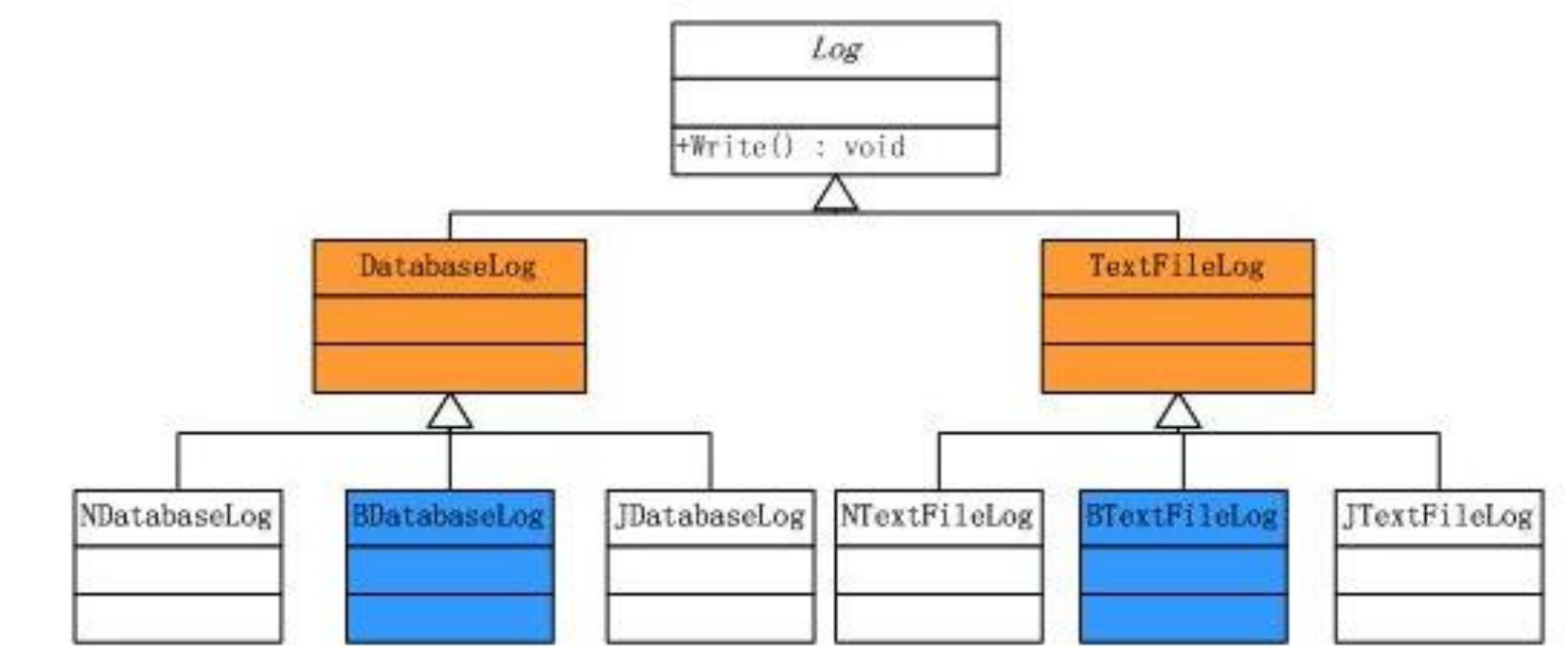
存在的问题

- 假如现在我们要引入一种新的xml文件的记录方式



存在的问题

- 如果我们引入一种新的平台呢？



存在的问题？

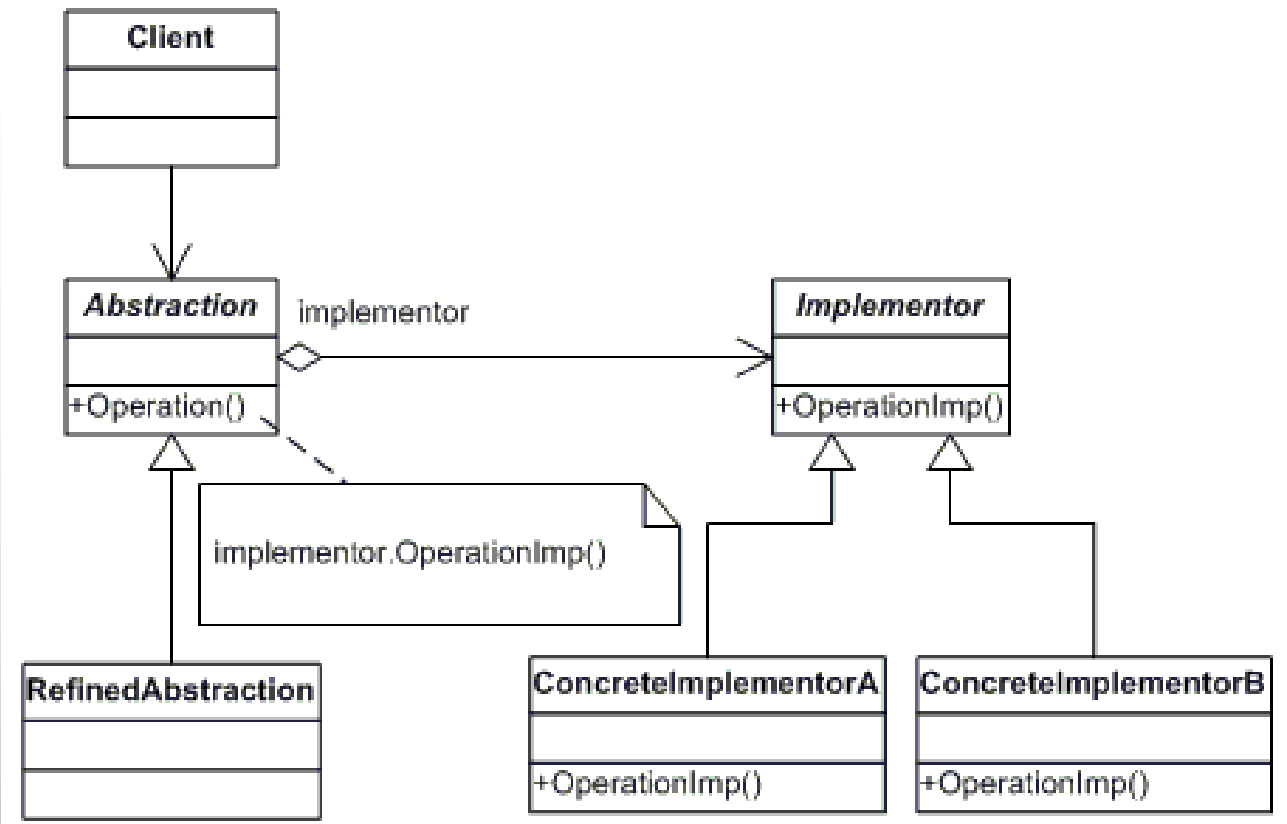
- 它在遵循开放-封闭原则
- 违背了类的单一职责原则
  - 即一个类只有一个引起它变化的原因（平台和日志记录方式都会引起类的变化）

课程内容

- 环境及问题
- 桥接模式详解
- 桥接模式实现
- 扩展练习

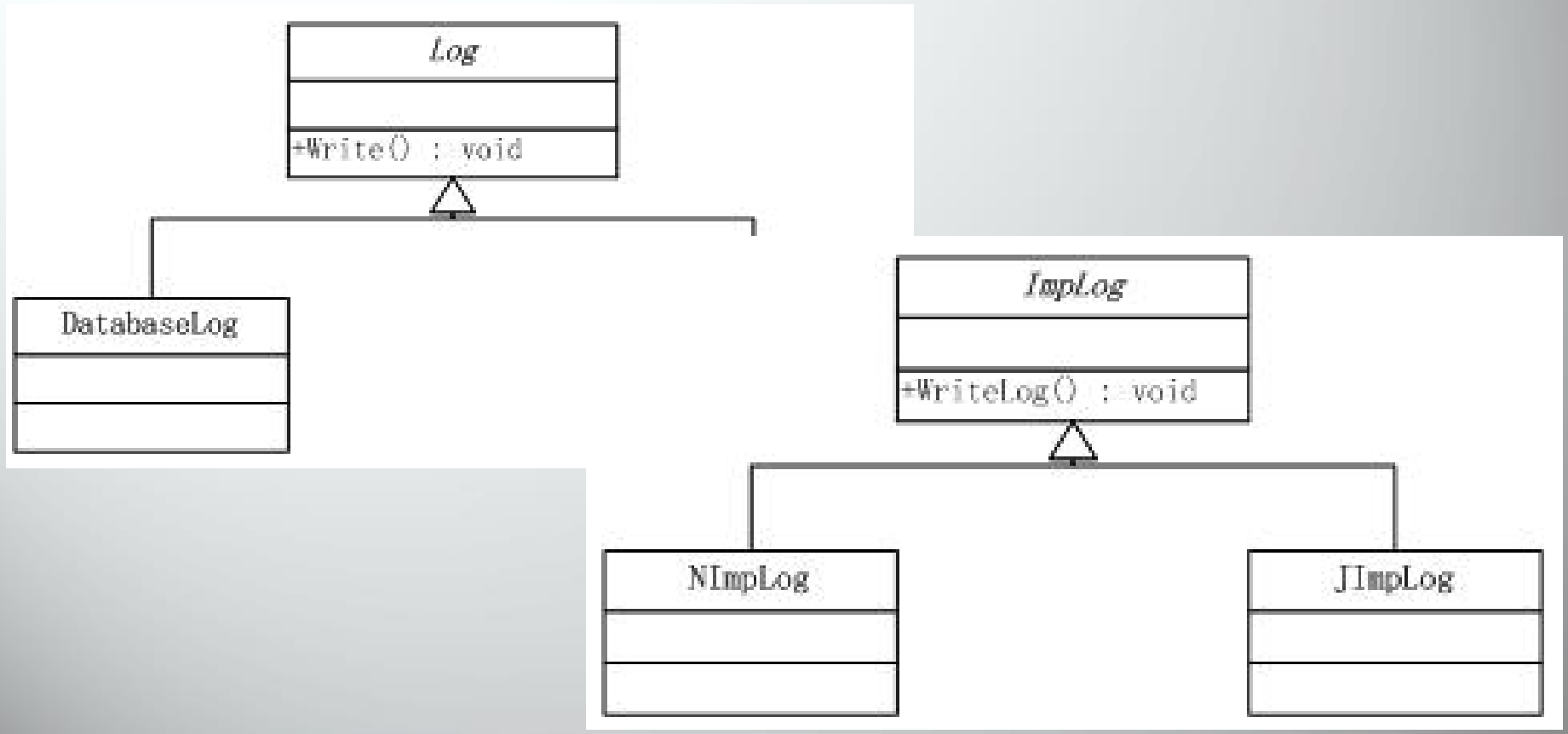
桥接模式详解

- 在软件系统中，如何应对“多维度的变化”？
- 将抽象部分与实现部分分离！



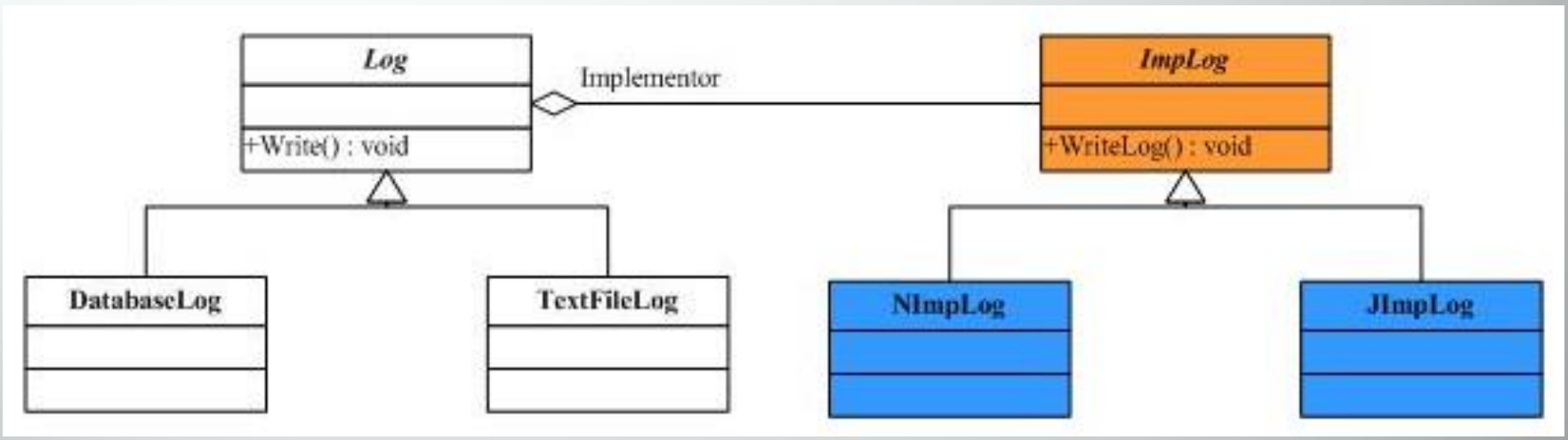
桥接模式详解

- 把日志记录方式和不同平台上的实现分别当作两个独立的部分来对待



桥接模式详解

- 我们要做的工作就是把这两部分之间连接起来
- Bridge使用了对象组合的方式

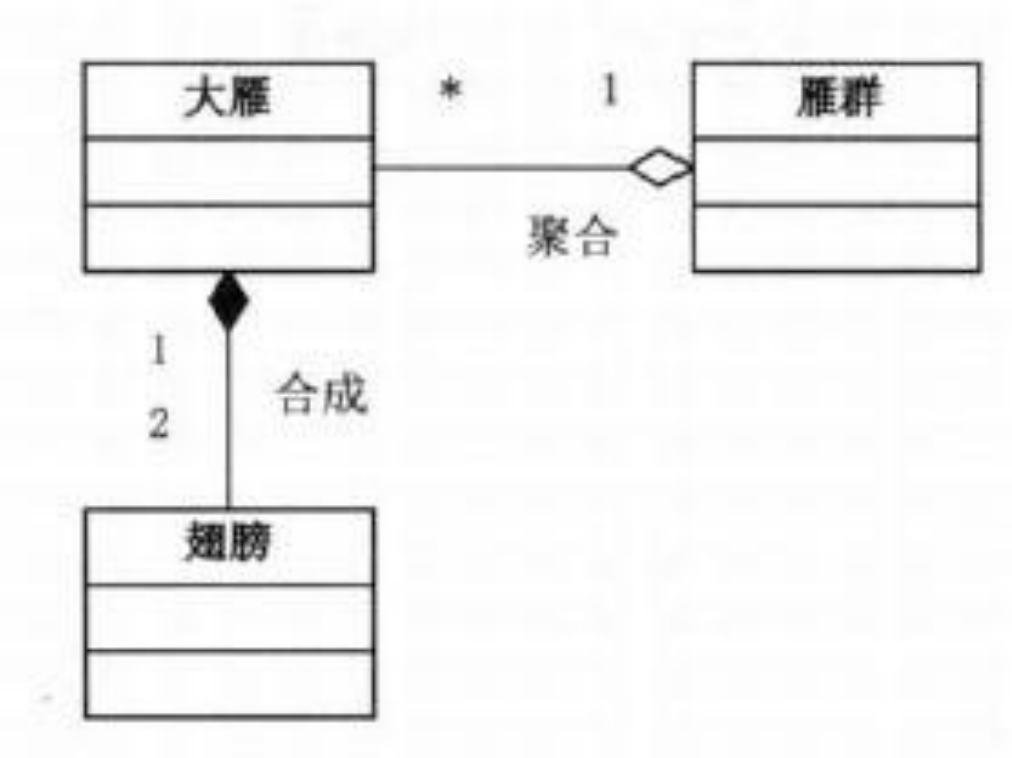


合成/聚合复用原则

- 合成（组合，Composition）和聚合（Aggregation）都是关联关系的特殊种类
- 聚合表示一种弱的“拥有”关系，A对象可以包含B对象，但B对象不是A对象的一部分
- 合成表示一种强的“拥有”关系，体现了严格的整体和部分的关系，部分和整体的生命周期是一样的。

合成/聚会复用原则

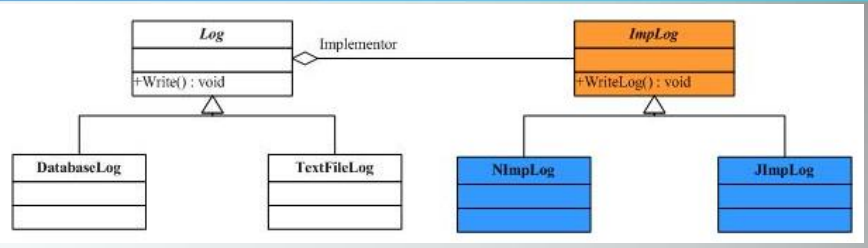
- 大雁和翅膀是什么关系；
- 大雁和雁群是什么关系。



课程内容

- 环境及问题
- 桥接模式详解
- 桥接模式实现
- 扩展练习

桥接模式实现代码

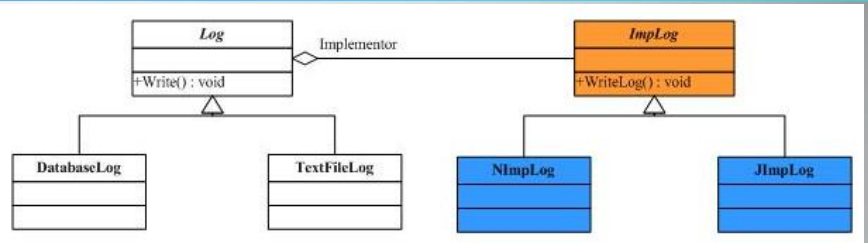



```
public abstract class Log
{
    protected ImpLog implementor;

    public ImpLog Implementor
    {
        set { implementor = value; }
    }
    public virtual void Write(string log)
    {
        implementor.Execute(log);
    }
}
```

```
public class DatabaseLog : Log
{
    public override void Write(string log)
    {
        implementor.Execute(log);
    }
}
public class TextFileLog : Log
{
    public override void Write(string log)
    {
        implementor.Execute(log);
    }
}
```

桥接模式实现代码




```
public abstract class ImpLog
{
    public abstract void Execute(string msg);
}
public class NImpLog : ImpLog
{
    public override void Execute(string msg)
    {
        //..... .NET平台
    }
}
public class JImpLog : ImpLog
{
    public override void Execute(string msg)
    {
        //..... Java平台
    }
}
```

```
class App
{
    public static void Main(string[] args)
    {
        //.NET平台下的Database Log
        Log dblog = new DatabaseLog();
        dblog.Implementor = new NImpLog();
        dblog.Write();

        //Java平台下的Text File Log
        Log txtlog = new TextFileLog();
        txtlog.Implementor = new JImpLog();
        txtlog.Write();
    }
}
```

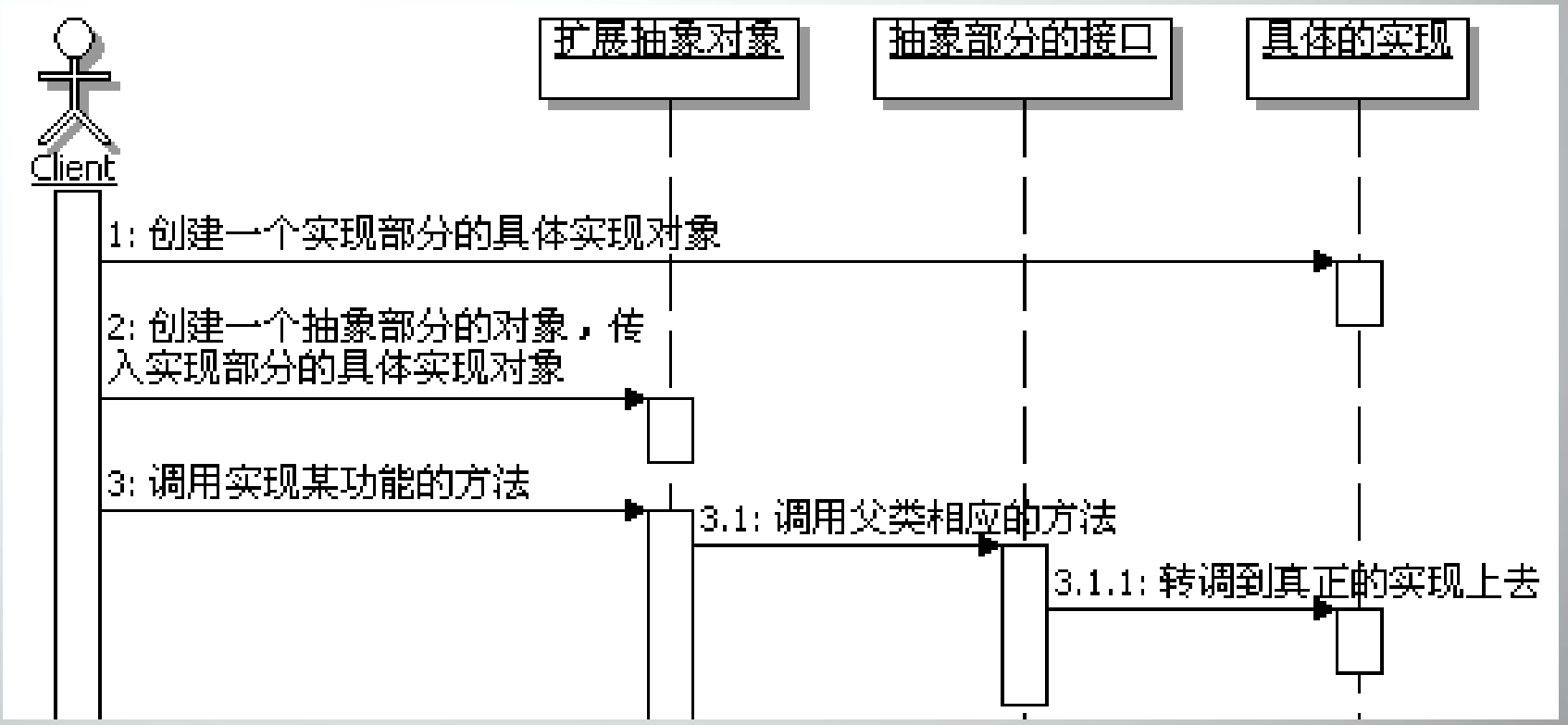
课程内容

- 环境及问题
- 桥接模式详解
- 桥接模式实现
- 扩展练习

扩展说明

- Bridge模式使用“对象间的组合关系”解耦了抽象和实现之间固有的绑定关系，使得抽象和实现可以沿着各自的维度来变化。
- 所谓抽象和实现沿着各自维度的变化，即“子类化”它们，得到各个子类之后，便可以任意它们，从而获得不同平台上的不同型号。

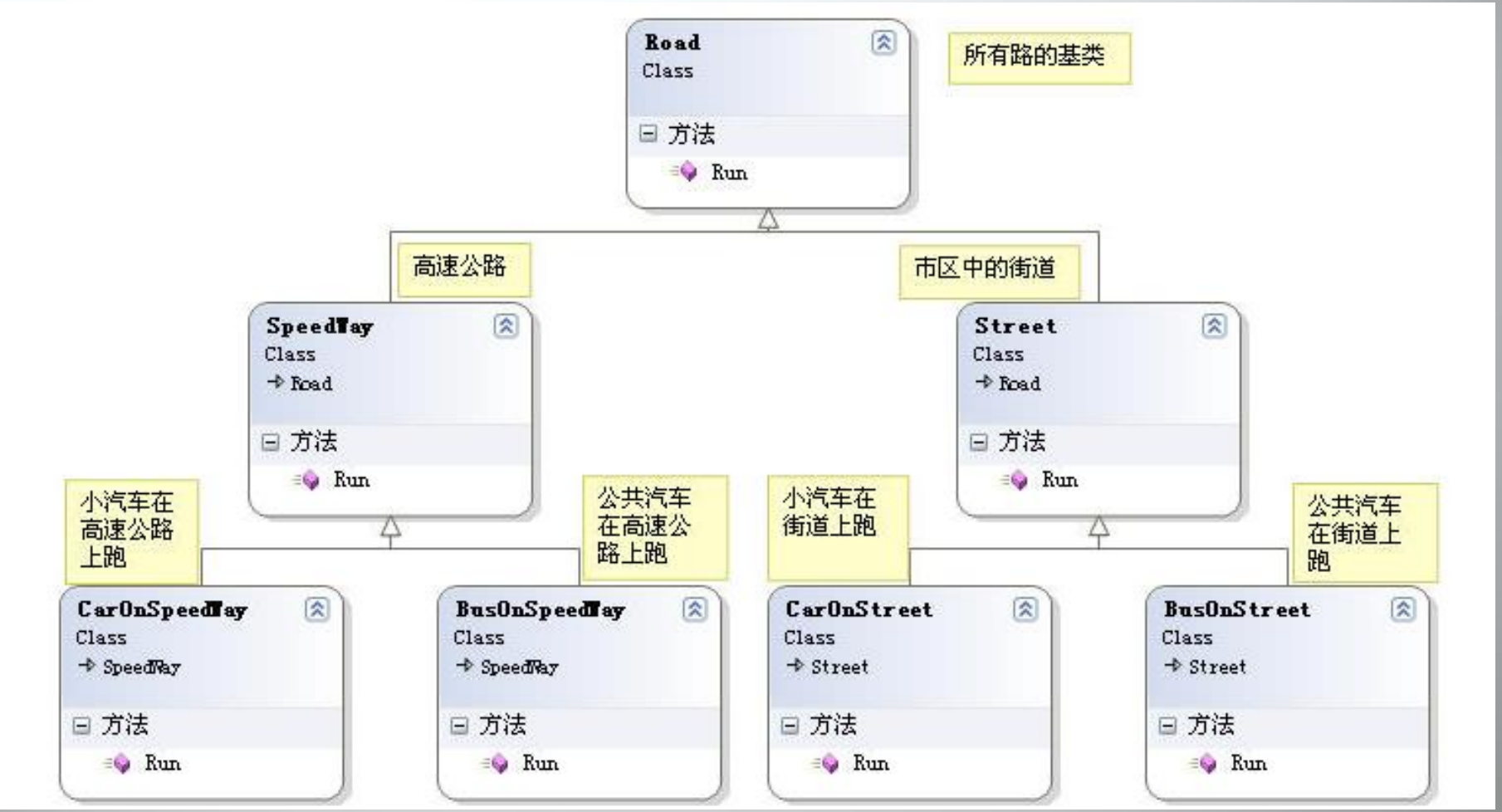
调用顺序图


扩展抽象对象
抽象部分的接口
具体的实现
Client
1: 创建一个实现部分的具体实现对象
2: 创建一个抽象部分的对象，传入实现部分的具体实现对象
3: 调用实现某功能的方法
3.1: 调用父类相应的方法
3.1.1: 转调到真正的实现上去

扩展案例

- 汽车在路上行驶。即有小汽车又有公共汽车，它们都不但能在市区中的公路上行驶，也能在高速公路上行驶。
- 交通工具（汽车）有不同的类型。
- 行驶的环境（路）也在变化。

错误方案


Road
Class
方法
Run
所有路的基类
高速公路
市区中的街道
SpeedWay
Class
Road
方法
Run
Street
Class
Road
方法
Run
小汽车在高速公路上跑
公共汽车在高速公路上跑
小汽车在街道上跑
公共汽车在街道上跑
CarOnSpeedWay
Class
SpeedWay
方法
Run
BusOnSpeedWay
Class
SpeedWay
方法
Run
CarOnStreet
Class
Street
方法
Run
BusOnStreet
Class
Street
方法
Run

正确方案


AbstractRoad
Abstract Class
抽象路
字段
属性
Car
方法
Run
SpeedWay
Class
AbstractRoad
高速公路
方法
Run
Street
Class
AbstractRoad
市区街道
方法
Run


AbstractCar
Abstract Class
抽象汽车
方法
Run
Car
Class
AbstractCar
小汽车
方法
Run
Bus
Class
AbstractCar
公共汽车
方法
Run

难度增加

- 桥接模式（Bridge）如何做(多维度变化)？

- 结合上面的例子
  - 增加一个维度"人",不同的人开着不同的汽车在不同的路上行驶(三个维度);

小结

- 合成/聚合复用原则
- 桥接模式是“将抽象部分和其实现部分分离，使它们都可以独立的变化”

Thank You，谢谢！