

# 面向 Aspect 编程的应用研究

李志纯<sup>1</sup>, 张南平<sup>1,2</sup>

(1. 武汉理工大学 计算机科学与技术学院, 湖北 武汉 430070;

2. 武汉菲旺软件技术有限责任公司, 湖北 武汉 430070)

**摘要:**介绍了面向方面编程(AOP)的应用背景和核心概念,从面向对象编程的不足谈起,引出“横切关注”行为,面向方面编程就是在此基础上应运而生。接着简述了面向方面编程的一些基本概念,然后针对一个简单案例,通过分别用传统编程方式和 AOP 方式实现,从中体现 AOP 编程的优势。

**关键词:**面向方面编程;面向对象编程;横切关注;关注点

**中图分类号:**TP311.1

**文献标识码:**A

**文章编号:**1673-629X(2006)05-0217-02

## Application and Research of Aspect-Oriented Programming

LI Zhi-chun<sup>1</sup>, ZHANG Nan-ping<sup>1,2</sup>

(1. College of Computer Science and Tech., Wuhan Univ. of Tech., Wuhan 430070, China;

2. Wuhan Philwong Software Tech. Co. Ltd, Wuhan 430070, China)

**Abstract:** Introduce the applying background and the core conception of AOP. Discussed from the disadvantage of OOP, crosscutting-concern is put into truth and AOP are based from it. Then presented some definitions of AOP. According to a simple case and accomplishing it with traditional programming and aspect-oriented programming, can know the superiority of AOP.

**Key words:** AOP; OOP; crosscutting concern; concern

### 0 引言

面向对象技术很好地解决了软件系统中角色划分的问题。借助于面向对象的分析、设计和实现技术,开发者可以将问题领域的“名词”转换成软件系统中的对象,从而很自然地完成从问题到软件的转换。人们认识到,传统的程序经常表现出一些不能自然地适合单个程序模块或者几个紧密相关的程序模块的行为,例如日志记录、对上下文敏感的错误处理、性能优化以及设计模式等等。将这种行为称为“横切关注(crosscutting concern)”<sup>[1]</sup>,因为它跨越了给定编程模型中的典型职责界限。如果使用过用于横切关注点的代码,您就会知道缺乏模块性所带来的问题。因为横切行为的实现是分散的,开发人员发现这种行为难以作逻辑思维、实现和更改。因此,面向方面的编程(AOP)应运而生<sup>[2]</sup>。AOP 为开发者提供了一种描述横切关注点的机制,并能够自动将横切关注点植入到面向对象的软件系统中,从而实现了横切关注点的模块化。通过划分 Aspect 代码,横切关注点变得容易处理<sup>[3]</sup>。开发者可以在编译时更改、插入或删除系统的 Aspect,甚至重用系统的 Aspect。更重要的是,

AOP 可能对软件开发的过程造成根本性的影响。

### 1 AOP 概述

#### 1.1 AOP 的概念

AOP 是 OOP 的延续,意思是面向方面编程。AOP 实际是 GoF 设计模式的延续,设计模式孜孜不倦追求的是调用者和被调用者之间的解耦,AOP 可以说也是这种目标的一种实现<sup>[4]</sup>。

#### 1.2 分散关注

将通用需求功能从不相关类之中分离出来;同时,能够使得很多类共享一个行为,一旦行为发生变化,不必修改很多类,只要修改这个行为就可以。AOP 就是这种实现分散关注的编程方法,它将“关注”封装在“方面”中。一个关注点(concern)就是一个特定的目的,一块人们感兴趣的区域<sup>[5]</sup>。从技术的角度来说,一个典型的软件系统包含一些核心的关注点和系统级的关注点。举个例子来说,一个信用卡处理系统的核心关注点是借贷/存入处理,而系统级的关注点则是日志、事务完整性、授权、安全及性能问题等,许多关注点——我们叫它横切关注点(crosscutting concerns)<sup>[5]</sup>——会在多个模块中出现,使用现有的编程方法,横切关注点会横越多个模块,结果是使系统难以设计、理解、实现和演进。

AOP 能够比上述方法更好地分离系统关注点,从而

收稿日期:2005-08-17

作者简介:李志纯(1979-),男,湖南益阳人,硕士研究生,研究方向为 J2EE、设计模式、框架设计、XP 编程、单元测试;张南平,教授,硕士研究生导师,研究方向为计算机网络、ERP。

提供模块化的横切关注点。

## 2 应用举例

### 2.1 案例

假设在一个应用系统中,有一个共享的数据必须被并发同时访问,首先,将这个数据封装在数据对象中,称为 Data Class,同时,将有多个访问类,专门用于在同一时刻访问这同一个数据对象。

为了完成上述并发访问同一资源的功能,需要引入锁 Lock 的概念,也就是说,某个时刻,当有一个访问类访问这个数据对象时,这个数据对象必须上锁 Locked,用完后立即解锁 unLocked,再供其它访问类访问。

### 2.2 传统编程实现

使用传统的编程习惯,会创建一个抽象类,所有的访问类继承这个抽象父类,如下:

```
abstract class Worker {
    abstract void locked();
    abstract void accessDataObject();
    abstract void unlocked();
}
```

缺点:

accessDataObject()方法需要有“锁”状态之类的相关代码。

Java 只提供了单继承,因此具体访问类只能继承这个父类,如果具体访问类还要继承其它父类,比如另外一个如 Worker 的父类,将无法方便实现。

重用被打折扣,具体访问类因为也包含“锁”状态之类的相关代码,只能被重用在相关有“锁”的场合,重用范围很窄。

仔细研究这个应用的“锁”,它其实有下列特性:

(1)“锁”功能不是具体访问类的首要或主要功能,访问类主要功能是访问数据对象,例如读取数据或更改动作。

(2)“锁”行为其实可与具体访问类的主要功能独立、区分开来的。

因此,一个新的程序结构应该是关注系统的纵向切面,例如这个应用的“锁”功能,这个新的程序结构就是 aspect(方面)。

在这个应用中,“锁”方面(aspect)应该有以下职责:

提供一些必备的功能,对被访问对象实现加锁或解锁功能。以保证所有在修改数据对象的操作之前能够调用 lock()加锁,在它使用完成后,调用 unlock()解锁。

### 2.3 AspectJ 实现

下面使用 AOP 的实现之一 AspectJ 来对上述需求改写。AspectJ 是 AOP 最早成熟的 Java 实现,它稍微扩展了一下 Java 语言,增加了一些 Keyword 等,pointcut 的语法如下:

```
public pointcut 方法名:call(XXXX)
```

建立一个类似 Class 的 Aspect,Java 中建立一个 Class 代码如下:

```
public class MyClass {
    //属性和方法 ...
}
```

同样,建立一个 Aspect 的代码如下:

```
public aspect MyAspect {
    //属性和方法 ...
}
```

建立一个 Aspect 名为 Lock,代码如下:

```
import EDU.oswego.cs.dl.util.concurrent.*;
public aspect Lock {
    ReentrantWriterPreferenceReadWriteLock rwl =
        new ReentrantWriterPreferenceReadWriteLock();
    public pointcut writeOperations():
        execution(public boolean Worker.createData()) ||
        execution(public boolean Worker.updateData()) ||
        execution(public boolean AnotherWorker.updateData());
    before(): writeOperations() {
        rwl.writeLock().acquire(); //上锁 advice body
    }
    after(): writeOperations() {
        rwl.writeLock().release(); //解锁 advice body
    }
}
```

上述代码关键点是 pointcut,意味切入点或触发点,那么在哪些条件下该点会触发呢?在执行 Worker 的 createData()方法和 Worker 的 update 方法时将触发。before 代表触发前做什么事情,答案是上锁。after 代表触发之后做什么事情,答案是解锁。

通过引入上述 aspect,那么 Worker 代码可以清洁如下:

```
public class Worker extends Thread {
    Data data;
    public boolean createData() {
        try {
            //对 data 实行写逻辑操作
        } catch() {
            return false;
        }
        return true;
    }
    public boolean updateData() {
        try {
            //对 data 实行写逻辑操作
        } catch() {
            return false;
        } finally {
        }
        return true;
    }
}
```

(多播)协议,使用户正常接收。

系统的工作过程如下:输入的视频和音频信号将送给编码器进行编码,编码器输出的节目流既可以存入存储设备也可以直接送给媒体服务器,媒体服务器的主要功能是完成节目流的播出。媒体服务器播出的节目有 3 个来源,它可能是保存在存储设备中的 ASF 文件,也可以是 Encoder 实时传送来的节目,它播出的节目还可以从其它的媒体服务器上获取。普通的用户可以通过 LAN 或通过无线网络接入到该系统之中。

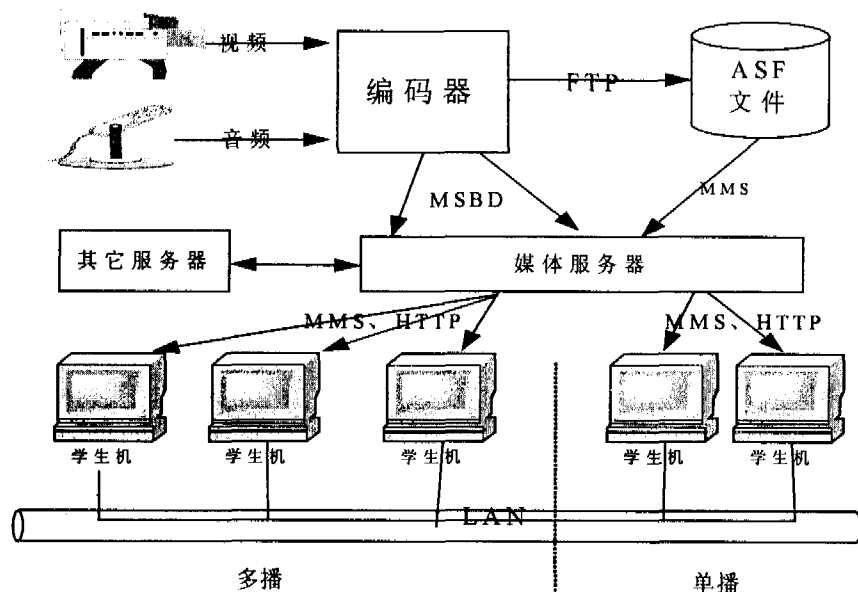


图 3 系统体系结构及工作流程

#### 4.3 系统开发的软件配置

Web 服务器: Microsoft Windows 2000 Server + Internet Information Server 5.0; 媒体服务器: Microsoft Media

Server; 数据库服务器: SQL Server 2000; 编码工具: Windows Media Encode 7.0 (编码生成的格式音频为 wma, 视频为 wmv); 客户端播放器: Windows Media Player; 系统管理、用户开发工具: ASP 3.0, Delphi 7.0。

#### 5 结束语

对于学校的大型活动,如名人讲座、文艺晚会等,由于场地空间的限制而将部分学生拒之门外。通过本系统可以让其他同学从校园网上看到实况广播,使每位同学都可以享受丰富多彩的校园生活。流媒体技术已经成为影响 Internet 的重大技术之一,可以预见,随着宽带网络的建设,流媒体技术在校园会有更广泛的应用前景。

#### 参考文献:

- [1] 钟玉琢,向哲,沈洪.流媒体和视频服务器[M].北京:清华大学出版社,2003.
- [2] 龚颜,姜昌金.流媒体技术及视频捕获和预览的研究[J].微机发展,2003,13(10):77-79.
- [3] 顾洪军.流媒体应用中的 QoS 问题分析[J].计算机应用研究,2003(11):117-119.
- [4] 马杰,田金文,柳键.流媒体技术及其文件格式[J].计算机工程与应用,2003(23):49-52.
- [5] 曹燕萍,谢剑英.Windows 流媒体技术及应用[J].计算机工程,2002,8(28):6-8.

(上接第 218 页)

```

}
public void run() {
    //执行 createData()或 updateData()
}
}

```

Worker 中关于“锁”的代码都不见了,纯粹变成了数据操作的主要方法。

#### 3 结束语

面向方面的技术具有很多潜在的优势。它为在系统中详细指定并封装横切点提供了方法。随着它们的发展,允许更好地进行系统维护——并且确切知道它们仍将继续发展。AOP 还将使我们在关注的形态中,对现存系统以一种有组织的方式增加新的特点。表达以及结构方面的提高允许保持系统运行更长的时间,并且不会带来完全改写的开销就可以增量地对其进行提高。

AOP 还是一个对质量专业人员的工具箱的重大增强。使用 AOP 语言,可以自动测试应用程序代码而不会对代码带来干扰。这将消除可能的代码错误<sup>[2]</sup>。

在理解 AOP 全部潜能中人们还处于一个初始阶段。看起来很明显,这项技术为保证未来的探索与实验提供了足够多的优点。

#### 参考文献:

- [1] 罗时飞.精通 SPRING[M].北京:电子工业出版社,2005.
- [2] Laddad R. AspectJ in Action[M]. America: Manning Publications, 2003.
- [3] 王欣轩.精通 AspectJ[M].北京:清华大学出版社,2005.
- [4] Pawlak R. Foundations of AOP for J2EE Development[M]. America: Apress, 2005.
- [5] Filman R E. Aspect - Oriented Software Development[M]. America: Addison - Wesley Professional, 2004.