第31卷 第20期 计 算 机 工 程 2005年10月
Vol.31 № 20 Computer Engineering October 2005
·软件技术与数据库· 文章编号：1000—3428(2005)20—0096—03 文献标识码：A 中图分类号：TP311

基于 J2EE 体系的 Web 应用框架整合

程 洪，钱乐秋，马舜雄
（复旦大学计算机科学与工程系，上海 200433）

摘 要：在研究了大量流行的 Web 应用框架的基础上，提出了一种 Web 应用框架整合模型，即 WAFC 模型(Web Application Framework Composition)，该模型基于分层思想，结合设计模式方法，给出了一组约束，增加了域对象层、服务定位层和数据接口层，有效地解决了框架整合过程中出现的功能冗余、层间通信不便、耦合度太高等问题。该模型充分挖掘了各个框架的长处，使它们以一种松散耦合方式结合，形成一个更高层次的应用框架。此外还结合了具体实例，分析和探讨了该模型的实例化过程。

关键词：框架；应用框架；复用；松散耦合；解耦

Web Application Framework Composition Based on J2EE

CHENG Hong, QIAN Leqiu, MA Shunxiong
(Dept. of Computer Science and Engineering, Fudan University, Shanghai 200433)

【Abstract】This paper gives a model of Web application framework composition on the base of research of many popular Web application frameworks. It is called WAFC model. It is based on layer architecture, combines with design pattern, gives a set of restrictions, and introduces three new layers which are domain object layer, service locator layer and data access interface layer. The model solves the problems such as function redundancy, inconvenience of layer communication and tight coupling in the process of frameworks composition. The model makes full use of strongpoint of each Web application framework, and combines them to achieve loose coupling. In addition, this paper discusses the process of instantiating the model with an example.

【Key words】Framework; Application framework; Reuse; Loose coupling; Decouple

框架是指一个特定领域中的一组相互协作的类，它是特定领域软件系统可以复用的设计和部分实现。该特定领域软件的开发人员能够运用继承、合成等方法定制和复用框架，开发特定的软件系统。可以说框架的复用是软件生产中最有效的复用方式之一，能尽可能地复用已有的比较成熟的应用框架也是最经济的开发模式。本文所研究的 Web 应用框架是一种面向 Web 应用领域的框架。当前在开放源代码运动的推动下，Web 应用领域基于 J2EE 体系的应用框架层出不穷，其中不乏优秀的设计，如基于 MVC 模式的 Struts、处理持续层的 Hibernate 以及服务于所有层面的 Spring 等。由于各种应用框架数目繁多，且没有公认的最好框架，因此如何更好地复用它们就成为我们面临的问题。

针对该问题，一个好的解决方法是选择其中优秀的框架，将它们整合，以发挥各框架的长处。目前在 Web 应用开发领域，也出现了一些整合方案，但是这些方案往往局限于特定框架的整合，框架整合后耦合度太高，如果需求有变动，想要替换掉某个已集成的框架代价太大。此外，对于整合过程中常见的功能冗余问题、层间框架通信问题也没有好的解决办法。

1 相关理论简介

1.1 3 层应用构架

大部分的 Web 应用在职责上一般被分成 3 层：(1)表示层：处理用户请求，数据信息格式化。它通常由 Web 容器运行，包括 JSP、 Servlet 等 Web 部件。(2)业务层：处理与领域相关的具体业务逻辑，事务管理等。(3)持久层：处理业务对象的持久化问题，通常与数据库相关。

1.2 当前流行的几种 Web 应用框架

(1) Struts

Struts 是一个免费的开源的 Web 应用框架，Struts 基于 MVC 模式，具有很高的可配置性。Struts 把 Servlet、JSP、自定义标签和信息资源整合到一个统一的框架中，开发人员利用其进行开发时不用再自己编码实现全套 MVC 模式，极大地节省了时间，所以说 Struts 是一个非常不错的应用框架，它适合处理表示层。

(2) Spring

Spring 是一个 2003 年 2 月才开始的开源项目，它是一个服务于所有层面的应用框架。Spring 有一个非常显著的特点：在某个层面上如果你不需要 Spring 的支持，就可以只使用它的某一部分的功能。Spring 以一种统一的、Ioc（控制倒置）的方式查找、管理、组装、使用系统的组件，取代一切工厂。这些特性使得 Spring 处理业务逻辑时具有很大的优势。

(3) Hibernate

Hibernate 是一个非常优秀的开放源代码的 O/R Mapping (对象关系映射框架)，在宣称支持 JDO 标准之后，它的生命力显得更加的旺盛了。它对 JDBC 进行了轻量级的对象封装，使 Java 程序员可以随心所欲地使用对象编程思维来操纵数据库。它是目前较优秀的一种持久层框架。

(4) Ibatis

Ibatis 也是一个 O/R Mapping 工具，但是它的定位和

作者简介：程 洪(1978—)，男，硕士生，研究方向：软件工程；钱乐秋，教授、博导；马舜雄，硕士生
收稿日期：2004-08-29 **E-mail：**alongwaych@yahoo.com.cn

Hibernate 不同，它不像 Hibernate 那么复杂，Hibernate 的目标是要提供一个完整的、自动化的 O/R Mapping 产品。但是 Ibatis 的理念不同，它提供的是一种简化的方式来使用 JDBC，和 Spring 的思路有些类似，但 Spring 采用代码重用的方式，而 Ibatis 采用的是映射文件的方式。所以，Ibatis 很容易理解，也很容易学习。Ibatis 是一个轻量级的持久层开发框架，目的也是提高持久层的设计与实现。

2 Web 应用框架整合模型（WAFC 模型）

2.1 WAFC 模型简介

在分析研究了大量 J2EE 体系的 Web 应用框架的基础上，提出了一种基于 J2EE 体系的 Web 应用框架整合模型即 WAFC 模型。它的模型图如图 1 所示。

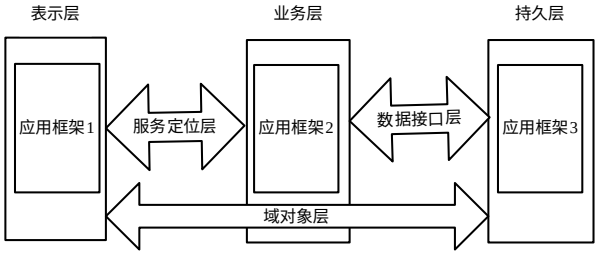

图 1 Web 应用框架整合模型图（WAFC 模型图）

通过上面的相关理论简介，知道大部分的 Web 应用在职责上一般被分成 3 层：表示层，业务层和持久层。而另一方面分层是分解复杂系统最常用的技术，将应用分层可以带来很多好处，可以很容易地改变某个层的实现，而不影响其它层，只要这些层的实现能支持相同的服务。

鉴于以上两方面因素，我们提出的 Web 应用框架整合模型（即 WAFC 模型）的基本思想是基于应用分层构架模式的，用户可以根据实际应用需求，选择相应层次（如表示层、业务层、持久层）的最适合框架，将它们整合，形成一个更高层次的框架，可以说 WAFC 模型是一种框架的框架。图 1 中应用框架 1、应用框架 2、应用框架 3 指适合于各个相应层次的 Web 应用框架，为了行文方便把它们称为子框架。特别地，WAFC 模型在传统的 3 层构架模式基础上，增加了两个中间层即服务定位层和数据接口层，这样做的好处是可以进一步降低层次间的耦合度，后文将有所讨论。此外，还引入了域对象层，它是联系各个层之间的纽带。

2.2 WAFC 模型特点分析

2.2.1 引入设计模式方法，能有效地解耦

所谓解耦就是降低耦合度。我们希望对框架进行整合后能得到一个具有好的开放性和灵活性的框架模型，这就要求所得到的框架模型的各个层面的子框架间能做到松散耦合，这样在实际应用中，可以根据需要很容易地替换掉某个层面的子框架而不影响其它层面。这一目标的实现关键就是子框架间的解耦问题，设计模式提供了一个好的解决方法。

设计模式是针对反复出现的设计问题给出的、经过实践检验的、可复用的解决方案。通常，设计模式与领域无关，可用于不同的应用系统和不同的领域，它记录了现有的、专家的设计知识，可以用来为具体的设计问题提供适当的解决办法；而框架是一种可复用的、可适配的软件，它要有灵活的结构以易于扩展。框架中通常可能包含若干个设计模式，而设计模式比框架具有更高的抽象性和更好的通用性。可见，设计模式和框架二者既有联系又有区别，正因如此，在 WAFC 模型中，可以将设计模式方法深化到框架整合中，从而有效地对持久层子框架和业务层子框架、表示层子框架和业务层子框架解耦，使得各个层面间子框架之间松散耦合。

(1)持久层子框架和业务层子框架解耦

为了对持久层子框架和业务层子框架解耦，WAFC 模型采用了 J2EE 设计模式中的数据访问对象模式(DAO)，在持久层子框架和业务层子框架间增加了数据接口层。DAO 模式通过接口和实现分离的方式，完全向上层隐藏了数据源实现细节。当低层数据源实现变化时，DAO 向客户端提供的接口不会变化，所以该模式允许 DAO 调整到不同的存储模式，而不会影响其上层。重要的是，DAO 充当了业务层和数据源之间的适配器，将低级别的数据访问逻辑与高级别的业务逻辑分离。因此通过在业务层子框架和持久层子框架间增加了数据接口层，能有效地降低两者的耦合度。

(2)表示层子框架和业务层子框架解耦

为了降低表示层子框架和业务层子框架间的耦合度，WAFC 模型采用 J2EE 另一个常用模式-服务定位器模式，提供了一个通用的服务定位层。服务定位器模式中有一个对象（即服务定位器）知道如何获得一个应用程序所需的所有服务。WAFC 模型中表示层子框架需要请求业务层子框架服务时，将向服务定位器层发出请求，间接获取相应的业务服务对象接口，而不是直接向业务层请求。WAFC 模型提供的这个服务定位层是与特定框架无关的，它根据 XML 配置文件里定义的服务映射关系来寻找表示层所需的业务服务对象，从而降低了业务层子框架和持久层子框架间的耦合度。

2.2.2 引入一组约束，能有效地解决功能冗余问题

Web 应用框架有的是针对多层次的，甚至是所有层次的，如 Spring 框架。但是仅仅选用其特定层面的功能，这时功能冗余问题就出现了。我们的 WAFC 模型针对应用于各个层次的子框架，提出了一组约束，将某个特定的框架功能限定于特定的层次，这样可以有效解决功能冗余问题。我们提出的约束如下：

(1)针对表示层子框架的约束

一个典型的 Web 应用的末端应该是表示层。我们规定表示层子框架的主要功能有：管理用户的请求，作出相应的响应；为显示提供一个模型。而不该在表示层子框架实例化编码中出现的，与表示层无关的部分，作出不可以出现的约束。不可以出现在表示层子框架中的有：直接地与数据库通信，例如 JDBC 调用；与应用程序相关联的业务逻辑以及校验；事务管理。如果不作出这些约束规定，在表示层框架编码中引入这些代码，则会带来高耦合和麻烦的维护。

(2)针对业务层子框架的约束

我们规定可以在业务层子框架中出现的主要功能有：处理应用程序的业务逻辑和业务校验；管理事务；允许与其它层相互作用的接口；管理业务层级别的对象依赖；管理程序的执行（从业务层到持久层）。

(3)针对持久层子框架的约束

可以在持久层子框架中出现的主要功能有：查询对象的相关信息语句；如何存储、更新、删除数据库记录；支持大部分主流数据库，并且支持 Parent/Child 关系，事务处理，继承和多态。

2.2.3 引入域对象层，能有效地解决框架通信问题

在框架整合过程中，层间框架通信问题也是一个难以解决的常见问题。为了有效解决框架通信问题，我们的 WAFC 模型引入了域模块层，域模块层由实际需求中的业务对象组成，它为联系各个层面提供了一个纽带，有了域模块层，业务领域对象信息就能够在层间进行传递，我们仅需关注域对象即可。

3 WAFC 模型实例化探讨

3.1 WAFC 模型实例分析

结合“学生成绩查询系统”来分析 WAFC 模型，主要目的是说明如何进行框架整合，因此对系统功能进行了如下简化：学生通过 Web 方式登录系统，系统对学生身份进行合法性验证；验证通过则显示该学生的成绩信息，否则提示未通过合法性验证。在分析了当前流行的 Web 应用框架后，决定这样来实例化我们的 WAFC 模型：表示层子框架采用前面介绍的 Struts，业务层子框架采用 Spring，持续层子框架采用 Hibernate。WAFC 实例化后的框架如图 2。

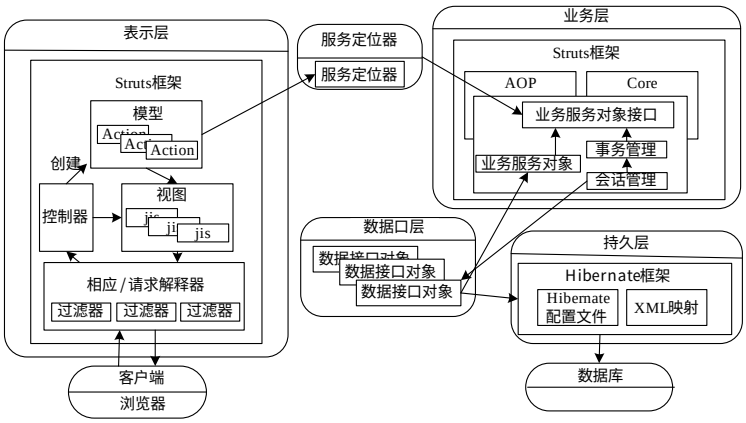

图 2 WAFC 模型实例化后的框架图

从图 2 可以看出确定了各个子框架，完成了 WAFC 模型的实例化后，得到了一个新的应用框架；下一步工作就是实例化这个框架，进行具体的应用开发了。

(1)域对象层

域对象层是连接各层的纽带，首先要创建域对象(即域对象层中的对象)并且确认域对象中哪些是需要持久化的，哪些是提供给业务层的，哪些是显示接口设计的。这层是编码的开始，我们创建了域对象层的两个对象，即

1)Cn.com.wingsoft.bo.Student.java 学生对象（包括学生学号、密码等信息）

2)Cn.com.wingsoft.bo.Grade.java 学生成绩对象（包括学生各科成绩信息）

(2)持久层

Hibernate 是一个设计良好的持久层框架，不存在功能冗余问题，所以持久层子框架采用它。Hibernate 通过 XML 文件来映射对象，Student.hbm.xml 文件和 Grade.hbm.xml 文件分别映射了 Student 和 Grade 对象。Hibernate 的 Session factory 对象是用来告诉程序应该与哪个数据库通信，该使用哪个连接池或使用了哪种数据源，应该加载哪些持久对象。而 Session 接口是用来完成增删改查这些操作的。

(3)业务层

Spring 框架拥有强大的 Bean 处理能力，对业务层的处理能力强，采用面向方面的 Spring 框架来实例化我们的业务层子框架。在我们的例子中，用一个业务服务对象接收一个数据接口层的数据访问接口对象，用它来控制域对象层对象的持久化。WAFC 模型在持久层和业务层中间增加了一个数据接口层，这样业务层子框架和持久层子框架间能做到松散耦合，所以虽然在这个例子中持久层使用了 Hibernate，但我们可以很方便地用其他适合持久层的框架替换它，而不影响业务层的实现。本例中业务服务对象提供学生身份验证和获取成绩信息服务，下面是它的代码片断：

```
public interface IStudentService {
    public abstract Grade getStudentGrade(String stuid)
        throws StudentException;
    public abstract Student LoginByStuid(String stuid,String passwd)
                throws StudentException;
    public abstract void setStudentDAO(
                IStudentDAO StudentDAO); }
```

(4)服务定位层

将我们的业务服务和数据接口层搭配起来，下面需要把我们的业务服务提供给表示层。WAFC 模型通过提供一个通用的服务定位器来返回给表示层业务服务对象接口，以降低表示层和业务层的耦合度。下面是服务定位器代码片断。

```
this. IStudentService = (IStudentService)
WSL.getBean("studentService")
```

(5)表示层配置

此例中表示层子框架采用了 Struts 框架。根据前文我们作出的约束，Struts 框架的功能应该是委派调用业务逻辑和处理显示，它不应该出现与持久层通信的代码（如 JDBC 调用）以及业务层应处理的职责（如与应用程序相关联的业务逻辑及校验）。Struts 最重要的配置文件是 Struts-config.xml，如下：

```
<action path="/getStuGradAction"
                type="cn.com.wingsoft.action.
GetStudentGradeAction "
        name="StudentForm"
        scope="request"
        validate="true"
        input="/login.jsp">
        <forward name="success" path="/ViewStudentGrad.jsp"/>
        <forward name="failure" path="/login.jsp"/>
    </action>
```

GetStudentGradeAction 这个动作是根据学生学号，从持久层获取学生成绩信息，然后再显示到 JSP 页面。

3.2 WAFC 模型开放性探讨

WAFC 模型中，由于各个层间子框架是松散耦合的，因此可以很容易地对某个层面的子框架进行替换而不影响其它层面，这给开发 Web 应用带来了极大的灵活性。如上例中可以将持久层的 Hibernate 框架替换为 Ibatis。由于在业务层中间加入了中间接口层，替换中只需用 Ibatis 框架实例化这些 DAO 接口，产生新的 XML 文件即可。类似地，也可以用 Struts 外的其他框架替换现在的表示层子框架，Spring 外的其他框架替换业务层子框架。

4 结束语

本文在研究了大量基于 J2EE 的 Web 应用框架基础上，提出了一种 Web 应用框架整合模型（即 WAFC 模型）。该模型引入了设计模式方法，增加了服务定位层和数据接口层，将设计模式方法深化到框架整合中，克服了目前框架整合过程中，不能有效地解耦，框架整合后耦合度高的不足；该模型引入了一组约束，有效地解决了目前框架整合过程中常见的功能冗余问题；该模型还引入了域对象层，有效地解决了框架整合过程中的框架通信问题。总之，我们的 WAFC 模型对多个 Web 应用框架按照一定的方法和规则进行整合，形成了一种更高层次的框架，整合后各个子框架间松散耦合，具有好的开放性和复用性。当然对于如何高效地为特定的层次选择合适的框架的问题还需进一步研究和探讨。

(下转第 120 页)

现模块，实现插接功能。

策略库：保存管理员制定的策略。

模块库：对网络上各层设备进行拓扑发现时可采用的发现技术集合，各模块输入为 IP 地址，输出为该 IP 与其它设备的连接关系。

适配器：分析当前节点设备的性质，当前网络类型及所支持的协议。

策略匹配：对适配器传入的参数进行分析，查找定位策略规则对象。

规则判决：获取规则判断策略条件是否为真。

规则执行：根据规则定义的动作执行范围控制或拓扑搜索。

4.2 主程序伪代码

L3 层拓扑发现

```
ScopePolicy scope=new ScopePolicy();//Step1
SearchPolicy search=new SearchPolicy();//Step2
RouterList rl=scope.getRouters_From_IPCondition();//Step3
SubnetList validatesnl=scope.getSubnets_From_SubnetCondition();
while(!rl.empty()){
Router cr=rl.shiftFirstElement();//Step4
String[]parameter=cr.ProtocolDetect();//Step5
String algorithm=search.getL3Policy(parameter);// Step6
Ventor link=cr.search(algorithm);// Step7
SubnetList snl=link.getR_SUB();//Step8
RouterList nextHop=link.getR_R();//Step9
Validatesnl.add(scope.check(snl));// Step10
rl.add(scope.check(nextHop));// Step11
}
```

L2 层拓扑发现

```
while(!validatesnl.empty()){//Step12
Subnet cs= validatesnl.shiftFirstElement();//Step13
String[]parameter=cs.ProtocolDetect ();//Step14
String algorithm=search.getL2Policy(parameter);//Step15
Ventor link=cs.search(algorithm);// Step16
}
```

程序说明：

Step 1 根据管理员的范围策略新建范围策略对象。

Step 2 新建一个搜索策略对象。

Step 3 从范围策略的 IP 限制条件获取待发现初始路由器列表或初始子网列表。

Step 4 取出路由器列表的首元素，作为当前正在搜索的设备。

Step 5 获取当前网络特征参数。

Step 6 将参数传入策略匹配点进行规则判决，对可用的发现算法模块进行选优。

Step 7 执行相应的拓扑发现算法，返回路由器与其周边设备的连接关系。

Step 8 获取与当前路由器直连子网列表。

Step 9 获取与当前路由器直连路由器列表。

Step10 将在管理员范围策略内的子网加入待发现子网队列。

Step11 将在管理员范围策略内的路由器加入待发现路由器列表。

Step12 进行子网拓扑发现。

Step13 选取子网列表中的首元素作为当前搜索的子网对象。

Step14 获取子网参数。

Step15 根据子网参数及管理员行为参数，匹配子网搜索算法。

Step16 进行子网拓扑发现，返回子网内交换机间，交换机与主机的连接关系。

5 总结

基于策略的拓扑发现技术采用可插接的模块化设计方法，通过策略技术将使用哪些发现技术、如何控制拓扑发现的决策权交付系统管理员，方便管理员将已知晓的网络知识应用到发现过程中，根据不同的网络条件选择不同的拓扑算法，根据不同的需要重构拓扑结构，为提高拓扑发现算法的广泛适应性和各种算法的优势互补提供了一种崭新的思路。

参考文献

1 陈彦铮，黄世育，温君伟. 网路拓扑探索界限机制. 2002 年台湾网际网路研讨会, 2002-11

2 李　可. 基于 Web 的网络管理中拓扑管理的研究与应用[硕士学位论文]. 上海：上海交通大学，2003

3 张少俊. 基于 CIM/WBEM 的网络管理策略研究与实现[硕士学位论文]. 上海: 上海交通大学，2002

(上接第 45 页)

参考文献

1 丘水生, 陈艳峰, 吴　敏等. 混沌保密通信的若干问题及混沌加密新方案. 华南理工大学学报(自然科学版), 2002, 30 (11): 75-80

2 郝柏林. 从抛物线谈起——混沌动力学引论. 上海: 上海科技教育出版社, 1993

3 陈鲁生, 沈世镒. 现代密码学. 北京: 科学出版社, 2002

4 Sobhy M I, Shehata A-E R. Methods of Attacking Chaotic Encryption and Countermeasures. In:IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001, 2: 1001-1004

5 冯登国. 密码分析学. 北京: 清华大学出版社, 2000

(上接第 98 页)

参考文献

1 Ana C V. Melo D, Moutinho M. One the Composition of Java Frameworks Control Flow. ACM, 2003

2 Mattsson M, Bosch J. Framework Composition: Problems, Causes and Solutions. In:Proc.of the Tools-23:Technology of Object-oriented Languages and Systems, 1997:203

3 Johnson R.Introducing to Spring Framework.http://xglw.51.net/5team/springframework/viewtopic.php?t=18

4 Eagle M. Wiring Your Web Application with Open Source Java. http://www.onjava.com/pub/a//onjava/2004/04/07/wiringwebapps.html, 2004

5 Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-oriented Software. New York: Addison-wesley, 1995