第41卷
2005年10月

南京大学学报(自然科学)
JOURNAL OF NANJING UNIVERSITY
(NATURAL SCIENCES)

Vol.41
Oct., 2005

基于构件组装的Web应用开发方法*

叶　菲,赵文耘,彭　鑫,张　志
(复旦大学计算机科学与工程系软件工程实验室,上海,200433)

摘　要： 把构件组装的思想引入到Web应用的开发中,提出了一种J2EE体系结构下基于构件组装的Web应用开发方法。该方法在Web层使用动作映射模型和转发映射模型来实现页面控制流建模,同时引入服务接口层作为门面向Web层提供业务服务,而业务层构件则按照服务接口层的需要进行组装。这样就可以分别对Web层和业务层进行建模,然后通过服务接口层把各个层次的构件组装成一个完整的可部署的Web应用。

关键词： 构　件,组　装,页面流转,Web应用
中图分类号： TP 311

A Component-Assembling Based Approach to Web Application Development.

Ye Fei, Zhao Wen-Yun, Peng-Xin, Zhang Zhi
(Department of Computer Science and Engineering, Fudan University, Shanghai, 200433, China)

Abstract: This paper introduces the concept of component-assembling into development of web applications and proposes a component-assembling based approach for J2EE web applications. An action mapping model and a forward mapping model are applied to model screen flow control on the web tier. A service interface tier is also introduced to act as the facade to provide services for the web tier. Thus business components can be assembled according to the interfaces defined on the service interface tier. Then components on various tiers can be assembled to generate the whole deployable web application.

Key words: component, assembly, screen flow, web application

软件复用是解决软件危机,提高软件生产效率和质量的有效途径[1].基于构件的软件开发(Component Based Software Development,简称CBSD)是近年来软件复用的研究热点之一.CBSD包括构件开发、构件管理和构件组装三个过程.在CBSD中,系统开发的重点在于构件组装[2].经过构件技术和Web应用开发的发展,在各个特定领域已经积累了大量的经实践检验的可复用构件,构件组装取代了传统的开发方式成为Web应用开发的主要方式.CBSD中的构件组装通常关注的是封装业务逻辑的构件的组装,而对于Web应用来讲,没有专门针对于封装表示逻辑的构件和封装业务逻辑的构件的整体组装的研究.北京大学提出的ABC方法[3]是一种基

* 基金项目:国家863计划(2004AA112070,2004AA113030,2004AA1Z2330),国家自然科学基金(60473061),上海市科委科研攻关项目(04DZ15022)

于体系结构的、面向构件的软件开发方法,把体系结构的概念与CBSD方法相结合,使得体系结构建模的思想贯穿需求、设计、实现等整个软件开发过程.ABC方法适用于所有的软件开发过程,但是没有特别针对于Web应用开发提供支持.本文提出了一种基于构件组装的Web应用开发方法(Component-Assembling Based Web application Development,简称CABWD),把ABC方法的思想运用到Web应用的开发中,在此基础上,针对于基于J2EE体系结构的Web应用,支持分别对表示逻辑和业务逻辑进行建模,引入了一个中间的服务接口层,服务接口层的构件作为Web层封装表示逻辑的构件与业务层封装业务逻辑的构件的组装关系的门面,从而实现整个Web应用的整体组装.

1 基于J2EE体系结构的Web应用开发

基于J2EE体系结构的Web应用一般是四层结构的,由上而下分别是客户层、Web层、业务层、企业信息系统层[4].客户层包括动态的Web页面和浏览器.Web层构件包括Servlet、JSP页面以及可选的JavaBeans构件.业务层构件封装了核心业务逻辑,通常是各种EJB构件.企业信息系统层通常是数据库和遗留系统,如ERP系统、CRM系统等等.

Web层和业务层通常采用JSP Model2体系结构.JSP Model2体系结构是符合MVC模式的,实现了模型model、视图View、控制器controller角色的分离和相互配合,已经成为整合运用Servlet和JSP技术的标准方式[5].JSP Model2体系结构是以Servlet为中心的,Servlet作为控制器,负责控制流转过程,接受用户提交的请求,把用户请求分派给相应的业务逻辑进行处理,然后根据处理的结果选择一个视图返回给用户.Bean作为业务模型,封装了业务逻辑,负责和数据进行交互.JSP中不包含业务逻辑,只包含表示逻辑,JSP负责生成视图返回给用户.JSP Model2清晰地分离了表示逻辑和业务逻辑,使得Web应用的表示逻辑和业务逻辑可以独立地进行演化.在传统的Web应用开发中,页面流转控制的代码通常是分散在各个Servlet和JSP代码中的,以硬编码的方式嵌入在表示逻辑中的,更改页面的流转需要改动Servlet或者JSP页面的代码.为了集中控制页面流转,业界涌现了不少的优秀的Web应用框架,如Jakarta Struts[6]、WAF[7]等,但是只局限于该框架.另外,传统的Web应用开发方法对于各个层次可复用构件的组装缺乏指导.

2 CABWD方法

CABWD为了通用地解决页面流转控制问题,把页面流转的相关信息抽取出来,提出了动作映射模型(Action Mapping Model)和转发映射模型(Forward Mapping Model)对页面流转关系进行建模.传统的Web应用开发中,Web层构件直接调用业务层构件,构件的修改和替换都引起调用它的构件内部代码的修改.CABWD方法在Web应用的层次结构中在Web层和业务层中间引入了一个服务接口层,作为Web层和业务层的中介,服务接口层将业务层构件提供的服务以一种方便Web层构件使用的方式进行包装,然后提供给Web层.CABWD方法的层次结构如图1所示:

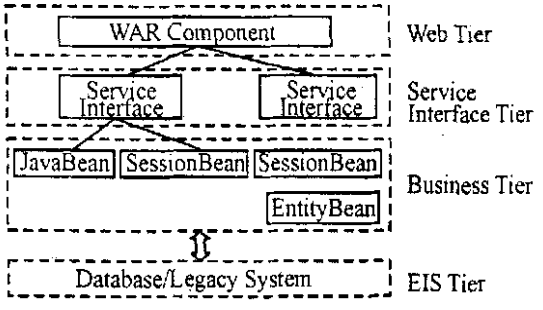

图1 CABWD方法的层次结构

CABWD方法对页面流转进行集中控制,对Web层构件和业务层构件的组装关系分别进行建模,然后对于整个Web应用进行系统组装.

2.1 CABWD方法的开发过程 从层次结构上看,CABWD方法是一个由上层到下层的设计和

实现的过程.从 Web 层设计建模到业务层设计建模,从集中控制页面流转到指定构件的实现实体,最后进行系统组装和部署,CABWD 有以下 5 个步骤:

第一,Web 层建模以及定义 Web 层构件和服务接口层构件之间的接口.根据用户需求,设计 Web 层构件,定义好响应客户请求的每个动作(Action)以及每个动作对外请求的服务.整个 Web 层作为一个 WAR 构件,把功能紧密相关或者有时序关系的服务封装在一个请求接口中 s.

第二,Web 层页面流转控制.定义每个动作的动作映射和转发映射.对于每个请求指定相应委派的动作路径,对于每个动作的处理结果指定下一步处理的动作路径或者视图.

第三,业务层建模以及定义业务层构件和服务接口层构件之间的接口.定义业务层构件之间的组装关系以及业务层构件与服务接口层构件之间的组装关系.

第四,指定构件实现实体.组成一个应用系统的构件分为三类:通用基本构件、领域共性构件、应用专用构件[8].通用基本构件和领域共性构件都存在构件库中,应用专用构件需要进行另外开发.具体的步骤是开发应用专用构件,从构件库中选定实现相应功能的构件,使构件适配,为 Web 应用整个系统体系结构中的每一个构件指定实现实体.

第五,系统组装和部署.验证构件实例之间的组装关系以及体系结构的完整性,按照组装关系生成粘合代码(glue code),打包 Web 应用,然后部署到指定的应用服务器上.

2.2　Web 层页面流转的集中控制　动作(Action)是页面流转控制的中心概念,是 Servlet 和业务逻辑之间的桥梁,Servlet 根据请求路径把处理任务委派给相应的 Action,Action 对外请求的服务作为 Web 层构件向服务接口层构件请求服务的接口.Action 辅助 Servlet 完成控制器的任务.页面流转控制的关键在于动作委派和转发选择两个部分.因此,要集中管理页面流转控制的信息,CABWD 方法抽取出动作映射信息和转发映射信息.动作映射信息是用户提交的请求所要执行的动作信息,用来把用户请求映射到相应的动作.转发映射信息是根据动作执行的结果所要返回的视图或者所要执行的下一个动作的信息,用来把执行结果转发给下一个视图或者下一个动作.这两个映射信息都是围绕动作(Action)的.

动作映射模型如下:

动作映射 :: = <请求,动作,表单,参数>

动作 :: = <动作路径,动作类型,动作的实现类>

表单 :: = <表单名,作用范围,提交的页面,是否需要验证>

一个请求只能对应一个动作,但是不同的请求提交给 Servlet,Servlet 可能会委派执行同一个动作.动作路径在整个 Web 应用中是全局唯一的,动作路径是一个动作的标识.尽管如此,不同动作路径的动作可能由同一个 Java 类来实现,路径不同的动作被看作是不同的动作.如果表单需要验证,验证又不通过的话,就要返回提交请求表单的页面.

转发映射模型如下:

转发映射 :: = <动作路径,动作结果,转发目标,转发类型>

转发目标 :: = <视图路径> | <动作路径>

一个动作可以有一个或者多个执行的结果,每个结果对应一个转发目标,转发的类型可以是直接转发(forward)或者是重定向(redirect),转发目标可能是视图路径,也可能是动作路径.也就是说一个动作执行之后的后续操作是返回下一个视图和执行下一个动作两者之一.控制器 Servlet 查看转发映射信息,根据动作路径以及动作执行的结果,选择转发目标,按照指定的转发类型,如果转发目标是视图,就返回视图给用户;如果转发目标是动作,就把请求转发给目标动作,然后执行目标动作.执行目标动作之后,同样由控制器 Servlet 来根据转发映射来选择下一个转发目标.

2.3　业务层构件组装关系建模　Web 层构件主要用来封装表示逻辑,业务层构件主要用来封

装业务逻辑以及持久化策略,服务接口层构件作为业务层构件为Web层构件提供的服务的门面(Facade),这种设计符合的门面设计模式[9].所有的构件都有一个通用的构件模型:构件对外请求的服务封装在构件的请求接口(Request Interface)里面,构件对外提供的服务封装在服务接口(Service Interface)里面.如图2所示,左边是构件模型,右边是构件例子:

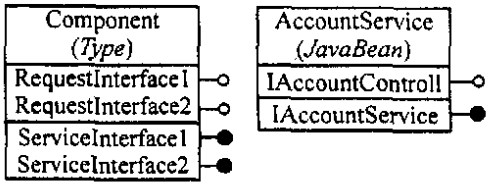

图2 构件模型

在CABWD方法中,整个Web层作为一个WAR构件,服务接口层构件是一个"虚拟"的构件,不需要用户从构件库中选择构件实体,也不需要用户开发一个特定构件实体来与之对应,通常是根据与Web层构件和业务层构件的连接关系由工具自动生成的.WAR构件把一组功能紧密相关的、或者有先后时序关系的请求服务的接口看作是一个服务的门面,由一个服务接口层构件完全提供这一组服务接口.一个服务接口层构件可以与一个或者多个(通常是多个)业务层构件进行连接,服务层构件对外请求的服务由业务层构件提供.WAR构件只对外请求服务,不对外提供服务.服务接口层构件既有向WAR构件提供服务的服务接口,又有向业务层构件请求服务的请求接口.业务层构件可以既有服务接口,又有请求接口,可以与向上层(服务接口层)或者本层(业务层)构件进行连接.在一个完整的构件组装模型里,必须实现的请求接口一定要与相应的服务接口进行连接,不能悬空.在构件组装生成粘合代码(glue code)之前必须验证模型的完整性.

构件之间的通信必须通过连接器来进行,连接器类型有方法调用式、消息传递式、面向方面式.所有的连接器都有一个通用的连接器模型:连接器有两个端口,一个端口叫做入口(In Port),用来连接充当请求服务角色的构件的请求接口;一个端口叫做出口(Out Port),用来连接充当提供服务角色的构件的服务接口.如图3所示,左边是连接器模型,右边是连接器例子:

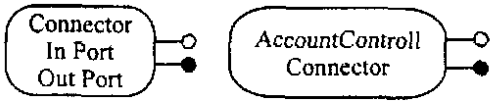

图3 连接器模型

一些基本的约束:

1. 在一个组装里面,构件的请求接口只能连接到一个连接器的请求端口.也就是说,确定了一个组装,一个构件所请求的封装在一个接口里面的服务,只能由确定的一个构件来实现.

2. 连接器的请求端口只能与一个构件的一个请求接口相连,连接器的提供端口只能和一个构件的服务接口相连.一个连接器上两个端口的连接就确定了一个构件的一个请求接口所要请求的服务完全由另一个构件的一个服务接口来实现.

3 工具支持及应用实例

为了支持CABWD方法,开发了一个可视化的建模工具BSAppBuilder Version2.0.BSAppBuilder Version1.0是2004年我们开发的一个基于C2风格的Java/EJB构件组装工具,为了实现基于接口调用方式的构件组装,开发了BSAppBuilder V2.为了便于实现集中的页面流转控制,BSAppBuilderV2在Web层采用了Jakarta开发的Struts框架,Struts框架的当前应用最广泛的Web框架之一.BSAppBuilderV2可以支持Web应用的可视化地集中控制页面流转,可视化地对于构件组装进行建模,生成构件组装的粘合代码,然后打包成可部署的Web应用,并支持把Web应用部署到指定的J2EE应用服务器上.

上海市成人高考网上报名系统是我们为上

海考试院开发的一个供考生在线填写志愿信息并缴纳报名费和考试费的 Web 应用系统. 各高校可以使用该系统对招生计划进行修改,对各个专业的考试科目、报名费、考试费进行设置,并查看报考考生的信息等等. 系统根据考生报考志愿所要求考试科目算出应缴的报名费和考试费,然后考生通过网上支付平台进行缴费. 图 4 和图 5 是使用 BUAppBuilderV2 画出来的页面流转图和构件组装图的一部分:

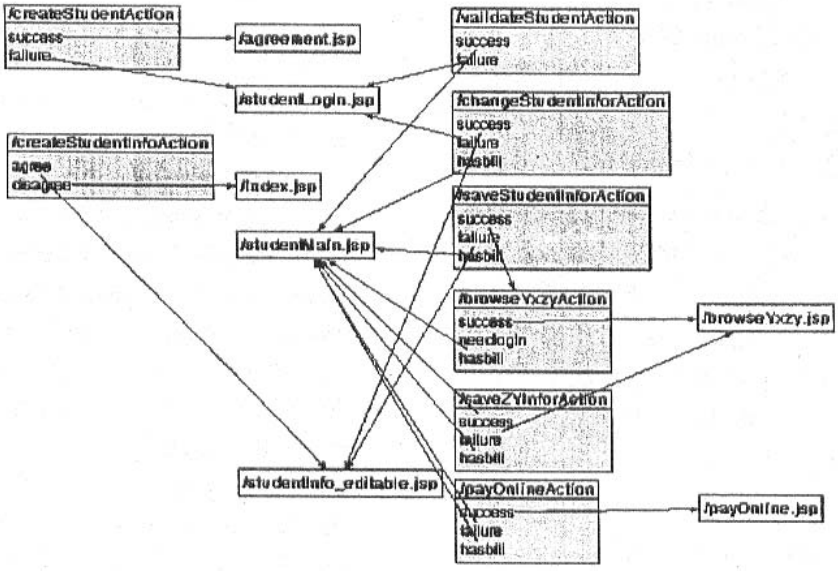

图 4　页面流转图实例

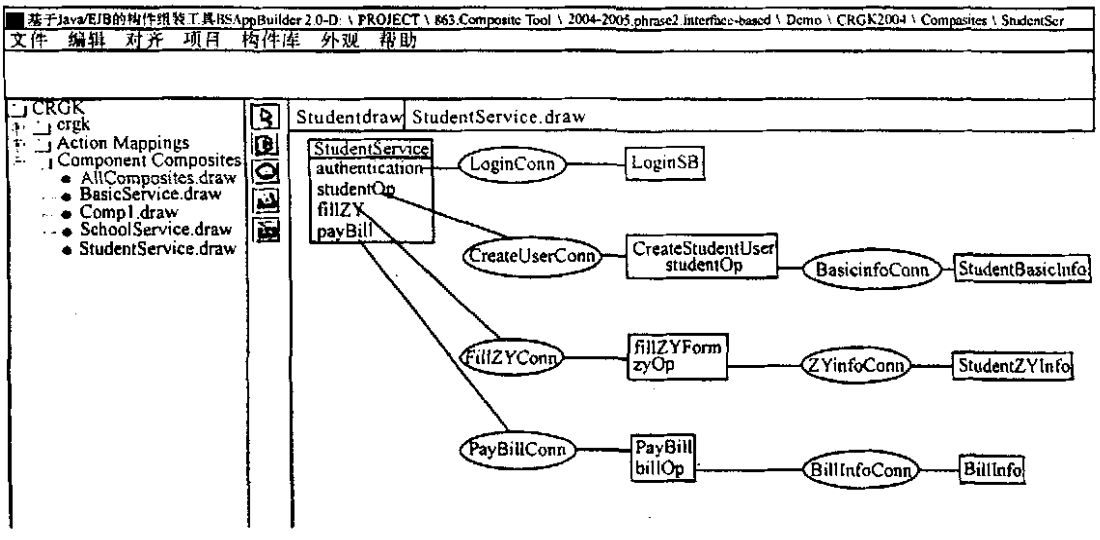

图 5　构件组装图实例

图 4 中 validateStudentAction 是考生输入用户名和密码之后提交表单所映射到的动作, createStudentAction 是创建考生用户的动作, createStudentInforAction 是创建考生用户详细信息的动作, saveStudentInforAction 是保存考生用户详细信息的动作, browseYxzyAction 是

浏览院校专业信息的动作，saveZYInfor是保存志愿信息的动子，payOnlineAction是网上缴费的动作.每个Action具有请求路径属性，以及一个或者多个转发名称(如：success、failure、hasbill等).转发目标可能是一个jsp文件，也可能是另一个Action.根据整幅页面流转图，把页面流转控制信息生成到Struts的配置文件中.

网上考试报名系统提供3类功能，(1)考生用户注册、登录、填写个人信息、填写志愿、在线付费等服务；(2)院校用户设置专业计划、设置专业考试科目、设置专业考试费和报名费、查看报名考生信息等服务；(3)一些公共服务如信息代码和名称对照等.图5描述的是网上考试报名系统考生服务部分的组装关系，考生服务构件作为考生服务的门面，有4个请求接口：身份验证、考生信息、考生志愿、网上服务.这4个请求接口所请求的服务分别由4个SessionBean构件来提供，SessionBean构件所请求的数据CRUD操作服务又由EntityBean构件提供.根据整幅构件组装图，首先为每个构件指定具体实现的构件实体，然后可以生成构件间的粘合代码，使系统成为一个可部署的Web应用.

4 总结和展望

对于在J2EE体系下的Web应用，本文提出的CABWD方法是一种基于构件组装的、多层次结构、分别对业务逻辑和表现逻辑建模的开发方法.CABWD方法，集中控制表现逻辑的页面流转，把整个Web层作为一个WAR构件，在层次结构中引入了一个服务接口层，服务接口层构件作为业务层构件向Web层构件提供服务的门面.CABWD方法使得Web应用各层次可以清晰明确地分别进行建模，集中控制页面流转逻辑，提高Web应用开发的效率和质量.进一步研究工作在于细化CABWD方法，兼容多种Web应用框架，支持多种构件组装的风格和方式.

References

[1] Mili H, Mili F, Mili A. Reusing software: Issues and research directions. Software Engineering, IEEE *Transactions*, 1995,21(6):528～562.

[2] Paul C. Clements from subroutines to subsystems: Component-based software development. Component-Based Software Engineering: Selected Papers from the Software Engineering Institute. Wiley-IEEE Computer Society Press,1996,3～6.

[3] Mei H, Chen F, Feng Y D, *et al*. ABC: An Architecture based, component oriented approach to software development. Journal of Software, 2003, 14(4):721～732.(梅 宏，陈 峰，冯耀东等. ABC:基于体系结构、面向构件的软件开发方法. 软件学报,2003,14(4): 721～732).

[4] Sun Microsystems, Inc. The J2EE 1.4 Tutorial (for the Sun Java System Application Server Platform Edition 8.1 2005Q1 UR1). Charter 1: Overview, Distributed Multitiered Applications. http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html, 2004-12-16

[5] Qusay H M. Servlets and JSP Pages Best Practices. http://java.sun.com/developer/technicalArticles/javaserverpages/servlets-jsp/, 2003-03.

[6] Jakarta Struts. http://struts.apache.org/.

[7] Sun Microsystems, Addison-Welsey. Designing Enterprise Applications with the J2EETM Platform, Second Edition. 2002. New York.

[8] Yang F Q, Mei H, Li K Q. Software Reuse and Software Component Technology. Acta Electronica Sinica,1999,27(2): 68－75.(杨芙清，梅 宏，李克勤. 软件复用与软件构件技术. 电子学报，1999,27(2): 68～75).

[9] Erich G, Ralph J, Richard H. Design Patterns: Elements of Reusable Object-Oriented Software. United States. Addison-Wesley,1994.