

支持MDA的J2EE PSM模型描述及代码生成方法

侯金奎, 万建成, 杨 潇

(山东大学计算机科学与技术学院, 济南 250061)

摘要:为解决模型驱动体系结构(MDA)中PSM模型描述存在的问题,以软件体系结构为基础,综合当前J2EE平台下广泛使用的多种应用框架,提取其共同特征并加以抽象分析,建立了一种包括领域构件模型、行为构件模型和界面展示构件模型3部分的J2EE平台模型描述方法。用该方法描述的模型不仅具有J2EE平台的特性信息,还便于向多种目标框架下的代码进行转换。以此为基础,引入抽象平台有关模型和具体平台有关模型的概念,进一步扩展了MDA开发框架。介绍了相应代码生成器的框架设计和实现过程。

关键词:模型驱动体系结构;软件体系结构;平台有关模型;Web应用框架;代码生成

MDA-supported J2EE PSM Model and Its Code Generation Approach

HOU Jin-kui, WAN Jian-cheng, YANG Xiao

(School of Computer Science and Technology, Shandong University, Jinan 250061)

【Abstract】In order to resolve problems in describing platform specific models of MDA, an architecture based modeling approach for J2EE platform specific models is proposed by analyzing and abstracting the common features of the widely-used Web application frameworks on J2EE. Application model built with this approach is composed of three parts: domain component model, action component model and UI presentation component model, which can be transformed into target codes based on many frameworks easily, while containing the specific information on J2EE platform. Based on this approach, the MDA-based development framework is extended by adding abstract platform specific model and concrete platform specific model. The corresponding code-generator's framework and its realization process are presented briefly.

【Key words】model driven architecture(MDA); software architecture; platform specific model; Web application framework; code generation

OMG提出并倡导模型驱动体系结构(model driven architecture, MDA)^[1],它通过体系结构和软件设计模型的可视化,将业务逻辑设计与不同的中间件平台分离。首先建立与具体技术无关的平台无关模型(platform independent model, PIM),然后通过模型转换得到加入了技术细节的平台有关模型(platform specific model, PSM),最后由代码生成器根据PSM生成目标代码。模型在软件开发过程中扮演了核心的角色,并在整个软件生命周期中得到复用。MDA能带来快速开发、可移植性、代码一致性、可维护性以及体系结构等方面的好处。

软件开发人员总结出了很多Web应用设计模式,这些模式不仅成为了软件开发的工具,同时也成为Web应用软件的评价标准。如J2EE平台上的Struts、Spring、JSF、Cocoon、Axis和Xalan等框架都得到了广泛的使用,这既降低了软件开发的成本,也保证了Web应用系统的质量。现有的MDA工具中,基于J2EE平台的Web应用代码生成器大多与特定框架绑定,如OptimalJ、Rational XDE、Arcstyler以及AndroMDA等,这不利于系统模型的重用,同时对于不熟悉该框架结构的用户而言,修改、扩展和维护目标代码都是很困难的。针对以上问题,本文基于目标平台体系结构风格,提出了一种支持MDA的J2EE PSM模型描述方法。

1 J2EE PSM模型描述方法

1.1 J2EE平台体系结构风格

J2EE应用体系结构一般分为4层^[2]:(1)客户端层,由直

接与用户交互的浏览器(或手机、PDA等)组成;(2)Web层,主要功能是使用JSP和Servlets生成Web页面及其中的动态内容,同时还负责把从客户端传来的请求打包发送到业务层中的EJB组件进行处理以及接收处理后的返回信息;(3)EJB层,又称业务逻辑层,主要负责数据处理以及与数据库或其他Java程序的通信;(4)EIS层,指企业信息系统(如数据库系统、文件系统)。

J2EE提供的构件主要有3类:(1)客户端的Applet和程序客户;(2)Web容器内的JSP、Servlet和JSTL等构件;(3)EJB容器内的EJB构件和资源连接构件。

1.2 DAU-PSM结构模型

本文综合当前J2EE平台下广泛使用的多种应用框架,提取其共同特征并加以抽象分析,建立了DAU-PSM结构模型。它包含J2EE平台的特性信息,支持多种目标框架下的代码生成,其宏观结构如图1所示。

DAU-PSM结构模型是一个3元组:<领域构件模型,行为构件模型,界面展示构件模型>。领域构件(Domain Component)的元模型描述如图2所示,主要包括Java应用程

基金项目:山东省科技发展基金资助项目“基于体系结构和模型驱动的Web应用建模及其系统生成”(20051014)

作者简介:侯金奎(1976-),男,博士研究生,主研方向:Web信息技术与应用工程;万建成,教授、博士生导师;杨潇,博士研究生

收稿日期:2006-09-18 **E-mail:**houjk@mail.sdu.edu.cn

序中的包(Package)、类(JavaClass)、接口(JavaInterface)、属性(Attribute)、方法(Method)以及类之间的关联关系等。

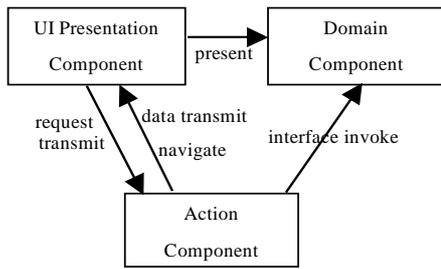


图1 DAU-PSM 结构模型

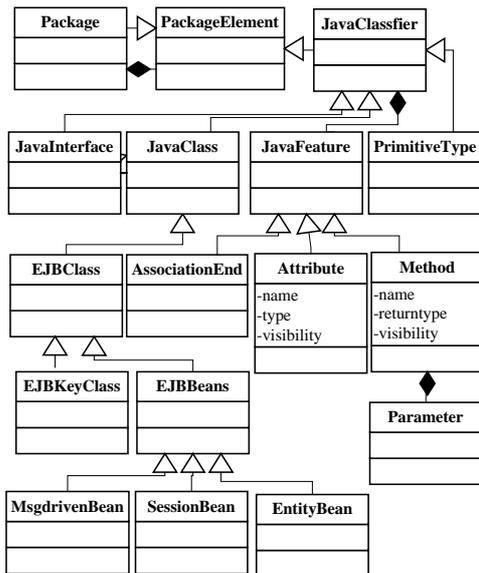


图2 DAU-PSM 领域构件元模型定义

DAU-PSM 行为构件模型以系统的任务处理为核心,从动态层面反映系统的信息。该模型以用户与系统的功能交互为依据,根据系统对用户请求的处理进行组织。其元模型构成如图3所示,主要由行为元素(WebAction)、导航目标(WebPage)、行为参数对象(ActionPara)、操作目标对象(DataObject)以及各种关联关系等组成。

系统行为元素通常包括2种:一种是系统与用户交互的接口,用户可以直接触发;另一种是系统内部行为元素,由第一种行为元素间接触发。在Web应用中,导航信息作为业务流程的主线贯穿系统与用户交互的全部过程,导航目标即代表功能请求处理完成后系统的进一步动作。行为参数对象代表在系统进行某种行为处理的过程中需要引入的参数对象。行为目标对象表示行为处理的过程中需要进行操作的对象集合。导航关联(navigation)表示导航源端到目标端的关联关系。调用关联(invoke)是行为处理过程中对实体对象方法的调用关系。

行为构件元模型的基本语义是:用户向系统发出任务请求,并传递 ActionPara 参数,系统识别请求类型并找到对应的 WebAction,由 WebAction 调用相应的 Invoke 关联对象的方法进行任务处理,任务执行完成后根据不同的处理结果进一步转发请求,转发的目标由 Navigation 的条件和目标确定。

DAU-PSM 界面展示构件模型通过树型层次关系反映 Web 应用所特有的界面元素之间的关系。简单的 Web 界面元素可以抽象成5类,包括数据对象、数据汇集、查询条件对

象、控制参数对象和行为触发对象。其中数据对象主要用于数据的输入和输出,对象的不同属性成分采用不同类型的控件来展示,一般为自由格式;数据汇集是指对数据对象集合集中进行展示的形式和操作,主要包括自由形式、表格形式、树型形式、图表形式和对象组形式;查询条件对象是指用于生成查询条件的对象,展示形式为简单控件组,通常有确认按钮或超连接与之组合;控制参数对象是指用于生成控制参数的数据对象,展示形式为下拉列表框、CheckBox 等简单控件组;行为触发对象代表和执行者相关的功能操作,其展示形式为普通按钮、图片按钮、菜单项和超链接等。

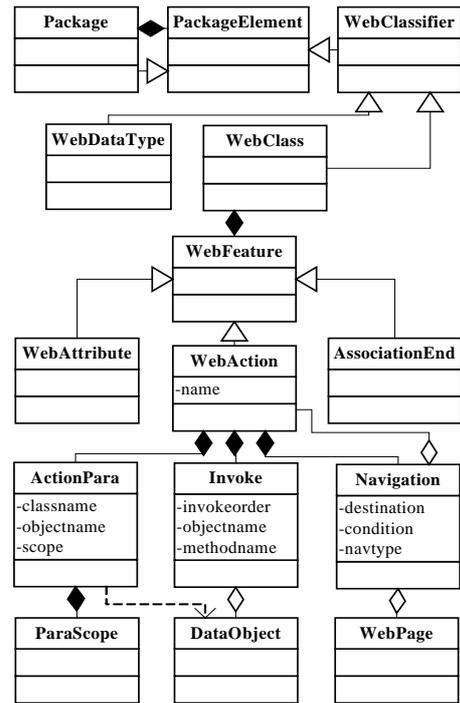


图3 DAU-PSM 行为构件元模型定义

DAU-PSM 界面展示构件的元模型定义如图4所示。

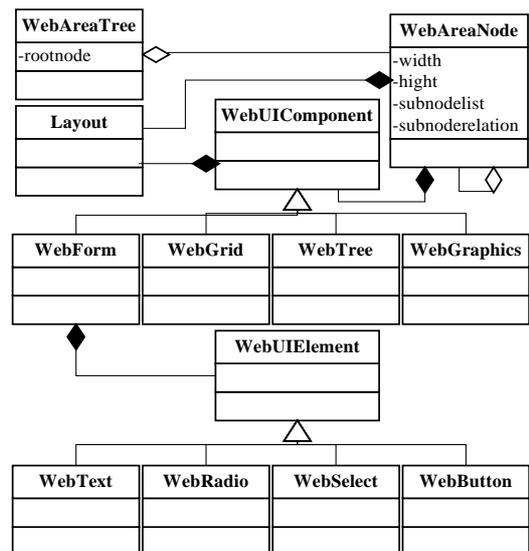


图4 DAU-PSM 界面展示构件元模型定义

一个展示页面表示为一棵展示树(WebAreaTree),它一般包含多个展示区域(WebAreaNode),并含有布局策略

(Layout), 用户可以在区域中嵌套划分子区域。每个展示区域又包含若干界面展示部件(WebUIComponent), 如表单、数据表格、行为触发点等。界面展示部件还指定了其与数据元素及行为触发点的绑定关系, 它是由不同的界面控件元素(WebUIElement)组成的。

2 支持多种目标框架的代码生成器

2.1 基于 MDA 的多目标框架开发模型

MDA中的平台有关模型是一个相对性的概念^[3], 当前多数Web应用开发方法将平台定义为具体的目标运行框架, 这使得基于某种平台的业务逻辑设计与目标运行框架产生过于紧密的耦合, 阻碍了遗留系统移植和演化过程中设计模型的重用。

本文将 WebApp 运行框架添加到开发模型中, 并且引入抽象平台有关模型(abstract PSM, aPSM)和具体平台有关模型(concrete PSM, cPSM)的概念, 从而扩展了 MDA 开发模型, 如图 5 所示。目标应用系统的需求可以分成功能需求和非功能需求, 功能需求反应在业务逻辑视图中, 而非功能需求反应在非功能需求。采用不同的视图分别对系统进行描述不仅可以提高模型的清晰性和灵活性, 还便于模型的重用。

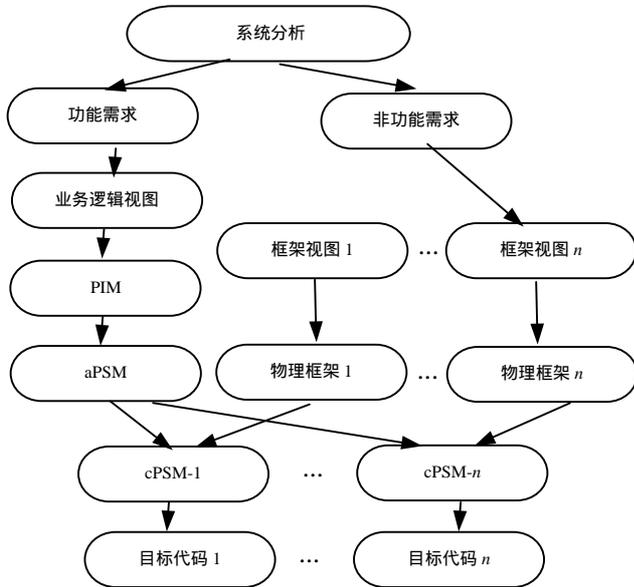


图 5 基于 MDA 的多目标框架开发模型

业务逻辑视图是对应用系统的业务域的描述, 根据它可以建立 PIM, 然后将 PIM 中的元素信息映射到 Web 应用的平台概念域中, 从而得到基于 DAU-PSM 的 aPSM。aPSM 既独立于各种具体的运行框架, 又可方便地转换为这些框架下的目标代码。

框架视图描述的是应用系统的逻辑组织结构, 如视图中的数据获取应采用的模式(Dispatcher View 模式或服务到 Worker 模式^[4])。基于框架视图可以选择出最合适的运行框架, 然后根据映射规则, 使用运行框架中的概念和术语对 aPSM 进行描述, 从而进一步得到工程的 cPSM。cPSM 的概念域是与具体的 WebApp 运行框架密切相关的。

代码生成器依据 cPSM 模型生成 Web 应用系统的目标代码, 这些代码的结构符合运行框架的规范和要求。

2.2 代码生成器框架

本文介绍的代码生成器是建立在 MOF 模型、WebApp 开发方法、J2EE 设计模式和多种运行框架的基础上的, 其结构

如图 6 所示。

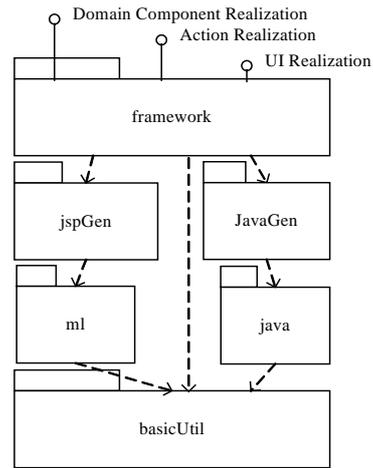


图 6 代码生成器框架

框架的最底层是 basicUtil 名空间(namespace), 包含了用于代码生成的基础支持类和方法。如代码输出的修饰部件, 提供了按名查询和换行缩进等功能, 以保证书写风格与逻辑设计分离。

基础层上面是元模型层, 包括 java 名空间和 ml 名空间。关于 Java 语言中的类(class)、属性(attribute)、方法(method)以及包(package)等概念的元模型都包含在 java 名空间中。这里的 Java 元模型设施参考了 JMI(Java metadata interface)规范^[5], 并进行了一定的简化, 同时还提供了一定的约束机制来保证模型的语法正确性, 例如类中属性和方法不能重复定义, 不能使用未定义的类型, 不能违背可见范围原则等。

J2EE WebApp 中的视图和配置文件是用标记语言(markup language)描述的, 需要标记语言元模型支持。这个元模型包含在 ml 名空间中, 包括节点(node)、元素(element)、文档(document)等概念的描述。J2EE WebApp 视图实现技术很多, 如 JSP、XSLT、Velocity 等, 它们的基础都是标记语言。目前, 本代码生成器仅支持 JSP 作为视图实现方法, 但是标记语言元模型也为其他实现方式的扩充提供了支持。

JspGen 名空间以 ml 名空间为基础, 主要根据 DAU-PSM 的界面展示构件模型生成 WebApp 视图的 JSP 文件, 包含 JSP 语法元素、界面的布局策略和界面组件生成模块。JavaGen 名空间依据领域构件模型和行为构件模型的信息生成相应的 Java 类文件。

处于整个生成器最上层的 framework 名空间负责根据用户选择的运行框架组织相应的代码和文档。framework 名空间中包含运行框架(Struts、Spring、JSF 等)的描述以及各种生成策略组件, 如命令对象的构建、映射关系的描述等。

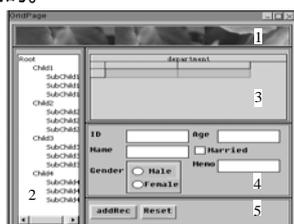
代码生成器提供了建立领域构件类、行为交互类和展示页面文件的 3 个接口, 通过这些接口传递来自于 DAU-PSM 模型的信息和数据, 以生成最终的目标代码。

2.3 代码生成的实现

J2EE Web 应用的组织结构包括 Java 类文件、动态页面文件、配置文件以及支持类库等。Java 类的构造包括创建业务对象类和创建数据访问对象类。这其中还包含正确性检验, 如类 A 要引用类 B, 但创建类 A 时类 B 还没有创建, 则产生类 B 未定义的异常。

生成器根据行为构件模型生成命令对象初始化模块、业

务逻辑执行模块、导航逻辑处理模块并填充控制器中的处理方法，还要在配置文件中填写映射关系。这些模块的形式是由运行框架决定的。



(a) 一个界面展示视图



(b) 运行实例-添加学生记录页面

图 7 展示界面的布局结构

由于大多数框架在配置文件中描述导航和映射关系，页面之间的耦合度很低，这样展示页面的生成便具有很高的独立性，依据界面展示构件模型中每一个 WebAreaTree 的信息就可以生成相应的动态页面文件。这主要包括页面布局和视图构件生成两方面的工作。DAU-PSM 界面展示构件模型中的信息采用树状存储结构，对应的页面区域划分算法就是树的广度优先遍历，如果遍历到的节点是叶节点，则根据构件类型创建相应的视图构件；如果遍历到的是非叶节点，则需

(上接第 71 页)

3 有关 IPv6 的考虑

目前对 IPv6 地址的路由查找算法都是基于 128 位的 IPv6 地址进行考虑的。事实上，IPv6 地址长度为 128 位并不意味着各级路由器都要考虑 128 位。128 位的 IPv6 地址被区分为单播 IPv6 地址、多播 IPv6 地址和泛播 IPv6 地址 3 种类型^[4,5]。除泛播 IPv6 地址目前还没有明确的定义外^[6]，其它两类均可通过最高位的几个比特来区分。就单播 IPv6 地址来说，以 6Bone、6Net 等几个 IPv6 骨干实验网的数据(尽管总的数据量还很少)来看，其前缀长度主要介于 19~64，不存在长度小于 16 和长度大于 64 的前缀，这实际上是 RFC 2373 所规定的。可见，核心路由器关心的是目标地址的高 64 位。据此，可结合本文方案设计出查找深度不大于 6 的二分查找方案，或者结合 DIR-24-8 结构^[7]设计查找深度仅为 3 的 DIR-24-24-16 的查找方案等。而将接口部分的 64 位交给子网路由器处理，这些路由器的负担往往比较轻，可根据子网的具体情况设计算法。

由于历史的原因，目前还存在像兼容 IPv4 地址、IPv4 映射地址、6over4 地址、6to4 地址等这样一些兼容地址，这些地址将来必定会被真正的 IPv6 地址所取代。笔者认为目前可以考虑将已延用多年的 IPv4 地址中的前缀部分划归到 IPv6 中的公共拓扑和站点拓扑中去，而将其余部分划归到接口标识中，不仅有利于各级路由器分工处理数据报的转发，也有利于路由器的设计和路由查找算法的设计。

对节点区域进一步划分，划分方式有垂直划分和水平划分两种。如图 7(a)所示的界面展示视图划分为 5 个区域，对应生成的 JSF 框架下的运行页面如图 7(b)所示。

目标代码的生成使用基于模板的方法(template-based approach)实现，这种生成方法在 AndroMDA、OptimalJ、ArcStyler 等 MDA 工具中广泛使用，在此不再赘述。

3 结束语

模型驱动的软件开发是近年来软件工程界研究的重点，本文介绍了一种基于 J2EE 平台的 PSM 模型描述方法和相应的代码生成方案。该方法支持多种目标框架下的代码生成，能够对模型驱动的软件开发提供有力的支持。今后的工作包括：进一步完善行为构件模型的形式化描述，增强其语义表述能力；对界面展示构件模型进行全面的抽取和描述，以加强所生成页面的视觉吸引力。

参考文献

- 1 Miller J, Mukerji J. MDA Guide Version 1.0.1(Document number omg/2003-06-01)[EB/OL]. (2003-09). <http://www.omg.com/mda>.
- 2 Seshadri G. Understanding JavaServer Pages Model 2 Architecture [EB/OL]. (2005). <http://www.javaworld.com/javaworld/jw21221999/jw2122ssj2jspmvc.html>.
- 3 Kleppe A, Warmer J, Bast W. MDA Explained, The Model Driven Architecture: Practice and Promise[M]. Addison-Wesley, 2003.
- 4 Martin F. Patterns of Enterprise Application Architecture[M]. Addison-Wesley, 2003.
- 5 Java Community Process. Java Metadata Interface (JMI) Specification Version 1.0, JSR 040[EB/OL]. (2002-09). <http://jcp.org/jsr/detail/40.jsp>.

参考文献

- 1 Waldvogel M, Varghese G, Turner J, et al. Scalable High Speed IP Routing Lookups[J]. Computer Communication Review, 1997, 7(4): 25-36.
- 2 Broder A, Mitzenmacher M. Using Multiple Hash Functions to Improve IP Lookups[C]//Proceedings of the IEEE INFOCOM'01, San Francisco. 2001: 1454-1463.
- 3 Venkatesh K, Aravind S, Ganapath R, et al. A High Performance, Parallel IP Lookup Technique Using Distributed Memory Organization[C]//Proc. of International Conference on Information Technology: Coding and Computing. 2004.
- 4 Hinden R. Deering IP Addressing Architecture[S]. RFC 2373, 1998.
- 5 Davies J. Understanding IPv6[M]. Washington, USA: Microsoft Press, 2003.
- 6 Satoshi D, Shingo A, Hiroshi K, et al. Design, Implementation and Evaluation of Routing Protocols for Ipv6 Anycast Communication[C]//Proceedings of the 19th International Conference on Advanced Information Networking and Applications. 2005: 1-6.
- 7 Gupta P, Lin S, McKeown N. Routing Lookups in Hardware at Memory Access Speeds[C]//Proc. of IEEE INFOCOM'98, San Francisco. 1998-04: 1240-1247.