第 27 卷 第 4 期
2012 年 8 月

天津科技大学学报
Journal of Tianjin University of Science & Technology

Vol. 27 No. 4
Aug. 2012

# 基于元模型的视图层 PIM 建模方法研究与实现

曹伯翰，李亚芬
（北京工业大学电子信息与控制工程学院，北京 100124）

**摘 要**：模型驱动架构（model driven architecture，MDA）的核心思想是将建模语言当作开发语言使用，基于 UML 的平台无关模型（platform independent model，PIM）建模方法主要关注于系统的体系结构与业务逻辑设计，对视图层支持较差. 针对此问题，研究了基于元模型的视图层 PIM 建模方法，给出视图层 PIM 模型中界面逻辑模型和界面布局模型的表示法及语义内容；基于 EMOF（essential meta object facility）对 UML 建模语言进行扩展，使其对上述两种模型提供语义上和表示法的支持；最后，基于 Eclipse GMF 框架开发了视图层 PIM 辅助设计工具 GMTP.

**关键词**：模型驱动架构；平台无关模型；元对象设施；元模型；可视化模型
**中图分类号**：TP311 **文献标志码**：A **文章编号**：1672-6510（2012）04-0069-05

# Research and Realization of Meta-model Based Modeling Method of View Layer PIM

CAO Bohan，LI Yafen
(College of Electronic Information and Control Engineering，Beijing University of Technology，Beijing 100124，China)

**Abstract**：The core idea of MDA（model driven architecture）is to use the modeling language as a development language. UML-based PIM（platform independent model）modeling method focuses on system architecture and business logic design. In order to solve the problem in describing view layer model in PIM，a Meta-Model based modeling method of view layer in PIM was researched into. The representation and semantic contents of the interface logic model and interface layout model in view layer PIM were given. The UML，based on EMOF（essential meta object facility），was extended so as to provide semantic support and representation to the above model. Finally，based on Eclipse GMF Framework，GMTP，the view layer PIM designing tool，was developed.

**Key words**：MDA；PIM model；MOF；meta-model；graphic model

MDA 的核心思想是将建模语言作为编程语言使用，建立模型以抽象表达系统设计结构与业务逻辑需求[1]；具有较高抽象层次的模型经过符合QVT标准的模型转换机制，向较低抽象层次的模型转换，并最终由代码生成机制[2]得到目标应用代码. 基于 MDA 的软件开发过程周期更短[3]，模型驱动的系统架构能够降低系统变更与扩展的成本[4]，被誉为未来十年内最重要的软件方法学. 在 MDA 中，系统可表示为应用逻辑的平台无关模型（platform independent model，PIM）. PIM 是 MDA 具有较高抽象层次，独立于平台及实现技术的设计模型，也是系统的业务逻辑模型，对系统的结构和功能进行完整的描述. 当前 MDA 中的 PIM 建模方法主要关注于软件体系结构与业务逻辑设计，对视图层建模支持较差，无法满足界面逻辑设计和界面布局的个性化定制的需求. 国内外很多学者在支持 MDA 的界面设计、建模方法相关研究领域进行了深入的研究，如：侯金奎等[5]提出 ASLP 模型，何啸等[6]通过分析总结UML 模式和表示法，提出一种基于表示法定义的元模型 NDM，Paterno 等[7]提出基于图形符号的 ConcurTaskTrees（CTT）任务模型表示法，并在此基础上设计实现了用于支持 MDA 界面建模的 CTTE/IERESA 系统等.

本文给出一种视图层 PIM 层次建模方法，通过建立界面逻辑模型及界面布局模型，对 Web 系统界

**收稿日期**：2012–02–16；**修回日期**：2012–03–23
**作者简介**：曹伯翰（1985—），男，北京人，硕士研究生，bestciel_12@hotmail.com.

面逻辑、界面展示内容和界面布局样式进行分离设计、建模.

## 1　建模方法

### 1.1　视图层 PIM 模型设计思路

在 MDA 中应用 MVC(model view controller)模式,其视图层 PIM 建模达到的预期目标是:通过建立层次模型,细化鲁棒分析得到的边界对象,对 Web 系统视图层界面逻辑和界面布局样式分别建立平台无关的模型,以描述视图层对象的设计结构和动态行为,并通过可视化建模方式,实现 Web 界面布局样式的个性化定制.

由此,给出视图层 PIM 建模过程如下:

(1)首现通过 Iconix 过程实现分析模型向设计模型的映射,建立鲁棒分析图,并在这个过程中分离出用例中的边界对象;

(2)根据鲁棒分析结果和用例规约中的描述,建立界面逻辑视图,描述 Web 界面逻辑,及与控制层的交互行为;

(3)建立界面布局视图,对 Web 界面布局样式进行个性化定制,并在界面布局视图中添加展示单元、框架单元、数据单元及控制单元,并与界面逻辑视图中的设计对象进行绑定.

### 1.2　基于 Iconix 实现分析模型与设计模型间的映射

在 MDA 中如何将分析模型映射为设计模型,没有一个统一的操作规范,Iconix 方法在这方面得到了很多成功的经验,它基于鲁棒分析方法对用例视图进行进一步细化、分离,建立起鲁棒模型,并根据鲁棒模型中 3 种对象的特点寻找与领域对象间的关系来对分析模型进行精化,以得到 PIM 设计模型.

采用 Iconix 方法中鲁棒分析方法分析用例视图,并结合 MVC 模式实现“分析对象到鲁棒分析对象,最终映射到 MVC 相应模型层的过程”,如图 1 所示.

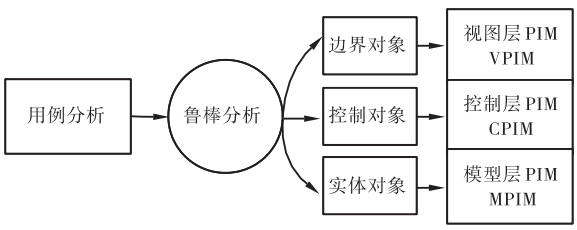


**图 1　分析对象与设计对象映射过程**

**Fig. 1　Mapping of the analysed and designed objects**

### 1.3　视图层界面逻辑模型

界面逻辑模型设计为面向 Web 界面对象的逻辑建模视图,其在语义上可以定义为一个四元组:

{uiObjects, uiRelations, uiAttributies, uiOperations}

uiObjects 为界面对象,描述了对象名称 name、唯一的 ID 序号及唯一性约束关系,对象的序号实现它在包内的索引;对象名在代码生成时,直接映射到界面对象属性 name 上. 对 Web 系统来说,由于 Html 与实现业务逻辑所依赖的技术实现平台无关,因此视图层 PIM 中界面对象 name、ID 等信息可以直接映射到 Html 中.

uiAttributies 为对象属性,对 Web 系统中视图层对象主要有展示、交互两个属性子集. 展示属性描述了该对象在界面上展示样式所依赖的样式列表;交互属性描述该对象在参与交互时所包含的属性.

uiOperations 为对象行为,根据 Web 系统中界面交互的特点,对象行为包含{主动行为、被动行为}两个子集. 主动行为对界面视图展示、动态效果所依赖的系统状态、对象属性进行建模;被动行为是由用户交互驱动的,如数据表单对象的提交操作 uiOperations:submit(),就是由用户在界面上操作参与人机交互完成的,表单对象在执行 uiOperations:submit()操作后会执行 setAttributies()操作,将表单内全部可见对象提交到 uiObjects:action 对象中,通过 uiObjects:action 对象向控制层传递;控制层在完成业务逻辑处理后,将处理结果返回到上一视图中,数据表单将执行 uiOperations:dispay()操作,通过判断该操作的先验条件,对处理结果数据进行展示.

uiRelations 为对象关联,表示对象间的各种引用关系,由(引用名称,引用类型、引用约束、引用源点、引用目标、转换关系)组成;在视图层界面逻辑模型中主要包含依赖、继承、关联、导航几种引用类型;uiRelations 继承自 UML 元-元模型 EMOF 中 EReference 对象,如图2所示,其中EStructural Feature 为 EReference 提供了建模所需的约束条件及操作.

### 1.4　视图层界面布局模型

采用界面布局视图建模可看作是将 Web 界面中布局样式与界面内容进行分离设计. 界面布局视图建模的任务是描述 Web 系统界面展示单元的布局结构,对 Web 界面个性化定制提供支持.

界面布局模型定义为一个四元组:

{Pc, Fc, Dc, Cc}

其中,Pc(present unit container)为界面展示单元容器,描述了界面展示单元 Pu(present unit)的相关信

息；Fc（frame unit container）对 Web 界面中框架布局进行建模，以实现 Web 界面布局的各式基本结构，如图 3 所示；Dc（data unit container）为数据单元容器，用以对 Web 界面展示具体内容进行描述，除包含静态数据单元（文本、图像、多媒体）内容外，还包含动态数据内容，Dc 中数据单元（data unit）可看作是实体对象与边界对象的接口，通过数据单元可实现界面逻辑模型中界面对象 display（）操作；界面布局模型的四元组 Cc 是 Web 界面中控制对象容器，由于界面数据内容（表单、链接等）、展示单元 Pu 的可见性和样式往往依赖于系统交互、系统状态等条件，因此在对界面布局进行建模时，仍要考虑到与控制层的交互过程进行描述.

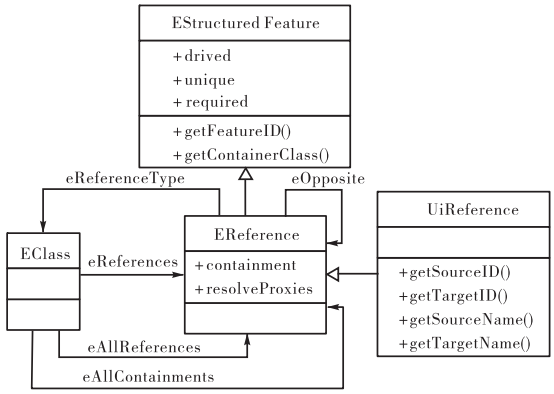


**图 2　扩展 UML 元–元模型中 EReference**

**Fig. 2　Ereference in extended UML meta-meta model**

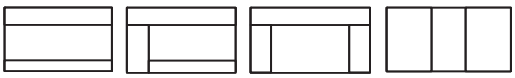

**图 3　界面容器布局模式**

**Fig. 3　Layout form of interface container**

## 2　视图层 PIM 模型的元模型实现

将界面逻辑模型与界面布局模型作为视图层 PIM 引入到 MDA 框架下，需要解决两个核心问题，即：建立两种模型的语义对应的元模型[8]及支持相应的表示法的可视化建模视图. 本文通过扩展 EMOF 元–元模型来构造界面逻辑模型和界面布局模型的元模型.

MOF（meta object facility）将建模语言划分为4 个层次，根据UML 自描述性特征，其中每一层都可看作是上一层模型的实例[9]. MOF 的核心规范主要是通过复用 UML2.0 infrastructure 的Core 包，并引入一组元语言构造机制（如Identifiers、Extension 及 Reflection）而形成的. EMOF（essential meta object facility）是 MOF 的一个轻量核心子集，提供了完整的 MOF 元–元模型概念和扩展机制，其简化结构如图 4 所示.

基于 EMOF 元–元模型进行扩展，得到界面逻辑视图与界面布局视图的元模型. 界面逻辑模型的{uiObjects，uiRelations，uiAttributies，uiOperations}四元组分别扩展自 EMOF 元–元模型中 EClass、EReference、EAttribute、EOperation 对象，在其基础上根据界面逻辑模型的语义重新进行定义. 界面布局模型的元模型结构如图 5 所示，其｛Pc，Fc，Dc，Cc｝4 种容器对象均基于EMOF 元–元模型中 EClass 对象扩展，后面以不同的表达法元模型予以区分. 图 6 展示的是界面布局模型中展示单元的元模型.

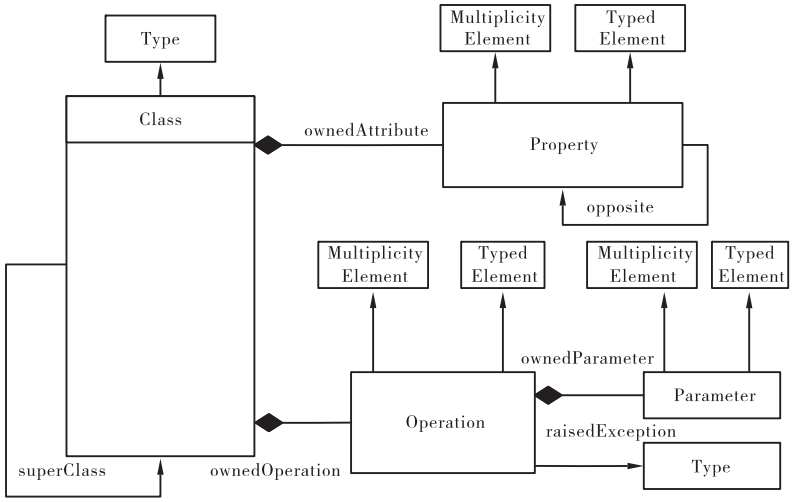


**图 4　简化 EMOF 元–元模型**

**Fig. 4　Simplified EMOF meta-meta model**

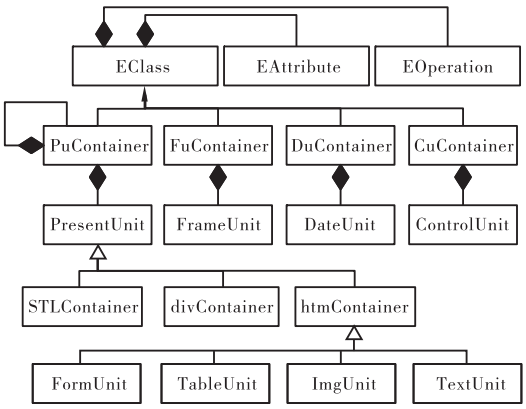


**图 5　界面布局视图的元模型**

**Fig. 5　Meta-model of interface layout view**

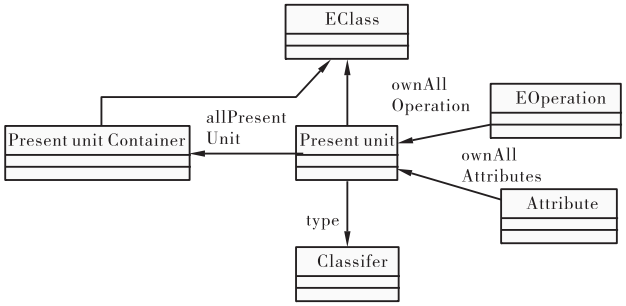


**图 6　实现展示单元及容器的简化元模型**

**Fig. 6　Simplified meta-model of display unit and container**

在建立了界面逻辑模型与界面布局模型的元模型后，要建立他们的表示法元模型，并建立两者的映射关系. 对于界面布局模型，界面的表示通常是一个矩形区间，称为展示区间. 为适应个性化界面布局需求，开发时通常要将展示区间划分为多个区域，称为展示单元 Pu，每个区域实现不同的展示、交互任务，各区域之间存在一定的交互关系. 为实现图 3 所示的各类界面布局模式，采用 UML 容器模式，该模式是一种图元间嵌套组合的方式. 通过建立 UML2.0 中提供的基本图元符号与界面展示元模型间映射，就可以对界面布局样式进行建模.

## 3　视图层 PIM 建模过程

### 3.1　界面逻辑模型的建立

在 Web 界面中的一个表单对象由＜Tag，Attribute，Event＞这 3 部分信息进行描述，该对象映射到界面逻辑模型中，能与 uiObjects，uiAttributies，uiOperations 一一对应，而界面描述逻辑模型中，uiRelations 为对象间的引用关系进行建模，表示为 dependence、navigation、association 等引用关系. 其中，navigation 定义为一个视图层边界对象由数据提交驱动的导航关系. 在元模型层，本文定义了 navigation 的规则：navigation：source，导航关系的根节点必须是数据提交的操作对象 uiObjects：ActionObject；navigation：Target，导航关系的目标节点端是一个独立的视图对象. 通过在建模工具上实例化 navigation 的元模型，就可以通过 uiRelations：navigation 实现“数据提交操作后导航到另一视图对象”这一界面逻辑的语义描述.

图7是一个分步填写考生调剂信息的界面逻辑.

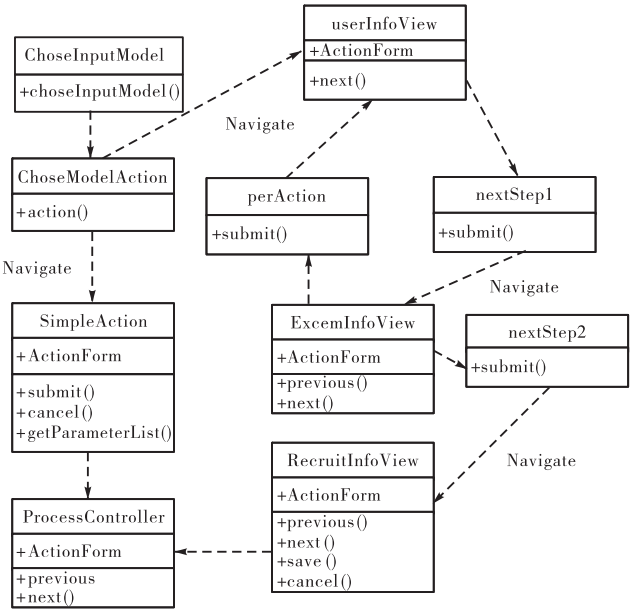


**图 7　申请调剂用例的界面逻辑模型**

**Fig. 7　Interface logical model of the example**

图中基于表单提交的数据操作是视图层与控制层协作的主要部分，也是业务逻辑得以实现的核心，界面中表单对象＜Form method = “get” Action = “DataInputService” Class = “DataInputForm”＞在执行 Submit()操作后，会以 get 方式向 DataInputService 控制对象发送 request 请求，传递表单内的数据，执行 DataInputService 中的方法，实现业务逻辑的转发、处理、页面跳转等功能. 因此，表单对象 uiObjects：Action 可视为 Web 系统视图层与控制层模型间的接口. 作为一个独立的设计对象，一方面声明了将要在控制层中实现的属性和方法；另一方面通过 uiRelations：navigation 引用实现了控制器处理后页面跳转这一过程的可视化描述. 通过这样的设计，就可将视图层界面逻辑建模与控制层建模分离开，开发人员只需关心界面逻辑模型中视图层的用户交互、数据处理逻辑与视图界面导航等关系，而不需考虑其在模

型层、控制层的具体实现. 通过 Iconix 过程，本文直接给出了其设计对象的形式.

### 3.2　界面布局模型的建立

拟建立的 Web 界面由页面框架单元 Fu 划分为左侧导航栏和右侧填报表格两个区域. 在框架单元 Fu 中设置展示单元 Pu，用于描述界面布局结构. 在代码生成时，Pu 直接映射为 div 单元，其属性映射为响应的样式列表 css. 在完成 Web 界面样式个性化定之后，通过在 Pu 中设置数据单元 Du 和控制单元 Cu，对界面展示内容进行建模. 如考生信息填报表单对应一个数据单元 Du，该单元提供了表单与其对应的实体对象信息，并与之进行绑定. 目标界面效果如图 8 所示. GMTP 中目标界面的界面布局模型如图 9 所示.


**图 8　目标界面效果**

**Fig. 8　Target interface effect**

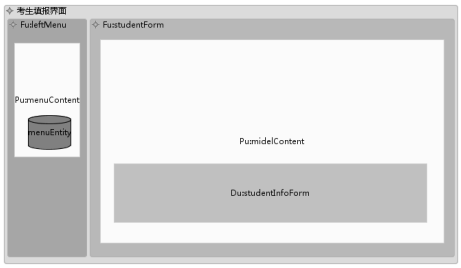


**图 9　GMTP 中目标界面的界面布局模型**

**Fig. 9　Interface layout model of the target interface in GMTP**

## 4　视图层 PIM 建模支持工具 GMTP 的实现

GMTP（graphic modeling tool for PIM）是本文基于 Eclipse GMF（Eclipse graphical modeling framework）框架开发的建模工具，用以支持视图层 PIM 的可视化建模. 它以 EMOF 作为自己的元–元模型，提供一套图形化编辑器以支持用户建立不同的建模视图. GMF 是 Eclipse 平台下的开源框架，以 Draw2D 为其提供图形表现功能.

GMTP 作为支持 MDA 的辅助设计工具，主要用于对系统进行分析设计，建立 PIM 模型，并将模型以序列化形式输出. 它支持的模型包含分析模型、设计模型两部分. GMTP 中实现不同视图的建模视图，可看作是对不同建模视图元模型实例化的过程. GMTP 运行时界面如图 10 所示，包含画布、工具栏、属性编辑栏、大纲索引栏 4 个区域.

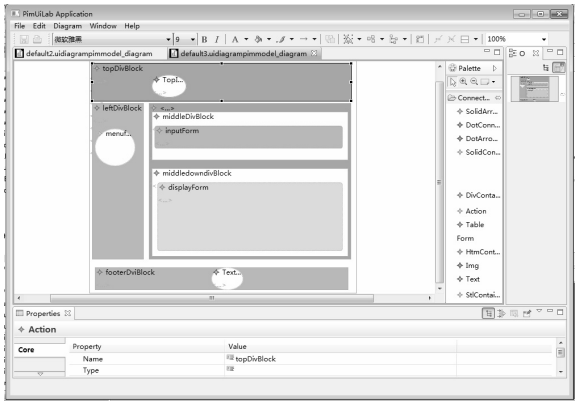


**图 10　GMTP 运行时界面**

**Fig. 10　Operation interface of GMTP**

开发人员可在画布区域可视化地进行建模工作（建立、拖动、修改模型）；工具栏区域提供了视图支持的建模元素列表，供开发人员调用；属性编辑栏用来编辑、修改画布区域中选中的建模元素属性，在未选中的情况下，默认选中画布图形元模型中的根元素，展示画布基础信息；大纲索引栏对画布区域提供对象到导航支持，并能够列出模型元素的层次关系. 基于 GMF 框架开发的视图层 PIM 建模工具为一个可独立运行的 Eclipse 程序，在使用过程中无需运行 Eclipse 即可使用.

## 5　结　语

为了更好解决视图层 PIM 建模问题，本文提出以界面逻辑模型和布局模型对视图层 PIM 进行描述的方法，利用元建模技术实现上述两种视图的元模型和表示法元模型，并在此基础上基于 GMF 框架开发了可视化建模工具 GMTP；下一步研究工作将精化视图层 PIM 模型的描述方法，改进 GMTP 中模型表示法的展示能力，以进一步提高工具的实用性.

（下转第 78 页）