

基于软件模式的 PIM 到 PSM 的模型变换

刘奎¹, 宋森², 陈一飞¹, 赵晓静¹

(1. 安庆师范学院 计算机系, 安徽 安庆 246011;

2. 复旦大学 信息与工程系, 上海 200433)

摘要:模型驱动体系结构(MDA)是一种以模型为中心的新的软件开发模式。MDA的基本思想是将模型主要分为平台无关模型(PIM)和平台相关模型(PSM),然后通过变换规则实现 PIM 到 PSM 的变换。文中将软件模式的概念引入到 PIM 到 PSM 模型变换中,从而提高模型变换效率和降低模型变换出错率。同时,设计了基于软件模式的 PIM 到 PSM 的模型变换方法的框架。

关键词:模型驱动体系结构;软件模式;模型变换

中图分类号:TP311.52

文献标识码:A

文章编号:1673-629X(2006)10-0074-03

Model Transformation from PIM to PSM Using Software Patterns

LIU Kui¹, SONG Miao², CHEN Yi-fei¹, ZHAO Xiao-jing¹

(1. Dept. of Computer, Anqing Teachers College, Anqing 246011, China;

2. Dept. of Information and Engineering, Fudan University, Shanghai 200433, China)

Abstract: Model driven architecture(MDA) is a new center - model software development pattern. The basic thought of MDA is that models are divided into platform independent model(PIM) and platform specific model(PSM), and that PIM is transformed into PSM using transformation rules. The paper introduces software patterns to the transformation from PIM to PSM, which can improve the efficiency of model transformation and reduce errors of model transformation. At the same time, the paper gives a frame of a method of transformation from PIM to PSM using software patterns.

Key words: model driven architecture; software pattern; model transformation

0 引言

早在 1990 年,OMG 就开始致力于解决企业间应用集成的问题,并制定了跨平台的应用集成和互操作的标准——CORBA。CORBA 允许对象在异构的平台上进行交互。然而 CORBA 并不是惟一的中间件平台,像 .NET, J2EE 都提供了类似的服务,所以目前的中间件技术并不足以解决企业间跨平台计算的要求。为此,OMG 在 2002 年提出了模型驱动体系结构(MDA, Model Driven Architecture)^[1]。

1 MDA 概述

MDA 是一种以模型为中心的新的软件开发模式。它是一种基于 UML 以及其他工业标准的框架,支持软件设计和模型的可视化、存储和交换。MDA 把建模语言用作一种编程语言而不仅仅是设计语言。

同时,MDA 以一种全新的方式将 IT 技术的一系列新的趋势性技术整合到一起,这些技术包括基

于组件的软件开发、设计模式、中间件、说明性规约、抽象、多层系统、企业应用集成,以及契约式设计等内容^[2]。MDA 的出现,为提高软件开发效率,增强软件的可移植性、协同工作能力和可维护行,以及文档编制的便利性指明了解决之道。

MDA 的核心思想是抽象出与实现技术无关、完整描述业务功能的平台无关模型(PIM),针对不同实现技术制定多个映射规则,然后通过这些映射规则及辅助工具将 PIM 变换成与具体实现技术相关的平台相关模型(PSM),最后,在一定程度上将 PSM 自动转换成代码。所以 MDA 的基本开发过程可以分为 3 个阶段:PIM 的获得,PSM 的生成,代码的产生,如图 1 所示。

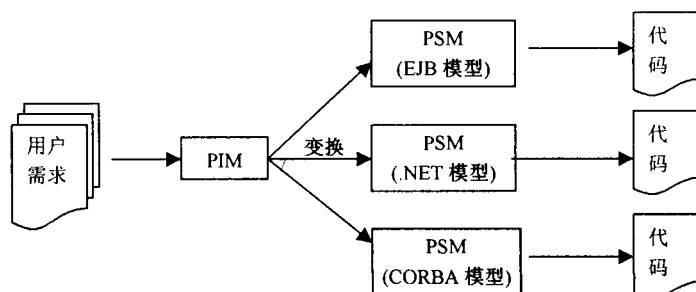


图 1 MDA 的开发过程

收稿日期:2005-12-27

作者简介:刘奎(1975-),男,安徽安庆人,讲师,研究方向为软件工程、软件开发方法。

现在 OMG 推荐应用 UML 来描述 PIM,并且开发者可以借助一些已有的工具形象化地描述模型,实现从用户需求中提炼出业务模型。从 PSM 生成可执行代码也比较容易实现,目前已经有一些这样的代码生成工具,如 Together, Rational Rose。但是从 PIM 到 PSM 的模型变换,目前还没有很好的方法和工具。所以文中研究基于软件模式的 PIM 到 PSM 的模型变换方法,是完全有价值的,可以提高模型变换效率和降低模型变换出错率,提高模型变换的质量。

2 软件模式

模式的概念最初是由建筑学家 Christopher Alexander 提出的^[3],即“每一种模式描述了一个在我们周围不断重复发生的问题,以及问题解决方案的核心。这样,我们就可以一次一次地重复使用该方案而不必做重复劳动。”后来,这种思想被引用到开发面向对象软件过程中来,成为当前软件工程研究领域的一大热点。

软件模式是用于捕捉一些成功的软件开发经验。这些开发经验可以用文档和模板的形式记录下来,在以后的开发中可以复用。软件模式的出现是面向对象技术的发展,是向知识与推理的回归,是对开发者的分析与设计知识的记录、提取和表示。

目前尽管存在多种描述方法,分别为:非形式化的文本描述方法,如自然语言;图形化的描述方法,如 UML;形式化的文本描述方法,如特定的形式化语言。但是这些描述方法都存在一个共同的缺陷:不能完整地、精确地与形式化地描述软件模式结构与行为方面的信息及其语义。

OMG 文档中描述的动作语义(Action Semantic)是对 UML 的扩展^[4],改善了 UML 在语义描述上的缺陷。动作语义为元模型提供了一个动作语言,能应用动作语义编写可直接执行的 UML 模型,这样就可以用动作语义来精确描述软件模式。

但是 OMG 文档中没有给出动作语义的具体语法,因此用户无法用标准化的方式编写该语言的任一语句。文中借助对象约束语言(OCL, Object Constraint Language)作为动作语义的动作说明语言。该动作说明语言由两部分组成:前置条件(pre),用于描述软件模式的结构信息;动作(action),用于描述软件模式的行为信息。

下面以 Proxy 模式为例,说明如何使用动作语义的动作说明语言来描述软件模式。

```
context Class::addProxy()
pre:
    let classname = self. package. allClasses -> collect (each: Class |
each. name in
    (classnames = excludes(self. name + 'Proxy') and
    classnames = excludes('concrete' + self. name))
actions:
    let name = self. name
    let self. name: = name. concat('Proxy')
```

```
let super = self. package. addClass(name, self. allSuperTypes(), {}
->
includes(self))
let concrete = self. package. addClass('Concrete'. concat(name),
{}->
includes(super), {}))
let Association = self. addAssociationTo('ConcreteSubject', concrete)
self. Operations -> forAll(op: Operation | op. moveTo(concrete))
```

3 基于软件模式的 PIM 到 PSM 的模型变换

基于 MDA 的软件开发方法最重要的活动是模型变换,而最重要的模型变换是如何把平台无关模型变换到平台相关模型,这关系到 MDA 的成败。文中重点研究了 PIM 到 PSM 的模型变换^[5,6],并提出了基于软件模式的 PIM 到 PSM 的模型变换。该方法与其他的方法相比,具有更高的模型变换效率和更低的模型变换出错率。

3.1 方法概述

从一个源模型(PIM)生成目标模型(PSM),必须把 PIM 中的每个元素同 PSM 中的一个或者多个元素建立联系。这样做的直接方式是把 PIM 的模型元素的元类(源元模型)同 PSM 的模型元素的元类(目标元模型)建立联系,这种联系就是变换规约^[2]。因为 UML 模型中的所有元素,如类、关联与属性都是用 UML 元模型定义的,模型元素与元模型之间有实例-类型关系,元模型在模型中每个实例都必须遵从对元模型设置的规则。也就是说,在源元模型与目标元模型之间建立通用的变换规约,在实际的模型变换中,首先匹配源模型中的元模型与变换规约中的源元模型,再生成符合变换规约中的目标元模型的目标模型。

变换规约由变换规则组成^[7],如图 2 所示。一个变换规则是个原子操作,它描述了一个变换步骤。一个变换规则由变换约束与变换操作(action)两个基本部分组成。变换约束是由 OCL 表达式组成,变换约束的前置条件(pre)决定执行变换操作时应该满足的条件,变换约束的后置条件(post)规定了执行变换操作后应满足的条件。如果输入模型符合 OCL 表达式的前置条件,则执行变换操作,由模式适配器从模式库中查找相应的模式,然后调用该模式实现模型的变换,生成符合目标模型的模型元素。例如在将 PIM 变换到与 EJB 平台相关的模型时,根据 PIM 中各数据项的构造类型,自动调用相应的模式绑定,实施 PIM 到 PSM/EJB 的变换,如对构造类型为《Entity》的实体类,绑定 EntityBean 模式,使类图变换为 Entity Bean。也可以是对某一个属性、方法实施变换,如对构造类型为《UniqueId》的属性(Association)变换为一个 EJB 主键类。但是,如果模式适配器在模式库中没有匹配到相应的模式,这时需要定义该模型变换规则,实现模型的变换,再将定义的模型变换规则封装为一个模式,存储到模式库中,以便以后复用。

3.2 方法构架

图 3 给出了该方法的架构^[7]。从图 3 可以看出,基于模式的 PIM 到 PSM 的模型变换的构架是符合 MOF 架构的元层次的。其中最抽象层次 M3 是 MOF。M2 层是 UML 元模型,M2 层的源元模型规定了描述 PIM 的 UML 的语法和语义。同样,M2 层的目标元模型规定了描述 PSM 的 UML 的语法和语义。M1 层源模型(PIM)与目标模型(PSM)都是 UML 模型。M1 层的模型变换引擎根据 M2 层的变换规约将 PIM 变换到 PSM。

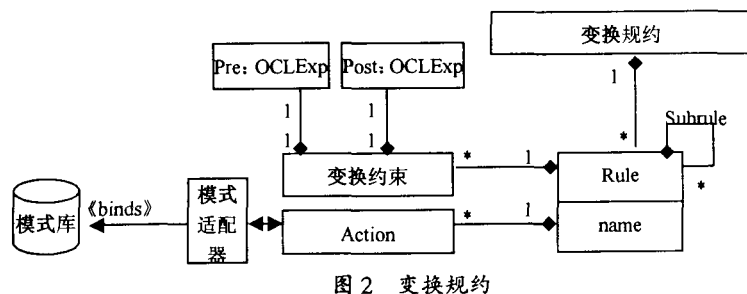


图 2 变换规约

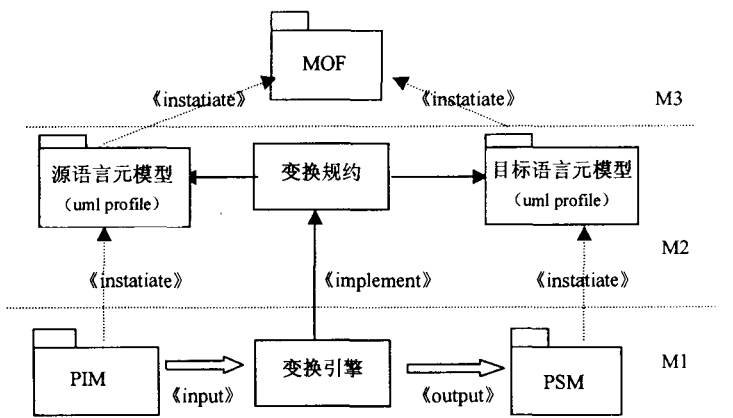


图 3 变换框架

3.3 实例

图 4 显示了一个基于模式的 PIM 到 PSM 的模型变换的例子。在这个例子中,左边的 PIM 中 Client 类与实现类 ImageImp1 直接相关联,这样不方便更改实现类。如果应用 Bridge 模式来优化左边的 PIM 得到右边的 PSM,这样可以在不需更改模型的情况下改变实现方式。注意,变换后的模型其实决定了模型的代码结构,属于平台相关模型。

下面给出应用 Bridge 模式的操作过程。

1) 创建两个 UML 元模型 Class 的实例,一个表示产品的抽象类,如 Image,另一个代表实现抽象类,如 ImageImp;

2) 删除 Client 类与 ImageImp1 类之间关联;

3) 创建一个 UML 元模型 Association 的实例,表示 Image 与 ImageImp 之间的关联;

4) 创建一个 UML 元模型 Generalization 的实例,表示具体实现类 ImageImp1 与抽象实现类 ImageImp 之间的继承关系。

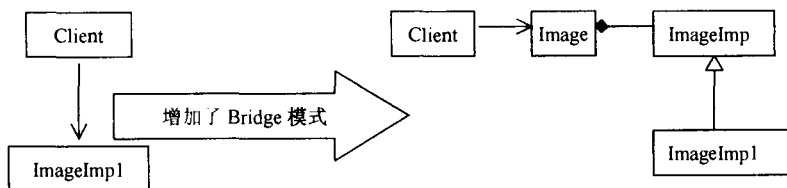


图 4 基于 Bridge 模式的 PIM 到 PSM 的模型变换

4 结论与展望

MDA 代表了一种新的互操作性标准,模型的形式化描述和模型变换始终是 MDA 的核心。然而,要实现 PIM 到 PSM 的模型变换,还有很长的路要走。文中研究的基于软件模式的 PIM 到 PSM 的模型变换只是为研究模型变换提供了一种思路,还有许多技术问题需要进一步研究,如模式适配器是如何选取适当的软件模式,模式库的组织问题等。另外,还应不断抽取新的模式,建立具有互操作性的软件模式库。

参考文献:

- [1] OMG document. MDA Guide Version 1.0[EB/OL]. www.omg.org/mda/,2003-05-01.
- [2] Kleppe A. 解析 MDA[M]. 鲍志云译. 北京:人民邮电出版社,2004.
- [3] Gamma E, Helm R, Johnson R, et al. 设计模式[M]. 李英军等译. 北京:机械工业出版社,2002.
- [4] OMG. Action Semantics for the UML[EB/OL]. http://www.omg.org,2001.
- [5] Sendall S, Kozaczynski W. Model Transformation - the Heart and Soul of Model - Driven Software Development[J]. IEEE Software, Special Issue on Model Driven Software Development,2003,20(5):42-45.
- [6] Caplat G, Sourrouille J. Model Mapping in MDA[A]. In Workshop in Software Model Engineering, Fifth International Conference on the Unified Modeling Language[C]. France: [s. n.],2002.
- [7] 刘奎. 基于模式的 PIM 到 PSM 模型变换方法的研究[D]. 合肥:合肥工业大学,2005.

2006 年起《微机发展》更名为《计算机技术与发展》
欢迎投稿, 欢迎订阅!