基于 EMF 和 OCL 的 MDA 软件工程方法研究*)

常浩浩　覃　征
(清华大学软件学院　北京 100084)

摘　要　随着基于模型的软件工程方法的兴起,模型逐渐地变为软件工程中的重要元素,介绍了 OMG 规范 MOF、XMI 和 OCL 及在 Eclipse 软件工程环境下的相关实现 EMF,说明了统一建模语言 UML 的使用方式和存在的问题,介绍了如何用 EMF 和 OCL 来设计建模语言,比较了基于 EMF 的建模语言和基于 EBNF 的计算机语言之间的区别与联系,给出了结合模板和 OCL 的模型转换方法,通过把 UML 类模型转换到 O/R 映射框架说明了此方法的有效性,分析了 MDA 软件工程方法的重用机制。

关键词　元对象设施,对象约束语言,EMF,模型驱动体系结构,UML 类模型,模型转换,O/R 映射

Research on MDA Software Engineering Based on EMF and OCL

CHANG Hao-Hao　QIN Zheng
(School of Software, Tsinghua University, Beijing 100084)

Abstract　With the advent of model based software engineering, model becomes an important part of software engineering. Introduce the MOF, XMI, OCL standards and the corresponding implementation EMF. Explain the way using UML and the needs to design new language with EMF and OCL. Compare the model language based on EMF and the language based on EBNF, then give a way to transform the EMF based model by combination of template and OCL and through this way transform the UML class model to O/R mapping framework. At last analyze the reuse mechanism of MDA.

Keywords　MOF, OCL, EMF, MDA, UML, Model transformation, O/R Mapping

1　MOF、OCL 和 EMF 简介

MOF[1]是 OMG 组织的元对象设施,它是 MDA[6,10~12]框架下用来创建建模语言的基础设施,是 MDA 软件工程方法的基础。使用 MOF 可以创建面向某领域的建模语言,比如 Petri 网建模语言和 UML[4,5,14]等。这些建模语言的建模元素是基于对象的且它们之间是结构化关系,即建模元素之间的包含和引用。而基于这些建模语言的实例模型所表达的语义是面向具体问题域的。与 MOF 相类似的分层结构有 EBNF 和 XmlSchema[16],其中 EBNF 用来描述计算机语言 XmlSchema 用来规范 xml 文档的格式。表 1 是这三种分层概念的比较。

表 1　分层概念比较

项目	MOF/EMF	EBNF	XmlSchema
元层	EClass EAttribute EReference	终结符 非终结符 产生式	Element Attribute Sequence
语言层	Uml Petrinet	C java fortune	订单格式 Soaps 格式
实例层	一个具体的 实例模型	一个 c 程序	一个订单的 xml 文档

XMI[2,9](XML Metadata Interchange)是 OMG 的规范,它定义了一种将结构化对象模型序列化到 XML 文档的格式,实现了对象的文档化存储,这种存储格式不仅可以反映对象的类型信息还能保存对象之间的包含和引用关系。

对象约束语言 OCL[3,8](Object Constraint Language)是一个作用在结构化对象模型上但不会改变模型状态的表达式查询语言,主要有两个作用:一是对模型进行语义约束,二是对模型的查询。

EMF[13,17]是 Eclipse 软件工程环境下对 MOF、XMI 和 OCL 等规范的实现,Eclipse 软件工程环境下的许多工具以 EMF 为基础,其中 org.eclipse.uml2[18]是 UML2.0 在 EMF 框架下的实现。Ecore 是 EMF 中对 MOF 核心的实现。图 1 是 EMF 的使用方式,首先描述一个结构化对象元模型,相当于用 EBNF 描述的文法,然后生成与元模型相对应的代码。图 2 是 Ecore 的元模型,相当于 EBNF 中的终结符、非终结符和产生式等文法描述元素。

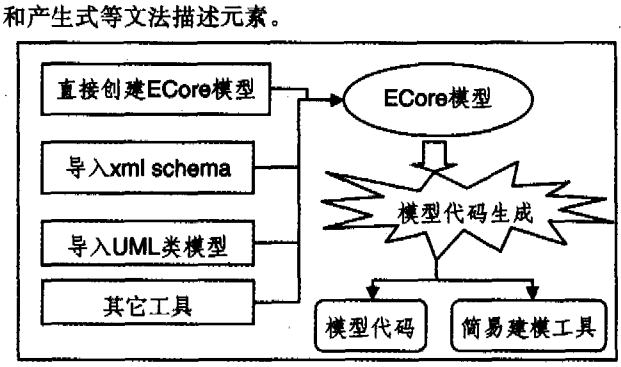

图 1　EMF 使用场景

*)国家 863 计划项目资助(2003AA412020)。常浩浩　硕士生,研究方向:软件工程、工作流;覃　征　教授,博士生导师。

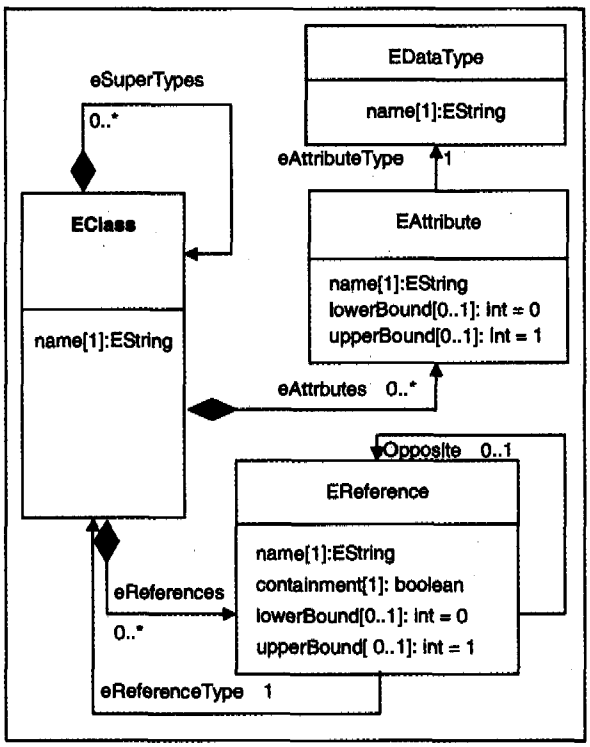

图 2 ECore 元模型片断

2 统一建模语言 UML 及其使用方式

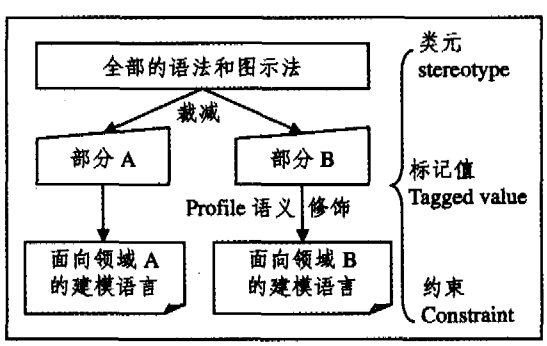

图 3 UML 语言的分层结构

表 2 UML 的使用方式

项目	主要目的	语言关注点	特点
草图	概要设计，交流，理解系统	图	探求性的，部分的，语义在交流中确认
蓝图	详细设计	图和元模型	定义性的，强调完备，语义不要求机器处理，手工翻译
编程	生成软件制品	元模型 语言的语义	编码性的，强调完备，语义可以机器处理，自动翻译

UML 是用 MOF 和 OCL 定义的一种建模语言，包含了一组抽象语法和相应的图示法，试图作为一个通用的建模语言来描述软件系统的静态结构特征和动态行为特征。UML 不是一个单一的语言而是多个子语言的组合，如类模型、活动图、状态图等。尽管 UML 用一组子模型来描述软件系统的多个方面，但对模型的结构完整性和语义一致性未作明确的规定，即无法回答一个 UML 模型是否合法。另外 UML 通过 Profile 机制来修饰通用建模元素以使其具有特定应用领域的语义，Profile 由类元、标记值和约束组成。但这种方式不能改变 UML 的语法结构，只能通过对 UML 原有语言元素的修饰来映射特定语义而缺乏对应用域直观地结构性支持，如图 3。

文[15]中把 UML 的使用方式分为三种：草图绘制语言，蓝图绘制语言，编程语言。表 2 做了进一步分析。

3 MDA 软件工程方法

MDA 是以模型为中心的软件工程方法，其理论和技术主要围绕模型的定义、元建模工具和模型的转换三方面进行。其中模型的定义即建模语言的创建，可以通过 MOF 和 OCL 来直接定义某种建模语言也可以通过裁减和修饰 UML 语言的方式进行，前者具有灵活的完全的定制能力，而后者是一种轻便的方法，可以重用 UML 中的语言元素；元建模工具指能够通过映射或者生成的方法从模型的定义得到与之相一致的可视化建模工具的工具，目前已有的元建模工具有 metaedit +[18]、GME[19] 和 Software Factories[20] 等，但这些工具没有遵守统一的标准，都有自己独立的元模型表示方法和模型处理方法；模型的转换包括模型到模型的转换和模型到代码的转换。OMG 组织为 MDA 软件工程方法提供了统一的标准[1~7]，而 Eclipse 为研究人员和开发者提供了基于插件的开放的统一工程环境。

4 使用 EMF 和 OCL 定义元模型

能够为不同的领域（应用域或技术域）创建建模语言是 MDA 软件工程方法的重要能力，这个任务主要由 MOF 和 OCL 来完成，前者描述类化的建模元素和他们之间的结构关系，后者通过约束建模元素之间的关系来表达语义。在 EMF 框架下目标模型的语法特征可以用 ECore 模型描述，语义约束可以通过把 OCL 表达式附加在 ECore 模型的相关元素上实现。

```
AddExpr → MulExpr ('+' MulExpr)*
MulExpr → Atom ('*' Atom)*
Atom → ('0'..'9')+ | '(' AddExpr ')'
```

图 4 表达式的 EBNF 范式

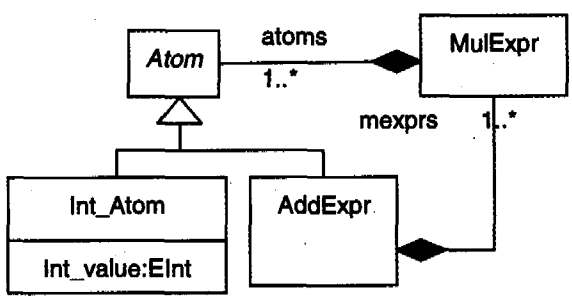

图 5 表达式的 ecore 模型

一个简单的表达式语言可以用 EBNF 如图 4 表示，之后通过 YACC/Antlr 等生成器可以生成相应的语言解析器。对于同样结构的语法也可由 ecore 模型来描述，图 5 是表达式的 ecore 模型，同 YACC 等语言工具类似 EMF 可以生成与 ecore 模型相对应的模型代码和简易的建模工具。尽管 EMF 与 YACC 的使用方式类似，但它们有着很大的区别，表 3 总结了其区别。表 4 分析了基于 EMF 的建模语言和基于 EBNF 的计算机语言的区别与联系。

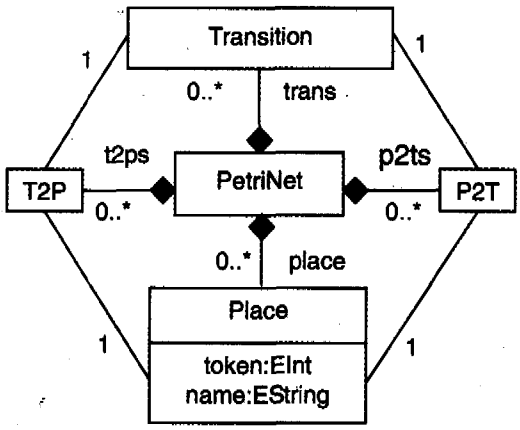

图 6　Petri 网的 ecore 模型

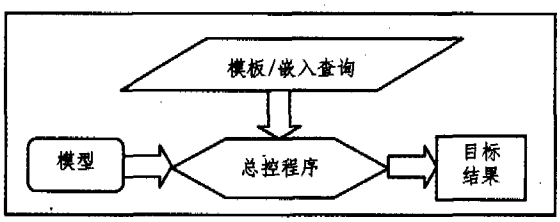

图 7　模型转换框架

表 3　解析器生成器(YACC/Antlr)和 EMF 的比较

项目	解析器生成器(如 YACC/Antlr)	EMF	联系与区别
语言设计元素	终结符 非终结符 产生式	EClass EAttribute EReference	前者基于 EBNF 泛式，后者基于 ECore 元模型
输入	文法文件和嵌入其中的代码	Ecore 模型	前者描述树形结构的语言，更复杂的结构需要语言本身的语义来表示，后者可以直接描述更加复杂的结构化对象模型
输出	带有语义动作的语言解析器	表示模型的类和一个基于树和表的简易建模工具	前者把无结构的字符流识别成有结构的语言单元，后者直接把结构化对象模型保存在 XMI 格式的文本中

OCL 用来约束模型的语义或查询模型，以下的 OCL 语句是针对图 6 的 Petri 网元模型。

约束示例-没有重名的库所：

```
Plcae. allInstances()->forAll
(p1,p2|p1<>p2 implies p1. name<>p2. name);
```

查询示例-所有可以触发的变迁：

```
Transition. allInstances()->select(t|
t. p2ts->forAll(p2t|p2t. place. token>0))
```

5　结合模板和 OCL 的模型转换方法

模型创建后需要进一步处理，最重要的操作是模型转换，模型转换分为模型到模型的转换和模型到代码的转换。OMG 发布的 QVT[7] 规范主要是针对模型到模型的转换。这里给出一种结合模板和 OCL 的模型到代码的转换方法，通过把 UML2 类模型转换到 O/R 映射框架 Hibernate 来说明此方法的有效性。

基于模板的转换在软件工程中的很多地方已经使用，比如 C 语言中的宏、C++中的模板技术、动态 Web 中的 Jsp 技术和 XML 文档转换技术(Xslt)，前两者通过简单的把形式参数代换为实际参数的方式把模板转换为目标结果，后两者需要使用特定的查询语言从相应的数据模型中提取所需的内容，图 7 是转换框架示意图，表 5 是基于模板的转换方法对比。

表 4　基于 EMF 的建模语言和基于 EBNF 的计算机语言的比较

项目	传统软件工程中的语言	MDA 中的建模语言	联系于区别
语法手段	EBNF (自描述的)	MOF/EMF (自描述的)	前者以字符集为基本构造单元，后者以可序列化对象为基本构造单元
语义手段	符号表/检验程序	OCL(约束)	前者对不同的语言要建立不同的符号表和检验程序，后者对不同的建模语言可以使用统一的 OCL 来约束语义
语言	C pascal java	Uml petrinet workflow ER	前者用文本编辑器来编写程序，后者通常采用可视化的建模工具
实例	一个具体的 c 程序或 java 程序	一个具体的模型	前者是通用编程语言，后者是面向某个特定问题域的模型
转换方法	编译器	模板 + OCL(查询)/ QVT 转换语言	前者对不同的语言要建立不同的转换程序，后者用统一的转换语言描述不同的转换
目标平台	CPU/操作系统/虚拟机	C/Java/ J2EE/. NET/ WebService/ 组件-框架-设计模式等	后者是比前者的抽象层次更高的技术框架

表 5　基于模板和查询的转换方法比较

数据模型	模板语言	所嵌查询语言
XML 数据	Xslt	Xpath
关系数据	Jsp	Sql
结构化对象数据	Jet	Ocl

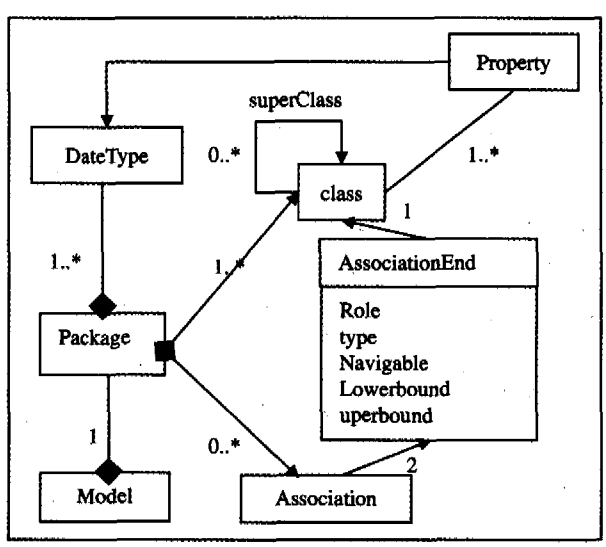

图 8　从 UML 中裁减出的建模元素

MDA 软件工程方法中可以用裁减并修饰 UML 语言的方式获得一个面向某个问题域的建模语言，这里针对数据建模 O/R 映射这个问题域。

(1)从 UML 中裁减出部分建模元素，如图 8。

(2)用 OCL 表达语义约束。

比如要求类属性名称中不能有'id_类名'以保留用来统一处理对象唯一标识可以表达为：

Class. allInstances ( )-) forAll (a | a. ownedAttribute-) forAll(b|b. name()'id_'. concat(a. name)))。

(3)写转换模板。

javaclass. jet 把一个 UML 类转换为一个 java 源文件，如图 9。hibernate. map. jet 把 UML 类及其继承和关联关系转换为 O/R 映射配置文件。

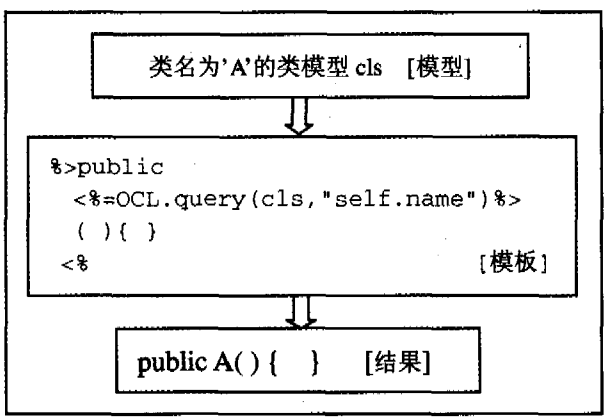

图 9 模板转换示意

转换流程包括：加载是以 XMI 格式存储的 UML2 类模型，对模型进行结构完整性和语义一致性检查，如果检查通过，使用模板编译后的模板类生成目标 java 源文件和 XML 配置文件，如图 10。OCL 在这里起到了两个重要作用：模型完整性和一致性检查；在模型转换中起到了模型查询的作用。

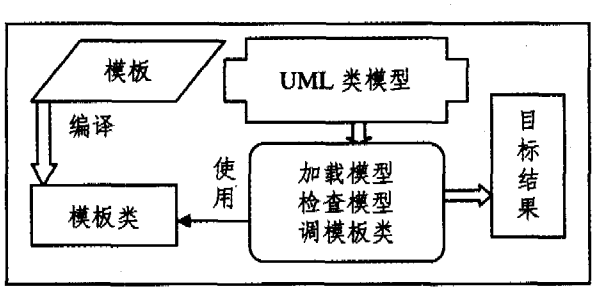

图 10 模型转换流程

由该例子认为尽管通过裁减和修饰 UML 语言元素能够得到面向问题域的建模语言，但这种扩展机制并不能改变 UML 语言本身的语法结构，只能通过在原来的语言元素上添加一些修饰语义，而不能从结构上直观反映问题域，所以对于 MDA 软件工程方法而言，能够使用 MOF 和 OCL 创建面向特定问题域的建模语言是至关重要的。

6 MDA 软件工程方法中的重用机制分析

软件工程的目的是为了解决需求表示到设计实现的映射。MDA 软件工程方法是一种域工程方法，以需求知识元模型化，解决方案模板化为原则，通过将过去成功的需求分析概念和设计实现方法分别表示成元模型(即建模语言)和模板的方式来保存过去的经验和成果。通过域工程和具体应用的迭代反馈不断地改进并积累元模型和模型转换规则从而达到部分软件自动化生产的目的，如图 11。

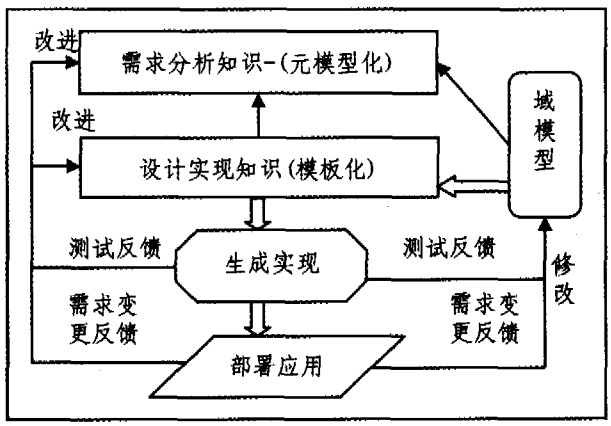

图 11 MDA 重用机制

结束语 MDA 软件工程方法还是一种不成熟的方法，在 Eclipse 软件工程环境下 EMF 框架对模型定义提供了成熟的支持，但对于把任意的元模型映射到可视化建模环境还没有成熟的工具，GMF[17]项目是把 Ecore 模型映射到可视化编辑框架 GEF[17]的尝试，目前还处于试验阶段。另外模型的转换方法也还未成熟，许多研究人员针对 OMG 组织模型的 QVT 模型转换规范做了有意义的尝试，并把其成果汇集于 Eclipse 下的 GMT[17]项目下。目前 MDA 软件工程方法主要用在嵌入式领域和企业软件领域，相信随着 Eclipse 软件工程环境下元建模和模型转换方法与工具的成熟，MDA 软件工程方法会得到广泛的应用。

参 考 文 献

1 Meta Object Facility 2. 0 Core Specification. OMG document 2004-10-05
2 MOF 2. 0/XMI Mapping Specification. OMG document 2005-09-01
3 OCL 2. 0 Specification. OMG document 2005-06-06
4 UML: Infrastructure. OMG document 2004-10-14
5 UML: Superstructure. OMG document 2005-07-04
6 MDA Guide 1. 0. 1. OMG document 2003-06-01
7 MOF QVT Specification. OMG document 2005-11-01
8 Warmer J, et al. The Object Constraint Language Getting Your Models Ready For MDA 2nd Edition . Addison Wesley, 2003
9 Grose T J. Mastering XMI-Java Programming with XMI, XML, and UML. Wiley, 2002
10 Kleppe A, et al. MDA Explained: The Model Driven Architecture: Practice and Promise. Addison Wesley, 2003
11 Stephen J, et al. MDA Distilled: Principles of Model-Driven Architecture. Addison-Wesley 2004
12 Frankel D S. Model Driven Architecture Applying MDA to Enterprise Computing. Wiley, 2003
13 Budinsky F, et al. Eclipse Modeling Framework: A Developer's Guide. Addison Wesley, 2003
14 Engels G, et al. UML-A Universal Modeling Language? Springer-Verlag Berlin Heidelberg, 2000
15 Fowler M, 徐家福译. UML 精髓 . 第 3 版. 清华大学出版社, 2005
16 http://www. w3. org/2001/XMLS chema. xsd
17 http://www. eclipse. org/[emf uml2 gmf gmt gef]
18 http://www. metacase. com
19 http://www. isis. vanderbilt. edu/projects/gme/
20 http://www. miscrosoft. com/