一个支持模型驱动开发的元建模平台的研制

麻志毅,刘 辉,何 啸,张 乐,吉 喆,戈 牧
(北京大学信息科学技术学院软件所,高可信软件技术教育部重点实验室,北京 100871)

摘 要: 由于当今软件系统的复杂性,模型驱动的软件开发中需要的建模语言和转换定义语言往往是多种多样的.本文阐述了对一个可满足这种需要的元建模平台的研制.其中着重论述了对建模语言的元模型和表示法进行建模的技术,建模语言质量保证机制,以及针对所建立的建模语言和转换定义语言提供自动生成相应工具的设施.

关键词: 元建模; 模型驱动开发; 平台
中图分类号: TP311 **文献标识码**: A **文章编号**: 071845-2112 (2008) 04-0731-06

Research and Design of a Meta-Modeling Platform for Model Driven Development

MA Zhi-yi, LIU Hui, HE Xiao, ZHANG Le, JI Zhe, GE Mu
(*Software Institute, School of Electronics Engineering & Computer Science, Peking University*
Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, Beijing 100871, *China*)

Abstract: Because of the complexity of software systems nowadays, the manifold modeling languages and transformation definition languages are needed in model driven software development. This paper presents a meta-modeling platform for defining and improving the needed languages. The paper primarily discusses the technology of modeling meta-models and notations of modeling languages, the mechanism for the assuring modeling language quality, and the facility for automatically generating the related tools aiming at the defined languages.

Key words: meta-modeling; model driven development; platform

1 引言

在模型驱动开发(Model Driven Development, MDD)中,模型已经成为首要制品.通常使用通用建模语言(如UML)建立软件系统的模型,但这样的建模语言不能完全满足对多数领域建模的需求,这需要对这样的建模语言进行扩展或建立新的建模语言.此外,已有的领域建模语言不断地在发展变化,新的领域建模语言也将不断地涌现.如何对建模语言的建立和维护提供技术支持,是一个急需解决的问题.

在MDD中,转换定义语言用于建立转换规则,转换规则用于转换用建模语言描述的模型.由于转换规则可能是复杂的,转换定义语言也可是种建模语言,用于对转换规则建模.

所产生的建模语言要有相应的建模工具的支持.建模语言正处于发展变化之中,新的建模语言也在不断地涌现.针对建模语言自动生成相应的建模工具,可解决手工开发或维护建模工具的高成本和高技术复杂性问题.由于要根据所建立的建模语言生成相应的工具,进而用于构建应用模型,因而要提供保证建模语言的质量的机制.元建模是定义建模语言和为应用建模语言提供必要的基础设施的过程.本文提出的平台要为元建模提供技术支持.

2 设计原理

上述的元建模语言、建模语言、转换定义语言、相应的工具、工具自动生成技术和语言质量保证机制为模型驱动开发提供了基础设施,它们之间的关系图1所示.

在图1中,元建模者在元建模工具的支持下,使用元建模语言建立建模语言和转换定义语言.为便于针对建模语言用转换定义语言构造转换规则,转换定义语言和建模语言应该是由同一元建模语言建立的[1].建模者

收稿日期:2007-10-08;修回日期:2008-02-03
基金项目:国家自然科学基金(No. 60773152);国家863高技术研究发展计划(No. 2007AA01Z127, No. 2007AA010301);国家科技支撑计划(No. 2006BAH02A02);国家973重点基础研究发展规划(No. 2005CB321805)

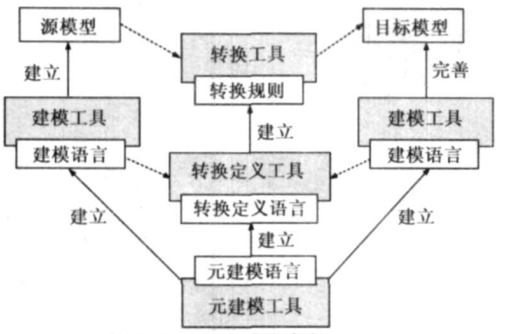

图 1 MDD 中的语言与工具

在建模工具的支持下,用建模语言建立或完善模型. 在转换定义工具的支持下,转换规则开发者针对建模语言用转换定义语言建立转换规则. 转换工具基于转换规则把一种模型(源模型)转换为另一种模型(目标模型),如把平台无关的模型转换为平台相关的模型.

3 元建模平台架构

元建模平台为模型驱动的软件开发提供语言和工具,以供建模工具开发者和应用系统开发人员使用. 元建模平台的架构如图 2 所示.

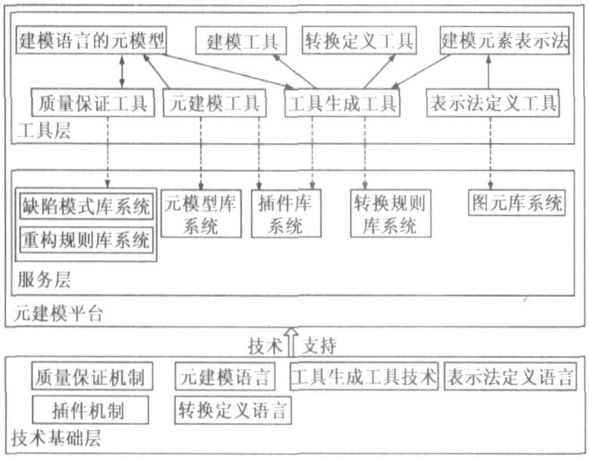

图 2 元建模平台的构架 *

该平台呈现给用户的是一组工具,这些工具构成了平台的工具层. 对于各种工具,要有相关的基础服务提供支持,这些基础服务形了平台的服务层. 工具层和服务层一起构成了元建模平台. 实现这个元建模平台要使用一系列的技术,这些技术构成了平台的技术基础层.

3.1 技术基础层

3.1.1 元建模语言

元建模语言是用于建立建模语言的. 一个建模语言包括抽象语法、具体语法和语义三个要素[2,3]. 一种主流的方法是采用元模型来描述建模语言的抽象语法和静态语义. 静态语义描述一般采用文本方式,其中的一些约束规则可以使用约束语言来表达. 具体语法一般采用图形化表示法,有关具体语法的内容见下节.

在对建模语言的抽象语法的定义上,本平台采用的是 OMG(Object Management Group)的 MOF2.0[4],它是一种元建模语言;对建模语言的部分语义的描述采用的是 OMG 的 OCL(Object Constraint Language)[5].

3.1.2 表示法定义语言

由于 MOF2.0 本身不含有对建模语言中的建模元素的表示法进行定义的语言,本平台采用了自己研制的建模元素表示法定义语言[6]. 该语言为用户提供了一组丰富的用于定义建模元素表示法的图形元素和图形编辑支持机制,其组织结构如图 3 所示.

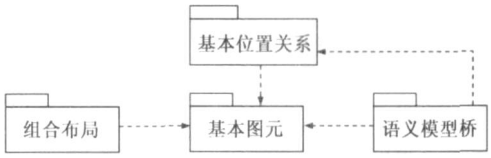

图 3 表示法定义语言的组织结构

图元:通过分析典型建模语言(如 UML 和 BPMN)中的建模元素的表示法,从易用性、简洁性和完备性考虑,确定出以下的用于表示法建模的基本图元:矩形、圆角矩形、菱形、三角形、多边形、椭圆形、圆形、直线、折线、弧线、文本对象和图片对象.

组合布局:通常需要把若干图元按照一定的方式组合起来定义建模元素的表示法. 组合布局反映了图元之间的组合关系. 虽然图元可以按任意的方式组合起来,但是通过对典型建模语言的表示法的总结,基本图元的组合可以分为以下 4 种模式:类模式、关联模式、关联类模式和活动者模式.

基本位置关系:组合布局反映了建模元素表示法的内部构造,而基本位置关系则定义了表示法之间可能存在的位置关系. 通过对典型建模语言中的建模元素表示法的分析,建模语言表示法之间的位置关系可以归结为 5 种:块状元素之间的嵌套、线状元素对块状元素的附着、块状元素附着在另一块状元素的边界上、块状元素对线状元素的附着、线状元素附着在另一线状元素上.

语义模型桥:语义模型桥是表示法和元模型之间的一种对应关系,这种关系描述了图形表示法及其构成元素与元模型中的元素是如何映射的. 用语义模型桥可建立三种映射:元模型映射、元属性映射和元关系映射.

在图 3 中,包“组合布局”、“基本位置关系”和“语义模型桥”都要使用包“基本图元”中的元素. 包“语义模型桥”还要使用包“基本位置关系”中的元素来建立表示法与元模型间的映射.

---

* 在图 2,3,6,7,9,10 和 11 中,带箭头的实线表示输入/输出,带箭头的虚线表示访问关系.

3.1.3　转换定义语言

OMG的QVT[1]是种转换定义语言,它是用MOF定义的,且可用它图形化地建立转换规则.这样,转换定义语言也是一种建模语言,针对QVT来自动生成图形化的转换定义工具是可行的.

然而,QVT对复合转换、动态行为和转换事务等进行图形化建模的能力是弱的,故要对QVT的这些方面要予以改进.在本平台中,丰富QVT了表示法,增加了转换概览图、转换顺序图、组合结构图以及对转换事务性的标注进行建模的机制.

3.1.4　建模语言质量保证机制

要保证建模语言的质量,首先要对其进行语法和语义检查,然后通过度量来评价其质量.对于度量后所发现的问题,通过对元模型进行重构以及对建模元素的表示法进行改进来解决.

(1) 建模语言的质量模型

质量模型由一组定义模型质量的要素和这些要素之间的关系组成.针对建模语言的质量度量所建立的质量模型[7~9]如图4所示.

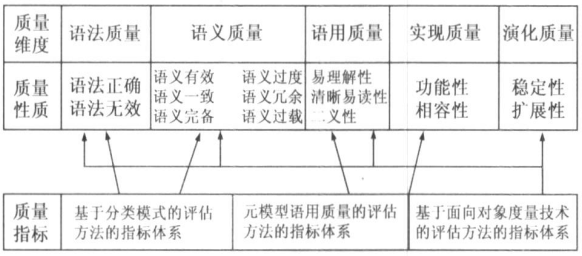

图4　建模语言的质量模型

图4所示的质量模型分为五个质量维度,每个维度具有若干质量性质.对于每个质量维度可由一些方法来度量.

语法质量刻画了建模语言的元模型遵循元建模语言的元元模型的程度.它的质量性质为语法正确和语法无效.

语义质量描述了建模语言的元模型与它规约的问题域的符合程度,即元模型中的建模元素到领域概念的映射情况.它的质量性质为语义有效、语义过度、语义一致、语义冗余、语义完备和语义过载.

语用质量反映了建模语言被其用户理解和使用的程度.它的质量属性为易理解性、清晰易读性和二义性.

实现质量反映了建模语言可以被建模工具支持的程度以及工具对建模语言的相容程度.它的质量属性为功能性和相容性.

建模语言的演化表示对建模领域的认识或者建模语言的内容发生了变化的程度.它的质量属性为稳定性和扩展性.

针对不同的评估方法有不同的质量指标体系.图4中列出了基于分类模式的评估方法、建模语言语用质量的评估方法和基于面向对象度量技术的评估方法.这三种方法所使用的质量指标体系请参加文献[8].

(2) 质量评价

虽然可以用建模语言的质量模型计算建模语言的质量,但是还难以直接回答一个建模语言的质量高低.为此,文献[7,8]通过一个建模语言成熟度模型来帮助解决这个问题.利用该模型可以有效地根据建模语言的质量表现对其进行质量分级.

(3) 对建模语言的重构

对建模语言重构是在保持建模语言的功能性的前提下,要对元模型中的元类和元属性等构造物进行重新组织和分配,并要对建模元素表示法进行再定义,以便于应用和扩展.由于对建模元素表示法再定义较为简单,以下讨论对元模型进行重构.

图转换有比较深厚的数学基础,很多研究已经将重构规则描述为图转换规则,并借助于图转换理论对重构规则的性质进行推导和证明,从而可以严格保证转换规则对软件某些特性(比如软件的外部行为特性)的保持.但是现有的基于图转换的软件重构描述语言依然存在表达能力不足的问题,以致于对许多复杂重构规则无法描述或者难以描述.

QVT结合了图转换与OCL.QVT没有注重考虑对转换规则的性质的推导和证明,并且对OCL的使用几乎没有任何限制.此外,QVT的表达能力很较弱,它只能表达一些简单规则.

本平台基于QVT给出一种重构描述语言[10].该语言对OCL约束进行了限制,其目的在于便于重构规则的性质推导,同时对QVT在动态行为、组合转换和图形化建模方面进行了增强.对建模语言重构的过程如图5所示.

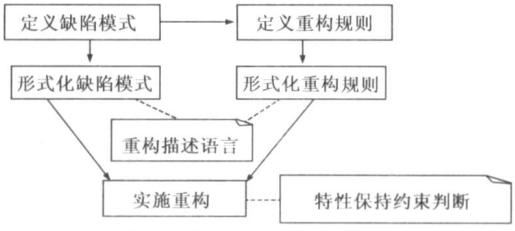

图5　建模语言重构的过程

首先定义元模型中的缺陷模式,并以此定义重构规则,然后用重构描述语言对它们进行形式化,进而对建模语言的元模型进行重构[10,11].

3.1.5　工具生成技术

该技术要解决把建模语言作为输入,生成相应建模工具的问题,且所生成的建模工具要具有图形化地进行模型编辑和操纵的能力.

图 6 所示的是本平台所采用的工具生成技术框架.

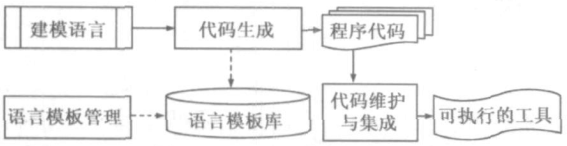

图 6 工具生成技术框架

语言模板管理部分用于编辑和管理语言模板. 语言模板库中的语言模板用于把特定的模型转换为指定的代码. 把建模语言作为输入,代码生成部件使用语言模板自动生成建模工具的主体程序代码. 代码维护与集成部件负责维护代码并把代码与预制的插件相集成,直至产生可执行的建模工具.

3.1.6 插件机制

插件是为系统提供功能的代码和数据的结构化包. 在插件实施的支持下,插件可协同工作. Eclipse 平台[12]具有发现、集成和运行插件的机制. 使用 Eclipse 平台,不存在表示集成问题;使用标准的 Eclipse 扩展机制能够解决控制集成问题;加之使用 XMI 提供了数据集成机制. Eclipse 平台还提供了丰富的插件. 特别是,当前的 Eclipse 技术是基于 OSGi 的,这使得它具有强的生命力. 本平台是基于 Eclipse 技术开发的,完全使用其插件机制.

3.2 平台的服务层

在上述的技术支持下,该层要为建立建模语言以及生成工具提供服务. 图元库系统管理用于构建建模元素表示法的基本图元. 元模型库系统用于管理元模型. 转换规则库系统用于管理生成代码用的转换规则. 在模型重构时,要使用一些缺陷模式和大量的重构规则,分别用缺陷模式库系统和重构规则库系统管理它们. 插件库系统用于对预定义和新开发的插件进行管理.

3.3 平台的工具层

3.3.1 表示法定义工具

表示法定义工具用于对建模语言中的建模元素的表示法进行定义. 图 7 给出了表示法定义工具的构成.

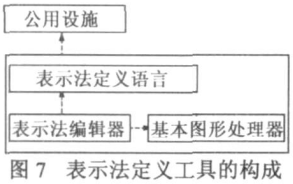

图 7 表示法定义工具的构成

部件"公共设施"提供了通用的图形编辑、XML 文件的导入与导出、抽象类、事件处理、资源管理、模型树管理、对象注册和属性编辑等功能. 元建模工具、代码生成工具以及建模工具也要使用该部件.

部件"表示法定义语言"把表示法定义语言代码化和数据化. 它还为该语言的元模型管理、元模型存储和元属性编辑提供了支持,这主要是通过实现公共设施的相关接口的方式完成的.

部件"表示法编辑器"依据部件"表示法定义语言"中代码化和数据化的表示法定义语言,生成工具栏. 用户通过使用工具栏上的建模元素,按照表示法定义语言的要求以所见即所得的图形化方式来定义表示法.

部件"基本图形处理"提供对各种基本图形元素的操纵功能,如缩放、定位点和着色等. 表示法编辑器使用该部件.

3.3.2 元建模工具、建模工具与插件

元建模工具用于建立建模语言的元模型以及描述其语义,它由一个核心部分和若干插件组成. 其核心部分要按照 MOF 对建模语言的元模型进行图形化编辑、管理和存储等,并用 OCL 描述建模语言的静态语义,其内要使用 3.3.1 节中的公用设施.

由于按当前的技术不能生成全部的建模工具代码,本平台能生成建模工具的核心部分,其余的一部分靠预制的插件提供,另一部分是靠建模工具开发者自己编制的插件提供. 图 8 描述了插件设施与工具的插件以及它们之间的关系.

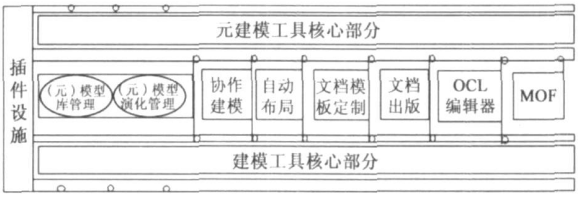

图 8 插件设施与工具的插件以及它们之间的关系

从图 8 中可以看出,元建模工具和建模工具都是基于插件机制的.

现已经对元建模工具预制了如下插件:元模型库管理、元模型演化管理、协作建模、自动布局(按照一定算法对元模型图进行自动布局)、文档模板定制(用户可用文档模板描述语言定义自己的文档模板)、文档出版(按选定的模板生成元模型的文档)、OCL 编辑器和 MOF(对 MOF 进行了代码化和数据化).

除了插件 MOF,上述的插件也用于所生成的工具,只是插件处理的对象是模型. 建模工具的核心部分具有通常的诸如模型编辑、管理和存储等功能,其内要使用 3.3.1 节中的公用设施. 所生成的工具也可以具有预制的插件.

3.3.3 质量保证工具

质量保证工具用于对建模语言进行语法和语义检查、度量以及重构,它由建模语言检查、度量与评价子工具和建模语言重构子工具组成.

(1) 建模语言检查、度量与评价子工具

该子工具的功能结构如图 9 所示.

对建模语言的元模型的语法正确性检查要依据 MOF 来进行,对语义正确性检查是要对其中的 OCL 表达式进行检查,这包括对 OCL 进行语法分析、与元模型信息一致的类型检查、逻辑表达的一致性检查. 为此,语

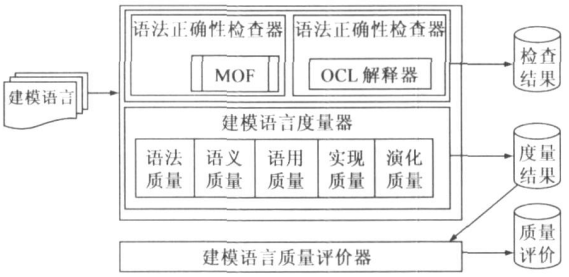

图 9　建模语言检查、度量与评价工具的功能结构

义正确性检查中内置了一个 OCL 解释器. 对于检查出来的问题需要人工予以解决.

建模语言度量器和质量评价器分别按照 3.1.4 节的质量模型和成熟度模型对建模语言进行质量度量和评价.

(2) 元模型重构子工具

该子工具输入一个建模语言或表示法定义语言的元模型,对其进行重构后,再予以输出. 其功能结构如图 10 所示.

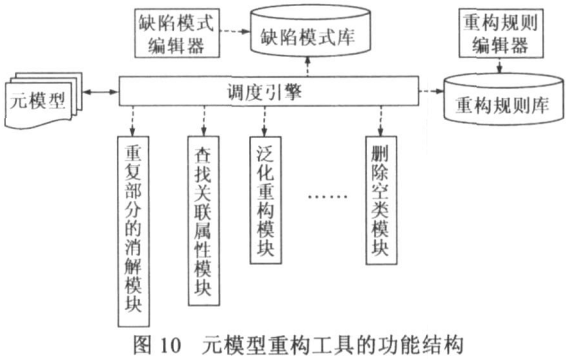

图 10　元模型重构工具的功能结构

该子工具的缺陷模式编辑器和重构规则编辑器分别用于定义或维护缺陷模式和重构规则,结果也分别放在缺陷模式库和重构规则库中. 调度引擎基于调度策略,通过对输入的元模型进行解析,调度相应的重构模块[11].

3.3.4　工具生成工具

该工具把合乎质量要求的建模语言转换为建模工具所需要的接口和实现代码,使得建模工具具有根据所定义的建模语言进行图形化模型编辑和操作的能力.

按照 3.1.5 中的工具生成技术,该工具的设计如图 11 所示.

在 3.3.1 节中定义了图 11 中的"公用设施",在 3.3.2 节定义了"MOF".

代码生成首先要利用公共设施中的 xml 文件导入功能获取元模型和表示法信息(建模元素表示法、编辑约束和视图信息). 然后对要生成的插件以及要生成的建模工具所支持的图的信息进行采集. 其中的一些信息是用户通过对话框输入的. 在得到了所需信息后,开始进行代码生成.

基于 MOF,利用公共设施以及 JET(Java Emitter Templates) 模板库系统中的 JET 模板,进行代码生成. 对于建模工具来说,需要生成有关元模型、表示法、编辑单元、模型树以菜单项等的代码. 这些部分各自可以看作一个模块. 在实际的代码生成过程中,经常要用到一些公共功能,如对元模型所在包的名字进行提取,得到待生成目录的绝对路径等,这些功能由基础模块进行提供.

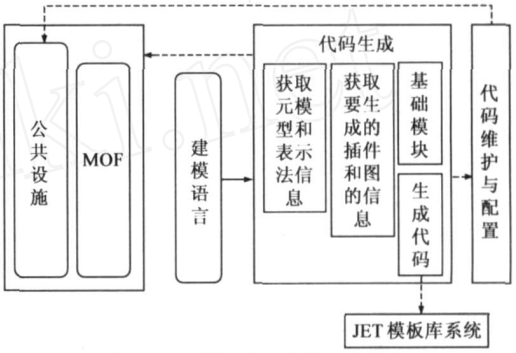

图 11　工具生成工具的功能结构

有些自动生成的代码可能还不完善或还不能完全满足用户的需求,因此需要对生成的代码进行修改和优化. 维护后的各模块共同构成一个建模工具插件,通过公共设施中的对象注册机制,可把该插件部署到运行平台上.

4　相关研究

MetaEdit + [13] 本身即能进行元建模,又能通过配置进行建模. 由于 MetaEdit + 采用的元元模型本身承担了过于复杂的语义,其建模功能不易扩展. 它的建模功能也有限. 此外,MetaEdit + 提供给用户使用的基本图元很少,且在表示法的构造上只支持简单地使用绝对坐标的布局(也即,在父图元内构造的内部成分,其大小必须和父图元一起成比例变化等),也不能表示不同建模元素的图形符号之间的位置关系. 在表示、数据和控制层上,它与其他工具的集成也存在着问题[14].

Eclipse 平台含有一些工具,可用于元建模和建模,如 Eclipse 建模框架 EMF、图形编辑框架 GEF、图形建模框架 GMF 和 Merlin 图形编辑编辑生成器.

用 GEF 构造的编辑器能无缝地与 Eclipse IDE 集成,如能直接使用 Eclipse 的代码生成插件. 但 GEF 提供的复用措施仍是低层次的,如缺乏多视图抽象,要用其构造高质量的模型编辑器,必须要进行复杂的编码.

GMF 是 Eclipse 平台上的一个开源项目[15],它结合了 EMF 和 GEF. 在 GMF 中,用户可以自己定义表示法,但是 GMF 没有一个明确的表示法元模型,而是使用了 Draw2D 中的大部分图元作为构建建模元素表示法的基本元素. Draw2D 所使用的图元多达七十余种,图元布局的种类也多达 6 种,这给用户增加了使用负担,实际上很多并不是必要的,应该对其进行重新设计. 此外,它不能说明所构建的建模元素图形符号之间的位置关系(如

UML 中的 Port 要放置在 Classifier 的边界上),而这种关系是相当常见的.

Merlin 图形编辑编辑生成器是一个开源的 Eclipse 插件[16].它用 EMF 数据模型去生成基本编辑器套件,从图语法生成 Eclipse 编辑器.Merlin 的元元模型采用了 Eclipse 的 Ecore[17].但是 Merlin 没有对表示法的定义提供支持,而只是对用户自定义的建模元素产生一个默认的矩形表示法,让用户自己去修改源代码来得到所需要的表示法.

本元建模平台 Meta-Modeler 采用的 OMG 的 MOF 不仅是标准的,而且表达能力也是强的.在对表示法的定义能力上,Meta-Modeler 采用了自定义的表示法定义语言,用户可用其定义所需要的建模元素表示法.该表示法定义语言的建模元素的数量适中且表达力优于其他几种.在易用性方面,Meta-Modeler 无论是对元模型还是对建模元素表示法的定义,均采用了惯用的图形化方式.特别是,仅 Meta-Modeler 提供了语言质量保证机制.

5 结束语

鉴于 MDD 在软件开发中的重要性,本文深入地阐述了一个元建模平台.该平台系统地提供了对建模语言的元模型和表示法的构建技术和工具,还系统地提供了语言质量保证机制和工具自动生成设施.

本平台中的对建模语言质量的评估模型还有待于进一步完善,尚需在实践中发现更多有意义的指标并积累更多的统计资料.随着模型重构技术的不断成熟,本平台要借鉴其成果对元模型重构予以加强支持.目前,所生成的插件的代码还不是很完善,对其中的一些还需要人工增补或修改,这也是今后需要重点研究的一个内容.

参考文献:

[1] OMG ptc/05-11-01. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification[S].

[2] Colin A, Thomas K, et al. The essence of multilevel metamodeling[A]. Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools[C]. London: Springer-Verlag, 2001. 19 - 33.

[3] David H, Bernhard R. Modeling Languages: Syntax, Semantics and All That Stuff, Part I: The Basic Stuff"[R]. Israel: Faculty of Mathematics and Computer Science, 2004.

[4] OMG ptc/04 - 10 - 15. Meta-Object Facility 2.0[S].

[5] OMG ptc/03 - 08 - 08. Object Constraint language[S].

[6] He X, Ma Z Y, Shao W Z, Li G. A metamodel for the notation of graphical modeling languages[A]. Proceeding of COMPSAC[C]. Bejing: IEEE, 2007. 219 - 222.

[7] 马浩海,麻志毅,吉哲,杨国东,张乐. 元模型可度量性及度量方法研究[J]. 电子学报, 2004. 32(12A): 211 - 214.
Ma H H, Ma Z Y, et al. A study of measurability and metrics for meta-models[J]. Acta Electronica Sinica, 2004, 32(12A): 211 - 214. (in Chinese)

[8] 马浩海. 面向 UML 建模语言家族的元模型质量评估技术研究[D]. 北京:北京大学信息科学技术学院, 2005.
Ma H H. Research on the Meta-Model Quality Evaluation for the UML family of languages[D]. Bejing: School of Electronics Engineering & Computer Science, Peking University, 2005. (in Chinese)

[9] Ma H, Shao W Z, Zhang L, Ma Z Y, Jiang Y B. Applying OO metrics to assess UML meta-models[A]. Proceeding. of the 7th International Conference of UML[C]. Lisbon: Springer-Verlag, 2004. 12 - 26.

[10] 刘辉,麻志毅,和云峰,邵维忠. 模型转换中的特性保持的描述与验证[J]. 软件学报, 2007, 18(10): 2369 - 2379.
Liu H, Ma Z Y, He Y F, Shao W Z. Description and proof of property preservation of model transformations[J]. Journal of Software, 2007, 18(10): 2369 - 2379. (in Chinese)

[11] Liu H, Li G, Ma Z Y, Shao W Z. Scheduling of conflicting refactorings to promote quality improvement[A]. Proceeding of International Conference on Automated Software Engineering[C]. Atlanta: ACM, 2007. 489 - 492.

[12] Eclipse Homepage. http://www.eclips.org/[DB/OL]. 2008.1.12.

[13] Juha-Pekka T, Matti R. MetaEdit+: defining and using domain-specific modeling languages and code generators[A]. Companion of the 18th annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications[C]. California: ACM, 2003. 92 - 93.

[14] John G, John H, et al. Generating domain-specific visual language editors from high-level tool specifications[A]. Proceeding of International Conference on Automated Software Engineering[C]. Tokyo: IEEE, 2006. 25 - 34.

[15] Eclipse. GMF. http://www.Eclipse.org/gmf/[DB/OL]. 2008.1.15.

[16] Eclipse. Merlin. http://sourceforge.net/projects/merlingenerator[DB/OL]. 2008.1.15.

[17] Dave S, Frank B, Marcelo P, Ed M. EMF: Eclipse Modeling Framework[M]. Boston: Addison Wesley Professional, 2006.

作者简介:

麻志毅 男,副教授,博士,主要研究方向为软件工程与软件工程环境、软件建模技术和面向对象技术等;E-mail: mzy@sei.pku.edu.cn

刘 辉 博士生,主要研究方向为模型重构技术.

何 啸 博士生,主要研究方向为模型驱动的软件开发.