

UML 类模型的 VDM++ 形式化规约

胡文生^{1,2,3}, 赵 明³, 杨剑峰^{1,3}, 贾国荣²

(1 贵州大学 计算机科学与信息学院, 贵州 贵阳 550025; 2 贵州商业高等专科学校 计算机科学与技术系, 贵州 贵阳 550005; 3 贵州省可靠性工程中心, 贵州 贵阳 550025)

摘 要: 利用目前使用最为广泛的形式化验证语言 VDM++ 在描述系统模型的语法和语义上的精确、一致的特点, 结合 VDMTOOLS 和 Rational Rose 工具把 UML 类模型中的各个元素转化成 VDM++ 表示, 从而实现对 UML 类模型中所包含的各个元素进行语法和语义的检查, 进一步提高 UML 的建模质量。

关键词: 统一建模语言; 类图; 类; 维也纳开发方法

中图分类号: TP311.1

文献标识码: A

文章编号: 1000-7180(2012)06-0104-04

VDM++ Formalization of UML Class Diagram

HU Wen-sheng^{1,2,3}, ZHAO Ming³, YANG Jian-feng^{1,3}, JIA Guo-rong³

(1 College of Computer Science & Information, Guizhou University, Guiyang 550025, China;

2 Department of Computer, Commercial College of Guizhou, Guiyang 550005, China;

3 Credibility Engineering Center of Guizhou Province, Guiyang 550025, China)

Abstract: The purpose of this paper is to use the most widely used formal language-VDM++ that has characteristics of accuracy and consistent in describing syntax and semantic of the system model, combined with VDMTOOLS and Rational Rose tool to transform the various elements of the UML class model into representation of VDM++. This method achieves the various elements contained UML class model about the syntax and semantics checking, and further improves the quality of UML modeling.

Key words: UML; Class diagram; Class; VDM++

1 引言

UML(Unified Modeling Language)是国际对象管理组织(Object Management Group,OMG)推出的用于对软件系统设计进行可视化、详述、构造和文档化的一套工业标准^[1]. UML 语言采用图形化的方式对系统的各个不同方面进行建模,有利于系统设计人员、系统开发人员和用户对即将开发的系统事先有一个直观、简单的了解.但是由于 UML 语言建立的模型图缺乏严格的语义定义,容易产生歧义,无法保证各模型图精确描述以及各模型图之间的一致性约束.因此,有必要对 UML 各模型图进行严格的形式化约束,以增强 UML 模型图的精确性,以及各模型图之间的一致性检测. VDM++是目前

比较流行的形式化描述语言之一,相比于其它形式化描述语言 Z、B 语言来说, VDM++ 具有更清晰的结构形式,利用 VDMTOOLS 工具可以很容易地实现由 VDM++ 转化为 C++、JAVA 源程序. 本文主要研究 UML 类图模型如何转化为 VDM++ 形式化规范的具体方法,利用 VDM++ 严格的形式化语义能够帮助发现 UML 建模过程中不容易发现的系统描述的不一致、不明确或不完整的地方,从而有助于增加软件开发人员对系统的理解,也是提高软件系统安全性与可靠性的重要手段.

2 VDM++ 和 UML 类图介绍

2.1 VDM++

VDM 也叫维也纳开发方法(Vienna Develop-

ment method),是用于开发大型软件系统的一种形式化方法.它以一阶谓词逻辑和抽象的数据类型为基础来对程序设计过程中所涉及到的每个运算、操作、函数的功能进行形式化的规格说明.

VDM的基本思想是:运用抽象的数据类型、数学概念和数学符号来规定运算或函数的功能,而这种规定的过程是结构化的,其目的要在系统实现之前简单而又明确地指出软件系统要实现的功能.其中,规范说明是VDM的核心.因此,VDM是一种规范说明技术,它充分地形式化.一个VDM规格说明一般是由types、values、functions、operations、state等不同的块组成的.而VDM++是在VDM-SL的基础上发展出来的,两者在语法和语义上稍微不同.在VDM-SL中的状态定义已经由VDM++中的实例变量定义所取代,在VDM++中表达式更加丰富,增加了new expression, self expression, isof-class expression^[2]等多个表达式.在VDM++中建立的主要模块是类,在类体内可以定义类的继承关系、类的各种属性、类的各种功能和各种操作.类的定义形式如图1所示.

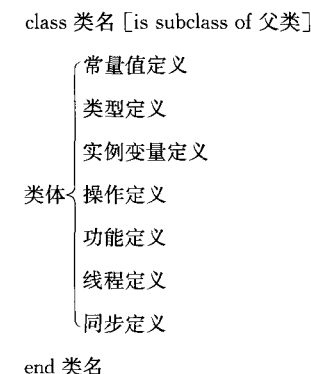


图1 VDM++中类的定义形式

2.2 UML类图介绍

UML类图主要用来反映系统所包含的所有类及类之间的关系,在绘制UML类图时,一般使用一个矩形来表示,该矩形被分割成三个部分:类名;属性;操作.单独一个类的描述如图2所示.

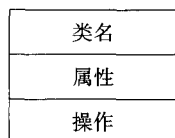


图2 UML中单独一个类的结构

UML类图中还可以反映出系统所包含的所有类之间的关系:关联、泛化、依赖以及抽象关系.

3 UML类图中的元素映射到VDM++对应的成员中

3.1 类在VDM++中的表示

(1) 类名的转化:UML单独一个类的类名转化为VDM++类定义形式下的类名.如图3所示.

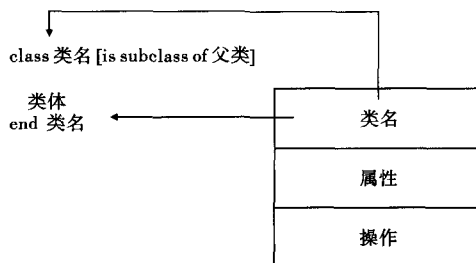


图3 UML单独一个类的类名的VDM++形式化

(2) 属性名的转化:UML单独一个类中的属性可以相应的转化为VDM++中的常量值、类型、实例变量的定义.在图4中用UML类图表示的一个类Teacher中包含工号(id)、姓名(name)、系部(depart)、授课名称(course)等属性,并为这些属性设置了可见性.

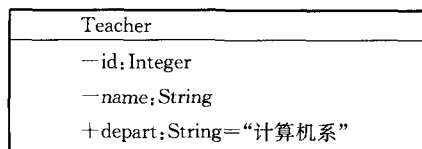


图4 UML中类的属性

在对类Teacher中的属性depart用VDM++形式化描述成如下形式:public depart: seq of char:=[‘计’,‘算’,‘机’,‘系’].也就是说在转换过程中,必须要注意各部分的对应,UML类属性的可见性要转化为VDM++形式化规范后的可见性,属性的可见性一般有private、protected、public,默认为private. UML类中的属性名转化为VDM++形式化规范后的属性名. UML类中的属性类型要成为VDM++形式化规范后的属性类型,但是在UML类中属性为String类型的要对应VDM++的seq of char,为float或double的要对应于VDM++中的real.

(3) 操作的转化:在UML类图中操作是用来表示类的行为和功能的,类中的操作是由操作名、访问权限、参数列表、参数类型、参数个数、操作的返回值等部分组成.在对UML类图中的操作进行VDM++形式化规约的过程中,需要注意的几点是:在

UML 中操作的访问权限可以对应 VDM++ 中操作的访问权限;UML 操作名对应 VDM++ 中的操作名;UML 中参数名对应 VDM++ 中的参数名;UML 中的参数类型对应 VDM++ 中的参数类型;UML 中的操作的返回值对应 VDM++ 中的返回值. UML 类图中操作的 VDM++ 形式化规约具体过程如图 5 所示

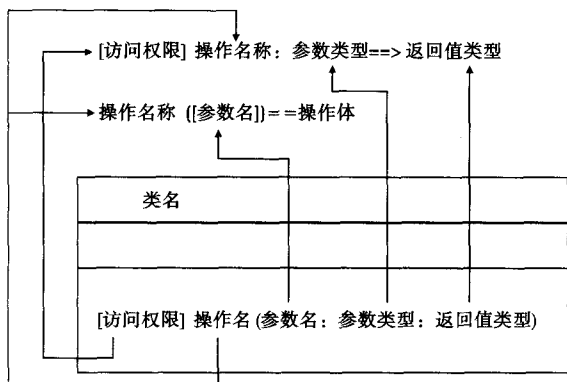


图 5 UML 中类的操作的 VDM++ 规约

3.2 UML 类图中各类之间的关系在 VDM++ 中的表示

在 UML 类图中各类之间的关系包括关联、泛化、聚合. 这些关系在 VDM++ 中的表示形式是不一样的.

(1) 继承关系的 VDM++ 形式化规约

继承(inheritance)主要是指由已有类创建新类的机制. 利用继承, 可以先创建一个具有公共属性的一般类, 然后根据该一般类再创建具有特殊属性的新类. 新类继承一般类的属性和方法, 并根据需要增加自己的属性和方法. 体现在 UML 类图中的继承关系是用一个带空心箭头把子类与父类连接起来, 其中空心箭头指向父类. 对 UML 类图中的继承关系进行 VDM++ 形式化规约过程如下所示:

class 子类 is subclass of 父类

类体

end 子类

也就是用关键字 is subclass of 来代替 UML 类图中的空心箭头

(2) 关联关系的 VDM++ 形式化规约

UML 类图中的关联用于表示类的实例对象之间相互引用或组合关系, 在 UML 类图中用一条实线连接两个类以表示两个类的对象之间存在链接. 在链接上可以设置它们的多重性, 如果他们是默认的值, 就是 1, 此时, 如果把该关联规约到 VDM++

的过程如下:

class 类名

实例变量部分

对象名: 关联类的类名

End 类名

如果关联上还带有序约束, 其 VDM++ 形式化规约如下: 默认情况下, 所有关联都是无序约束的, 当关联的另一端表示为 $0 \cdots n$ 时, 在 VDM++ 中表示为 set of 关联类的类名: $= \{ \}$. 如果关联上带有序约束, 其 VDM++ 形式化规约如下: 当关联为一个有序约束时, 关联的另一端表示为 $0 \cdots n$ 时, 在 VDM++ 中表示为 seq of 关联类的类名: $= []$.

4 实例应用

利用 Rational Rose 工具对文献[3]提出的一个关于图书管理系统的类图进行绘制. 在该类图中 Title 类是书库里的记录, 而 Item 类则是指具体的书籍, 在书库中每条记录都会有多本书存在, 所以 Title 与 Item 之间存在一对多的关系. Title 与 Reservation 之间也是一对多的关系, 也就是说 Title 可以有多个预订记录, 但是也可以没有预订记录. Item 与 Reservation 之间是一对一的关系, 不可能存在同一本书被两个人预订的情况. Borrower 与 Load 以及 Borrower 与 Reservation 之间也是一对多的关系^[4-5]. 如图 6 所示.

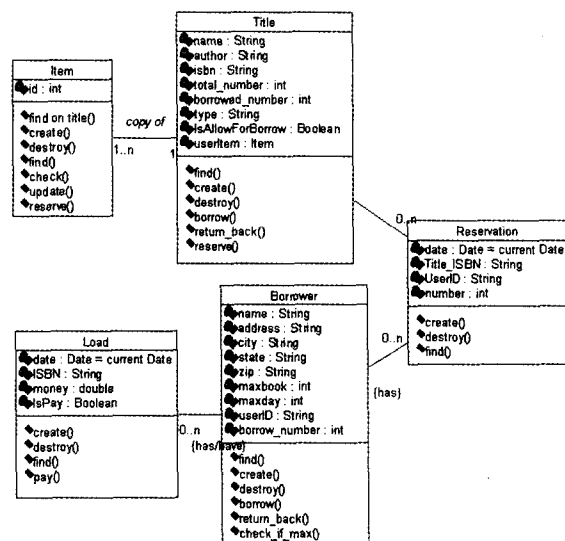


图 6 Rational Rose 工具下的类图表示

利用 VDMTOOLS 工具^[6-7], 调用 Windows 菜单中选择 Call Rose Link 子菜单, 弹出 Rose Link Windows 子窗口, 该窗口会自动调用 Rational Rose

中当前的模型窗口。选择 Map 按钮即可实现由窗口中显示的 UML 中设计的各个类向 VDM++ 转换,在转换的过程中对各个类的定义进行语法和语义等方面的检查,并给出相应的提示,同时转化为 VDMTOOLS 工具下的各个文件。

在利用 Rose-Link-Tools 工具对 UML 类^[8]的转换过程中,对数据类型都是直接转换而来,利用 VDMTOOLS 对文件进行类型、语法检查时,会报告一些错误。所以对相应的文件按照 VDM++ 的规则进行改进。改动后的文件形式如图 7 所示。

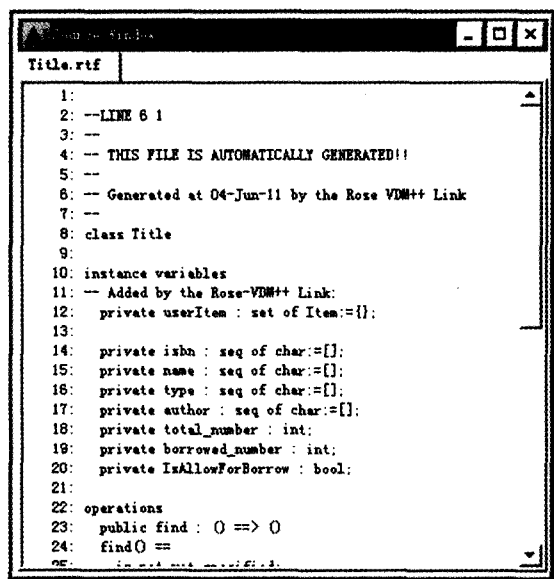


图7 VDMTOOLS 工具下对 Title. rtf 文件的修改

5 结束语

本文采用 VDM++ 方法对 UML 的类图所包含的各种元素进行了形式化描述。下一步的研究工作结合 VDMTOOLS 和 Rational Rose 工具对 UML 模型中静态和动态结构进行一致性验证和分析;把可视化建模和形式化分析结合起来;从而确保系统开发之前建立的系统模型的可靠性和完

备性。

参考文献:

- [1] Hans-erik Eriksson, Maguns Penker, Brian Lyons, et al. UML 2 toolkit[M]. 安萍余,俊平俞,译. 北京:电子工业出版社,2004:80-130.
- [2] 古天龙. 软件开发的形式化方法[M]. 北京:高等教育出版社,2005:202-234.
- [3] 蔡敏,徐慧慧,黄炳强. UML 基础与 Rose 建模教程[M]. 北京:人民邮电出版社. 2006:93-114.
- [4] 赵正旭,张常有,刘立嘉,等. 信息对象开放式采集系统的形式化描述[J]. 微电子学与计算机,2010,27(8):20-23.
- [5] 钟金琴,辜丽川,张友华,等. 设计模式的一种形式化描述方法[J]. 微电子学与计算机,2009,26(6):150-153.
- [6] Edwin MIT. Developing VDM++ operations from UML diagrams[D]. UK: University of Salford, 2008: 67-159.
- [7] Peter D Mosses. VDM semantics of programming languages: combinators and monads[J]. Formal Aspects of Computing, 2011,23(2):221-238.
- [8] 肖健宇,张德运,董皓,等. UML 状态机到 B 形式化规约的转换[J]. 微电子学与计算机,2005,22(8):80-84.

作者简介:



胡文生 男,(1969—),讲师.研究方向为软件可靠性。

赵明 男,(1962—),博士,教授,博士生导师。

杨剑峰 男,(1985—),博士研究生.研究方向为软件可靠性。

贾国荣 男,(1958—),教授.研究方向为电子电路。