

可视化UML模型验证环境研究

付茂沼¹, 王 悠²

(1. 中国民航飞行学院计算机学院 四川 广汉 618307; 2. 中国民航飞行学院航空工程学院 四川 广汉 618307)

【摘要】针对非形式化的描述可能导致的不明确和不一致问题, 提出并设计实现了一种基于OCL的、能对UML模型中类之间的关联及约束进行分析与验证的OUVE, 使得在编写代码之前就能很好地验证类之间的关联关系, 以及约束条件是否与实际的要求相符合, 降低了由于不明确和不一致导致的设计、编程的错误以及所要付出的修改代价。

关键词 对象约束语言; 规约; UML模型; 验证

中图分类号 TP311

文献标识码 A

doi:10.3969/j.issn.1001-0548.2010.02.029

Research on Visual UML Model Verification Environment

FU Mao-ming¹ and WANG You²

(1. Department of Computer, Civil Aviation Flight University of China Guanghan Sichuan 618307;

2. Aeronautical Engineering Institute, Civil Aviation Flight University of China Guanghan Sichuan 618307)

Abstract Aiming at the problem of inaccurate and discordant arithmetic caused by informal software specification, this paper puts forward an OUV (OCL-based UML verification environment) setting which is based on Object Constraint Language. OUV can verify whether the constraints and relationship among classes correspond with practical requirements before coding. Comparing with traditional method, OUV reduces the cost of modification caused by improper design and programming fault.

Key words object constraint language; specification; unified modeling language; verification

随着软件开发规模的不断扩大, 开发软件系统的复杂度也越来越高、周期也越来越长, 基于面向对象思想的建模技术成为解决复杂问题的关键。UML已经成为面向对象领域事实上的标准建模语言, 模型驱动在开发过程中扮演非常重要的角色。

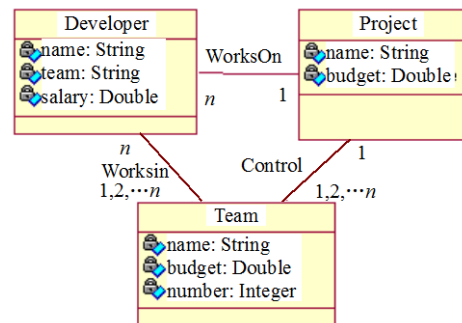
分析与设计是软件开发过程中的重要阶段, OMG的统一建模语言UML已经被广泛接受成为一种软件系统建模标准。UML支持面向对象的软件开发方法, 包含丰富的建模概念。UML的图形符号包括用例图、类图、顺序图、状态图以及活动图等9种。这些图形符号既可以用于描述系统的静态方面, 也可用于描述系统的动态方面^[1]。

软件建模曾经被视为一个图纸生产过程, 绝大多数模型是由许多方框、箭头和一些说明性的文本所组成, 所传达的信息不够完整、非正式、不够精确, 有时甚至自相矛盾。模型中的许多缺陷是因为所使用的图形表达能力有限而造成的。仅仅只有一个图时并不能表达一些条件约束。本文通过采用OCL规范对UML模型进行约束定义, 研究其分析验证环境, 以降低模型中的不明确和不一致导致的问

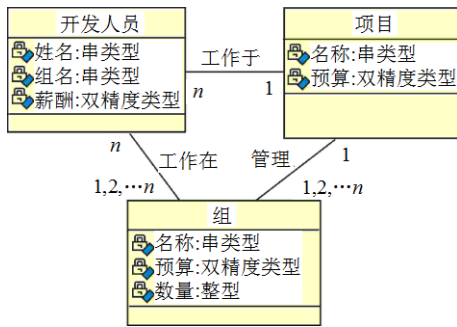
题, 以及由此造成的设计、编程错误。

1 非形式化的UML约束描述^[2]

UML类图如图1所示。图中包含项目(Project)、小组(Team)及开发人员(Developer)等3个类。其中, 每个项目(Project)可能需要1个或多个小组(Team), 以及多名开发人员(Developer); 每个小组(Team)可能需要配备若干名开发人员(Developer); 每名开发人员(Developer)可能属于不同的小组, 每个项目(Project)需要多名开发人员(Developer)。



a. UML类属性



b. UML类属性中文图

图1 UML类图

除了上述类之间的关联关系之外, 还有如下约束条件。

(1) 某一组内的开发人员数量应小于或等于项目的总开发人员数量;(2) 所有组员人数不超过项目总人数;(3) 开发人员数量越多, 组内的经费就越多;

(4) 每个组的经费都不能超过项目总经费;(5) 所有组的经费总和不能超过项目总经费;(6) 每名开发人员如果属于1个以上的组, 则其薪水大于只属于1个组开发人员。

设计人员使用常用建模工具Rational Rose、Borland Together或uml_designer(北航软件所自行研发的UML建模工具)建立模型, 并对约束以非形式化的语言进行描述, 但非形式化的描述很可能导致描述的不明确和不一致。如果描述的不明确和不一致导致设计、编程的错误, 将来的修改所要付出的代价就非常大。如果导致的错误没有被发现, 则影响程序的可靠和使用^[3]。

2 研究与实现

本文研究并设计、实现一种基于OCL的、能对UML模型中类之间的关联及约束进行分析与验证的环境OUVE。

可视化是指采用图形界面的方式对计算机系统或者软件进行描述表达, 可视化UML模型验证环境是一种能让用户在计算机上交互动地使用图形界面, 验证UML模型是否与实际要求相符合的软件平台。基于OCL的含义是指采用OCL规范对UML模型进行约束定义。

OCL作为一种表达式语言, 通过使用该语言可以在模型上编写表达式, 如属性的派生规则、查询操作体、不变式、先验条件及后置条件等。OCL可以用于UML和MOF模型。使用OCL可以扩展UML/MOF的表达力, 使创建的模型更精确、更详尽^[4]。

任何OCL表达式都会被计算成一个值。OCL是一种说明性的表达语言, OCL表达式描述值的属性, 但却不能规定如何计算该表达式。但是, OCL表达式可以被翻译成某种高级编程语言, 以便说明如何执行表达式^[5]。

在实际软件开发过程中, OCL能从以下4个方面提高软件质量^[6]。

(1) 分析阶段, 业务规则得以精确表达。通常情况下, OCL不变式约束于UML类图, 使其含义更精确、更完整。

(2) 设计阶段, OCL可被应用于类图中的操作, 约束既可以精确说明操作的先验条件, 又可以精确说明操作的后置条件, 为操作的实现提供精确的规约。

(3) 开发阶段, OCL可以作为简单的查询与浏览语言使用。

(4) 在UML模型中的其他图符(如状态图、活动图)中, OCL约束都可以用于精确说明。

3 形式化约束

首先需要将上述非形式化的描述转换为形式化的描述, 可采用对象约束语言OCL(object constraint language)形式化地说明UML模型上的约束^[7]。如上述6个约束条件的OCL表达如图2所示。

```
-- OCL constraints
constraints
context Team
  inv LessDeveloperThanProjects:
    self.developoer->size<=self.project->size
  inv MoreBudget:
    Team.allInstances->forAll(t1,t2|
      t1.developoer->size>t2.developoer->size
      implies t1.budget>t2.budget)
  inv LessBudgetThanProjects:
    self.budget<=self.budget
context Developer
  inv MoreSalary:
    Developer.allInstances->forAll(d1,d2|
      d1.team->size>e2.team->size
      implies d1.salary > d2.salary)
```

图2 基于约束条件的OCL表达

如果采用rose或together工具进行建模, 则可以将上述OCL附加至UML类图, 并通过工具的导出功能, 将类的属性、操作、类之间的关联关系、约束条件以及操作的先验和后置条件导出, 作为基于XMI标准的XML文件, 然后通过XMI技术获取模型信息, 如图3所示。

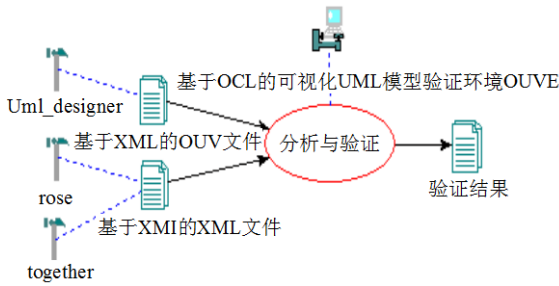


图3 基于OCL的可视化UML模型验证环境使用流程图

如果采用uml_designer工具进行建模,由于当前版本还不能直接导出XML文件,无法通过XMI技术获取模型的信息。为此,本文专为uml_designer等不能导出XML文件的工具定制一种规约文件OUV。该规约文件OUV也基于XML标准,其中包含模型名称、类名称、属性名称及类型、操作名称、参数及返回类型、类关联信息、约束条件等信息,根据使用uml_designer工具绘制的模型信息,可以手工创建相应模型的OUV文件^[8]。

3.1 体系结构

基于OCL的可视化UML模型验证环境体系结构如图4所示。

该结构分为3层,上层处理模型信息,模型信息经过解析后产生UML元素信息和OCL约束条

件信息;中间层主要处理系统数据实例化,为下层验证工作做准备;模型验证工作集中于下层,分为类之间关联关系的验证以及OCL约束条件的验证^[9]。

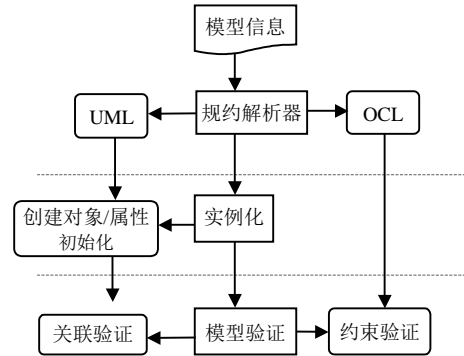


图4 基于OCL的可视化UML模型验证环境体系结构

3.2 技术方案及实现

基于OCL的可视化UML模型验证环境包含3个用例,分别是:

- (1) 描述对OUV规约文件或Rose/Together导出的XML文件检查的用例;
 - (2) 描述模型实例化的用例;
 - (3) 描述验证模型是否符合实际要求的用例。
- 用例图如图5所示。

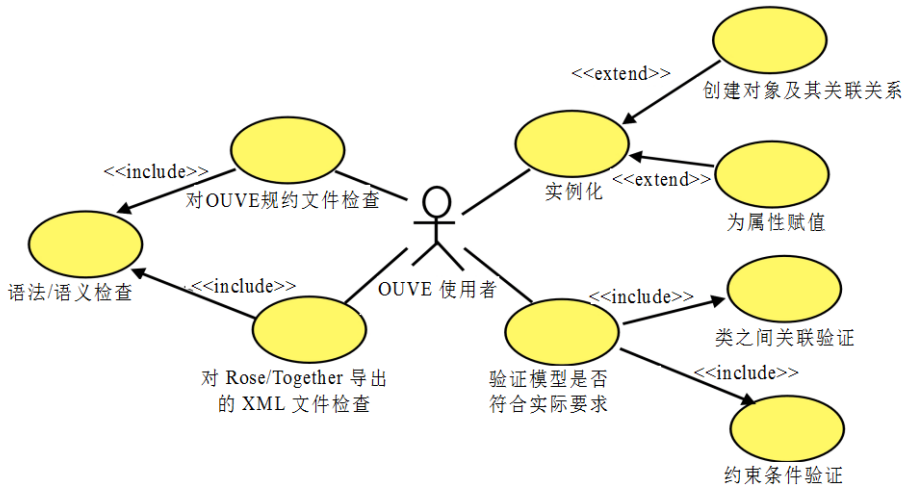


图5 用例图

技术方案如下。

(1) 使用工具ANTLR(ANother tool for language recognition)^[10],采用一个文本文件描述OUV规约。如果把ANTLR指向该文件,那么ANTLR将生成一个能够处理OUV规约的词法分析器(lexer)和解释器(interpreter)。词法分析器会把字符翻译成为语言符号,而解释器则利用语言符号完成任务。

(2) UML模型交换使用XML元数据交换(XML metadata interchange, XMI)技术,可以将XML文档直接映射为Java定义的对象,或与其他软件工具交

换这些对象,并且在XML应用程序中实现建模。

4 结论

尽管UML与早期的OMT对象方法技术相比,具有很强的功能,但它的语法与语义还不完善。也就是说,UML提供了一套丰富的概念和图形集,但UML在精确描述模型语义以及关联约束时仍存在问题。作为UML的一个重要部分,对象约束语言OCL可以使创建的模型更精确、更详尽。从软件开发全过程的角度来看,UML模型和OCL约束具有精

确的语义是非常重要的,这是因为精确描述UML模型的语义和OCL约束对于模型的分析、验证、转换、代码自动生成起着重要的作用。

通过使用上述基于OCL的可视化UML模型验证环境OUVE,可以在编写代码之前就很好地验证上述类之间的关联关系和约束条件是否与实际要求相符合,降低由于不明确和不一致导致的设计、编程错误,以及修改错误的代价。

基于上述手工生成的OUV文件规约,可在实现上加以改进,先生成文本文件,再自动读取,从而提高自动化程度,这也是要进一步研究的问题。

参 考 文 献

- [1] HEATON L. OMG unified modeling language specification[DB/OL]. [2003-03-01]. <http://www.omg.org>.
- [2] GRADY B, JAMES R, IVAR J. UML用户指南[M]. 邵维忠, 麻志毅, 张文娟, 等译. 北京: 机械工业出版社, 2001.
GRADY B, JAMES R, IVAR J. The unified modeling language user guide[M]. Translated by SHAO Wei-zhong, MA Zhi-yi, ZHANG Wen-juan, et al. Beijing: China Machine Press, 2001.
- [3] MELLOR S, BALCER M. Executable UML: a foundation for model-driven architecture[M]. Beijing: Science Press, 2003: 217-225.
- [4] NAYAK R, LEE B. Web service discovery with additional semantic and clustering[C]//Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence. Washington. D C, USA: IEEE Computer Society, 2007: 555-558.
- [5] Object Management Group, Inc. OCL 2.0 Specification[DB/OL]. [2006-05-01]. <http://www.omg.org/>.
- [6] 刘超, 张莉. 可视化面向对象建模技术: 标准建模语言UML教程[M]. 北京: 北京航空航天大学出版, 1999: 98-120.
LIU Chao, ZHANG Li. Visual object-oriented modeling—the unified modeling language tutorial[M]. Beijing: Beijing University of Aeronautics and Astronautics Press, 1999: 98-120.
- [7] JAMES R, IVAR J, GRADY B. UML参考手册[M]. 姚淑珍, 唐发根, 译. 北京: 机械工业出版社, 2001: 131-143.
JAMES R, IVAR J, GRADY B. Unified modeling language reference manual[M]. Translated by YAO Shu-zhen, TANG fa-geng. Beijing: China Machine Press, 2001: 131-143.
- [8] 高仲仪, 金茂忠. 编译原理及编译程序构造[M]. 北京: 北京航空航天大学出版社, 1990: 268-291.
GAO Zhong-yi, JIN Mao-zhong. Compiler: Principles and program constitution[M]. Beijing: Beijing University of Aeronautics and Astronautics Press, 1990: 268-291.
- [9] 田丽从, 周伯生. UML可视化建模工具中模型一致性检查机制的研究与实现[J]. 计算机应用与软件, 2005, 22(1): 24-26.
TIAN Li-cong, Zhou Bo-sheng. Research of model consistency checking mechanism in UML visual modeling tools[J]. Computer Applications and Software, 2005, 22(1): 24-26.
- [10] WARMER J, KLEPPE A. Octopus: OCL tool for precise UML specifications[DB/OL]. [2005-04-01]. <http://www.klasse.nl/octopus/index.html>.
- [4] BATTISTA G D, PATRIGNANI M, PIZZONIA M. Computing the types of the relationships between autonomous systems[C]//Proc of IEEE INFOCOM 2003, San Francisco, California: IEEE Press, 2003, 1: 156-165.
- [5] ERLEBACH T, HALL A, SCHANK T. Classifying customer-provider relationships in the Internet[R]. Technical Report TIK-Nr. 145, ETH, 2002.
- [6] MAO Z M, QIU L L, WANG J, et al. On AS-level path inference[C]//Proc of ACM SIGMETRICS, Canada, Banff: ACM Press, 2005, 339-349.
- [7] DIMITROPOULOS X, KRIOUKOV D, FOMENKOV M, et al. AS relationships: inference and validation[J]. SIGCOMM Computer Communication Review, 2007, 37(1): 29-40.
- [8] WANG Y X, WANG Y, CHEN M K, et al. Inter-domain access volume model: Ranking autonomous systems[C]//Proc of International Conference on Information Networking(ICOIN 2007), Estoril, Portugal: Springer, 2007, 482-491.
- [9] MEYER D. University of oregon routeviews project [EB/OL]. [2008-05-07]. <http://www.routeviews.org/>.
- [10] San Diego Supercomputer Center. Cooperative association for internet data analysis [EB/OL]. [2008-05-07]. <http://www.caida.org/>.

编辑 张俊

编辑 蒋晓

(上接第270页)