角色模型化:一个软件模式知识级的建模方法*)

Role Modeling: A Modeling Method for Software Pattern at Knowledge Level

何克清[1] 应 时[2] 魏 峻[3] 田中茂[4] 冈本泰次[4]

(武汉大学计算机科学系[1] 软件工程国家重点实验室[2] 武汉430072)

(中国科学院软件研究所 北京100080)[3]

(日本富士通株式会社 软件事业本部 MiddlewareSoft 事业部)[4]

Abstract Based on dominant degree of role model among the viewpoints for object oriented modeling process, this paper presents a graphic notation of role model, and dissertates that role modeling is a modeling method for software pattern at knowledge level. After gaving some examples for modeling analysis pattern and design pattern at knowledge level using role model, this paper presents the process for refining design pattern from role model to class model and event trace diagram of UML. In this paper, we advocate the opinion that role modeling before object modeling of UML. Through combining role model and UML, we construct a more comprehensive modeling method for object-oriented modeling process.

Keywords Object oriented, Role modeling, Software pattern, UML

近年来,软件模式已成为软件工程研究和应用的一个热点。1995年 Erich Gamma 等在名为"Design Patterns, Elements of Reusable Object-Oriented Software"[1]的专著中首先提出了设计模式的概念,并给出了23个典型的设计模式,文[1]在学术界和产业界都产生了很大的影响,极大地促进了近代软件工程的发展,它获得了"Software Development Productivity Awad"大奖,成为近30年来最优秀的软件工程著作。此后,Martin Forwler 于1998年在名为"Analysis Patterns: Reusable Object Models"[2]的专著中又提出了分析模式的概念,进一步为软件系统的分析提供了一些可重用的对象模型、系统分析经验和知识。Frank Buschmann 等于1996年在名为"Pattern-Oriented Software Architecture: A System of Patterns"[1]的专著中提出了软件体系结构模式的概念,同时提供了一些可重用的软件体系结构设计经验、方法及其模式。OMG(Object Management Group)提供的UML(Unified Modeling Language)[4]已被国际软件界广泛应用,为软件模式的结构描述和应用提供了有力的手段。但是,如何描述软件模式中所包含的经验和知识,为用户提供标识、认识和运用软件模式的有效方法还是一个没有真正解决的问题。

我们认为:软件模式所表达的是软件解决方案中可重用的经验和知识,因此软件模式的描述首先应该在知识级上进行,其后才是在结构级上进行。角色模型RM(Role Modeling)作为一个对象的概念性模型化方法,近年来已经在国际上引起了关注[5~8]。如何将RM技术与UML有机地结合起来,并将它们用于软件模式和软件框架的知识级描述中?这是一个重要的、有待深入研究的课题。本文在讨论了面向对象模型化视点中角色模型的主导作用后,首先给出了对象的角色、角色模型RM的可视化图式法,随后给出了描述设计模式和分析模式的角色模型,本文同时还提出了将RM和UML相结合的更为完全的描述体系,并给出了由设计模式的角色模型求精到UML类模型的一个实例。

1. 对象的角色模型

1.1 对象模型化的4个视点

构造对象模型必须根据使用者的立场和视点来认识对象。如图1所示,我们可以将这些视点归纳为角色视点、类视点、接口视点、数据-实体视点四个方面[8]。

*)本文研究得到日本国富士通公司软件事业本部 Middleware Soft. 事业部的委托研究项目、中国国家教育部科学技术重点项目的资助,何克清 教授,日本国论文博士、主要研究领域为面向对象的软件工程方法论、软件开发环境、软件模式和软件框架等。应 时 副教授,博士,主要研究方向:软件工程、组件重用等,魏 峻 博士、研究方向:分布对象计算,移动 agent 计算和软件形式化方法,田中茂、冈本泰次 富士通公司软件事业本部 Middleware Soft. 事业部负责人。

(1)角色视点:在对象系统中,对象间的协调行为是模型化首先要考虑的重要因素。在协调的对象模型中,一个对象所具有的角色必须与被协调对象的角色相匹配。因此,角色视点在面向对象模型中描述对象存在的理由,角色模型抽象地描述对象在系统协调行为中的作用。角色和类是两个不同的概念,角色从概念上描述了类的行为,一个角色可以由多个类来实现,一个类也可以实现多个角色。

(2)接口视点:描述对象在外观上的操作和行为的接口。

(3)类视点:描述对象如何抽象和实现。着眼于对象的属性和操作等结构规范的描述。

(4)数据-实体视点:视对象为数据-实体,描述实体及其之间的关系。

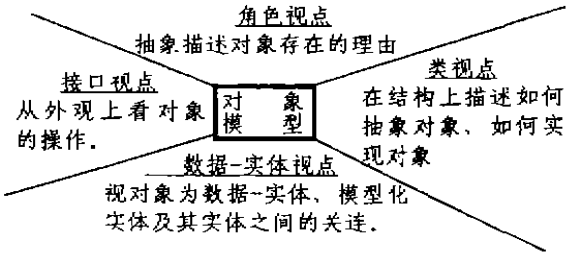

图1 对象模型化的4个视点

对象模型化的4个视点中,角色视点是对象抽象的、本质的概念描述。但是,我们在对象模型化时,一般只是注重于对象的类结构模型化、或物理的接口模型化,或实体模型化,往往容易忽视对象的角色模型化这一根本立场。基于RM识别对象及其协调关系,并展开到类模型、接口模型、实体模型是本文提倡的对象模型化方法。对于标识、描述和应用像软件模式这类可重用的软件知识和经验,这样做是更加必要的。

1.2 角色模型

角色模型用于描述角色对象及其相互间的协调行为。一个角色描述一个对象在系统行为中所承担的责任。角色对象描述一个或一个以上的责任。我们给出RM静态结构图的基本元素[5]如图2所示:(1)角色对象用圆角矩形表示,角色用附在圆角矩形上的黑点表示。如果角色只有一个,且角色名与角色对象名相同,则角色名可省略。(2)角色间的协调关联用直线表示、角色间的使用关系用带箭头的直线表示,角色间的继承关系用带黑箭头的直线表示、角色间协调的多重关系用数字表示。图2中所表示的是1个角色对象R1与1个或1个以上角色对象R2之间的协调关联。(3)角色约束的描述一般使用自然语言,随着角色的细化,也可使用OMG/OCL(Object Constraint Language)[4]进行描述。用同样的方法也可以描述RM中角色之间的约束关系。除静态结构图之外,角色模型还包括描述角色对象之间动态关系的行为协调图等[5]。图3给出了用角色模型描述Cash Withdrawal的事例[7]。

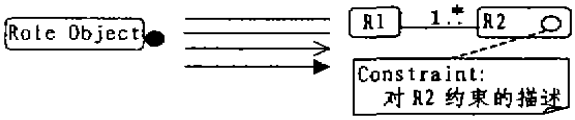

图2 RM的基本图元素

Cash Withdrawal:

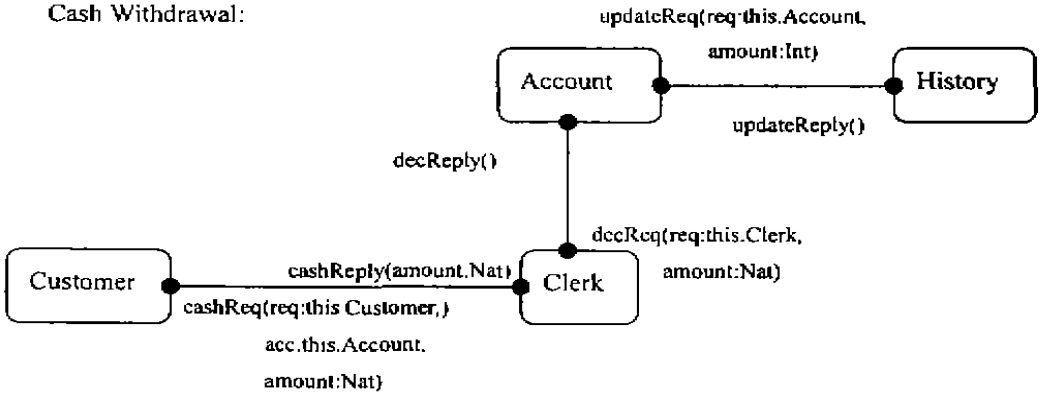

图3 "Cash Withdrawal"Role Modeling

```
Cash Withdrawal={Customer,Clerk,Account,History}
Cash Withdrawal. Customer
   = {Clerk, {cashReq (req: this. Customer, acc, this. Account,amount:Nat)}}
Cash Withdrawal. Clerk
   = {Account·{decReq(req·this. Clerk, amount, Nat), InspectBal()→Int}, //this is function for messages//
   Customer:{cashReply(amount·Nat)}}
Cash Withdrawal. Account
   = {History: {updateReq (req· this. Account, Amount: Nat)},
   Clerk·{decReply()}}
Cash Withdrawal. History={Account:{updateReply()}}
```

2. 设计模式的RM例子

文[1]给出了描述软件设计模式的模板,模板中包括的项目有:目的、别名、动机、可适用性、结构、构成要

素、协调关系、结果、实现、样本代码、使用例子、关联模式名等。我们认为模式的描述可分为知识级的描述和操作级的描述两个部分，前者包括模式的目的和动机的抽象描述，后者包括模式的可适用性，结构、构成要素、协调关系、实现等的具体描述，前者定义了后者。

依存管理模式的角色模型定义了二个对象间的依存关系，如果一个对象的状态发生了改变，应自动地通知与它具有依存关系的所有对象。Observer 设计模式[1]就是描述对象间依存关系的一个范例，图4所示的就是这个软件模式的 RM 描述。

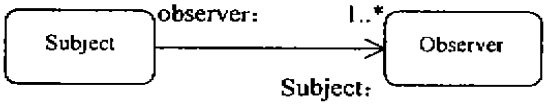

图4 依存管理模式的角色模型

结合应用的目标领域，图4的 RM 描述可分别求精为图5所示的 RM1、图6所示的 RM2、图7所示的 RM3。RM1应用于两个对象间依存状态的管理和维护。RM2定义事件通知模式中的知识，角色对象 Subject 定义状态空间角色对象 StateChange，角色对象 Observer 通过角色对象 EventPot 使用状态空间角色对象。通过确立 StateChange 和 EventPort 之间的通信协议，就可达到隔离角色对象 Subject 和角色对象 Observer 的目的，从而使角色对象 Subject 和角色对象 Observer 不需要关心它们之间通信的细节。实际上，RM2已被 CORBA 用于设计和实现事件通道，RM3定义了一种通知服务模式中的知识。角色对象 Subject 不是将事件通知直接发送到角色对象 Observer，而是发送到介于两者之间的通知服务角色对象，由通知服务角色对象按预先定义的方式处理事件队列，异步地发送事件。RM3同样也达到隔离角色对象 Subject 和角色对象 Observer 的目的。

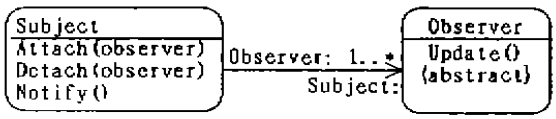

图5 Observer 设计模式的角色模型 RM1

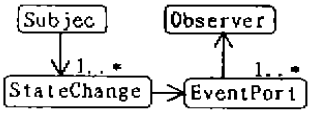

图6 事件通知模式的角色模型 RM2

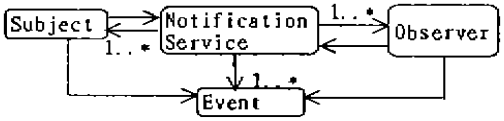

图7 通知服务模式的角色模型 RM3

在建立设计模式的角色模型之后，我们可以进一步使用 UML 及相应的工具，完成细化和实现设计模式的任务。图5所示的角色模型 RM1可以用 UML 方法细化（求精）为如图8所示的类结构。我们将图5所示的 RM1和图8所示的的类结构实际应用于设计和实现计时器管理，就可以得到如图9所示的类结构图、图10所示的消息追踪图。此后，借助 Rational Rose/UML 代码生成功能，就可以方便地得到该应用实例的源程序。

3. 分析模式的 RM 例子

我们认为：与设计模式一样，分析模式的描述也可分为知识级描述和操作级描述。下面我们以观测（observation）模式[2]为例，论述 RM 在分析模式的知识级

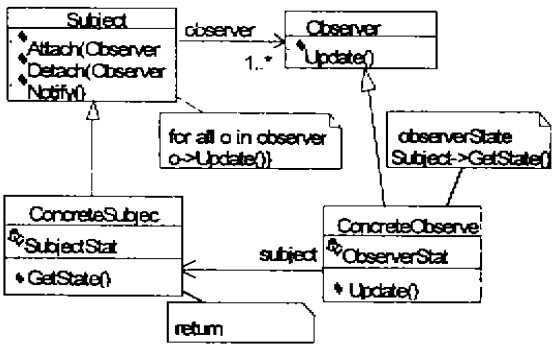

图8 Observer 模式的类结构图

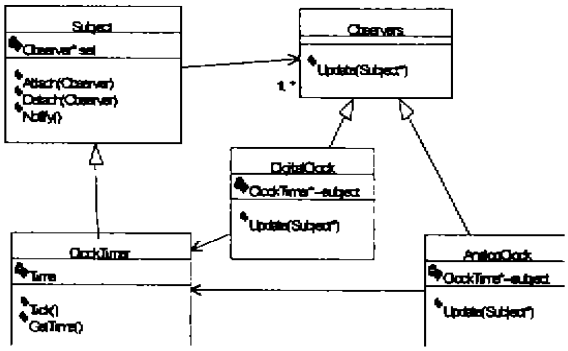

图9 计时器管理的类结构图

描述中的作用。所谓观测是人或事物以各种方式对现象的分析和测定。观测模式的主要知识包括观测的类型，观测的规程以及现象的类型等。如图11所示，我们在上半部分使用 RM 描述了观测模式的知识模型；在下半部分使用 UML 描述了观测模式的操作模型。

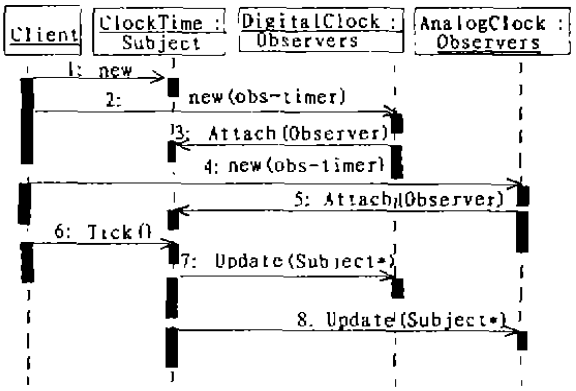

图10 计时器管理的消息追踪图

在操作模型中，Observation 类是抽象类，Measurement 和 Category Observation 是其子类。知识模型 RM 定义了观测模式的本质，决定了操作模型 UML 的框架。在本模式中，观测可以在任意级别上抽象，所以可以在观测模式的知识模型中导入类层次。如果一个观测类的所有观测子类都存在，则它们的观测父类存在。如果一个观测不存在，则它所有的观测子类都不存在。因此，在这个类层次中，存在关系是向上传播的，不存在关系是向下传播的。

4. RM 与 UML

目前 UML 已经在世界上广泛使用，成为面向对象模型化方法和可视化语言研究和应用的主流。在世界软件业界中，IBM，MicroSoft，SunSoft，Fujitsu，Hitach，NEC，TOSHIBA，…，等公司已在软件开发现场中使用。目前使用 UML 开发的组件软件、框架软件，以及应用软件产品已经进入市场流通。到1999年为止，

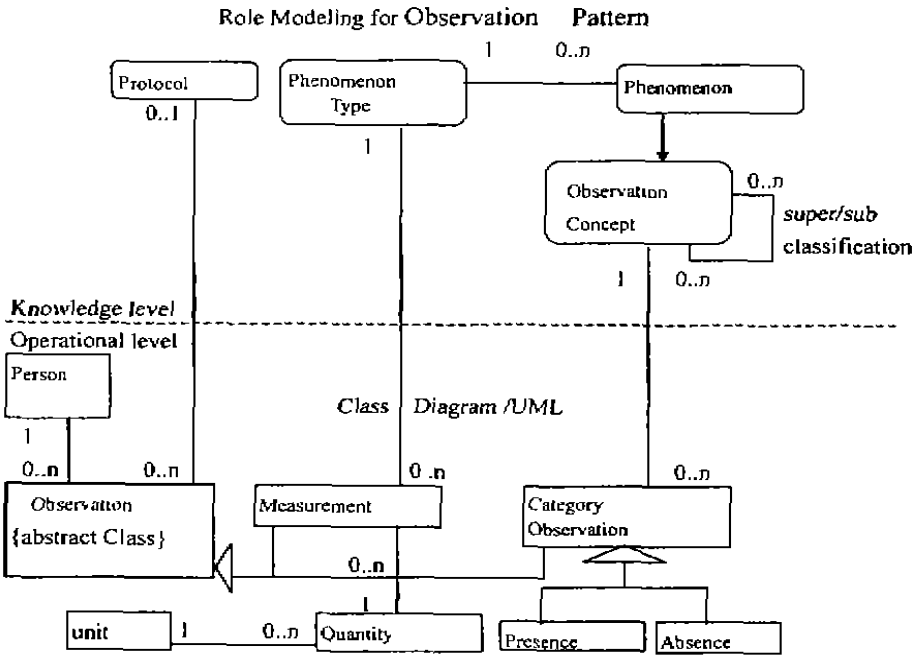

图11 观测模式的知识级和操作级描述

OMG 已发表了 UML1.4 版。目前，从 UML1.x 到 UML2.0扩充的研究和提案工作引人注目，但是，目前使用 UML 还限于软件工程的专业人员，在普及上还存在技术的困难。我们认为其主要原因在于 UML 本身主要是一个设计和实现对象模型结构的模型化语言，对于设计和实现需求规格说明已经明确的对象模型，UML 是十分有效的，但是 UML 在系统分析和概要设计方面能力比较弱。

本文针对 UML 描述能力的不足，提出了在使用 UML 描述方法之前，建立角色模型并构造角色对象类层次的基本思想。如图12所示，我们首先使用 RM，对角色对象及其相互间的协调关联进行分析和抽象。然后，面向应用领域，对角色对象进行类型化[9]，并定

（下转第80页）