角色建模语言及其工具的实现

Role Modeling Language and Implementation for its Tool

沈剑波　潘金贵

（南京大学软件新技术国家重点实验室　南京大学多媒体计算机研究所　南京210093）

Abstract　RML is a visual modeling language suitable for system requirement analysis and conceptual design. Some notation of RML is introduced here and together with the extension and mapping relationships from RML to UML. And the software designed for RML in application domain is also introduced

Keywords　Role modeling, Role modeling language, RML, UML

1. 引言

角色建模（Role Modeling，简作 RM）是一种基于角色抽象的面向对象建模方法，角色抽象不同于通常建模机制（如 OMT、Booch 方法等）所支持的类抽象，它用一个角色刻画一个特定的对象，角色具有可标识性和封装性；角色模型包含了一系列角色的集合，角色模型刻画对象交互作用的主题、对象间的关系、对象协作时向协作者传送的消息以及模型信息的处理过程。

这种抽象机制有如下几个特征：角色维持了对象的可标识性，角色模型刻画了相互协作的对象结构，这样的模型有利于我们研究系统的综合行为；RM 支持 Dijkstra 的"分而视之"原则，这一特点使它非常适用于复杂的系统建模，而其综合技术更使之具有强大的重用能力；角色的概念重视一个对象在系统内存在的理由、责任和位置，角色模型规定了对象的作用及其作用的协调关系（相比而言，统一建模语言 UML 则是规定了每个对象的能力和构造），因此 RM 适用于系统的需求分析和概要设计；RM 是在角色协作中对对象的语义进行描述的，它提供了对对象的一种新的抽象机制，这种描述方式对用户来说是有意义的，可以让用户成为信息系统的一个部分；RM 与模式技术具有很好的一致性；RM 中接口描述提供了 RM 抽象和类抽象的兼容。

正是因为这些特点，RM 受到越来越多的重视，欧美一些研究者如 Dirk Richle（德国）和 David Notkin（英国）正在研究一种新的建模语言规格说明 RML（Role Modeling Language）：在着眼于对象的角色之间关系的基础上描述业务模型的一种语言，用类似于 UML 的图来表达模型，适合于描述业务领域的分析模型。

RML 模型可涉及软件开发过程中的分析和设计阶段的概念模型构造，而当前大多数面向对象程序设计语言都是基于类抽象的，所以 RML 模型无法直接进入实现阶段；而 UML 在软件构造能力方面表现得很出色，然而对需求分析的能力还是有点不足。因此，将 RML 与 UML 结合起来进行企业复杂系统的分析建模直至完备的实现便成为当前迫切需要规范开发、降低成本并缩短开发周期的企业的一个新的尝试。

2. 角色建模语言 RML（Role Modeling Language）简介

2.1 RML 的建模元素

Role 的概念是以对象行为所扮演的角色来定义的，可以反映相关的行为（概念行为/责任/动作的特点），它从本质上描述了一个对象在系统中存在的理由，在 UML 的对象构造之前去分析对象的角色，是一种对对象的认识和抽象的自然的思维顺序。RML 的 Role 有两种：Behavior Role 和 Artifact Role（静态的无生命的，例如资源）。Behavior Role 必须和 Artifact（business entity）Role 协调好。Behavior Role 可有三种层次的划分：目标（Goal）、责任（Responsibility）和动作（Action）。相应的便有角色建模的三层视图：面向目标的角色建模（Goal Oriented RM）建立一个概念抽象（Conceptual Abstraction）的角色模型、面向责任的角色建模（Responsibility Oriented RM）建立精化的（Refined）概念抽象角色模型、和面向动作的角色建模（Action Oriented RM）建立实例对象协作（Instance Object Collaboration）角色模型。RML 中定义的 Role 的构成如图1(a)所示：Role 名（保持系统内身份可标

沈剑波　硕士研究生，主要研究方向为组件对象技术、可视化建模及应用。潘金贵　教授，主要研究方向为中间件、Agent 技术、多媒体远程教学系统等。

识性)、Role Type 名(应用程序领域保持身份可标识性)、抽象状态属性和抽象行为语义。在需求分析的最初阶段只能识别单个角色对象,所以可以只有 Role 名的定义,其他可以省略;而精化后的 refined Role 就需要详细定义《role type》、role name、State Attribute 和 Behavioral Semantics 了。

Role Type 是和 UML 中《Stereo Type Name》联系的桥梁,而 Type Model 则是元对象模型,是系统中可重用的对象模型。Role Type、Role Type Model 的例子分别如图1(b)和图1(c)所示。业务领域的分析过程中角色 Behavioral stereotypes(Behavioral Role types)包括 Information holders、Role names according to a Pattern's Role Object、Structures、Controllers、Coordinators、Service providers 和 Interface 等7种;而特别对于商业应用领域,以上的 Role types 可分成三个种类:《Requirement-Role》、《Service-Role》和《Interface-Role》。企业应用集成(EAI)中定义的基本类型有:Publishing、Subscription、Mapping、Augmentation、Correlation、Exception、Compensation、Fanout 和 Deadline 等。

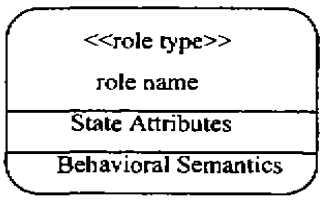

图 1 (a) Role 的构造

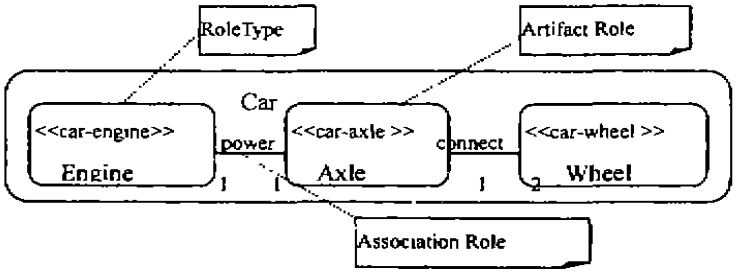

图 1 (b) Role Type 的例子

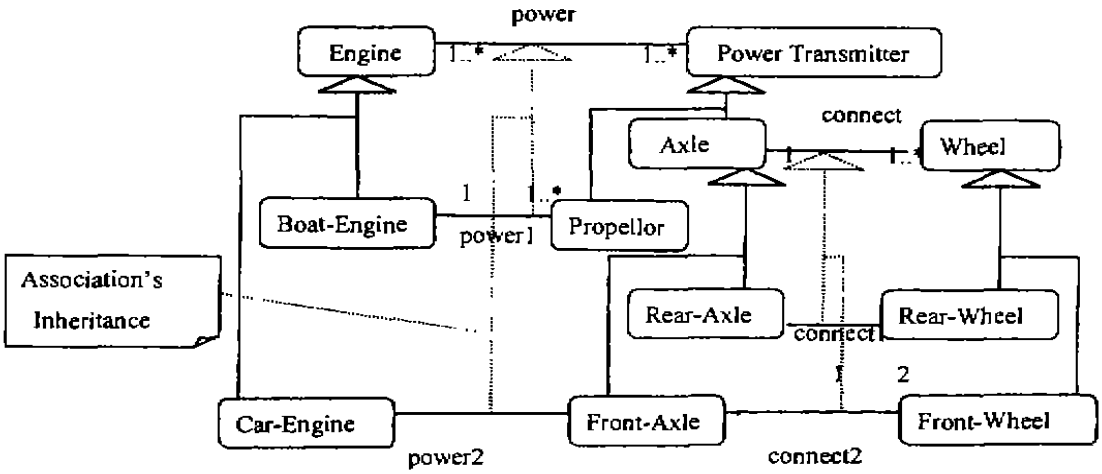

图 1 (c) Role Type Model 的例子

关联表示概念之间、概念的类型之间以及类型的实例之间的关系。如同 UML 中定义的关联表示对象之间或类之间的关系,RML 的关联表示 Role 之间或 Role type 之间的关系。关联的一个重要方面是多重度。

RML 的继承有两种:Role 的继承和关联的继承。根据 Brendan McCarthy 的关联的继承和合成一文[3],RML 中定义了9个关联继承的模式。

角色暗示(Implied Role)和角色相当(Role equivalent)是 RML 中定义以用于模式演化的。类似于 UML 定义的记号概念,RML 中也定义了包(Package)、集约(Aggregation)、交互(Interaction)、消息(Message)、注释(Note)等概念,另外还定义了 multi role-interface, Inaggregation 和 Association Aggregation 等概念。

RML 是可视化建模语言,如同 UML 一样用"图"来建立模型,表达 RML 模型的图可以有 Static Diagram(Role Diagram)、Interaction Diagram、Collaboration Diagram 和 Event Diagram 等。

类似 UML 的 Use Case,对于需求分析,RML 有 Use Role-Case Modeling。参照 Mark Collins-Cope 对 UML 的 Use Case Model 构造的改善提案 RSI 中提出的模式,日本富士通有关专家提出了适合角色建模的、针对商业应用领域需求分析的 Use Role-Case Pattern,并提议将这些模式作为今后商用领域的基本 Use Role-Case 模式。这些 Use Role-Case 分为三类:

《Requirement-Role》Case、《Service-Role》Case 和《Interface-Role》Case，这三个 Role-type 分类是 UML 的 stereotype 的扩展。

2.2 RML 和 UML

从角色模型过渡到 UML 模型需要经过精化，图2中以 Static Diagram 和 Collaboration Diagram 为例来说明精化的 RML 对 UML 的映射关系(图2)。

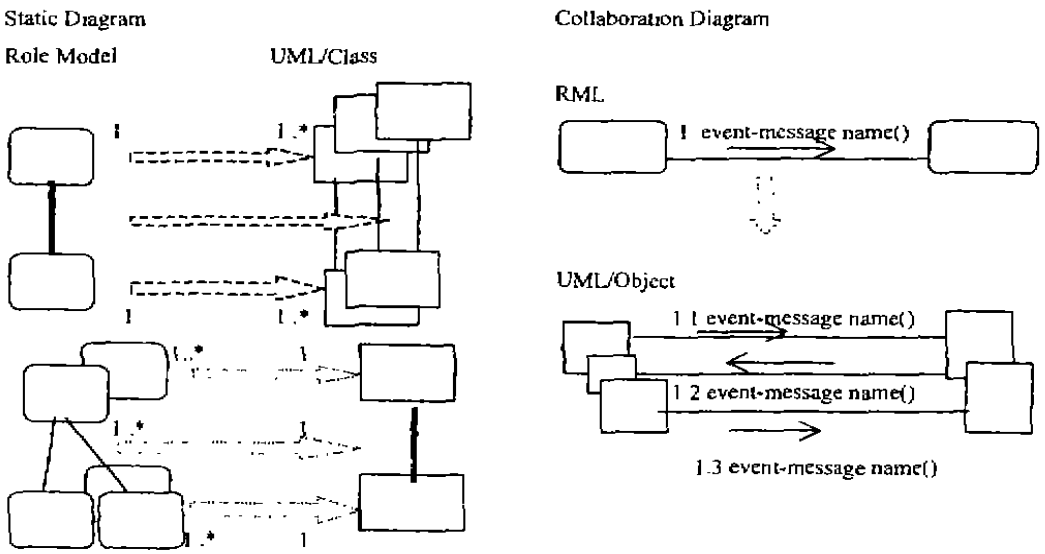

图2 RML 对 UML 的映射示例

以 RML 扩展 UML1.x 的 Meta-Model 从而对概念抽象建模、系统需求分析和概念设计有了一个更完善的对象建模规约；通过角色建模，对象的识别和分析有了更完整的语义；对 Role(Type) Object 和 Class 之间的关系有了更完备的规约；对对象和关联的精化、合成等有了更完整的语义；对角色建模的模式(Pattern)有更好的定义和使用；为描述业务过程增加了事务图；对企业应用集成(EAI)，增加了事务-消息图。RML (Vol.0.76)的扩展 UML 规约是以 UML1.3 draft R5 Semantics Meta-Model 为对象，对如下几处进行了扩展：UML Behavioral Elements Packages、Core Package-Classifier (eg.)、Core Package-Backbone、Core Package-Relationships、Extension Mechanisms、Collaborations、State Machines-Events、Activity Graphs，并增加了 Use Role-Case Package、Event Diagram Meta-Model、Event-Message Diagram for EAI。这里对具体的扩展规约不再细述。

以 RML 扩展 UML 而进行更为完善的对象建模，RML 中提出了对对象建模的三层模型规约，即 Role→Type→Class，如图3所示。

3 支持角色建模的工具 RMLEditor 的设计及实现

3.1 系统概况

在开放的网络环境中，企业信息系统的开发过渡到了以 business object(BO)为目标，从而实现降低成本和缩短开发周期。为建立以 business object 流通结构的国标，日本信息处理协会(IPA)主持的由包括富士通公司在内的四家公司参与的"Business Object 联合系统开发"项目即应用 RML 对企业资源的各部门(设计信息管理系统、销售管理系统和物流管理系统等)建立概念模型，然后将概念模型通过 UML 转换形成结构模型和实现模型，并作为"Business Object 分析设计平台"中的共用组件在日本全国范围内的各软件企业之间流通，组件的执行环境是在 CORBA 等分布式对象环境上构筑的 EJB，今后一个企业的各部门、乃至不同软件商和企业开发各自的应用程序(Business Object)时通过 XML 数据取得平台中的组件并对其进行重用，通过建立固有的逻辑来开发实用的程序。该系统包括设计信息管理系统、销售管理系统和物流管理系统以及系统间接口的开发，并且包括一个名为"BO 分析设计平台"部分的开发。在该系统的分析设计中，有必要在保持每一个 BO 内部构造隐蔽的基础上来对其任务和接口进行分析设计，所以"BO 分析设计平台"尝试采用了 RML 这种着眼于对象任务分析的语言，以 RML 和 UML 的结合来负责共用组件模型的分析设计。作为富士通公司的合作伙伴，我们参与了 IPA 的该项目的研究与开发工作，并在"BO 分析设计平台"中设计并开发了 RML 的可视化编辑工具 RMLEditor。

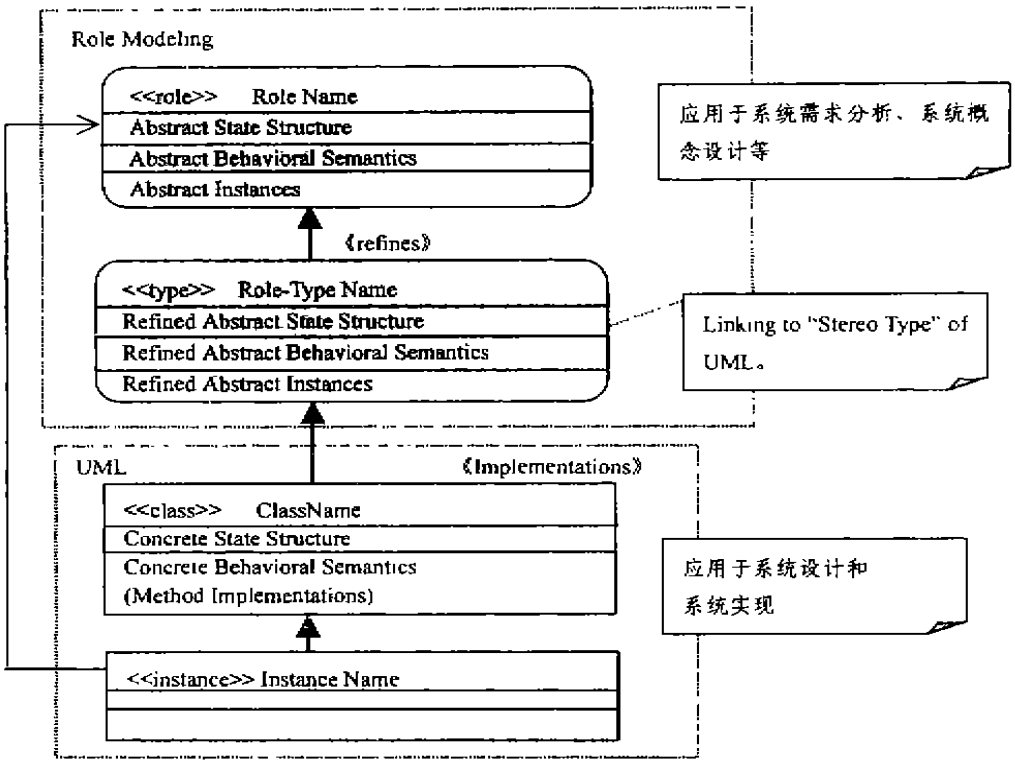

图3 完整的对象建模规约 Role→Type→Class Specification

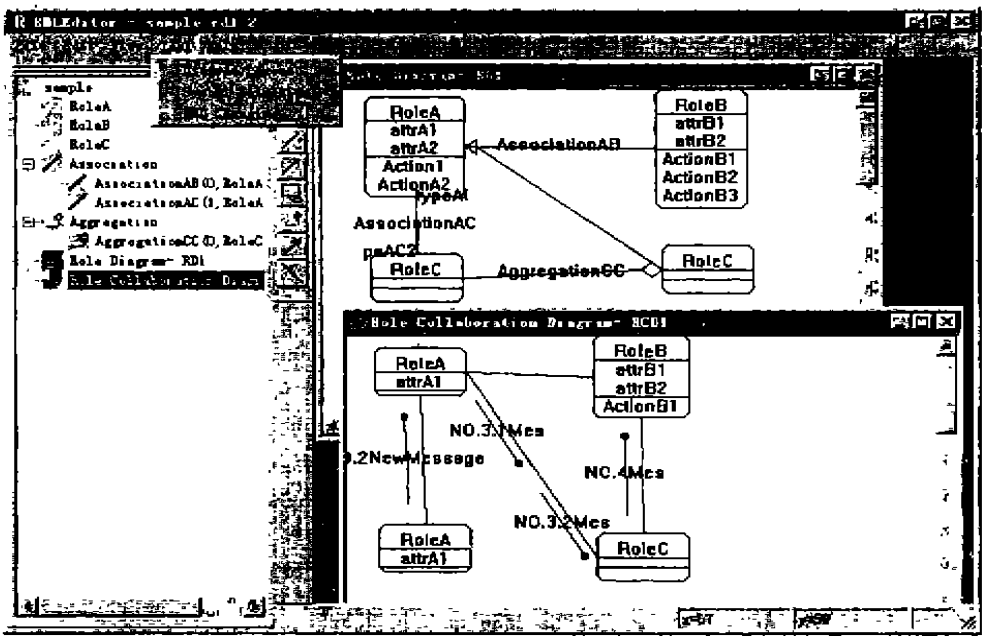

图4 可视化 RML 建模工具 RMLEditor 使用示例

3.2 RMLEditor 简介

RMLEditor 是为 RML 的一个实用子集提供的可视化编辑器。该子集针对本系统的应用领域即设计信息管理系统、销售管理系统和物流管理系统，采用了 RML Specification(vol.0.76)的一个子集，如模型图由两种图 RML Diagram 和 Collaboration Diagram 构

成，图式记号集合由 Role、Note、Association、Aggregation、Generalization、Interaction 和 Message 等构成，而去除了 Package、Actor、Multi Role-interface、Association Aggregation 概念等。为方便用户学习和使用，实现 RML 和 UML 在使用上的"一致性"，我们借鉴了最广为接受的 UML 编辑工具 Rational Rose 的设计风格，为用户提供了建立 RML 模型的可视化编辑器及工具，作为 RML 模型向 UML 模型转换的结果，我们设计直接调用 Rose'98或 Rose'2000作为 UML 工具，图4是用我们设计的 RMLEditor 工具做的一个简单例子。

3.3 RMLEditor 结构与功能介绍

该项目被设计为三个部分来实现：RML 工具部、模型控制部和模型管理部，图5表示了三个部分之间的关系，对 UML 的转换和对 UML 工具的调用由模型控制部实现，而对模型信息的存储管理由模型管理部完成。

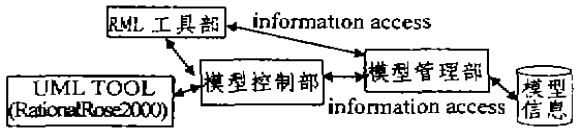

图5 RMLEditor 的结构

3.3.1 RML 工具部 是通过 RML 进行模型设计的工具，包括图式图管理功能、图式记号描画/操作功能和模型图管理功能，主要由从 MFC 继承来的 CRMLEditorView 及其子类、CMainFrm 等类和从 RMLViewObject 衍生来的如 RMLRoleView、RMLAssociationView 及 RMLDiagram 等视图处理类完成其功能。

图式图管理提供的功能有：调用图式记号描画/操作功能来进行图式记号的描画和操作；调用模型图管理功能来进行模型管理；进行和模型控制部之间的模型信息交换。

图式记号描画/操作提供的功能有：接受图式图管理功能的请求进行对图式记号的描画和操作；进行和模型图管理功能的信息交换以进行模型管理。

具体包括：RML 图式记号的描画、创建、属性编辑、删除、移动、复制、剪裁、粘贴和 Resize。

记号集合包括：Role、Association、Aggregation、Generalization、Note、Interaction 和 Message。对应每个记号元素，我们设计了相应其中对 Role、Association、Aggregation 以及 Generalization 的删除分为从图上删除以及从模型中删除。

模型图管理提供的功能有：接受来自图式图管理功能的模型管理请求；进行和图式图管理功能之间的模型信息交换；进行和图式记号描画/操作功能之间的图式信息交换；把用图式记号描述的 RML 模型作为模型图来管理；进行和模型控制部之间的模型信息交换；在模型管理数据库中进行存储管理。

3.3.2 模型控制部 主要由 RMLObject 类衍生的 RMLRole、RMLAssociation、RMLAggregation、RMLGeneralization、RMLNote、RMLInteraction、RMLMessage、RMLProject 和 RMLControl 等类构成。它包括模型控制功能和对外联合功能。

◇模型控制功能，包括：对 RML 模型信息的处理和保存、对 RML 模型语法和语义检查，以及将 RML 分析模型转换成 UML 设计模型。

◇对外联合功能，包括：调用 UML 工具(Rose)和对 UML 信息的访问。

3.3.3 模型管理部 包括项目管理功能和模型信息存取功能，主要由从 MFC 继承的 CRMLEditorDoc、CMainFrm 等类及 RMLProject 类等完成。项目是本系统中定义的用于 RML 向 UML 过渡的单位，它类似于 UML 的 Package 概念。

◇项目(Project)信息管理功能，包括：Project 浏览处理、新建 Project、删除 Project、编辑 Project、Project 打开和关闭以及保存处理。

◇模型信息存取功能，包括输入/检索系统功能和输出系统功能，即：打开模型处理、关闭模型处理、模型保存、Import 的模型信息获得、Import 模型和模型检索。

结束语 RML 语言是在着眼于对象的角色之间关系的基础上描述业务模型的一种语言，非常适合系统的需求分析和概要设计；RML 语言基于角色抽象，因此无法直接进入实现阶段，而统一建模语言 UML 虽在软件构造方面表现出色但对系统的需求分析尚嫌不足，因此将 RML 和 UML 结合起来进行优势互补是一个值得不断研究的课题。我们在实际应用中为 RML(vol.0.76)设计和开发的编辑工具 RMLEditor 具备了 RML 向 UML 简单结合的功能，目前在针对特定商用领域的"BO 联合系统"中测试良好，并被正式投入使用。

参考文献

1 McCarthy B. Association Inheritance and Composition. JOOP, 1997, 10(4)
2 Andersen E P. Conceptual Modeling of Objects. A Role Modeling Approach. [Dr Science theses]. Dept. of Informatics, University of Oslo. Nov. 1997
3 OMG Unified Modeling Language V1.3 Specification (draft). March 1999
4 IBM. Architectures for Enterprise Application Integration. Version 1.0, 1999
5 Keqing H. RML(Role Modeling Language) Specification. Version 0.76, Fujitsu Limited, Jan. 2000
6 Links C. 18,501 Links on Objects and Components/Business Objects http://www.cetus-links.org/oo-business-objects.html
7 Reenskaug T Working with Objects: A three-model architecture for the analysis of information systems. JOOP May 1997