协同系统体系结构模型的形式化语义

侯金奎[1,2]
(1. 潍坊学院计算机与通信工程学院,山东潍坊 261061; 2. 山东大学计算机科学与技术学院,山东济南 250101)

摘 要: 针对模型驱动的协同应用系统开发,将范畴理论、代数规范和进程代数相结合,为软件体系结构模型提出了一种新的语义描述方法. 该方法在构件规约描述的基础上,用态射表示构件之间的关系,态射类型蕴含了构件关系的不同语义,从而用类型范畴图表来描述软件体系结构模型,用函子描述体系结构模型之间的映射关系. 体系结构模型的形式化描述可用于判断一个转换是否满足某些特性或约束. 以一个协同编著系统为例说明了该方法的应用.

关键词: 模型驱动开发; 协同系统; 软件体系结构; 形式化语义
中图分类号: TP311 **文献标识码**: A **文章编号**: 0372-2112 (2009) 4A-106-06

Formal Semantics of Architecture Model of Collaborative Systems

HOU Jin-kui[1,2]
(1. *School of Computer and Communication Engineering, Weifang University, Weifang, Shandong* 261061, *China*;
2. *School of Computer Science and Technology, Shandong University, Jinan, Shandong* 250101, *China*)

Abstract: Focusing on model-driven development for collaborative systems, a new description approach for the formal semantics of architecture models is proposed by combining category theory with algebraic specification and process algebra. On the basis of component specification, morphisms are used to describe the relationships between components, and the morphism types imply the different semantics of component relations. Thus architecture models are described within typed category diagrams, and functors are used to describe the mapping relations between different levels of models. The formal approach can be used to judge whether a transformation satisfies some property preservation constraints or not. A collaborative editing system is given as a case to illustrate the application of this approach.

Key words: model-driven development; collaborative system; software architecture; formal semantics

1 引言

CSCW(Computer Supported Cooperative Work)系统庞大的需求和复杂的设计开发过程对系统开发者而言是一个巨大的挑战,诸如多用户、分布的界面、协同感知、协作等特征,使得协同系统的设计和开发非常的困难[1]. 模型驱动开发(Model-Driven Development, MDD)已成为软件工程技术的研究热点和发展趋势[2],它通过提升抽象层次来应对软件开发的复杂性. 采用构件化技术,并在软件体系结构(Software Architecture, SA)的指导下对系统进行分析、验证、实现和维护是保证复杂系统开发效率和质量的一种有效方法[3]. 将构件化技术和 MDD 方法应用于 CSCW 系统开发中,可充分借助于这两项技术的优势,降低 CSCW 系统开发的难度. 开发人员可充分复用已存在的各关键技术部分的代码或构件,提高系统的开发效率和质量,以及可重用性、可伸缩性和灵活性等.

收稿日期:2008-09-02;修回日期:2008-12-26
基金项目:山东省科技攻关项目(No. 2008GG10001026)

模型转换是 MDD 中的一项关键技术,目标代码或具体模型可以由抽象模型经过一系列的转换来生成,每一步转换都添加了层次细节. 语义一致性是模型转换正确性的一个重要标准[4],但目前模型转换中语义特性保持的定义、描述和验证仍是一个尚未解决的难题[5,6]. 本文针对协同应用系统开发,将范畴理论[7]、代数规范和进程代数[8]相结合,为软件体系结构模型建立了一种形式化的语义描述方法,可用于判断一个转换是否满足某些特性或约束,从而为模型驱动的协同系统开发提供支持.

2 体系结构模型

CSCW 系统中,每个协同应用处理特定的协同任务,并以特定的协作模式支持小组共同完成协作任务[9]. 本文中,各协同应用及其支持平台的分析和设计都采用面向构件的方法,系统的各功能都以构件的形式进行封装,构件间通过消息进行通信. 用户通过代理构

件访问系统,用户间的交互在系统中表现为各代理构件的交互. 遵循 MDD 的思想,在对系统进行分析设计时,可从问题空间出发,独立于具体的实现平台和环境,定义构件规约的功能与接口以及它们间的关系和通信机制,然后由这些抽象的构件描述,根据具体平台和环境的特点,通过一系列的模型转换得到特定平台的构件实现. 本文以范畴理论作为体系结构模型语义描述的基本框架,有关的基础知识参见文献[7].

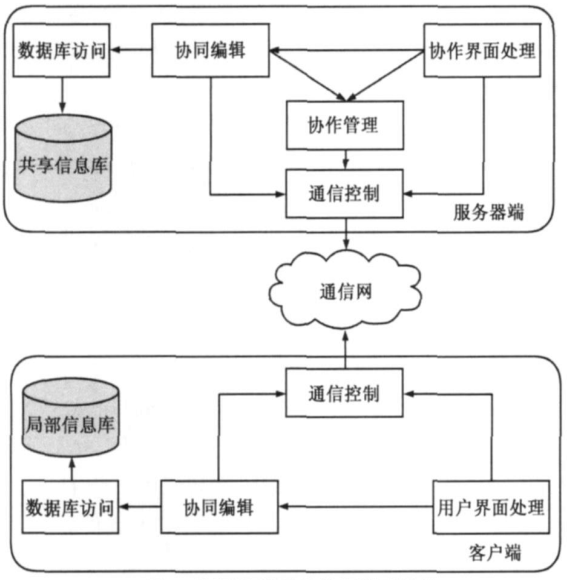


图1　协同编著系统的源模型结构

图 1 描述了一个基于客户机-服务器的协同编著系统的基本结构,该实例将贯穿全文来说明相关的概念和模型. 在高层模型体系结构描述中,有一个协同管理中心(服务器)和多个协同客户端,各组成构件功能如下:(1)通信控制构件提供对信息传输的支持及数据的加密和解密,是协同工作的基础;(2)协同编辑构件负责编辑系统的管理及维护,用户的认证及权限分配,文档的存取控制、锁定及编辑,用户动作事件的协同感知等;(3)协作管理构件负责协调各成员的协同工作,包括会话的发起与维护,各协同应用的管理和拆卸,以及各协同应用中操作的协调;(4)数据库访问构件负责信息的访问和存取;(5)协作界面处理构件负责协作信息的显示、操作响应、信息一致性维护等功能;(6)用户界面处理构件负责本地消息的处理和信息更新以及共享界面显示的更新等.

2.1　构件规约描述

CSCW 系统模型描述中,各运行部件的行为是时间相关性很强的时序行为,并且其关系本质上是并行的,为此我们采用进程代数(Process Algebra, PA)[8]来形式化描述构件的对外行为协议.

定义 1　构件规约(Component specification)

是一个八元组 $CP = (Cid, A, \Gamma, fa, fp, BP, \Phi, I)$,其中

(1) Cid 为构件的惟一标志;

(2) A 是构件的属性符号的集合;

(3) Γ 是构件的端口标识符号的集合;

(4) fa 是一组用于描述属性的全函数集合;

(5) fp 是一组用于描述端口的全函数集合;

(6) BP 是表示构件外部行为的 PA 描述;

(7) Φ 是代表构件的功能目标和非功能目标的公式集合;

(8) I 对每个端口 $p \in \Gamma$ 指派一个作为卫条件的公式集合,表示实现构件目标所应满足的条件和约束,如协同的子构件之间的协作策略和控制规则,行为约束等.

```
Component UIManager
 Attributes
      Cid: String;//构件的唯一标志
      Private CompStatus: StatusType; //构件所处的状态
      Private UIEventSet: Collection; //用户界面事件集合
      …
 Ports
      In PT1{
      Boolean collectEventInfo(); //收集用户界面事件
      }
      In/Out PT2{
      Boolean sendShareEvent(event:EventType);
                                    //发送共享操作事件信息
      Boolean receiveUpdateInfo(message:InfoType);
                                    //接收共享界面处理信息
      }
      In/Out PT3{
      Boolean sendLocalEvent (event:EventType);
                                    //发送本地操作事件信息
      Boolean receiveLocalInfo(message:InfoType);
                                    //接收本地界面更新信息
      }
      …
 Axioms
      (collectEventInfo()==True)⇒(sendShareEvent() ∨ sendLocalEvent());
      receiveLocalInfo(message)⇒Redisplay();
      receiveUpdateInfo(message)⇒Redisplay();
      …
 Behavior
      BP_UI ≜ clEvtInfo.(sdShrEvt.BP_UI+sdLocEvt.BP_UI) + rcShrIf.rdsp.BP_UI
+rcLocIf.rdsp.BP_UI,
 …
```

图2　用户界面处理构件UIManager的规约描述

图 1 中客户端用户界面处理构件 UI Manager 的规约描述 $CP_{UI} = (Cid_{UI}, A_{UI}, \Gamma_{UI}, fa_{UI}, fp_{UI}, BP_{UI}, \Phi_{UI}, I_{UI})$ 如图 2 所示,其中 Cid_{UI} 为构件的惟一标志;$A_{UI} = \{UIEventSet, CompStatus\}$;$\Gamma_{UI} = \{PT1, PT2, PT3\}$;$fa_{UI}$ 描述了属性的类型等信息,此处定义包括:$type(UIEventSet) = Collection$, $type(CompStatus) = StatusType$;$fp_{UI}$ 描述了端口的类型、所接受的消息类型等信息,此处定义包括:

$type(PT1) = In$, $type(PT2) = In/Out$,

$type(PT3) = In/Out$, $MessageType(PT1) = \{\}$,

$MessageType(PT2) = \{EventType, InfoType\}$,

$MessageType(PT3) = \{EventType, InfoType\}$;

Φ_{UI}, I_{UI} 如公理(Axiom)部分所述;外部行为描述 BP_{UI} 中,各事件定义如下:

$clEvtInfo$:收集用户界面发生的事件;

sdShrEvt:向通信控制构件发送共享操作事件信息;
sdLocEvt:向协同编辑构件发送本地操作事件信息;
rcShrIf:从通信控制构件接收共享界面处理信息;
rcLocIf:从协同编辑构件接收本地界面更新信息;
rdsp:更新用户界面显示.

2.2 构件规约态射

构件交互机制是由构件的模型或平台所提供和决定的,特定构件间的交互受到构件类型和关系机制的约束.构件交互关系的类型代表了构件通信、协同和仲裁决策的一种抽象,刻画了构件交互的协议.本文用范畴图表中的态射来描述构件规约之间的关系,态射类型蕴含了这些关系的不同语义.

定义 2 构件规约态射(Component specification morphism)

给定两个构件规约 $CP_1 = \langle Cid_1, A_1, \Gamma_1, fa_1, fp_1, BP_1, \Phi_1, I_1\rangle$ 和 $CP_2 = \langle Cid_2, A_2, \Gamma_2, fa_2, fp_2, BP_2, \Phi_2, I_2\rangle$,从 CP_1 到 CP_2 的态射 $\omega: CP_1 \to CP_2$ 包括:

(1) 一个属性映射 $\omega_{at}: A_1 \to A_2$,满足 $\omega_{at}(A_1) \subseteq A_2$,并且 $\exists a \in A_1$,有 $fa_1(a) = {}_D fa_2(\omega_{at}(a))$. $= {}_D$ 表示特性描述一致;

(2) 一个端口映射 $\omega_{ac}: \Gamma_1 \to \Gamma_2$,满足 $\omega_{ac}(\Gamma_1) \subseteq \Gamma_2$,并且 $\exists p \in \Gamma_1$,有 $fp_1(p) = {}_D fp_2(\omega_{ac}(p))$;

(3) 一个行为描述映射 $\omega_{BP}: BP_1 \to BP_2$;

(4) $\exists q \in \Phi_1, \omega(q) \in \Phi_2$;

(5) $\exists p_1 \in \Gamma_1, I_2(\omega(p_1)) \supseteq \omega(I_1(p_1))$.

条件(4)说明构件规约中功能目标和非功能目标的保持;条件(5)允许强化卫条件,但不能弱化.

2.3 体系结构范畴

陆汝钤院士针对知识的抽象描述和知识处理,组合范畴理论的探讨能力和领域知识构造和推理的能力,提出了类型范畴的概念和理论[10].类型范畴在范畴理论的基础上增加了表达和推理能力,但没有违背范畴理论的基本框架[10].我们将陆汝钤院士的类型范畴理论作进一步的扩展,为对象和态射都定义类型,且每一种类型都可以定义一系列的属性.于是可以将体系结构概念和类型范畴概念做一个一一映射:构件实例↔对象,构件关系实例↔对象态射,构件规约↔对象类型,构件规约之间的关系↔态射类型,构件的属性↔对象类型属性,构件关联属性↔态射类型属性,那么一个软件体系结构模型就可以表示成为一个类型范畴,称之为体系结构范畴.

定义 3 体系结构范畴(Architecture category)

体系结构范畴是一个五元组 $AM = \langle CO, CR, CT, RT, RuleS\rangle$,其中:$CO$ 是由作为对象的构件实例组成的集合,CR 是由作为对象态射的构件关系实例构成的集合,CT 是构件类型描述规约的集合,RT 是作为构件关系类型的规约态射集合,$RuleS$ 是构件间关系类型合成机制的集合,除满足范畴定义的基本条件[7]外,还要符合以下特性:

(1) $CO = \{o_i \mid 1 \leq i \leq n, sort(o_i) \in CT\}$;

(2) $CR = \{r_j \mid 0 \leq j \leq m, \exists a, b \in CO, r_j = (a, b, t), t = sort(r_j) \in RT\}$;

(3) $RuleS: RT \times RT \to RT$,对任意 $(t, s) \in dom(RuleS)$,类型 $w = t \times s$ 称作是 t, s 的合成类型,并且对给定的 $r_i, r_j \in CR, r_i = (a, b, t), r_j = (b, c, s)$,存在态射的复合 $r_k = (a, c, w) = r_j \overset{\circ}{} r_i \in CR$;

(4) 对任一 $a \in CO$,有单位态射 $r_a = (a, a, u), u = sort(r_a) \in RT$,并且对任一 $t \in RT$,有 $u \times t = t \times u = t$;

(5) 对任意 $r_i = (a, b, t), r_j = (b, c, s), r_k = (c, d, q)$,有 $r_k \overset{\circ}{} (r_j \overset{\circ}{} r_i) = (a, d, (t \times s) \times q) = (a, d, t \times (s \times q)) = (r_k \overset{\circ}{} r_j) \overset{\circ}{} r_i$;

(6) 对任一 $r_i = (a, b, t) \in CR$,总有 $r_i \overset{\circ}{} r_a = r_b \overset{\circ}{} r_i = r_i$,其中 $r_a = (a, a, u), r_b = (b, b, u)$.

定义 4 子结构范畴(Sub-architecture category)

一个体系结构范畴 $AM_1 = \langle CO_1, CR_1, CT_1, RT_1, RuleS_1\rangle$ 称作是另一个体系结构范畴 $AM_2 = \langle CO_2, CR_2, CT_2, RT_2, RuleS_2\rangle$ 的子范畴,必须符合以下条件:

(1) CO_1 是 CO_2 的子集;

(2) CT_1 是 CT_2 的子集;

(3) RT_1 是 RT_2 的子集;

(4) 对任何 $a, b \in CO_1, t \in RT_1$,若 $(a, b, t) \in CR_1$,则必有 $(a, b, t) \in CR_2$.

另外,如果还符合下面的条件,则 AM_1 称作是 AM_2 的满子结构范畴:

(5) 对任何 $a, b \in CO_2, t \in RT_2$,若 $(a, b, t) \in CR_2$,则必有 $(a, b, t) \in CR_1$.

一个复合构件可描述为由构件规约和规约态射构成的范畴图表.体系结构模型中构件规约之间的态射大致可分为交互态射和组合态射两大类:交互态射用于描述构件间的各种交互和依赖关系,它确定了一个构件提供的服务如何与系统中的其它服务组合在一起;组合态射作为结构保持的映射,建立了两个构件描述之间的组合关系,其中一个可以看作是另一个的子构件.

每个构件实例的行为阐述都可由该构件规约的行为描述到其交互的投影来定义,这样的投影行为可通过对相应的 *PA* 术语序列应用隐藏算子[8]来描述,从而复合构件的行为语义就可以由其结构配置和子构件的行为计算得出.这样,一个体系结构模型就是由构件规约和构件规约态射及其实例构成的类型范畴,其语义是系统各个组成构件依据交互协议协同执行而得到的

整体抽象.

2.4 体系结构映射函子

模型驱动开发过程中,较高抽象层次的构件规约描述到低抽象层的构件规约描述之间的对应转换关系,同样用范畴理论的态射来表示,称为 mapping 态射.同一系统不同抽象层次的两个体系结构模型间的映射关系可用范畴理论中的函子(Functor)[7]来描述.

定义 5　体系结构映射函子(Architecture mapping functor)

从体系结构范畴 $AM_1 = \langle CO_1, CR_1, CT_1, RT_1, RuleS_1\rangle$ 到 $AM_2 = \langle CO_2, CR_2, CT_2, RT_2, RuleS_2\rangle$ 的结构映射函子 $Fu: AM_1 \rightarrow AM_2$,是满足以下条件的映射:

(1) 对所有的 $cs \in CO_1$,都有 $Fu(cs) \in CO_2$;

(2) 对所有的 $cp \in CT_1$,都有 $Fu(cp) \in CT_2$;

(3) Fu 是从 RT_1 到 RT_2 的同态映射,并满足以下特性:

(a) 对所有的 $t \in RT_1$,都有 $Fu(t) \in RT_2$;

(b) 对所有的单位态射类型 $u \in RT_1$,有 $u = Fu(u) \in RT_2$;

(c) 对所的 $t, s \in RT_1$,都有 $Fu(t \times s) = Fu(t) \times Fu(s)$;

(4) Fu 是从 CR_1 到 CR_2 的同态映射,并满足以下特性:

(a) 对所有的 $cs, cs' \in CO_1, (cs, cs', t) \in CR_1$ 都意味着 $(Fu(cs), Fu(cs'), Fu(t)) \in CR_2$;

(b) 对单位态射 $r_a = (a, a, u) \in CR_1, u = sort(r_a) \in RT_1$,有 $Fu(r_a) = (Fu(a), Fu(a), Fu(u)) = (Fu(a), Fu(a), u) \in CR_2$;

(c) $Fu(f \circ g) = Fu(f) \circ Fu(g)$,若 $f, g \in CR_1$ 且定义了 $f \circ g$.

命题 1　对每个体系结构映射函子 $Fu: AM_1 \rightarrow AM_2$, $Fu(AM_1)$ 是 AM_2 的子范畴.

证明　由定义 4 和定义 5,易知该命题成立.

软件体系结构模型的形式化描述可用于判断一个转换是否满足某些特性或约束,这将在下一节通过实例进行阐述.体系结构映射函子实质上是模型之间转换关系的描述,它保持了源模型的设计决策和语义特性.从而,模型驱动开发可以看成是由多层抽象程度不同的 SA 模型形成的体系结构空间,开发过程则是一系列以体系结构为中心的模型转换.

3 实例研究

我们仍用图 1 所示的协同编著系统来阐述本文所提理论和方法的应用.假定经过转换后的目标模型结构如图 3 所示:在服务器端,协同编辑功能由系统管理子构件、协同感知子构件和锁定及编辑子构件三者组合而成的复合构件实现;源模型中协作界面处理构件的功能由全局消息处理构件和全局显示更新构件组合

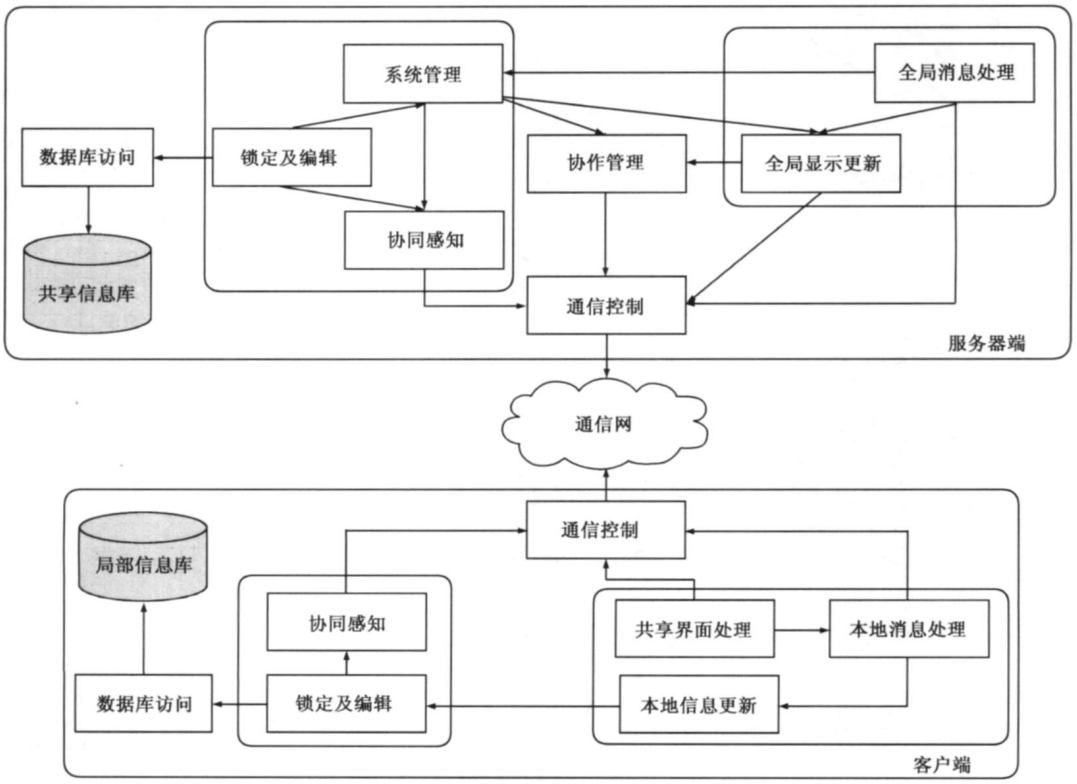


图3　协同编著系统的目标模型结构

实现;在客户端,协同编辑功能由协同感知子构件和锁定及编辑子构件两者组合实现;源模型中用户界面处理构件的功能由共享界面处理构件、本地消息处理构件和本地信息更新构件组合实现.

在这个模型描述中,系统管理构件负责编辑系统的管理及维护,处理来自用户和操作系统的各项命令和消息,管理用户身份认证,权限分配,当有新成员加入或退出时,随时将系统中的成员变化反映给各个客户端.锁定及编辑构件定义了存取权限,用户操作前的文档锁定功能,并提供编辑环境.协同感知模块定义了协同感知权限,描述用户动作被多个参与者感知的能力.当协作成员对某共享对象进行操作时,首先由本结点的本地消息处理构件截获该事件,并判断是否为共享操作.当操作的是本地的私用对象或信息时,消息处理构件就将此消息转发给本地信息更新构件,由其负责进一步的响应.否则,将该消息经通信控制模块发送至服务器的全局消息处理构件.全局消息处理构件负责系统各结点所有共享操作的响应和处理,它及时地从消息队列中取出消息,并根据消息的类型判断其是否需要访问共享信息库.若该共享操作将影响共享信息,则将消息转发给系统管理构件,并将结果信息发送给全局显示更新构件.当共享操作不影响共享信息时,全局消息处理构件直接将此消息转发至全局显示更新构件,由后者将此共享操作的消息广播发送至各个协作结点.最后由各协作结点的共享界面处理构件响应此消息,从而使得该共享操作为各结点的协作成员可见.

系统源模型 AM_S 的范畴图如图 4 左半部分所示,

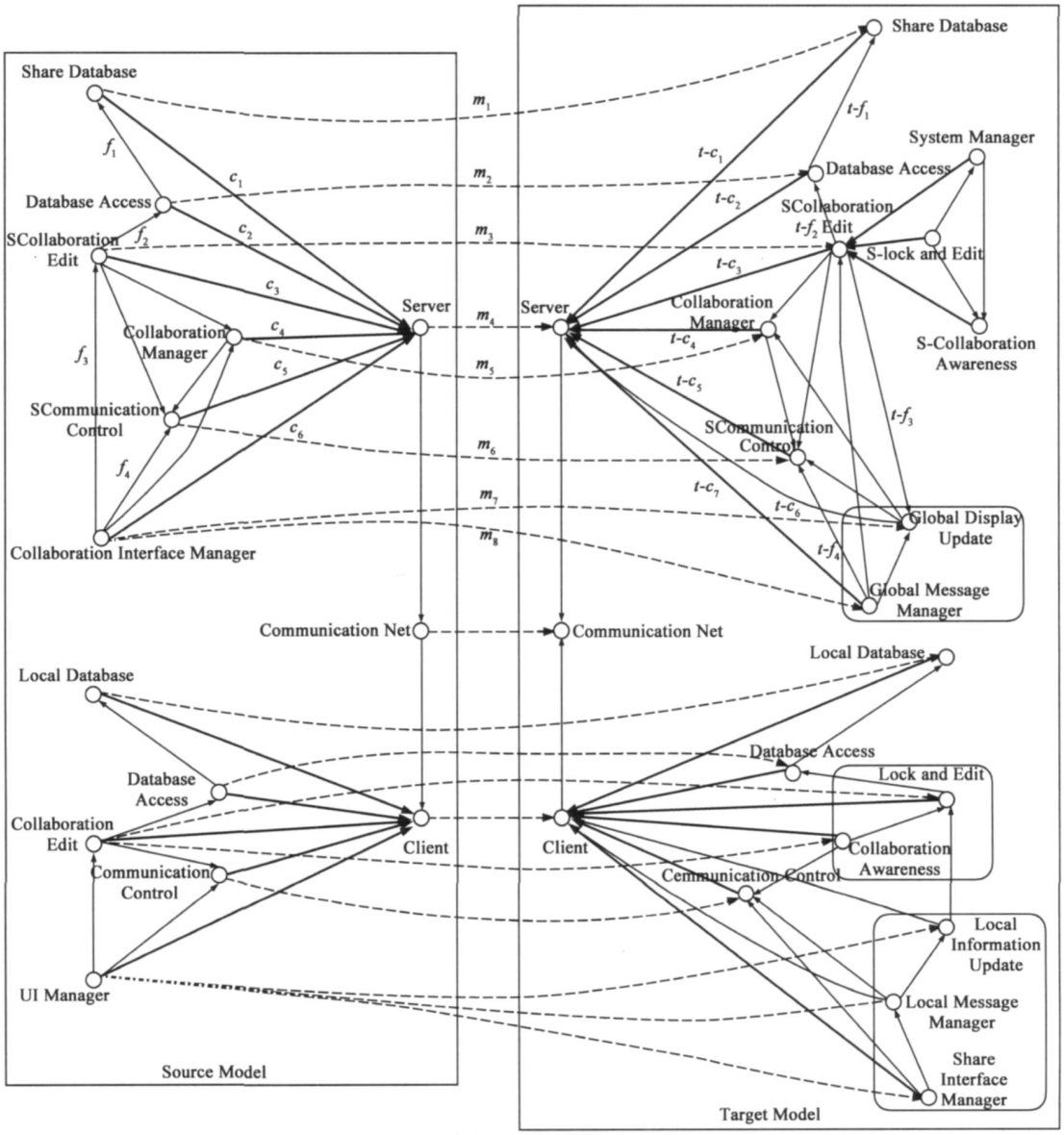


图4 协同编著系统体系结构模型映射图

c_1、c_2 等粗箭头代表组合态射,f_1、f_2 等细箭头为交互态射,目标模型的范畴图如图 4 右半部分所示. 源模型到目标模型的构件映射关系如图中虚线箭头所示. 构件规范间的组合态射、交互态射以及 mapping 态射满足范畴图的可交换性[7],如 $t\text{-}c_1^{\circ}m_1 = m_4^{\circ}c_1$, $t\text{-}f_1^{\circ}m_2 = m_1^{\circ}f_1$, $t\text{-}f_4^{\circ}m_8 = m_6^{\circ}f_4$ 等,表明该转换保持了构件间依赖关系的一致性. 源模型中用户界面处理构件映射为目标模型中共享界面处理构件、本地消息处理构件和本地信息更新构件三者的组合,限于篇幅,本文仅给出由这三者组成的复合构件规约(T-UIManager),如图 5 所示,这里省略了其部分内部端口描述.

```
Component T-UIManager
  Attributes
      Cid:String;//构件的唯一标志
      Private CompStatus: StatusType;//构件所处的状态
      Private UIEventSet:Collection;//用户界面事件集合
      Subcomponent locIfUpdate: LocalInfoUpdate;//子构件
      Subcomponent locMsgManager: LocalMessageManager;//子构件
      Subcomponent shrItfManager:ShareInterfacrManager;//子构件
      …
  Ports
      In PT1{
      Boolean collectEventInfo(); //收集用户界面事件
      }
      Inner PT2{
      Boolean isShareOperation(event:Event);//是否为共享操作事件
      Boolean void transLocalEvent(); //传递本地操作事件
      }
      Out PT3{
      Boolean sendShareEvent(event:EventType);
                                  //发送共享操作事件信息
      }
      Inner PT4{
      void preLocalInfo(); //预处理本地操作事件
      }
      In PT5{
      Boolean receiveUpdateInfo(message:InfoType);
                                  //接收共享界面处理信息
      }
      In/Out PT6{
      Boolean sendLocalEvent (event:EventType);
                                  //发送本地操作事件信息
      Boolean receiveLocalInfo(message:InfoType);
                                  //接收本地界面更新信息
      }
      …
  Axioms
      (collectEventInfo()==True)⇒isShareOperation();
      (isShareOperation()==True)⇒sendShareEvent();
      (isShareOperation()==False)⇒sendLocalEvent();
      receiveLocalInfo(message)⇒Redisplay();
      receiveUpdateInfo(message)⇒Redisplay();
            …
  Behavior
      BP_T-UI ≜ colEtIf.(isShrOp.sdShEt.BP_T-UI+isLocOp.sdLcEt.BP_T-UI)+
rcShIf.rdsP.BP_T-UI+rcLcIf.rdsp.BP_T-UI;
      …
```

图5　共享界面处理构件、本地消息处理构件和本地信息更新构件的复合规约描述

考察复合构件 T-UIManager 的行为语义描述 $BP_{T\text{-}UI}$,和源模型中构件 UIManager 的行为语义描述 BP_{UI},则显然有 $BP_{T\text{-}UI}$ 弱等价[8] BP_{UI}. 这种情况下,我们称目标构件模型的外部观察行为弱模拟源构件模型,或称转换 $M: M_{UIManager} \rightarrow M_{T\text{-}UIManager}$ 保持了行为语义一致性. 构件 UIManager 的源规约描述到其映射目标(复合构件 T-UIManager)之间的详细映射关系可由属性、端口以及公理描述中的名字对应得到.

4　结论

本文针对协同应用系统开发,将范畴理论、代数规范和进程代数相结合,为软件体系结构模型建立了一种新的语义描述方法. 使用范畴理论作为数学框架,使得所讨论的问题可以用与特定 ADL 无关的术语来形式化描述. 范畴理论支持图形化建模,可以使模型中的构件关系以及结构特征可视化,有利于对模型转换的理解和追踪. 范畴理论和进程代数相结合,既使得系统行为和系统拓扑的描述清晰分离,又可以一种统一的方式来处理构件模型的各种语义特性.

进一步的工作将从应用实际出发,对协同系统的体系结构描述进行全面分析和归纳,制定出一个细致、全面的特征模型来形式化定义构件规约和体系结构模型,从而提高模型的语义表述能力和一致性验证能力,使其更加准确、有效地反映并支持工程化的模型驱动开发.

作者简介:

侯金奎　男,1976 年 1 月生于山东寿光,2008 年 12 月获山东大学博士学位,潍坊学院讲师. 主要研究方向为模型驱动开发、形式化方法和软件体系结构等.

E-mail:hjk @mail. sdu. edu. cn

参考文献:

[1] 杨胜文,史美林. 协同服务:从群件到面向服务的协同系统[J]. 通信学报,2006,27(11):148 - 153.
Yang Shengwen, Shi Meilin. Co-service: from groupware to service-oriented collaborative systems[J]. Journal on Communications, 2006, 27(11): 148 - 153. (in Chinese)

[2] Brent Hailpern, Peri Tarr. Model-driven development: The good, the bad, and the ugly[J]. IBM Systems Journal, 2006, 45(3): 451 - 461.

[3] 梅宏,申峻荣. 软件体系结构研究进展[J]. 软件学报,2006,17(6):1257 - 1275.
Mei Hong, Sheng Junrong. Progress of research on software architecture[J]. Journal of Software, 2006, 17(6): 1257 - 1275. (in Chinese)

[4] Beydeda, S, Book M, Gruhn V. Model-Driven Software Development-Volume II of Research and Practice in Software Engineering[M]. Berlin: Springer, 2005.

(下转第 105 页)

vances in Eurocrypt 2004[C]. Berlin: Springer-Verlag, 2004. 171 - 188.

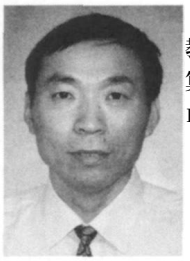
[8] Canetti R, Goldreich O, Halevi S. The random oracle methodology, revisited[A]. Proc of the 13th Annual ACM STOC[C]. New York: ACM Press, 1998. 209 - 218.

[9] 张乐友,胡予濮,刘振华. 标准模型下基于身份的可证安全门限签名方案[J]. 西安电子科技大学学报(自然科学版),2008,35(1):81 - 86.
Zhang Le-you, Hu Yu-pu, Liu Zhen-hua. Provable secure ID-based threshold signature scheme without random oracle. Journal of Xidian University, 2008, 35(1): 81 - 86. (in Chiese)

[10] Kenneth G Paterson, Jacob C N Schuldt. Efficient identity-based signatures secure in the standard model[A]. ACISP 2006[C]. Berlin: Springer-Verlag, 2006. 207 - 222.

[11] Feldman P. A practical scheme for non-interactive verifiable secret sharing[A]. Proc. of the 28th IEEE Symp on the Foundations of Computer Science[C]. New York: IEEE Computer Society, 1987. 427 - 437.

作者简介:



蔡永泉　男,1956 年生于安徽合肥,博士、教授、博士生导师. 主要研究方向为信息安全、计算机网络.
E-mail: cyq@bjut.edu.cn

张雪迪　男,1984 年 8 月生于山东淄博. 在读硕士研究生,主要研究方向为密码学.
E-mail: sk218zxd@163.com

姜　楠　女,1977 年 8 月生于山东,博士,讲师. 主要研究方向为信息安全.

(上接第 111 页)

[5] Mens T. A survey of software refactoring[J]. IEEE Transactions on Software Engineering, 2004, 30(2): 126 - 139.

[6] 刘辉,麻志毅,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 Hui, Ma Zhiyi, Shao Weizhong. Description and proof of property preservation of model transformations[J]. Journal of Software, 2007, 18(10): 2369 - 2379. (in Chinese)

[7] Barr Michael, Wells Charies. Category Theory for Computing Science[M]. New Jersey: Prentice-Hall, 1990.

[8] Bernardo M, Ciancarini P, Donatiello L. Architecting families of software systems with process algebras[J]. ACM Transactions on Software Engineering and Methodology, 2002, 11(4): 386 - 426.

[9] 顾宁,杨江明,张琦炜. 协同组编辑中基于地址空间转换的一致性维护方法[J]. 计算机学报,2007,30(5):763 - 774.
Gu Ning, Yang Jiangming, Zhang Qiwei. Consistency maintenance based on the address space transformation technique in group editors[J]. Chinese Journal of Computers, 2007, 30(5): 763 - 774. (in Chinese)

[10] Lu Ruqian. Towards a mathematical theory of knowledge[J]. Journal of Computer Science and Technology, 2005, 20(6): 751 - 757.