BDL模型到UML状态图的可视化方法研究

马 丽[1,2] 毋国庆[2,3] 黄 勃[2,3] 程 铭[2] 崔梦天[4,5]
(平顶山学院软件学院 平顶山 467000)[1] (武汉大学计算机学院 武汉 430072)[2]
(武汉大学深圳研究院 深圳 518057)[3] (西南民族大学计算机科学与技术学院 成都 610041)[4]
(电子科技大学计算机科学与工程学院 成都 610000)[5]

摘 要 针对复杂软件系统需求模型难以理解的问题,提出了一种UML状态图描述需求模型的可视化方法。该方法基于行为描述语言(Behavior Description Language,BDL)构建的需求模型,通过定义映射规则,将BDL模型中的行为、行为间关系与UML状态图中的迁移相关联,并将行为执行后产生的状态与UML状态图中状态相关联。然后根据转换算法自动提取各结点信息,输出完整的状态图,从而实现BDL需求模型的可视化。最后通过实例,验证了该方法的有效性。

关键词 行为描述语言,需求建模,映射规则,模型转换,状态图
中图法分类号 TP311.5 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.7.009

Visualization Method of BDL Model to UML State Diagram

MA Li[1,2] WU Guo-qing[2,3] HUANG Bo[2,3] CHENG Ming[2] CUI Meng-tian[4,5]
(School of Software,Pingdingshan University,Pingdingshan 467000,China)[1]
(School of Computer,Wuhan University,Wuhan 430072,China)[2] (Shenzhen Institute,Wuhan University,Shenzhen 518057,China)[3]
(School of Computer Science and Technology,Southwest University for Nationalities,Chengdu 610041,China)[4]
(School of Computer Science & Engineering,University of Electronic Science & Technology of China,Chengdu 610000,China)[5]

Abstract Aimed at finding a solution to the difficulty understand for the requirement model a complex software,this paper presented a visualization method by using UML state diagram discribing the requirement model. This method is based on system behavior sequences,which are indicated by behavior description language(BDL). It builds up a system behavior model. Mapping rules which are defined construct the relation between behaviors of the BDL model and the migrations of UML state diagram,and the state of UML state diagram associates the state produced by behavior. Then,according to the transformation algorithm,the information for each node is extracted automatically to visualize BDL requirement model. At last,the instance was presented to verify the effectiveness of the method.

Keywords Behavior description language,Requirements modeling,Mapping rules,Model transformation,State diagram

1 引言

随着大数据时代的到来,软件系统日益大型化、复杂化,而软件的质量远未达到用户的期望。很多研究发现软件需求问题是软件开发项目失败的主要原因[1]。需求工程是软件生命周期的起点,关系到软件开发后继各阶段的成败。需求建模是软件需求工程过程中的关键活动,它从不同角度为系统用户建立一个能观察到的概念模型,并提供一个验证沟通与需求信息正确性的平台,从而获得高质量的需求。

从软件需求建模理念的角度,比较受关注的需求建模的指导思想主要有面向主体的方法、面向目标的方法、问题框架方法等[2]。从需求建模方法的角度,可将其粗分为基于数学符号的形式化方法和基于图形符号的半形式化方法。形式化方法以Z语言和维也纳开发方法(VDM)为典型代表[3],半形式化方法中基于统一建模语言(UML)的面向对象方法和基于数据流图(DFD)的结构化方法最具代表性[4]。需求建模的主要目标是建立需求模型,明确软件系统"做什么",其过程可看做通过分析或预测目标软件的运行时行为及其作用效果判断是否能够满足用户需求[5]。因此软件行为的正确性决定了软件能否满足用户需求,而软件系统的完整性、一致性、可信性、安全性等非功能需求也需要通过运行软件来验证。目前与软件行为相关的研究多是对系统是否存在行为冲突的检查或对系统某些特性的检测,较少系统地建立基于软件行为的需求建模方法。

另外,分而治之策略是一种解决复杂问题的较好方法。需求分析中,从不同侧面多视点对系统所关注的问题进行描

到稿日期:2014-07-01 返修日期:2014-10-06 本文受国家自然科学基金(91118003,61003071),深圳战略性新兴产业发展专项资金(JCYJ20120616135936123),国家自然科学基金面上项目(61379019),国家留学基金(201206070041),四川省学术和技术带头人培养资金联合资助。
马 丽(1968-),女,硕士,教授,主要研究方向为软件需求工程、模式识别与智能控制,E-mail:malixnc@126.com;毋国庆(1954-),男,教授,博士生导师,主要研究方向为软件工程、软件形式化与自动化;黄 勃(1985-),男,博士生,主要研究方向为软件需求工程、形式化方法;程 铭(1985-),男,博士生,主要研究方向为软件需求工程;崔梦天(1972-),女,博士后,教授,硕士生导师,主要研究方向为可信软件、优化理论和算法。

述，即为多视点的需求分析方法[6,7]。该方法所蕴含的独立视点的思想[8,9]，不仅能降低需求描述的复杂度，而且还能减少潜在需求的遗漏，面向多视点的需求工程特别适合大型、复杂的软件系统。但目前还没有一套有效的基于多视点的完整的需求分析以及建模方法。

基于上述观点，对于复杂软件系统，文献[9]提出一种基于软件行为和多视点的需求工程建模方法。在软件需求建模过程中，由于存在使用多种建模技术建立软件需求模型的情况，因此在此把利用文献[9]之外的建模技术建立的视点需求模型通称为异类视点需求模型，目前有关异类视点需求模型的转换研究还比较少。UML 是一种可视化的面向对象建模语言，已成为复杂系统建模的工业标准，并可借助代码自动生成工具实现从分析到编码的开发过程自动化[10]。

为使文献[9]提出的需求建模方法更具有通用性，本文在上述研究的基础上研究基于行为和多视点需求模型的异类模型转换。本文第 2 节简要介绍 BDL 行为需求模型的构建；第 3 节以 UML 状态图为例，详细介绍异类视点需求模型的转换方法研究；第 4 节为 BDL 需求建模工具应用及转换仿真结果；最后为结束语。

2 BDL 行为需求模型的构建

基于软件行为的需求模型（简称行为模型）建模方法，利用行为描述语言 BDL 把目标软件系统行为表示为行为表达式，通过其变化来刻画系统的状态变化，建立系统的行为模型[4]。建立基于行为和多视点的需求模型的基础是视点、场景、软件行为等概念，行为描述语言 BDL 利用形式化技术将上述概念符号化和规范化。

2.1 行为描述语言 BDL

BDL 是一种基于多视点描述软件系统需求模型的语言，主要用于描述软件系统的行为，也是研究软件系统性质的基础。BDL 以场景作为视点需求描述的基本单位，以组成场景的原子行为作为最小单位。需求模型建立后，可依据其动态语义对模型做多种特性检测，以检验需求的正确性、有效性等。

2.1.1 行为模型结构

行为模型主要由场景行为模型和视点行为模型构成，其结构可表示如下。

（1）场景行为模型结构

场景 ID[场景 ID]:
BEGIN
[ABEH:(原子行为)
ABehID:原子行为 1;
……
ABehID:原子行为 n;]
BEH:(复合行为标识前缀必须有 Beh)
BehID=场景行为表达式;
[BehID=子行为表达式 1];
……
[BehID=子行为表达式 m];
END

其中，一个或多个子场景行为表达式按其关系可组合为一个场景表达式。

（2）视点行为模型结构

视点 ID:
VPBEGIN
[视点内共享数据存储池 ID;]
场景 ID_1 的行为模型;
…
场景 ID_n 的行为模型;
VPBehID＝视点行为表达式
＝场景标识 场景间关系符 场景标识[场景间关系符 场景标识……]
VPEND

视点内共享数据存储池 ID 主要用于存放视点内各场景共享的数据或由其它视点传入的数据等。

（3）系统行为模型结构

系统名:
视点 ID_1 的行为模型;
……
视点 ID_n 的行为模型;

2.1.2 BDL 语法结构

令 ABehID 为原子行为标识，BehID 为行为标识，f 为主体 sub 施于客体 obj 的服务、操作或动作。BDL 语法结构定义如图 1 所示。

(1)原子行为
①原子行为表达式
ABehID:f(sub,obj [&obj 的补充说明])
[When 前置条件]
[INFrom (ID_i)(u_1,…,u_n)]
[OUTTo (ID_k)(v_1,…,v_m)]
②空动作:ABehID:Idel
③复合行为结束动作:ABehID:Return (ABehID) 或 Return ( )
(2)简单行为
|－ ABehID;(原子行为构成简单行为)
(3)复合行为
①顺序行为:

a) $\dfrac{|-ABehID_1 \,\&\, |-ABehID_2}{|-ABehID_1;ABehID_2}$

b) $\dfrac{|-ABehID \,\&\, |-BehID_1}{|-ABehID;BehID_1}$

c) $\dfrac{|-BehID_1 \,\&\, |-ABehID}{|-BehID_1;ABehID}$

d) $\dfrac{|-BehID_1 \,\&\, |-BehID_2 \,\&\, \cdots \,\&\, |-BehID_n}{|-BehID_1;BehID_2;\cdots;BehID_n}$

②确定选择行为:

$$\frac{|-BehID_1 \,\&\, |-BehID_1 \,\&\, b\ 布尔表达式}{|-If\ b\ Then\ BehID_1\ Else\ BehID_2\ Fi}$$

注:b 为 $BehID_1$ 中最开始原子行为的前置条件。

③未确定选择行为:

$$\frac{|-BehID_1 \,\&\, |-BehID_2 \,\&\, \cdots \,\&\, |-BehID_n}{|-BehID_1+BehID_2+\cdots+BehID_n}$$

④并行行为:

$$\frac{|-BehID_1 \,\&\, |-BehID_2 \,\&\, \cdots \,\&\, |-BehID_n}{|-BehID_1 \parallel BehID_2 \parallel \cdots \parallel BehID_n}$$

图 1 BDL 语法

图 1 所示的 BDL 语法中，when 中的条件表示行为执行的前提条件（缺省为真）。带 when 的原子行为相当于语句“If 前置条件 Then 行为 Fi”。

INFrom 表示从 ID_i 获得数据，OUTTo 表示将数据输出到 ID_k，$u_1,\cdots,u_n$ 和 $v_1,\cdots,v_m$ 为行为输入、输出值的有限集，ID_i、ID_k 为 ABehID 或视点内共享数据池 ID 或外部实体名、或其它视点 ID。

复合行为结束动作(Return)的参数可选:若有参数则转到原子行为;否则结束其执行。

复合行为由一组连续执行的原子行为构成,也可通过顺序、并行、选择算子连接复合行为与简单行为构成更为复杂的表达形式。

2.2 行为描述语言的动态语义

令 $B,B',B_i,B_i'(i=1,2)$ 为行为表达式,ABehID 和 α 为原子行为标识。行为描述语言 BDL 的动态语义表述如图 2 所示。

(1)顺序行为:

① $B \xrightarrow{ABehID} B'$

② $B \xrightarrow{ABehID(v_1,\cdots,v_n)} B'$

③ $B \xrightarrow{Idle} B'$

(2)非确定选择行为:

① $\dfrac{B_1 \xrightarrow{\alpha} B_1'}{B_1+B_2 \xrightarrow{\alpha} B_1'}$

② $\dfrac{B_2 \xrightarrow{\alpha} B_2'}{B_1+B_2 \xrightarrow{\alpha} B_2'}$

(3)并行行为:

① $\dfrac{B_1 \xrightarrow{\alpha} B_1'}{B_1 \parallel B_2 \xrightarrow{\alpha} B_1' \parallel B_2}$

② $\dfrac{B_2 \xrightarrow{\alpha} B_2'}{B_1 \parallel B_2 \xrightarrow{\alpha} B_1 \parallel B_2'}$

(4)确定选择行为:

① $\dfrac{B_1 \xrightarrow{\alpha} B_1',\text{b 的值为真}}{\text{If b Then } B_1 \text{ Else } B_2 \text{ Fi} \rightarrow B_1'}$

② $\dfrac{B_2 \xrightarrow{\alpha} B_2',\text{b 的值为假}}{\text{If b Then } B_1 \text{ Else } B_2 \text{ Fi} \rightarrow B_2'}$

图 2 BDL 语义

图 2 所示的 BDL 动态语义中,顺序行为规则①(②)描述 B 执行一个无参(有参)原子动作后对应的行为表达式为 B';非确定性选择规则描述如果 B_1(B_2)执行一个动作,则 B_2(B_1)被舍弃;并行规则描述了 B_1、B_2 独立执行;确定性选择规则描述了如果 B_1(B_2)执行一个动作且 b 为真(b 为假),则确定选择执行 Then(Else)分支。若把行为表达式看作一个进程以及把原子行为看作进程间的迁移动作,则图 2 为 BDL 行为迁移系统的规则描述。因此行为表达式的迁移变化可由 BDL 的动态语义描述,由此获得视点与场景行为模型的迁移变化,从而为检测行为需求模型所描述的多种系统特性奠定了基础[11]。

3 异类视点需求模型的转换研究

模型转换起源于软件工程,是模型驱动体系结构(Model Driven Architecture,MDA)的核心技术。MDA 从定义转换规约的角度定义模型转换,提出了一系列从一个模型转换为另一个模型的转换规则的转换定义。Shane Sendall 从映射的角度定义模型转换:一个模型的集合到自身或到另一个模型集合的映射。该映射定义了源模型元素与目标模型元素间的对应关系。模型转换有源模型与目标模型、映射关系或转换规则两个关键点。一方面,虽说 BDL 模型能够捕获和形式化自然语言描写的系统需求的动态行为,但其在可读性和可理解性方面的效果较图形方式差一些;另一方面,UML 建模技术在软件开发过程中使用得较为广泛,BDL 建模方法若能与其结合不仅更具有实用性与通用性,还促进了需求验证和到模型驱动软件开发方法的转变[12]。本节以 UML 状态图为例,将 BDL 行为模型转换为 UML 状态图。

3.1 UML 状态图的相关定义

UML 状态图是 UML 中系统动态建模图形工具之一。目标系统中各元素(单个实体或实体间的交互活动)的离散行为通过有限状态——迁移系统给予描述。状态图主要用于描述从一个状态到另一个状态的控制流。UML 状态图在 3 个方面扩充了传统的有限状态机:(1)层次(hierarchy),通过状态嵌套实现,一个状态可由多个状态组成;(2)并发(concurrency),允许多个复合状态并发执行;(3)广播通信(broadcast communication),并发状态的通信通过事件广播实现。

UML 状态图由状态、迁移、初始状态、终止状态、判定、简单状态、复合状态等基本要素点组成,各要素符号表示如图 3 中的符号所示,其中:

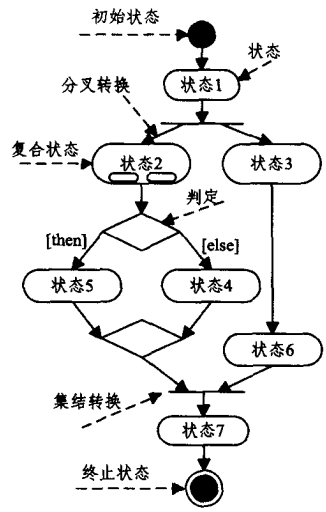


图 3 UML 状态图层次结构示例

状态:在对象的生命周期中的某个条件或者状况,在此期间对象将满足某些条件、执行某些活动,等待某个事件的过程中的一次交互。

迁移(转换):两个状态之间的关系,表示系统对象在某个条件满足且某个事件发生时,由一个状态进入另一个状态。UML 状态图的迁移包括源状态、触发事件、卫式条件、动作、目标状态等元素。转换用带箭头的直线表示,一端连源状态,一端连目标状态。当源状态接收一个事件后,执行相应的动作,转换到目标状态。若转换上无标注事件,则表示此转换自动进行。

初始状态:此状态代表状态图的起始位置,初始状态只能作为转换的源,不能作为转换的目标,一个状态图有且只有一个初始状态。

终止状态:模型元素的最后状态,是一个状态图的终止点。其只能作为转换的目标,而不能作为转换的源。终止状态在状态图中可有多个。

判定:当工作流按某一条件的取值而产生分支时,就要使用判定。

简单状态:不包含其他状态的状态。简单状态没有子结构,但可有内部转换。

复合状态:一个嵌套了若干个状态的状态,被嵌套的状态称作子状态。简单状态和迁移可通过初始(initial)、终止(final)、分叉(fork)、汇聚(join)、选择(choice)等连接子组装为复合状态。

UML 状态图主要由简单状态与迁移（转换）构成，使用 UML 状态图可对一个对象（类）的行为建模，也可对一个子系统或整个系统的行为建模，故 BDL 模型可转换为 UML 状态图。为使 BDL 模型转换的 UML 状态图能满足 UML 状态图的标准规范，约定 UML 状态图的层次状态图规范，即结构化的图层嵌套关系，如图 3 所示。图中的图形符号含义及其与 BDL 对应关系见下文说明。

3.2 BDL 结构与 UML 状态图中符号的对应（映射）关系

图 3 主要使用了 UML 状态图中的初始状态、终止状态、状态、复合状态、顺序迁移、分叉和集结迁移、判断迁移等符号（暂且不考虑其它符号）。BDL 中各结构与 UML 状态图中的符号映射关系定义如下。

映射规则 1（初始状态和终止状态）：每个状态图或子状态中只允许有一个初始状态和最终状态，且成对出现。BDL 中的视点行为开始标记 VPBEGIN 或场景行为开始标记 BEGIN 对应状态图中的初始状态；BDL 中的视点行为结束标记 VPEND 或场景行为结束标记 END 对应状态图中的终止状态。

映射规则 2（迁移）：BDL 中的原子行为或复合行为对应状态图中的迁移。

映射规则 3（状态也称简单状态）：BDL 中原子行为（动作）执行后产生的状态，即 BDL 的原子动作 ABehID、空动作 Idle、返回动作 Return 等执行后产生的状态对应状态图中的状态，状态名按 BDL 中原子行为执行顺序自动生成。BDL 场景行为表达式中的第一个行为（一般为原子行为 ABehID）执行后产生的状态对应 UML 状态图中的第一个状态，BDL 场景行为表达式中的最后一个行为（一般为原子行为 Ruturn）执行后产生的状态对应 UML 状态图中的最后一个状态。

映射规则 4（顺序迁移）：BDL 中的顺序关系“;”对应 UML 状态图的顺序迁移。

映射规则 5（分叉迁移和集结迁移）：BDL 中的并行关系，即“‖”的 begin、end 分别对应状态图中的分叉迁移（并行开始）、集结迁移（并行结束）。

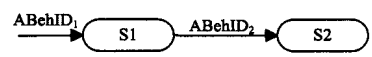
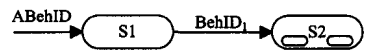
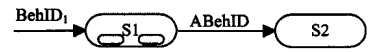
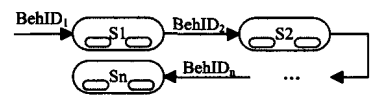
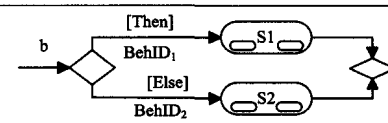
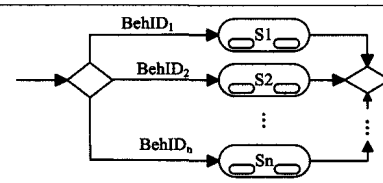
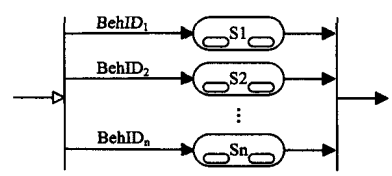
映射规则 6（判定迁移）：BDL 中的选择行为对应 UML 状态图的判定迁移。BDL 中的确定性选择关系 IF() THEN ELSE FI，根据入度、出度来判断判定对应的是 IF 还是 FI。如果入度为 1（转换引入），出度为 2（分别为判定肯定与否定分支判定），则为判定的开始符 IF，反之为判定结束符 IF。BDL 的确定性选择关系“IF() THEN ELSE FI”中的 THEN 分支与 ELSE 分支分别对应 UML 状态图中的判定肯定分支与否定分支迁移。BDL 中的非确定性选择关系，对应 UML 状态图的多（3 个及以上）事件（条件、动作）引起的判定迁移。

映射规则 7（复合状态）：BDL 中除场景行为表达式外的其它复合行为执行后自动产生的状态对应状态图中的复合状态，其状态名可按 BDL 中行为执行顺序自动生成。BDL 复合行为表达式中的第一个行为（一般为原子行为 ABehID）执行后产生的状态对应 UML 状态图中复合状态中的第一个子状态，BDL 场景复合行为表达式中的最后一个行为（一般为原子行为）执行后产生的状态对应 UML 状态图中复合状态的最后一个子状态。

映射规则 8（转换单位）：以场景为单位，即将 BDL 中场景行为表达式转换为 UML 状态图。

依据上面的映射规则和图 3 中约定的符号，图 1、图 2 中 BDL 语法及语义中的顺序行为、确定选择行为、非确定选择行为、并行行为到状态图结构表示的映射如表 1 所列。

表 1　BDL 模型中的复合行为到状态图结构的映射

	BDL 模型中的复合行为	状态图结构片断
顺序行为	$\frac{\|-ABehID_1 \&\|-ABehID_2}{\|-ABehID_1;ABehID_2}$	
	$\frac{\|-ABehID\|-BehID_1}{\|-ABehID_1;BehID_1}$	
	$\frac{\|-BehID_1 \&\|-ABehID}{\|-BehID_1;ABehID}$	
	$\frac{\|-BehID_1 \&\|-BehID_2 \&\cdots\&\|-BehID_n}{\|-BehID_1;BehID_2;\cdots;BehID_n}$	
确定选择行为	$\frac{\|-BehID_1 \&\|-BehID_2 \&\ b\ 为布尔表达式}{\|-If\ b\ Then\ BehID_1\ Else\ BehID_2\ Fi}$	
非确定选择行为	$\frac{\|-BehID_1 \&\|-BehID_2 \&\cdots\&\|-BehID_n}{\|-BehID_1+BehID_2+\cdots+BehID_n}$	
并行行为	$\frac{\|-BehID_1 \&\|-BehID_2 \&\cdots\&\|-BehID_n}{\|-BehID_1 \parallel BehID_2 \parallel \cdots \parallel BehID_n}$	

3.3 BDL 需求模型到 UML 状态图的转换算法描述

BDL 行为模型→UML 状态图转换算法基本思想为：第一步，根据上下文，从 BDL 行为需求模型中提取出数据信息（原子行为、复合行为、场景行为）、结构信息（行为间的关系）；第二步，通过对所有复合行为的直接子行为等信息进行分析，提取出该复合行为的层次信息；第三步，根据上面两步提取出的数据信息结构信息、层次信息，按照前面提出的 BDL 模型到状态图转换规则，提取相应状态名称和转换行为，统计状态层次数量，动态计算画板尺寸以及各状态的纵、横坐标，实现状态图的整体输出。本节以场景 BDL 模型为单位转换状态图，依据前面描述的 BDL 行为模型，视点 BDL 模型转换成状态图可由场景 BDL 行为模型转换成的状态图按场景间的非确定性选择、确定性选择、并行与顺序关系组成。场景 BDL 行为模型转换成 UML 状态图算法的具体描述如下。

```
输入：BDL 行为模型，定义转换规则
输出：当前场景对象的状态图
Define Object TransArrayOfKey[]；//根据转换规则创建转换规则关键字存储结构
Define class NodeInfo[]；//定义状态图中各结点结构
//遍历 BDL 行为描述的行为序列，并创建状态图各结点和存储各信息
For each rowObject Do compare with
  TransArrayOfKey[]
  getNodeInformationFrom rowObject()
  Create the state Si //创建状态
  set NodeInfo[i]={Si,
  rowObject. Substring(),
  level of TransArrayOfKey[],
  Type of TransArrayOfKey[],
  Xi,Yi}；//设置状态图的状态名称、类型、转换行为、坐标等信息
EndFor
//根据结点分析得到的结点数量初始化结点间距、总体画板大小等信息
Bitmap bmpT = new Bitmap(dtW * (nodeTotalNum - subNodeNum),400)；
Graphics gh=Graphics. FromImage(this. pictureBox1. Image)；
//创建状态图，计算各结点坐标和相对布局位置，计算
For each object in NodeInfo[] Do by The Type of NodeInfo[]. NodeInfo
  GetFirstState(NodeInfo[])/取得状态图初始状态，并在画板上显示初始状态结点
  SearchSC (NodeInfo[])//搜索状态图其他状态结点，并在画板上显示相应的状态结点
  gh. DrawRectangle(new Pen(Color. Black,3),x,y,nodeW * 2,nodeH)；
  If NodeInfo[i]. NodeInfo==“LNode”‖“RNode” then //如果当前结点为选择语句的结点
    GetLeftFirstNodeXY()；
    SearchSC (NodeInfo[])；
  SetRigthFirstNodeXY(GetLeftFirstNodeXY()；)
    DrawNode()；
    GetLastIfElseNodeXY()；
  EndIf
  …//其它结点情况类似，限于篇幅省略
  DrawNode(GetLastIfElseNodeXY())；
  GetLastState(NodeInfo[]) //取得接收对象状态图的终态
EndFor
```

建立在顺序状态机与行为代数 Σ 之上的 BDL 模型，利用有穷状态迁移系统来描述其动态执行过程。而状态图用于显示状态机，故可先给出一种状态机结构的结构式形式化描述，然后据此研究 BDL 行为表达式与状态结构式的强互模拟以及二者之间互模拟关系之间的关系；研究 BDL 模型与其转换后的状态模型的互模拟等价性；文献[13]详细阐述了行为模型到状态模型转换的正确性。

4 BDL 需求建模工具应用及模型转换仿真

本节通过 ATM 实例，具体描述如何构建 BDL 行为需求模型以及将其转换为 UML 状态图。首先划分系统的问题域，将 ATM 系统相关的问题域划分为 ATM、总行、分行 3 个。然后标识视点，假设每个问题域至少有一个视点，分别标识为 ATM 视点（VPATM）、总行视点（VPHeadBank）、分行视点（VPLocalBank）。最后为每个视点划分场景并建立场景行为模型。标识视点后，项目负责人指定编辑视点内需求的视点责任人。基于上面分析，建立的基于行为的 ATM 系统需求模型如图 4 所示。

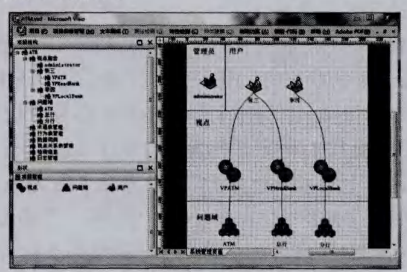


图 4　ATM 系统需求模型

鉴于整个系统的复杂性和文章篇幅，本文仅给出 ATM 系统的总行场景 BDL 模型的部分描述。

```
BEHHeadBank：//总行场景
BEGIN
  ABEH：
Hbank1：拥有（总行，多台 ATM）
Hbank2：组成（多个分行，总行）
Hidel：idel()；//idel 等待 ATM 请求或分行回答
//总行接收 ATM 请求或分行回答
Hreceive：响应（总行，ATM 请求或分行回答）
      OUTTo(VPdatacell)（响应类型）
//接收 ATM 验证请求
Hresponse：接收（总行，ATM）
      OUTTo (VPdatacell)（ATM 号）
…
BEH：
  BEHHeadBank=
    Hbank1；
    Hbank2；
    Hidel；
  If 响应类型=ATM 请求
    Then Behhbank1；
    Else Behhbank2；
  Fi
  Returnto；
Behhbank1=
  Hresponse；
```

```
If 验证请求
  Then Hreceive1;
    Discern;
    TranRequest1;
  Else//中央计算机接收 ATM 传来的事务请求
    Hreceive2;
    TranRequest2;
Fi;
Behhbank2=
If 回答验证请求
Then //中央计算机将验证结果发送到 ATM 机
    RecRespon1;
    Tranresult1;
Else //中央计算机将分行答复发送到 ATM 机
    RecRespon2;
    ranresult2;
Fi;
END
```

在场景 BDL 模型建立后，即可按照上面给出的方法将其转换为 UML 状态图。BDL 模型须通过一系列检测，以确保 UML 状态图的正确性。该算法利用队列和栈结构实现对 BDL 文档的遍历，其开放性较好，能够为今后扩展或二次开发提供较为便捷的接口参数。以 ATM 系统 BDL 需求模型中的总行场景为例，将其转换为 UML 状态图，运行界面如图 5 所示。

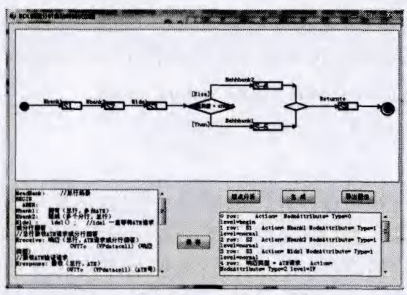


图 5　总行 BDL 场景模型转换生成的状态图

结束语　本文基于软件行为的需求建模方法与工具，提出了一种基于行为的需求模型的异类模型转换方法。通过 ATM 系统实例，实现了该实例从 BDL 需求模型到 UML 状态图的转换。该方法通过定义映射规则将 BDL 模型中的行为及其行为执行后产生的状态与 UML 状态图中的迁移与状态关联；将 BDL 模型行为间的顺序、选择、并行等关系与 UML 状态图中的顺序、选择、分叉和汇聚等迁移关联。设计了 BDL 模型到 UML 状态图的转换方法，根据映射规则自动提取各状态相关信息，且能依据 BDL 模型规模自动调整各节点间距和空间布局，实现状态图的整体输出。该研究不仅扩展了基于软件行为需求建模方法的通用性，而且对开发过程中保持模型间的一致性也具有重要意义。将来的研究方向是实现该转换与基于软件行为的需求建模方法和工具的无缝对接，以及实现 BDL 模型与其它建模方法的转换。

参 考 文 献

[1] 练红. 软件开发项目需求分析研究[D]. 北京：北京邮电大学，2008
Lian Hong. Research of Requirement Analysis in Software Development project[D]. BeiJing: Beijing University of Posts and Telecommunications, 2008

[2] 赵也非. 动态 UML 子图的形式语义研究[D]. 上海：华东师范大学，2010
Zhao Ye-fei. The Study on Formal Semantics of Dynamic UML Diagrams[D]. ShangHai: East China Normal University, 2010

[3] 金芝，刘璘，金英. 软件需求工程：原理和方法[M]. 北京：科学出版社，2008
Jin Zhi, Liu Lin, Jin Ying. Software requirements engineering: principles and methods[M]. Beijing: Science Press, 2008

[4] 毋国庆，梁正平，袁梦霆，等. 软件需求工程[M]. 北京：机械工业出版社，2013
Wu Guo-qing, Liang Zheng-ping, Yuan Meng-tin, et al. Softwarer requirements engineering[M]. Beijing: China Machine Press, 2013

[5] Qu Yan-wen. Software behaviou [M]. Beijing: Publish house of Elcetronics Industry, 2005

[6] Ross D T. Structured analysis a language for communicating ideas [J]. IEEE Transactions on Software Engineering, 1977(SE-3): 16-34

[7] Schoman, Ross D T. Structured analysis for requirements definition[J]. IEEE Transactions on Software Engineering, 1977(SE-3): 6-15

[8] Wan L, Wu G, Wu H. BDL-behavior description language[C]// Proceeding of 2009 International Conference on Software Technology and Engineering, 2009. World Scientific, 2009: 37-41

[9] 万黎，毋国庆，吴怀广. 面向行为的需求建模研究及实现[J]. 计算机科学，2011，38(4)：175-181
Wang li, Wu guo-qing, Wu Hai-guang. Research and Implement of behavior-oriented requirments modeling[J]. Computer Science, 2011, 38(4): 175-181

[10] 万小平，李蜀瑜. 基于 XML 的 UML 模型向 AADL 模型的自动转换[J]. 计算机技术与发展，2014，24(3)：71-73，78
Wan Xiao-ping, Li Shu-yu. Automatic Conversion of UML Model to AADL Model Based on XML[J]. Computer Technology and Development, 2014, 24(3): 71-73, 78

[11] 吴怀广，毋国庆，陈曙，等. 面向软件行为的需求模型及特性检测[J]. 计算机研究与发展，2011，8(5)：869-876
Wu Huai-guang, Wu Guo-qing, Chen Shu, et al. A software behavior oriented requirements models and properties verification [J]. Jouranl of Computer Research and Development, 2011, 8(5): 869-876

[12] 解方，段富. 从行为树转换到 UML 状态机来验证系统需求[J]. 计算机工程与设计，2013，34(10)：3710-3716
Xie Fang, Duan Fu. Validation of system requirements from behavior tree to UML state machine[J]. Computer Enginnering and Design, 2013, 34(10): 3710-3716

[13] 李琳，毋国庆，黄勃，等. 基于行为模型的需求可视化研究[J]. 计算机学报，2013，36(6)：1312-1323
Li Lin, Wu guo-qing, Huang Bo, et al. Behavioral model based requirements visualization method[J]. Chinese Jouranl of Computers, 2013, 36(6): 1312-1323