使用单一多层级图示重新构建需求建

萨巴赫·阿尔-费达吉*科威特大学计算机工程
系 科威特 salfedaghi@yahoo.com,
sabah.alfedaghi@ku.edu.kw

摘要 – 基于软件的系统日益复杂，尤其是在捕获需求及其相互依赖关系方面。概念模型是一种思考并理解**现实世界**复杂性的方法。本文聚焦于这一背景下的两种方法：（a）对同一系统采用多个模型，并同时使用不同的表示法；（b）采用单一表示法框架的单一模型。第一种方法中，模型之间会产生不一致，因此需要投入大量精力，以严谨的方式协调各模型。多模型理念基于这样的主张：无法用单一表示呈现应用的所有视图，因此需要采用不同模型，分别表示不同视图。本文倡导第二种方法，即采用具有多层级（静态/动态和行为）规范的单一模型。为证明**这一方法的**可行性，我们采用仅发生模型，该模型包含（a）斯多葛本体论、（b）物化机（TM）语言和（c）卢帕斯库逻辑。本文聚焦于 TM 建模，将其作为构建单一模型的机制。我们认为，TM 可以成为几乎所有现有建模语言的统一图示语言。为展示这一主张，我们重新绘制了国际需求工程委员会《需求建模手册》中几乎所有图示表示。这些重新绘制的内容包括上下文图、类图、活动图、用例图、数据流图和状态图。结果似乎表明，使用 TM 表示所有视图并不存在困难。

索引词 – 概念建模、不一致性、斯多葛本体论、需求规约、事件、状态

I. Introduction

现实世界日益增加的复杂性表明，有必要重新思考如何开展概念建模[1]。概念模型是—一种思考现实世界的复杂性并理解其意义的方式‖ [2]。在这种情况下，保持一致性是模型驱动软件工程的一个关键方面 [3]。

此外，基于软件的系统复杂度增长速度显著加快，

地分析和

需求定义不当导致的错误，会增加重新进行需求定义、

会大幅增加。

这些事实清楚地表明，精确的需求分析与捕获是多么

系统开发需要多方利益相关者（包括软件工程师）共同参与，并且需要对系统需求达成共识 [6]。当不同利益相关者针对同一问题设想出不同的解决方案时，就会产生需求问题，结果是他们为该问题指定了彼此不同、

需求 [6]。

需求建模有助于促进利益相关者之间的共识。建模是一种通过对待创建的现实部分进行抽象表征，来确认明确需求的特定方式。模型有助于推断粗略概念，并提高正确解读这些概念的可能性，从而促进形成恰当的共识 [7]。为了指定此类模型，会创建一种图示语言，以支持需求的获取、分析和记录处理。该语言专为便于管理、业务和技术利益相关者理解而设计 [8]。

A. 问题

在开发多个模型以处理同一系统并同时使用不同表示法时，我们面临着保持各种系统描述彼此一致的真正困难。其他困难包括可追溯性、过程映射、可变性、完整性、过度规范、一致性和文档 [9]。它们需要模型之间大量艰苦的纪律和同步。这种多模型方法基于这样的主张：不可能在单个模型中呈现所有应用视图，因此使用多样化的表示，每个模型代表不同的视图。根据 [6]，—如果使用多个模型来描述需求，重要的是保持这些模型彼此一致。这需要模型之间大量的纪律和协调。‖ 对 UML 的深入研究 [10] 表明，UML 2 的官方定义包含不一致的 UML 模型。

*2021年6月退休，2021/2022学年秋季学期借调

尽管如此，当前软件系统开发的主流路线提倡统一化的概念，如万物皆对象 [11], 和统一多重建模。统一化是指一种统一表示法，它将多种其他表示法的优点以及当前最佳实践融入一种普遍适用的表示法中 [12]。

另一种方法是采用单一模型的集成方法，以统一的图示语言编写需求规格说明。为此，人们已进行了多次尝试。本文使用一种称为事物化机（TM）的模型进行建模 [13–14]。为简洁起见，接下来我们将重点讨论行为建模，即系统各方面随时间发生的变化。

B. 动机：行为模型

我们将概念化过程具体划分为两个不同阶段： 潜在性的静态刻画，以及现实性的动态呈现。潜在性指作为事件而存在的可能性。这里使用的术语基于文献 [15], 中提出的本体模型，即仅发生模型，本文稍后将对此作简要回顾。潜在性是一个基础的哲学概念，包含许多相关概念（例如遥远性和纸性；又如精液、胚胎和婴儿 [16]）。在此背景下，我们将尽量避免在本研究的这一阶段深入探讨哲学层面的问题。本文的内容旨在尽量减少对这些方面的讨论，并将进一步探索留待未来研究。

我们采用斯多葛学派意义上的“潜在性”概念，将其视为事物的存续而非存在（见 [14]），并在建模的静态、无时间层面加以运用。例如，我们将这种静态性与状态概念（如状态机中的状态）联系起来，以支持我们的主张：状态图并非严格意义上的动态描述。按照这一思路，即使我们知道一个对象的状态，仍需了解它在特定时间的状态版本。一个人处于恋爱状态是一种（静态）状态条件，但在动态层面，我们仍必须为这一陈述补充一个具体的人物实例（例如，威廉·莎士比亚，时间：1582–1616）。此外，我们还需要知道他/她何时坠入爱河（例如，时间：20岁及以后），才能将静态陈述转化为动态描述。这一主张意味着，诸如 UML 状态图、活动图和顺序图这样的（软件）图示，还不足以构成行为模型。在两层建模中，静态描述与动态描述分别置于无时间层面和充满时间因素的层面，如下所示。

根据 [17], , UML 有三种描述行为的方式。每种方式都侧重于不同方面：活动用于描述动作之间的输入和输出及其时间顺序；状态机用于响应外部事件的通知；交互则用于描述对象之间的消息。参考文献 [17] 解释道：“行为语言中所述内容在现实世界中的含义，取决于行为实际发生时会出现什么情况。”例如，每天许多工厂都会多次执行一种改变对象颜色的工厂操作，每次操作涉及不同的对象、不同的颜色等。行为每发生一次，就构成一次独立的行为发生。因此，[17] 似乎主张，行为是按照反复实例化的类型来描述的。行为语言的语义规定了哪些行为发生（图 1 右侧）无法被完整表达（图 1 左侧）。此外， [17] 提出描述行为事件之间的先后关系，如图 2 所示。

正如本文将要展示的，TM 建模填补了图 1 左右两个图示之间的图示表达空白。TM 将活动、对象和状态统一视为事件。我们将说明，TM 为当前大多数行为建模方法提供了另一种本体论基础和建模语言。

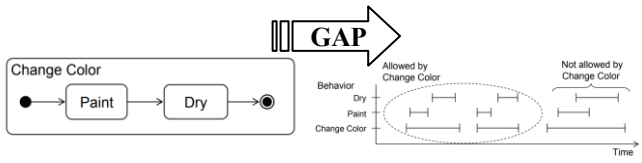

图 1 （左）行为，（右）发生情况（摘自 [17]）

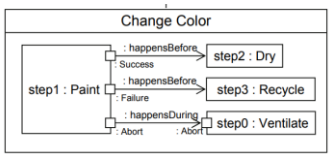

图 2 行为事件之间的先后关系 [17]

C. 两层建模

本节阐述单一模型上的垂直表征这一概念。

不同于将静态特征与动态特征分别用图示表示的常见方法（例如，类图与状态图），TM 语言构建的模型在静态表示之上具有垂直的动态表示。静态性指无时间性。静态 TM 模型由存续区域构成，并由潜在流和触发关系赋予逻辑顺序。静态模型包含固定部分，并且只是单纯地存续，例如——“交通流取决于汽车 [和流]，除此之外并无其他”[18]。交通本身并非实体，但它仍然是真实存在的，因为它的存续依赖于汽车；这种存续由静态区域捕捉。汽车是作为物理事物存在的实体。交通是一个处理，作为现存交通的区域而存续。即使没有汽车，交通仍作为一种潜在事物而存续。假设尽管有汽车和道路，交通却“消失”了（例如，由于 COVID-19）。没有交通可能意味着，

(a) 有一种称为交通的事物，但它目前并未发生（已停止）。

(b) 根本不存在所谓的交通，正如人死后没有生命、也不存在圆形的正方形，尽管确实有正方形和圆形。

显然，（a）是合理的答案。如果有一种称为交通、但当前尚未现实化（存在）的事物，那么它在哪里？斯多葛学派的回答是：它处于存续状态。

为了让计算机科学家更清楚地理解存续这一概念，我们不妨考虑加法处理，而不是交通，并用代码 ADD A,B 表示。指令 ADD A,B 当然只是加法处理的一种代码。加法处理存续于 ADD A,B 中，并在 ADD A,B 执行时现实化而存在。未执行时，加法处理仍存续于 ADD A,B 中。我们将 ADD A,B 归类为加法的静态描述；执行时，它会以动态方式实现，例如，将操作数 A 取入 ALU，再将操作数 B 取入 ALU。因此，代码 ADD A,B 并不是加法；它是加法存续之处。当加法不存在时，也就是说，在执行模式下，它并非处于“非存在”模式，因为它仍真实地存在于那里，等待执行。另一方面，ADD A,B 不存在，则意味着它已消失或不复存在，例如，它被擦除，或从未被写成代码。这就是斯多葛学派认为意义（Lekta）存续于语言表达式中的原因。因此，ADD A,B 的意义当然不是这些语言符号。它可以被描述为存续于 ADD A,B 之中、能够在现实中现实化的东西。

存续与存在同样真实。处理的停止并不意味着它被毁灭，正如发动机停止运转并不意味着它消失。发动机仍然在那里，只是不再运行。运行（工作）的存在与不存在取决于发动机，因为没有发动机，运行就无法独立存在。在这种情况下，如果运行不存在，它便以存续状态存在，随时可以再次存在。运行的存续并不意味着它被毁灭；这不同于人类和动物的生存与非生存（死亡）：一旦停止，便不会再次发生于生物（潜在性）。同样，交通停止意味着汽车和道路仍然存在，但交通处理作为一种存续现象仍然真实存在，随时可以再次出现。在这种情况下，根据斯多葛学派的观点，现实可以描述为存在与存续交替的形式。

这里我们看到，潜在性是比可能性更弱的特征。存在的潜在性意味着某个事物独立存在，或存在于某个已存在的事物之中。可能性则意味着某个事物独立存在。因此，汽车和道路是可能的，而交通则具有潜在性（它需要汽车和道路才能存在）。由于交通的存在以“汽车和道路”的存在为基础，而对交通存续的描述是一个包含汽车和道路的区域，因此整个区域具有潜在性，其中包含了更强的可能性特征。

图3展示了作为车流重复事件的现有交通。图4展示了作为处理（事件）区域的静态交通。静态交通的存续成为一个现有事件。交通区域以潜在性的形式存续。本文第二节将讨论这两幅图中的概念。

构建动态模型需要表示现存的事物（例如，物理事物、地震、庭院和外层空间）。这里的存在源自静态图所具有的潜在性。这种两层建模基于斯多葛学派的两层存在观，该观点采用两种现实形态：存在与存续。大多数现有建模方法都缺少这种两层模型。

示例：请看图

5 所示的状态图，该[19],

图对用于控制地铁出入的旋转闸机进行建模。将一枚硬币投入旋转闸机的投币口会解锁闸杆，允许一位客户推入通过。客户通过后，闸杆会再次锁定，直到投入另一枚硬币。

根据[20], —理解状态是成功建模的基础。我们需要了解的一切以及希望完成的一切，都可以用受控系统的状态来表达，因为归根结底，那些正是我们希望控制的事物。‖然而，（软件建模中的）状态通常以非正式的方式定义。OMG 的 [21] 《UML 状态机语义》文档提到“状态”一词 2,462 次，却没有对其进行定义。

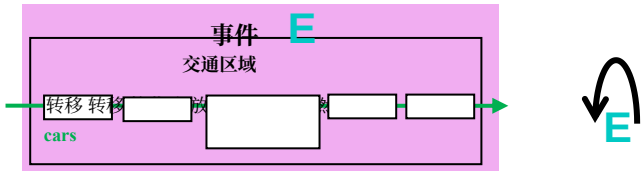

图3 事件 交通作为一种现有事物。



图4 作为持续存在事物的区域交通。

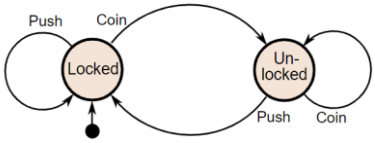

图5 旋转闸机状态机（摘自 [19]）。

从语义上说，行为指系统所做且可被公开观察到的一切。它是对做某事的描述[22]。状态则是对系统状态的描述，此时系统正等待执行一组操作。它是对某事已完成的描述[22]。

旋转闸机是一个包含乘客和机器的系统。乘客执行动作，而机器则被执行某些操作。因此，是机器而不是系统在等待，是乘客而不是系统在执行动作。因此，状态图似乎涉及缺失的隐含知识，混合了谁做什么以及做某事与被做某事之间的模糊性。相比之下，TM 在任何系统中只包含动作——实际上只有五个动作：创建、处理、释放、转移和接收。

静态描述是系统所经历的一系列状态，例如上一小节讨论的模型 [17] 。该模型由涂漆、干燥等类型构建，这些类型属于非个体（非对象）概念。涂漆和干燥的发生情况在另一张图中指定。

相比之下，动态模型描述这些状态如何随时间展开[23]。它们用于理解系统如何随时间变化。在旋转闸机示例中，图5由类型（而非实例）构建：硬币、锁定、推入等。因此，它是一种静态表示。静态描述是系统所经历的一系列状态，例如上一小节讨论的模型 [17] 。该模型由涂漆、干燥等类型构建，这些类型属于非个体（非对象）概念。相比之下，动态模型描述这些状态如何随时间展开 [23]。它们用于理解系统如何随时间变化。

静态 TM 模型：图6展示了静态 TM 模型。接收硬币会触发（虚线箭头编号1）释放锁，使其处于开启位置（2）；推入闸机（3）会触发锁移动到锁定位置（4）。

动态 TM 模型：要构建此模型，我们需要定义什么是事件。TM 事件由静态图的一个子图（称为区域）和时间构成。图7展示了硬币投入旋转闸机的事件。为简化起见，我们用该事件的区域来表示事件。图8展示了包含四个事件的动态系统，E_1–E_4。图9展示了以事件序列表示的旋转闸机系统行为模型。

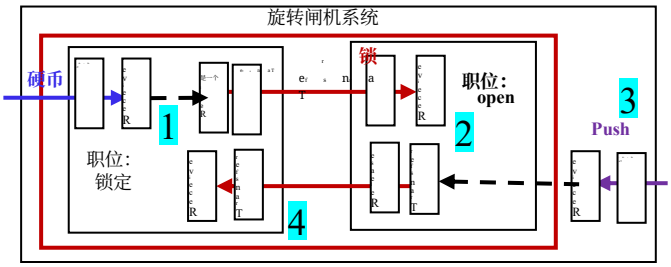

图6 旋转闸机系统的 TM 静态模型。

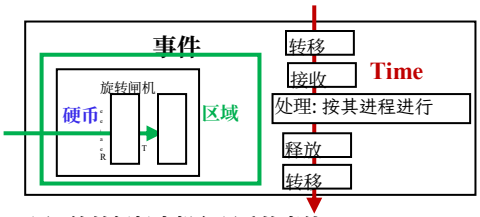

图7 旋转闸机中投入硬币的事件

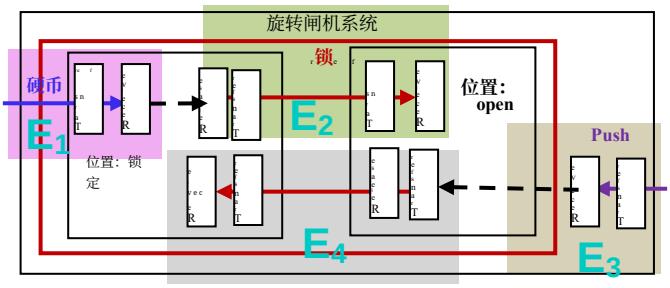

图8 旋转闸机系统的TM动态模型。

D. 解答：单一多层模型 Th

垂直建模的理念 ion 本文的工作采用基于“一切皆为事件”而非“一切皆为对象。”这种方法称为仅发生建模，其中发生指的是一个事件或处理；处理（首字母大写，以区别于 TM 中作为动作的“过程”）被定义为一组构成整体的事件（即一个高层事件）[15]）。事件的’存在（表示一次发生）由时间和“区域

在 TM 中，对象不过是持续时间较长的事件。其底层范式包括：(a) 斯多葛本体论，包含存在与存续两种存在类型；(b) TM 将活动限定为五种通用动作；以及 (c) 卢帕斯库逻辑，用于处理负事件。

本文的工作聚焦于 TM 建模。我们认为，作为一种单一多层模型，TM 有望成为几乎所有现有建模语言的统一图示语言。为证实这一观点，我们在 [4] 中几乎用 TM 重新绘制了所有图示表示。图 10 展示了重新绘制后的图示。结果似乎表明，用 TM 表示这些图示并不存在困难。请注意，图 10 中还包括 BPNM、实体关系图和顺序图，这些图示来自先前的出版物，且此前已用 TM 表示。

$$E_1 \longrightarrow E_2 \quad E_3 \longrightarrow E_4$$

图*9* 旋转闸机系统的*TM*行为模型。

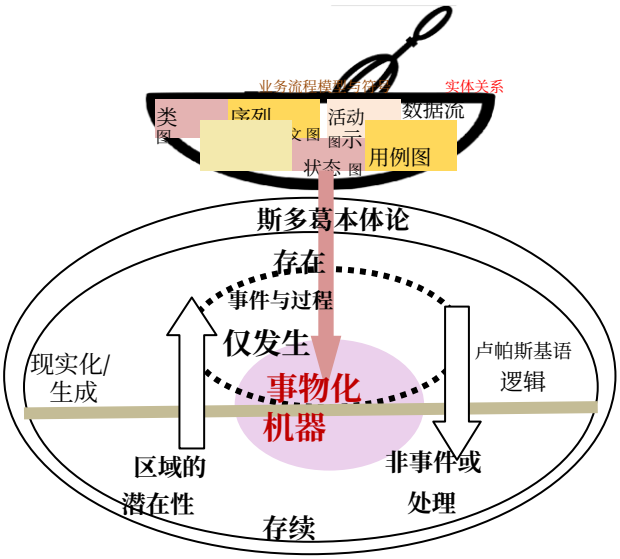

图10 将图表转换为TM的通用框架。

E. 论文结构Th

本文包含对仅发生建模框架（包括 TM）的一般性回顾。第 3 节和第 4 节将以下建模工具图转换为 TM：文本需求、上下文建模、类图、用例、数据流图、活动图和状态图。

II. 仅发生CCURRENCE-ONLY 建模ODELING

仅发生概念建模建立在三个基础之上： [15], TM、斯多葛本体论和卢帕斯库逻辑。本文重点介绍 TM 材料。有关斯多葛本体论和卢帕斯库逻辑的更多信息，请参阅 l器），因为它被概念化为兼具事物与机器的双重性质。机器由五

TM 机器的操作描述如下。

1) 到达：一个事物到达一台机器。

2) 接受：一个事物进入机器。为简化起见，我们假设到达的事物都会被接受（见图 11）；因此，我们可以将到达和接受阶段合并为接收阶段。

3) 释放：一个事物已准备好转移到机器之外。

4) 处理：一个事物被改变、处理和检查，但不会产生新的事物。

5) 转移：事物输入到机器中或从机器中输出。

6) 创建：在机器中显现一个新事物

此外，TM 模型包含一种触发机制（在本文的图中以虚线箭头表示），它会启动从一台机器到另一台机器的（非顺序）流。此外，每个动作都可以拥有自己的事物存储器（在 TM图中以圆柱体表示）。为简化起见，在某些图中可以省略创建，因为表示事机的方框已隐含其存在（于模型中）。

三、STATIC MODELS

从本节开始，我们将引言中讨论的图示转换为 TM 图。我们的目的是证实这一主张：作为单一多层模型，TM 有可能成为几乎所有现有建模语言的统一图示语言。

据 [11], 所述，自“万物皆对象”这一概念诞生以来，统一化原则一直是推动简单性、通用性和集成能力的最有力引擎。对象这一概念是一种统一数据与处理的方案。据[4], 所述，为了使建模的复杂性易于管理，系统的各种视图通过图来表示。每种图都基于特定的图类型，而图类型又通过建模语言来定义。需求建模可采用多种图类型及其相应的建模语言。

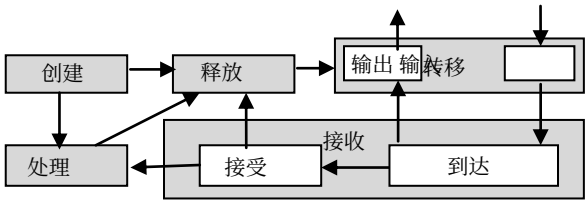

图 11 物化机。

图类型的选择取决于其用途，而用途决定了应记录哪些具体的系统需求，以及哪些人是需求模型的“目标受众”。所采用的图类型包括可用于建模处理相关方面的图，例如业务分析中的事件驱动过程链或BPMN图，以及UML活动图[4]。

A. 文本与图表

[4] 中的图 12 展示了“文本”需求与“建模”需求之间的区别，图 13 展示了相应的静态 TM 模型。在图 12 中，左侧面板展示了文本需求，说明了如何通过录入界面输入数据所需的行为。右侧面板展示了建模后的需求，其中“模型被视为系统属性的一种抽象映像”[4]。

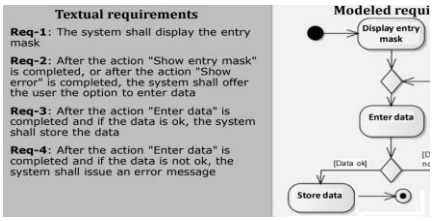

图 12 [4] 给出的需求示例。

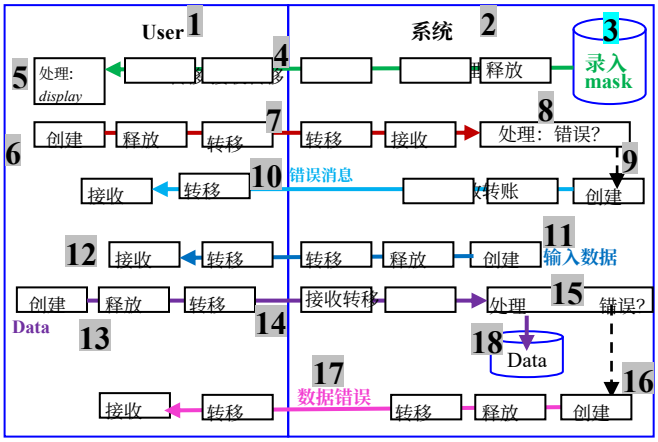

图 13 给定需求的静态 TM 规范。

这个示例表明，“建模需求明确呈现了所需系统行为各个方面之间的交互，而这些信息在文本需求中仅隐含存在”[4]。

图13是与图12对应的静态TM图，涉及用户（图中编号1）和系统（2）。录入掩码（3）被检索并发送给用户（4）以显示（5）。用户应输入某些内容（6），该内容流到系统（7）进行检查（8）。如果输入错误，则触发（9）发送给用户的错误消息（10）。否则，要求用户（11）输入数据（12）。用户输入数据（13），该数据移动到系统（14）进行检查（15）。如果数据中有错误，则创建数据错误消息（16）发送给用户（17）；否则存储数据（18）。

图13是基于静态规范的工程模式，构成动态模型的基础。如前所述，为简化起见，事件将由其区域表示。以下更有意义的事件比图14所示的通用事件（五个通用动作的事件）更优；因此，我们列出以下事件。

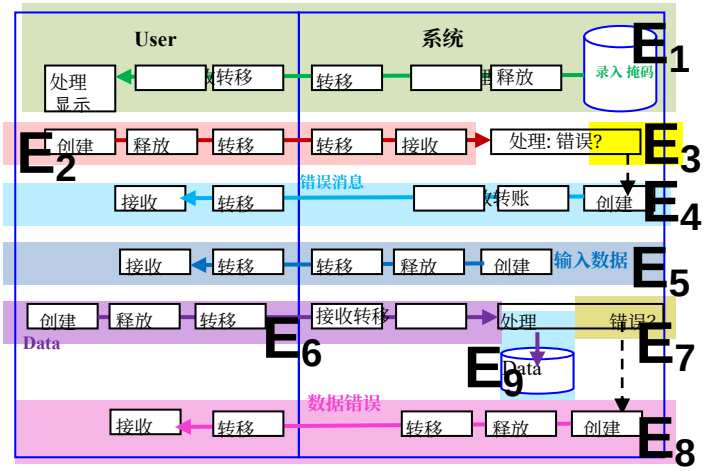

图 14 动态图

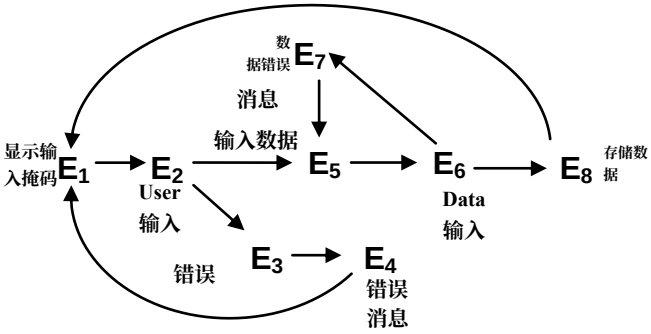

图15 行为图

E_1：系统向用户发送要显示的录入掩码。E_2：用户向系统发送输入。E_3：系统在输入中发现错误。E_4：向用户发送错误消息。E_5：系统请求录入数据。E_6：用户发送数据。E_7：系统在数据中发现错误。E_8：系统发送数据错误消息。E_9：系统存储数据。

Fi图 15 展示了行为图。

B. 上下文建模 A

需求工程的challenge一项工作是理解待开发系统的上下文。上下文图用于识别待开发系统与其上下文之间必要的接口。上下文是系统环境的一部分，与系统及其需求的沟通相关 [6]。

图 16 展示了一个使用结构化分析绘制的上下文图示例。图 16 中的整体系统（矿业预警系统）以圆圈表示，置于图中央。人以简笔人物表示，组织和技术方面的相邻系统则以方框表示。接口通过与相邻系统之间的数据流来建模 [4]。 图 17 展示了使用 TM 建模绘制的相应上下文图。我们采用了一种简化方式，没有显示操作“发布”“转移”和“接收”。

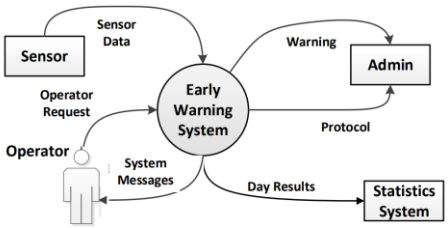

图 16 上下文图（摘自 [4]）。

C. 类图

根据类[4]是一种模板，用于定义多个对象的共同属性。这些对象随后被称为这些类的实例。图18中的类图包含一种“部分/整体”UML 关系。为求简洁，请勿将起始点（见图18）纳入相应的模型（图19）。

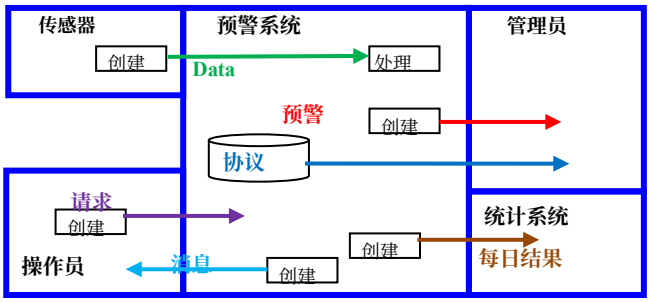

图 17 对应的上下文图

1) 静态模型

静态 TM 模型中，路线包含目的地，作为其组成部分，不能脱离特定路线而独立存在。图中的兴趣地点与路线 之间涉及所谓的聚合关系：一条路线可以有零个（空值）或多个兴趣地点，而一个兴趣地点至多对应一条路线（在同一时间）。

构建路线的过程从粉色数字1,开始，并假定路线 2包含数据：路线3的标识符，以及其他属性4（图中未显示），还有作为路线的子 Thimac 的目的地 5。构成路线复合部分的这些字段6 将由兴趣点 7 补全。因此，目的地是路线的一部分，会随路线的存在而存在，也随其消亡而消亡。

兴趣地点 7是路线的一部分，并按如下方式注入路线中。兴趣地点被8接收（独立于路线），并经9 处理，以提取**兴趣地点**标识符10及其路线标识符11。我们假设兴趣地点数据包含其将被插入的路线标识符。这些数据是必要的，目的是，

1) 将兴趣地点插入其路线12 （标记为分配路由地点的棕色框）。

2) 按如下方式实施这些约束。

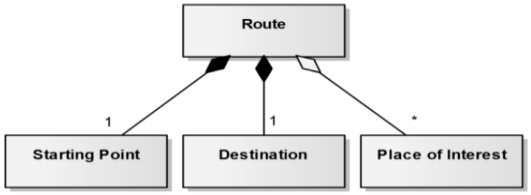

图 18 示例类关联（来自 [4]）

强制要求一个兴趣地点只能关联一条路线： 路线标识符和兴趣地点标识符 13 经处理 14 后，创建元组（标识符，识符）15，并将其流向系统16。在那里，该元组（标识符，标识符）会与表18中的类似元组进行比较17；该表记录了路线标识符与兴趣地点标识符之间所有当前已存在的关系。因此，如果给定的新目的地已经关联了另一条路线，就会发生错误 19；否则，20如果该元组（标识符，标识符）尚未存在于表中，就将其添加进去。因此，向路线中插入兴趣点的操作由此完成

图19可以更详细地展开（例如，搜索（标识符，标识符）表），也可以轻松简化。

2) 对应的动态模型

该模型假设路线数据存储在一个文件中，其中包含兴趣点的多个字段。

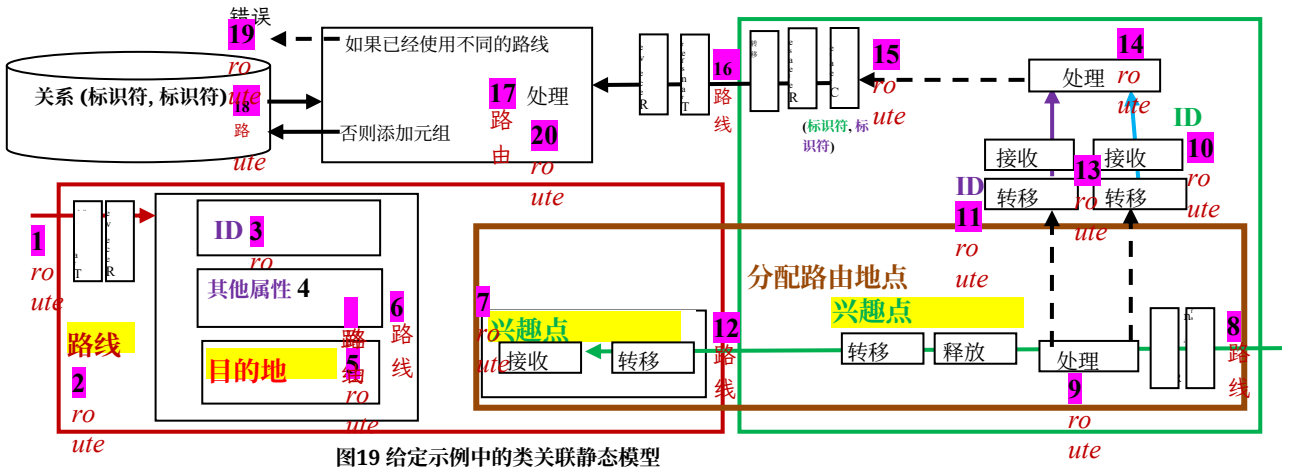

图19 给定示例中的类关联静态模型

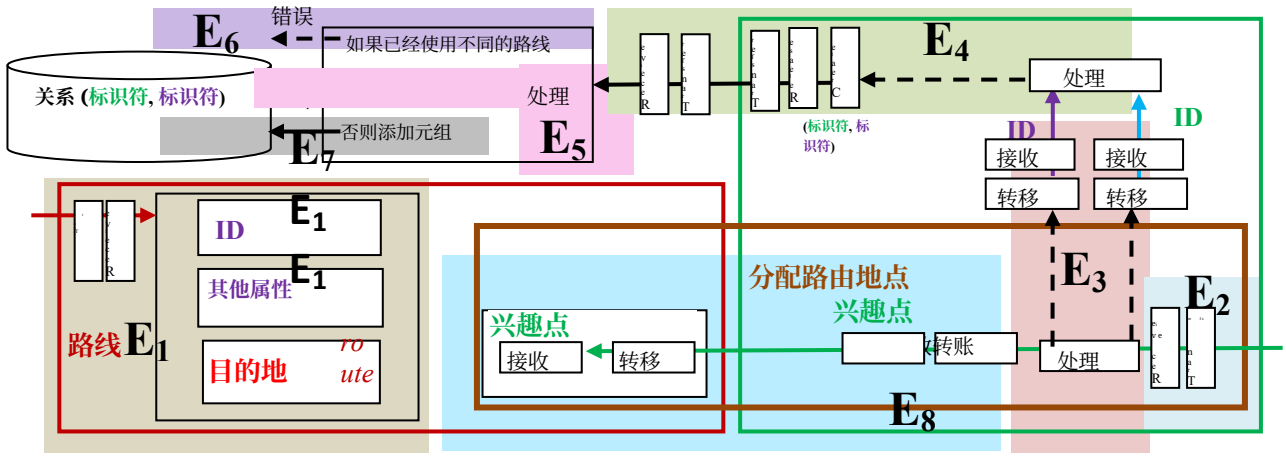

图 20 给定示例中的类关联的动态模型。

动态模型描述了依次输入一条包含兴趣点的路线。所选事件如下：

E_1：输入路线，包括目的地，E_2：输入兴趣地点，E_3：提取路线及其兴趣地点的两个标识符，E_4：构建标识符元组，E_5：在标识符元组中搜索，确认兴趣地点是否已与其他路线关联，E_6：如果该兴趣地点已与某条路线关联，则发出错误消息，E_7：如果标识符元组尚不存在，则将其添加到表中，E_8：将兴趣地点插入路线中。图 21 展示了此处理的行为模型。

IV. Dynamic Modeling

根据[4]，，动态视图“为指定功能的不同方面提供了许多不同的抽象标准。”文献 [4] 提供了以下动态视图类型：用例视图、面向控制流的视图、面向数据流的视图、面向状态的视图和场景视图。

A. 用例

图 22 展示了一个用例. 图 23 展示了相应的 TM 模型。值得注意的是，这两种表示形式之间的差异并非只是将简笔人替换为方框。在 TM 建模中，图 23 中的图示是基础模型，随后会在行为层面不断增强；相比之下，简笔人表示法会在后续建模阶段消失。

B. 数据流图

根据 [4]，，数据流图常用于从数据流导向的视角对需求进行建模。此类图可以在 TM 模型的不同层级中轻松构建。

C. 活动图

根据 [4]，，UML活动图可以从控制流的角度对需求进行建模，且这些图主要用于相关人员之间的沟通：“通过补充活动描述，可以确保规格说明的完整性。”

文献 [4] 介绍了一种活动图模型，该模型与需求工程相关：可中断活动区域，即用户通过点击“取消”来终止活动。不过，该示例并不清晰；因此，我们用一个取自 [24] 的示例来说明可中断活动的概念，如图 24 所示。

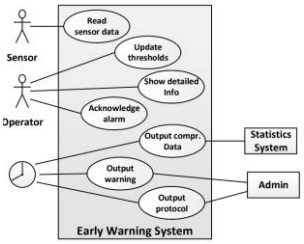

图 22 示例用例（来自 **[4]**）。

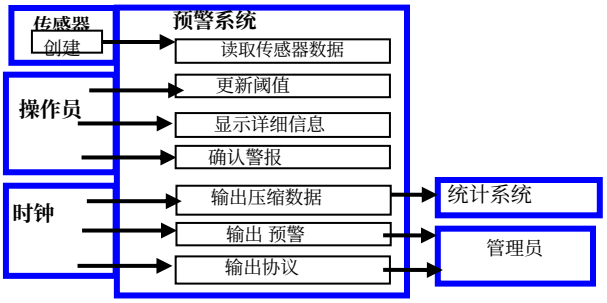

图 23 与给定用例图相对应的 TM 模型。

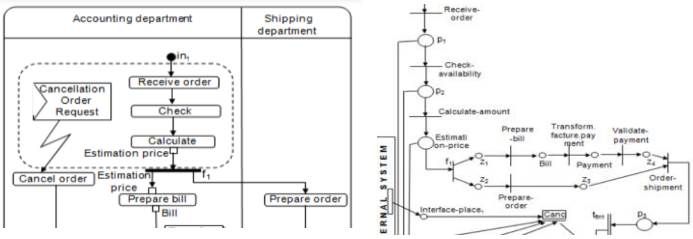

图 24处理订购商业活动的活动图，包括可中断区域（左侧）及其佩特里网取消事件。

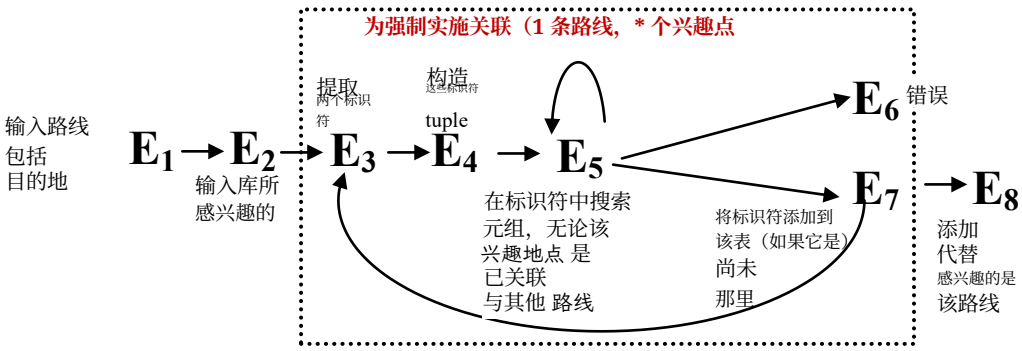

图 21 构建一条路线的行为模型。

该示例涉及一个商业订单流程，其中属于该系统的客户可以介入并停止订单处理程序。文献 [24] 使用订单处理活动的活动图对这一情形进行建模，其中包括可中断区域（左侧）及其对应的佩特里网取消事件（右侧）。

图 25 展示了相应的 TM 模型。首先，一项订单请求到达系统（粉色数字 1）。系统会处理该请求以进行检查（2）。假设请求没有问题，系统便创建订单（3），并据此创建一份发送给客户的估算价格（4、5）。为节省篇幅，本文不再描述订单流程的其余部分。此外，客户可以通过取消信号（6）中断订单操作；该信号经过处理（7）后，可在任一阶段取消订单进程。

图 26 展示了该订购系统的动态。图 27 展示了行为系统。请注意，图 27 中的 R_i 表示 E_i 的区域，即负事件“非（或停止）R_i”；如第 2 节所述，该事件会回退至静态层级。在行为模型中，如果取消事件 R_7 与任何其他事件同时发生，该事件就会回退至静态层级，从而停止订单处理的所有进程。图 28 展示了此类系统中的事件进展。

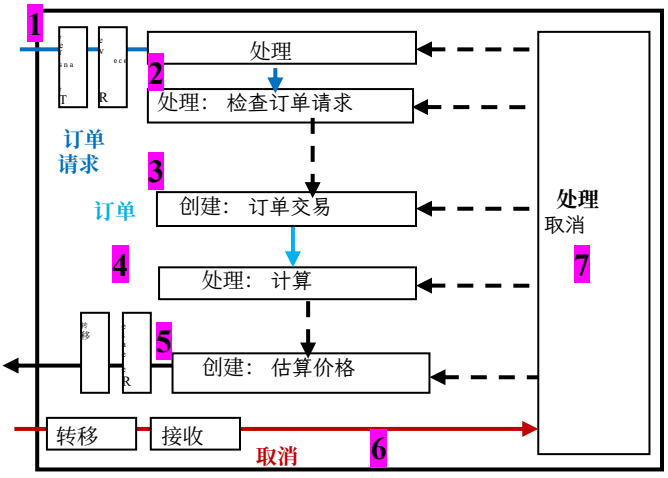

图 25 客户订单各阶段的静态模型

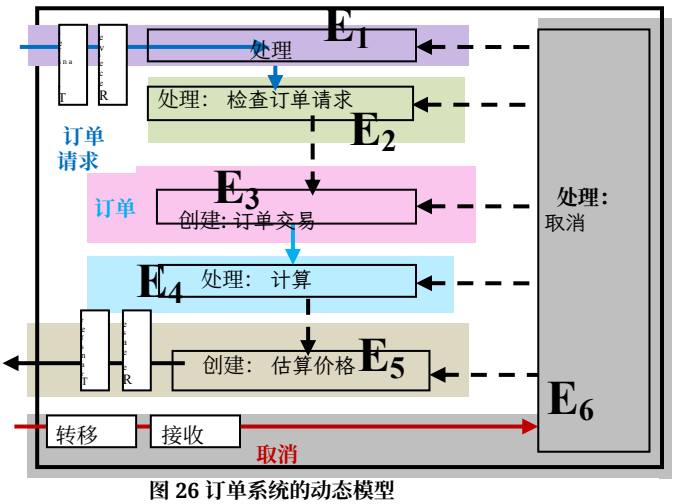

图 26 订单系统的动态模型

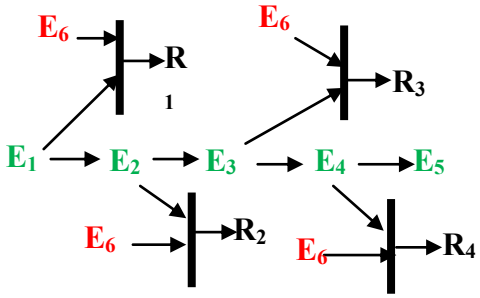

图 27 订单系统的行为模型。

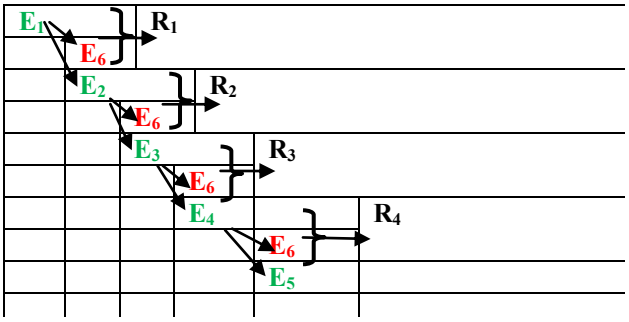

图 28 事件进程示意图

D. 状态图

根据 [4]，，系统的状态空间在动态视图中的面向状态的视图内建模。此视图对系统与系统上下文之间的接口处可观察到的状态及状态变化进行建模。系统的状态变化可由事件触发。参考文献 [4] 给出了面向业务的系统中请假申请对象状态的一个典型示例。本文作者无法仅凭该示例的有限细节完全理解其内容。然而，文献中包含许多面向业务的系统中的请假申请状态图，我们选取了一个示例（图 29），该示例改编自 [25]，以展示我们的目标。

图 29 中的状态图涉及以下过程：员工提交请假申请，主管批准工时表，人力资源部门审核并最终作出决定。状态包括：正在起草申请、等待经理批准、已退回、等待人力资源部门批准、已批准或已拒绝。

图30显示了相应的 TM 模型。首先，员工提交休假申请（蓝色编号1），并将其发送给管理器（2）。

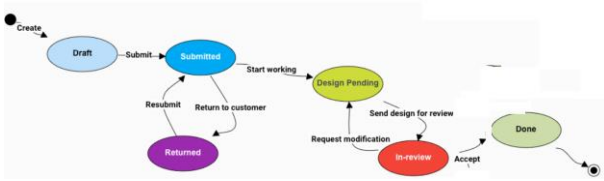

图 29 请假申请的状态（引自 [25]）。

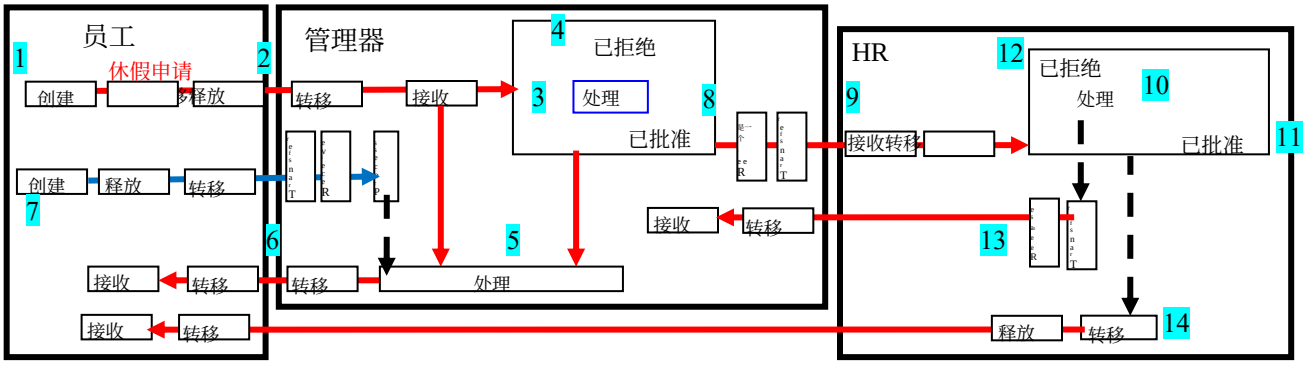

图30 静态模型

管理器处理（3）该申请，可以拒绝（4）该申请，也可以将其退回给员工（5和6）。员工也可以撤回（7）休假申请，并将其取回。我们假设此撤回操作会中止申请处理。管理器可以批准（8）休假申请，并将其转交给人力资源（9）。人力资源处理（10）休假申请，可以批准（11）或拒绝（12）该申请，然后将其退回给管理器（13），或发送给员工（14）。

图31和图32分别展示了动态模型和行为模型。

V. 结论ONCLUSION

统一化在需求工程建模中的重要性几乎无需论证。这种统一建模能够使领域系统化，并最大限度地减少内部不一致。占主导地位的 UML 方法旨在“统一表示法，将若干其他表示法中的精华以及当前最佳实践纳入一种普遍适用的表示法”[26]。本文试图展示另一种统一化：通过统一模型连贯地描述彼此无关的概念，并揭示人们认为互不相关的现象之间的联系。

在本文中，我们重点关注了这一背景下的两种方法：应用同时使用不同表示法的多建模与利用单一表示法框架的统一建模。当然，该问题需要进一步研究，但 TM 模型显然是实现替代类型建模统一化的可行方法。

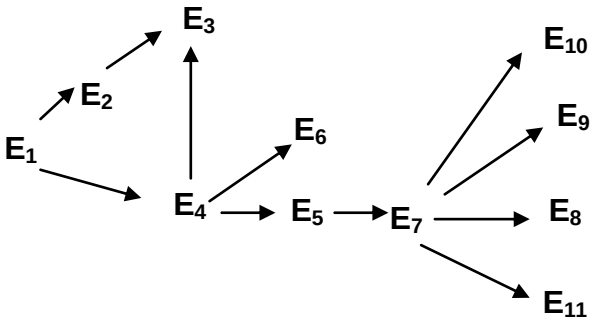

图32 行为模型

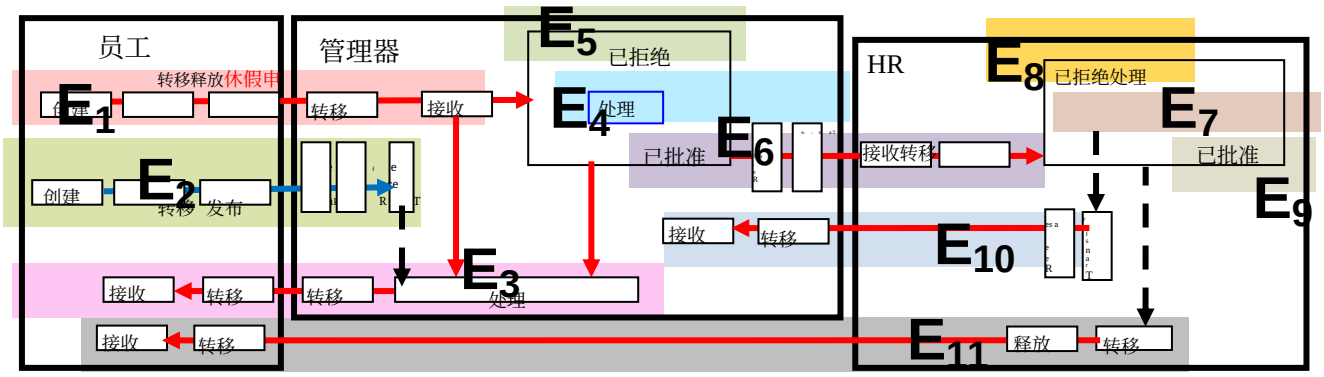

图31 动态模型

References

[1] R. Lukyanenko、V.C. Storey 和 O. Pastor，“系统：一种用于捕获复杂性的核心概念建模构造”，《数据知识工程》，第 141 卷，2022 年 9 月。[2] M.Q. Pattor，《定性研究与评估方法》，第 3 版，千橡市：Sage 出版社，2002 年。[3] [illegible] aeten 和 M. D'Hondt，“使用转换依赖分析检测并解决模型不一致性”，载于《模型驱动工程语言与系统》（MODELS 2006，《计算机科学讲义》第 4199 卷），O. Nierstrasz、J. Whittle、D. Harel 和 G. Reggio 编，Springer，柏林，海德堡，2006 年，第 200–214 页。[4] T. Cziharz、P. Hruschka、S. Queins 和 T. Weyer，《根据 IREB 标准的需求建模手册：IREB 认证需求工程专业人员高级水平“需求建模”的教育与培训》，第 1.3 版，（译：E. van Akkeren 等）IREB，2016 年 8 月。https://www.gasq.org/files/content/gasq/downloads/certification/IREB/IREB%20AL/ireb_cpre_handbook_requirements- modeling_advanced-level-v1.3.pdf [5] T. von Ramsch，“经典需求与测试分析：一种已停用的模型？”，《需求工程杂志》，2023 年 1 月 25 日。[6] M. Glinz、H. van Loenhoud、S. Staal 和 S. Bühne，《根据 IREB 标准的 CPRE 基础水平手册》，第 1.0.0 版，2020 年 11 月。可在 https://www.gasq.org/files/content/gasq/downloads/certification/IREB/IREB%20FL/cpre_foundationlevel_handbook_en_v1.0.pdf 获取 [7] M. Glinz 和 S. Fricker，“软件工程中的共享理解”，在德国亚琛 GI 软件工程研讨会上发表，2013 年 3 月。[8] J. Beatty 和 A. Chen，《需求建模语言导论：[illegible]). Keler[illegible]，工业工程[illegible]，2013 年。[10] G. O'Keefe，《UML 模型的含义》，博士学位论文，澳大利亚国立大学，2008 年 9 月，2010 年 2 月修订。https://openresearch-repository.anu.edu.au/bitstream/1885/49319/2/02whole.pdf [11] J. Bézivin，“论模型的统一化力量”，《软件与系统建模》第 4 卷，第 171–188 页，2005 年。[12] G. Karsai，《统一化还是集成？异构建模中的语义挑战》，发表于第二届全球建模语言国际研讨会，西班牙瓦伦西亚，2014年9月28日。[13] S. Al-Fedaghi 和 G. Aldamkhi，《IP 电话通信系统的概念建模：案例研究》，《国际交互式电信网络期刊》，第13卷第3期，第83–94页，2021年7月。[14] S. Al-Fedaghi，《基于卢帕斯库逻辑，使用物化机对系统事件和负事件进行建模》，arXiv 预印本，arXiv:2211.12586，2022年11月22日。[15] S. Al-Fedaghi，《对象导向的对立面：应用于工程与医学的仅发生建模》，arXiv 预印本，arXiv:2302.07087，2023年2月14日。[16] A. Shihadeh，《可能性的本体论》，载于《〈对伊本·西那的质疑：谢拉夫·丁·马苏迪对〈指示〉的评注研究与校订本》，第95卷，波士顿：Brill，2016年[17] C. Bock 和 J. Odell，《本体行为建模》，《对象技术期刊》，第 10 卷第 3 期，第 1–36 页，2011年。[18] V. de Harven，《虚无如何成为某物：斯多葛学派的虚空理论》，《古代哲学》，第35卷第2期，第405–429页，2015年。[19] LibreTexts，《有限状态机概述》，载于《离散结构》（在线版），圣路易斯大学，2021年4月30日。可访问：https://eng.libretexts.org/Under_Construction/Book%3A_Discrete_St ructures/09%3A Finite-_ State Automata/9.01%3A Introduction/9.1.01%3AFinite-_ _State Machine Overview [20] D. Wagner、N. Rouquette、S. Jenkins、M. Bennett、M. Ingham 和 R. Karban，《状态分析本体：形式化映射至 SysML》，发表于 IEEE 航空航天会议，美国大天空，2012年3月3日至10日。[21] OMG，《UML 状态机的精确语义（PSSM）》，版本1.0，2019年5月。可访问：https://www.omg.org/spec/PSSM/1.0/PDF [22] J. Kim，《心智与行为》，纽约：Routledge，2018年。[23] E. Boccardi，《计算外在论：实现的语义图景》，博士论文，伦敦政治经济学院，2008年1月。[24] S. Boufenara、K. Barkaoui、F. Belala 和 H. Boucheneb，《使用 TPNet 对 UML2 活动进行形式化：案例研究》，发表于第五届计算机通信系统验证与评估国际研讨会，突尼斯，突尼斯，2011年9月15日至16日。[25] RethinkWorkflow，《工作流》，可访问：https://support.rethinkworkflow.com/2018/12/ [26] J. Hunt，《统一过程指南：UML、Java 与设计模式》，第2版，伦敦：Springer，2003年。