计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.201200044

从信息物理融合系统问题模型到 UML 用例图的变换方法

李 智[1,2,3] 邓 杰[1,2,3] 杨溢龙[4] 韦尚锋[1,2,3]
1 广西师范大学计算机科学与信息工程学院 广西 桂林 541004
2 广西多源信息挖掘与安全重点实验室 广西 桂林 541004
3 广西区域多源信息集成与智能处理协同创新中心 广西 桂林 541004
4 北京航空航天大学软件学院 北京 100191
(zhili@gxnu.edu.cn)

摘 要 问题框架(Problem Frames,PF)方法在需求工程研究中已经获得广泛重视,特别是应用于基于环境建模的信息物理融合系统中,但如何有效地把问题模型(问题图及相关描述)转换为软件设计和实现,仍是一个亟待解决的问题。文中提出了一种问题框架与模型驱动技术相结合的方法,其可将问题模型转换为 UML(Unified Modeling Language)需求模型中的用例图和概念类图,进而指导软件系统的设计和实现。所开发的 CASE 工具,通过支持领域涉众和软件设计人员合作建模来提高需求质量,并允许从问题空间的需求模型平滑过渡到解空间的软件设计。通过文献[1]中一个基准案例(邮件分拣控制问题)的扩展版本,展示了其可行性及在实际应用场景下的使用,从而对推动问题框架方法从理论研究走向实际应用具有重要意义。

关键词:信息物理融合系统;问题框架;统一建模语言;需求工程;模型驱动工程
中图法分类号 TP311

Transformational Approach from Problem Models of Cyber-Physical Systems to Use Case Diagrams in UML

LI Zhi[1,2,3],DENG Jie[1,2,3],YANG Yi-long[4] and WEI Shang-feng[1,2,3]
1 College of Computer Science and Information Technology,Guangxi Normal University,Guilin,Guangxi 541004,China
2 Guangxi Key Laboratory of Multi-source Information Mining & Security,Guilin,Guangxi 541004,China
3 The Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing,Guilin,Guangxi 541004,China
4 College of Software,Beihang University,Beijing 100191,China

Abstract Problem Frames (PF) have attracted extensive attention and research from the requirements engineering community, particularly in the environment-based modeling of cyber-physical systems (CPS). However, effectively transforming the problem models (i.e., problem diagrams with associated descriptions) of PF into the design and implementation of software is still an open problem to be solved. This paper proposes an approach to automatically transforming problem diagrams into UML (Unified Modeling Language) conceptual class diagrams and use case diagrams, which can directly guide downstream designs and implementations. A method of combining PF and model-driven technology is proposed. This method, together with the developed tool support, improves the quality of requirements through collaborative modeling by stakeholders and software designers, thus allowing for the smooth transition from modeling in the problem space to software design in the solution space. This method is applied to the package router control problem, a benchmark case study in the PF literature, to illustrate its feasibility and how the work can be used in a practical setting, which plays an important role in promoting the PF approach from theory further into practice.

Keywords Cyber-physical systems, Problem Frames, Unified Modeling Language, Requirements Engineering, Model-driven Engineering

到稿日期:2020-09-04 返修日期:2020-10-28 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金项目(61862009);广西自然科学基金项目(2018GXNSFAA281314);广西研究生教育创新计划项目(JXXYYJSCXXM-001)
This work was supported by the National Natural Science Foundation of China (61862009), Provincial Natural Science Foundation of Guangxi (2018GXNSFAA281314) and Innovation Project of GuangXi Graduate Education(JXXYYJSCXXM-001).
通信作者:杨溢龙(yilongyang@buaa.edu.cn)

1 引言

信息物理融合系统(Cyber-Physical System,CPS)的一个显著特征是,除了包含起控制协调作用的软件部分(即 Cyber 部分),还包括将要被软件所控制或干预的物理世界(即 Physical 部分),如机电设备或人等。正如软件工程著名学者 Jackson[1] 指出,信息物理融合系统的原始需求存在于现实世界,而需求工程的主要任务是从这些原始需求出发导出描述软件部分的规格说明[1)]。Jin[2] 进一步提出把软件的物理环境作为需求工程建模的主要任务。鉴于此,学术界和工业界都急需一种能够对 CPS 的物理世界和信息世界的内在联系进行建模、分析、设计和自动生成软件原型的方法并提供工具支持。

一方面,Jackson 提出的问题框架方法[3] 充分考虑了上述 CPS 的特征,并提供问题图模型来着重刻画物理世界和信息世界的内在联系。在这方面最具代表性的是 Hall 等在文献[4]中所做的工作,所用的案例是 PF 文献中著名的基准案例-邮件分拣问题[1]。该工作把问题模型变换成为一个描述控制软件的结构模型——UML 类图。另一方面,近年来 Yang 等[5-7] 提出了 RM2PT 工具来支持快速生成软件原型的方法。该方法使用标准的 UML 需求模型作为输入来自动化生成原型系统。UML 需求模型由用例模型和领域模型组成,其中用例模型包括用例图(Use Case Diagram)、系统顺序图(System Sequence Diagram)和系统操作合约(Contracts of System Operations);领域模型由概念类图(Conceptual Class Diagram)来刻画。UML 需求模型的精准性将直接影响到生成的原型系统的精准性,由于其是通过软件设计者进行建模得到的,如果软件设计者对系统领域的上下文不够了解,所生成的 UML 模型与实际的领域涉众(Domain Stakeholders)的需求有偏差,则可能会导致系统开发成本增加。本文认为,RM2PT 为融合以上两个世界提供了一种切实可行的方法,即本文的工作通过工具支持进一步将 Hall 等的工作[4] 向前推进,不仅生成了概念类图,还生成了用例图,这样更有利于领域涉众确认并保证需求的准确性。

2 领域涉众和软件设计者合作建模过程

在本文讨论的软件建模过程中,领域涉众指具有软件应用领域专业知识的专家、工作人员等,他们对软件应用的上下文环境的内在规律十分熟悉,但往往不具备 UML 的知识;软件设计者指具备 UML 专业知识和软件开发及设计背景的软件从业人员。

2.1 上下文图

PF 是由软件工程著名学者 Jackson 提出的一种软件开发方法,其关注计算机外部的世界,并提供了一种图形化方法来描述软件与环境之间的交互。问题框架用问题图来描述问题,采用带有特殊含义的节点和边来展示问题,符合一般大众对复杂事物的描述和认知,因此是一种适合领域涉众对问题上下文建模的方法。一般通过上下文图来确定其位置,上下文图将现实世界划分为机器领域和问题领域,用实线表示领域之间的接口,即机器领域如何作用于问题领域,问题领域如何影响机器领域,上下文图中的所有领域都是物理的。机器领域在图中用带有双竖线的矩形框表示;设计领域是信息的物理表示,例如存储在磁性条码上或硬盘上的信息,在图中用带有单竖线的矩形框表示;问题领域在问题图中用矩形框表示。

下面以一个例子进行说明。本案例是各大超市普遍适用的一种销售点(Point-of-sale,POS),在该系统正常运行的情况下,购买商品的客户直接与 POS 进行交互而无需收银员的干预,其问题陈述如下。

"某一超市需要安装使用一个全新的销售点系统,该系统包括将要开发的软件和从第三方购买的一些硬件设备,包括读卡器、具有验钞功能的现金接纳设备和找钱等硬件设备,还有触摸显示屏以及打印收据的设备等。要解决的问题是,当客户把一系列要购买的商品呈现给该销售点系统后,他/她付钱后应该得到一个与所呈商品等价的收据"。抽象出客观事物构成上下文图,如图 1 所示,图中的连线表示领域间的共享现象,具体解释如表 1 所列。

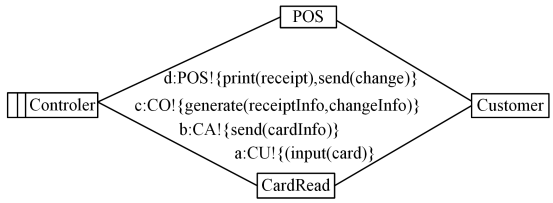

图 1 POS 上下文图

Fig. 1 Diagram of POS context

表 1 问题图 1 上的共享现象及描述

Table 1 Shared phenomena and their descriptions in problem diagram of Fig. 1

共享现象	描述
a	顾客将银行卡插入读卡器中
b	读卡器读取银行卡信息并传入系统
c	系统将找零信息和收据信息传给 POS
d	POS 打印收据并找零

在问题框架中,将问题领域分为 3 种类型[1]。第一是因果(Causal)领域,其特性在于其因果现象之间存在着可预见的因果关系,根据这些因果关系可以将复杂的问题拆分为多个子问题[8-9]。第二是可叫牌(biddable)领域,一般由人组成,虽然没有明确的可预见的内部因果性,但可通过培训使其外部行为具有因果性。第三是词法(lexical)领域,它是数据的物理表示,因此也具有因果特性,例如它包含的数据可被写入和读取。

2.2 问题图

问题框架的上下文图描述了问题在何处,通过添加需求来扩展成为问题图。而问题图主要回答了问题是什么,从而为问题的分析建立一个初始点。需求在问题图一般用虚线椭圆框表示,其与领域间的关系用虚线表示,虚线描述了引用关系,而带箭头的虚线表明不仅存在引用关系,还存在约束关系。因此,问题图是领域涉众对问题上下文和需求两者内在关系的一种建模方法。

下面给 POS 上下文图添加收银找钱需求(REQ),顾客先将银行卡插入读卡器中或将现金放入 POS 收银系统终端中。

1) 对很多软件设计者来说,规格说明常被称为"需求",实际上是一种描述软件应该做什么的"软件需求",本文用"UML 需求模型"来表示

收银系统获取到卡信息或现金后，对照用户购买的商品信息，将处理后的收据和找零交于用户，POS 问题图如图 2 所示。

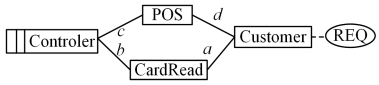

图 2　POS 问题图

Fig. 2　Diagram of POS problem

2.3　UML 图

UML[10]是软件开发中常用的一种统一建模语言，用于说明、可视化、构建和编写正在开发的、面向对象的软件密集型系统的开发方法。UML 为软件开发和设计人员提供了一种建模方法，本文从中选择最常用的类图和用例图对软件需求进行建模。

本文将问题框架方法与模型驱动思想结合。抽象外部客观事物构成上下文图，并从领域涉众和软件设计者的角度分别构建问题图和 UML 图。首先，构建包含有问题图和 UML 图的元模型。然后，利用 EMF[11]（Eclipse Modeling Framework）和 Sirius[12]技术将元模型视图化，从而生成面向领域涉众和软件设计者两种不同建模者的合作建模工具平台，合作建模工作完成后会生成 XMI 格式的文档。最后，利用基于 ATL[13]的模型转化技术，将其转换成 UML 需求模型中的用例图和概念类图，从而实现了用户需求到软件设计与实现的过渡。本合作建模思想如图 3 所示。

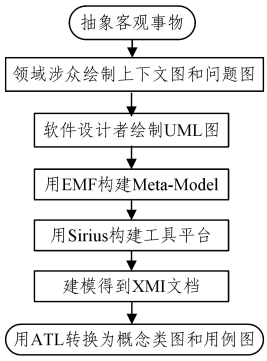

图 3　合作建模流程图

Fig. 3　Flowchart of cooperative modeling

3　合作建模工具平台

本节主要介绍了模型驱动工程（MDE），以及利用 MDE 的思想构建合作元模型和合作建模工具平台。

3.1　模型驱动工程

为了降低大规模软件系统开发的成本和应对企业技术的快速变化，2002 年由 OMG（Object Management Group）组织构建了模型驱动开发的软件设计架构（Model Driven Architecture，MDA[14]）。使用 UML 和其他相关的 OMG 建模标准来构建独立于开发平台的模型（Platform Independent Model，PIM），然后再设计出适用于特定平台的模型（Platform Specific Model，PSM），从而将软件系统的业务和应用程序逻辑与底层平台技术分离开，降低了两者的耦合。

3.2　构建合作建模元模型

在问题框架方法中，用问题图对问题进行刻画，抽象出共性，构建出元模型。问题图由节点和边构成，每条边包含两个节点，边的端节点用 to 表示，而源节点用 from 表示。节点包含需求（Requirement）、机器领域（Machine）、领域（Domain）、词法领域（Model）和用于存储信息的属性（Attribute），都带有 name 和 type 属性。其中，领域包含因果领域和可叫牌领域，故其类型是一个枚举类型。属性用虚线矩形框表示，用来存储问题图中其他节点的详细信息。边包含 3 个子类，其中实线（Shared_line）用来描述领域之间的关系；虚线（Reference_line）用来描述领域和需求之间的引用；带箭头的虚线（Constraint_line）用来描述领域和需求之间的约束引用。UML 需求模型中的概念类图显示了模型中存在的类、类的内部结构以及与其他类之间的关系，与问题领域存在相似性，故概念类图可以从问题图的领域部分获取。我们给 Domain 添加与类有关的属性（ClassAttribute）和关联关系（Reference）元模型元素，并为其设置详细的值。

UML 中的用例图由参与者（Actor）、用例（Use Case）、边界以及它们之间的关系构成，用于描述系统功能。在问题框架中，Actor 被包含在问题图的领域中，Use Case 可以抽象为需求，故在元模型中增加 Domain 和 Requirement 的关系即可。合作建模元模型如图 4 所示。

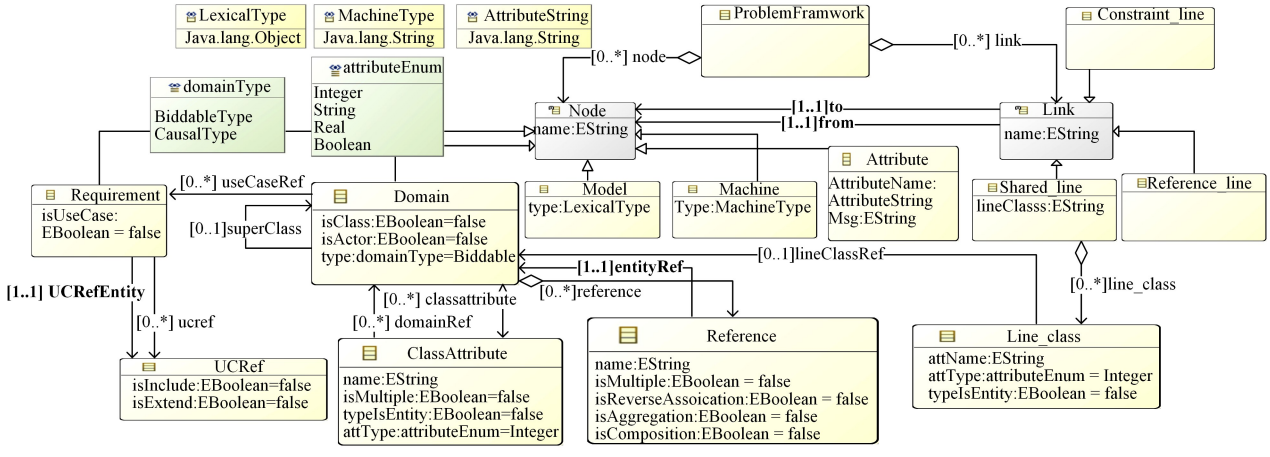

图 4　合作建模元模型

Fig. 4　Cooperative modeling metamodel

3.3 使用 EMF 和 Sirius 实现工具平台

MDE 定义了许多与模型有关的规范，而 EMF 为这些模型规范提供了一种 Java 语言的实现方式。EMF 是一种强大的架构和代码生成工具，可用于构建模型对应 Java 的应用程序，在建模人员和 Java 程序员之间架起了一座桥梁。EMF 还提供了一种可扩展的插件开发方式，后续使用的 Sirius 和 ATL 技术都是基于 EMF 扩展的插件。

近年来，许多学者[15-17]为设计可执行的领域特定建模语言(Domain Specific Language，DSL)做了大量的工作。在 MDE 中创建 DSL 通常称为领域特定建模(Domain Special Modeling，DSM)，因为 DSM 构建图形工具通常是一个非常复杂的工作，从零开始构建编辑器几乎是不可能的。Viyović 等[18]开发用于 DSM 图形编辑器的 Sirius 框架，然后对比了 Graphiti 和 Sirius 的框架[19]，发现基于 Sirius 的框架有着更好的特性。这意味着 Sirius 框架是创建 DSM 编辑器的更好选择，因此选择 Sirius 技术来解决我们的问题。使用 Sirius 技术生成编辑器主要分为以下 3 步。1)创建 EMF 模型，即领域特定建模 DSM。该步骤在第 3.2 节中已经完成，并得到了合作建模元模型。2)生成 Model，Edit 和 Editor。创建了元模型后，可以根据元模型文件生成. genmodel 文件。genmodel 是元模型文件的映射，利用其可以生成. model，. edit 和 . model. editor 项目包，再以 Eclipse Application 方式启动项目。最后可以在 Runtime 工作空间中测试生成的模型是否完整无误。3)构建图形编辑器。根据问题图的绘图标准，用 Sirius 技术来构建图形编辑器。Sirius 的最大优势在于其允许开发人员以图形化的方式来指定编辑器，而不需要了解任何后端的过程。

4 ATL 转换规则

本节主要介绍利用 ATL 技术将合作建模元模型转换成概念类图和用例图的转换规则。

4.1 ATL 转换规则

ATL[20]是 AtlanMod 研究组受 OMG 的 QVT(Query/view/Transformation)标准启发而创立的一门面向模型到模型(Model to Model，M2M)转换的领域语言。

本文用的源元模型如图 4 所示，转换的目标元模型是 RM2PT 元模型。转换后生成的概念类图和用例图用于辅助 RM2PT 生成原型系统。为此，我们需要从问题图中筛选出 UML 用例图及概念类图，具体做法如下：用例图中的参与者(Actor)可以从问题图中的问题领域选择，这需要软件设计者判断哪些领域适合作为用例中的参与者，同时用例名称可以从问题图中的需求获得，这样我们就可以得到用例图；概念类图中的类可以从问题图中的问题领域中选择，软件设计者需要判断哪些问题领域适合作为概念类图中的实体类，把领域的名称及属性作为概念类的名称及属性，这样就得到了概念类图。

4.2 概念类图转换规则及算法描述

这里的源元模型是上文中的合作建模元模型，目标元模型是来源于 RM2PT 工具元模型中的关于概念类图部分，用 EcoreTool 展示，如图 5 所示。该模型用一个 DomainModel 包含所有的 Entity，每一个 Entity 相当于概念类图中的类，包含有 name 和 isCRUD 属性。类与其他类的关系用 Reference 进行描述，关系分为单向关联关系(isAggregation)、双向关联关系(isReverseAssoication)、组合关系(isCompostion)、一对多关系(ismultiple)，此外还有类与类的继承关系(superEntity)。类中还包含属性(Attribute)，属性类中有 name 和 isMultiple 属性，属性类有属性对应的类型 typeCS，其包含 3 个子类型：集合类型(CollectionTypeCS)、实体类型(EntityType)和枚举类型(EnumEntity)。

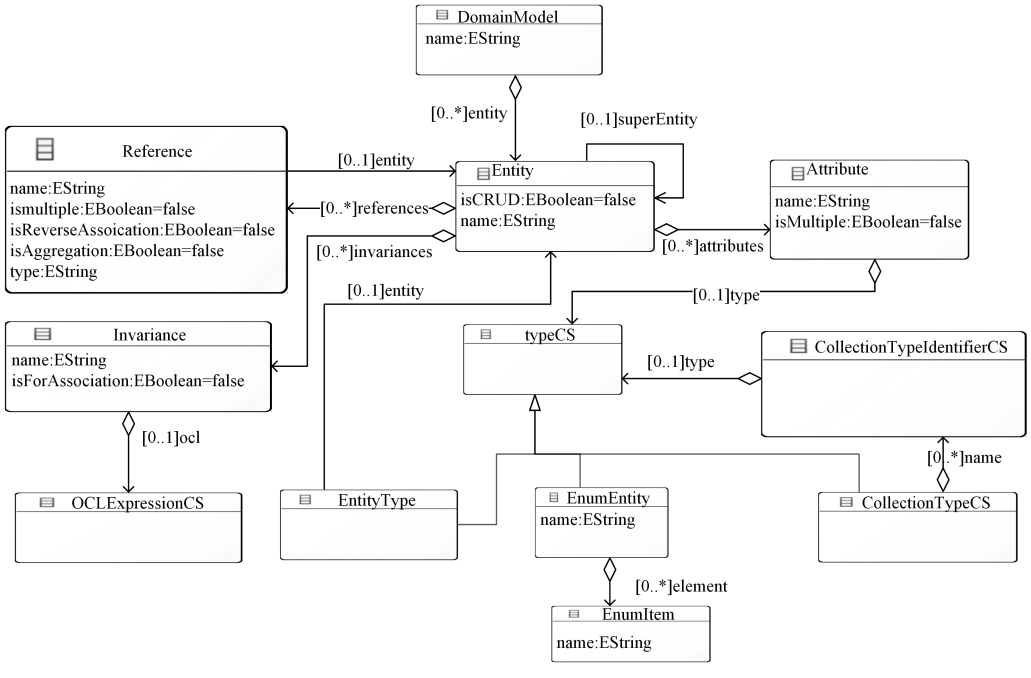

图 5 概念类图目标元模型

Fig. 5 Target metamodel of conceptual class diagram

获取到源元模型和目标元模型后，分析真实业务需求，设计对应的转换规则。首先获取所有领域并分别判断是否将其划分为一个类，这里使用了布尔型值 isClass 进行标识，如果为 true 则进行下一步，软件设计者填充其属性和与其他类的关系。在填充类的属性时，先确定属性的名称，再确定对应的类型（包含 Integer，Real，String，Boolean 及自定义的枚举类型等）。在判定类之间的关系时，确定其与其他类的关系（包括双向依赖关系、聚合关系、组合关系及数量上的一对多和多对多关系等）。PF2ClassDiagram 算法如算 1 所示。

算法 1　PF2ClassDiagram

```
1. --@path RE=/PF2UML/metamodels/REMODEL. ecore
2. --@path PF=/PF2UML/metamodels/pf. ecore
3. module PF2ClassDiagram
4. create OUT:RE from IN:PF
5. helper context Domain def:getAllClassAttri():Sequence(Attrib-
   ute)=self. classattribute
6. rule One
7.     from ProblemFramewrk(pf) to DomainModel(re)
8.       re. name<-re. name
9.       re. entity<-Domain. allInstances()->select(a|a. isClass).
         asSequence()
10. end rule One
11. rule Two
12.   from Domain (pf. isClass) to Entity(en)
13.     en. name <-pf. name
14.     en. attributes <-pf. classAtttibute
15.     en. reference <-pf. reference
16.     en. superEntity <-pf. superClass
17. end rule Two
18. rule Three
19.   from Domain (pf. superClass. oclIsKindOf(Domain)) to Entity
      (en)
20.     en. name <-pf. superClass. name
21.       end rule Three
22. rule Four
23.   from ClassAttribute(Domain. allInstances()->collect(a|a.
      getAllClassAttri(). includes(pf))) to Attribute(att)
24.     att. name <-pf. name
25.     att. type <-pf. attrType
26.   from ClassAttribute(pf) to EnumEntity
27.     name<-if(pf. attType. toString()='Integer')
28.       then 'Integer
29.       else if(pf. attType. toString='Real')
30.         then 'Real'
31.         else 'String'
32.         endif
33.       endif
34. end rule Four
35. rule Five
36.   from Reference(pf) to Reference(re)
37.     re. name<-pf. name
38.     re. isAggregation<-pf. isAggregation
39.     re. isComposition<-pf. isComposition
40.     re. isReverAssoication<-pf:isReverAssoication
41.     re:ismultiple<-pf:Ismultiple
42.     re. type<-pf. entityRef. name
43. end rule Five
```

4.3　用例图转换规则及算法描述

源元模型是上文构建的合作建模元模型，目标元模型来源于 RM2PT 工具元模型中关于用例图的部分，用 EcoreTool 展示，如图 6 所示。其中，UseCaseModel 表示用例图，包含有零至多个用户（Actor）和用例（UC）。用户有 description 属性，指向自身的超类（superActor）表示用户与用户可能会存在继承关系，指向 UC 表明 UC 可以关联多个 Actor，用例包含 description，name 和 UCRelation 3 个属性。

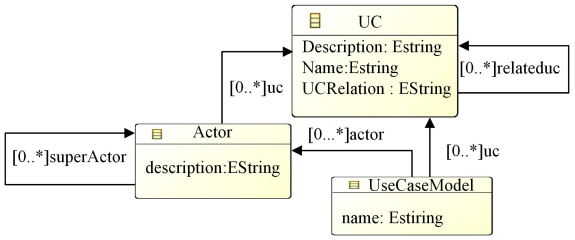

图 6　用例图目标元模型

Fig. 6　Target metamodel of use case diagram

得到源元模型和目标元模型后，构建对应的 ATL 转换规则，具体描述如下：首先获取所有的领域并判断其是否为 Actor，如果为 true 则进行下一步，根据与其连线的需求来确定其包含的用例，再根据问题图中需求之间的关联关系来确定用例之间的包含或扩展关系。PF2UseCaseDiagram 算法如算法 2 所示。

算法 2　PF2UseCaseDiagram

```
1. -- @path RE=/PF2UML/metamodels/REMODEL. ecore
2. -- @path PF=/PF2UML/metamodels/pf. ecore
3. module PF2UseCaseOne
4. create OUT:RE from IN:PF
5. rule One
6. from Domain(pf. UseCaseRef. notEmpty()) to Actor(ac)
7.       ac. name<-pf. name
8.       ac. uc <-pf. UseCaseRef
9. end rule One
10. rule Two
11. from Requirement(pf. isUseCase) to UC(uc)
12.       uc. name <-pf. name
13.       uc. relateduc <-pf. ucref
14. end rule Two
15. rule Three
16.   from UCRef(pf) to UC(uc)
17.       uc. dedescription <-if pf. isExtend then 'extend' else 'in-
          clude' endif
18.       uc. name <-pf. UCRefEntity. name
19. end rule Three
```

5　案例研究

下文通过包路由问题的例子进行案例分析，展示如何使用建模工具平台进行需求分析以及生成概念类图和用例图。

5.1 问题背景描述

包路由器[1)]是一个大型机械装置，邮政等投递组织用其来把包裹按照目的地分到各个邮箱中。每个包裹在进入传送装置前就贴上了条形码，然后被放置到传送装置上，跟随着传送带移动到读码工作站。读码工作站会读取条形码以获取包裹的目的地，最后由重力作用滑到顶部和底部都装配有感应器的管道上。这些管道由双向开关连接起来，计算机能够控制这些开关将包裹移动到目标邮箱中。包路由机械装置的设计如图 7[1]所示。

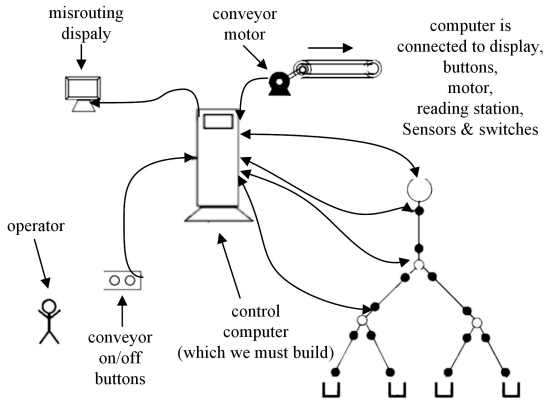

图 7 包路由器设计图

Fig. 7 Design diagram of package router

包裹有序有距地通过管道上的感应器，按照“大不压小、重不压轻、先进后出、易碎件单独摆放”的原则通过传送装置，且都不能超过前面的包裹，以保证感应器能独立检测到每一个包裹。然而，在现实世界中，包裹由于大小和重量等原因可能会进入错误的路由，从而被分配到错误的邮箱中，这时需要一个终端控制按钮，让操作员控制机械装置让其停止或启动。为了节能，该传送带还配有测重功能，正常情况下传送带马达以低速运作，当检测到包裹的重量超过一定量时，会启动高速运转模式。

5.2 领域涉众建模

包路由问题在客观世界里包含有控制开关、显示装置、读码器、传送带、包裹、通道开关和目的邮箱等物理设备，还包含有操作员。传送装置用于运输包裹，且尾端有一个读码器用于读取包裹上条形码的信息。根据对该问题的描述，将问题图分为 4 个子问题。1)服从操作者命令子问题。机械装置在运转过程中遇到紧急事件时，操作员应能启动或停止它。2)路由包子问题。传送装置能控制包裹经过正确的管道开关并保证其能到达正确的邮箱。3)节能子问题(该子问题是本文对案例的一个扩展)。为了响应节能环保政策，传送带配有智能检测系统，在正常情况下以低能耗运转，包裹放入传送带后，会自动检测其重量，超过一定值时，再启动高能运转模式。4)报告错误子问题。当包裹由于某些因素干扰导致其进入了错误的邮箱，机器立即检测错误路由，并将它们在显示器上显示出来。包路由问题图如图 8 所示。

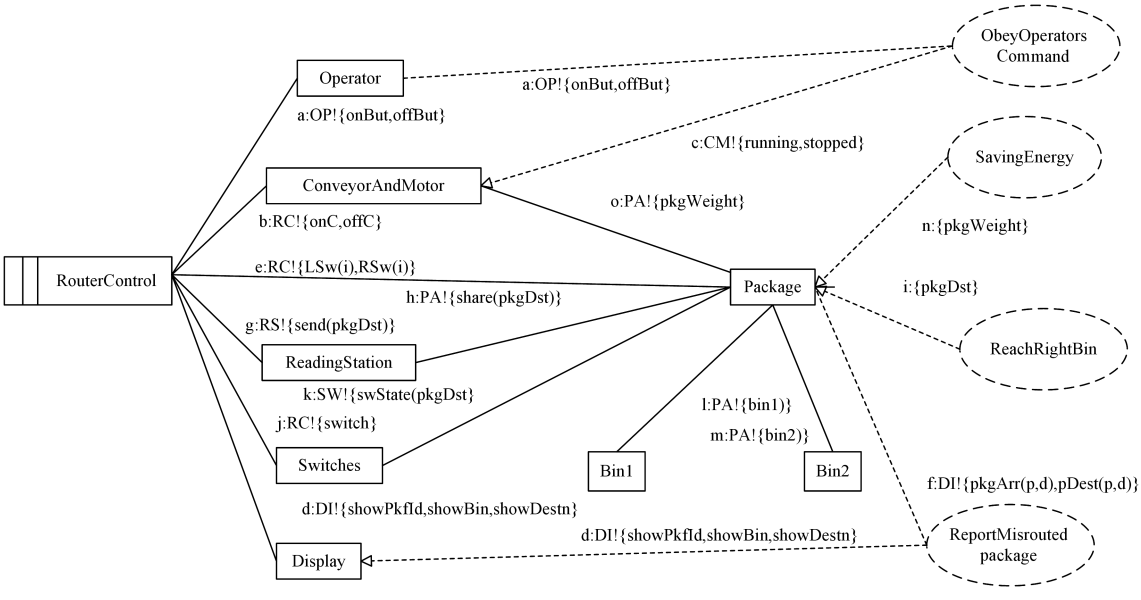

图 8 包路由问题图

Fig. 8 Problem diagram of package router

5.3 设计人员建模

领域涉众完成问题建模后，软件设计人员开始建模，设计人员根据开发经验和问题图上的提示来填充概念类图和用例图所需要的信息。

5.3.1 填充概念类图信息

设计人员依据经验依次判断每一个问题领域能否被划分为类，若能则将标签中的 isclass 属性改为 true，根据与问题领域相连的属性框的提示信息给问题领域添加相应类的属性，再根据与其他问题领域相连的信息来填充类的关联。例如包路由例子中，设计人员将 Operator 划分为一个类，将其标签 isclass 设置为 true，再为其添加 operatorId，name 和 turnOn 属性。根据与之相连的领域 Display 为其添加关联并将其判断为一对多的关联关系。按照相同的方式添加 Switches，Display，Bin 和 Package 类。添加细节如图 9 所示。

1) 在此指用于分拣包裹的装置，我们沿用文献[1]中的译文

其中，name 表示转换后的类名，isClass 和 isActor 分别用于判定该领域能否被划分为一个类和用例图中的 Actor。Type 用于选择该领域属于因果领域还是可叫牌领域。操作者是人，故应为可叫牌领域。SuperClass 用来选择该领域的父类，Classattribute 和 Reference 分别用于给类添加属性和依赖。将转换后 XMI 文件的信息导入 RM2PT 工具中，如图 10 所示。

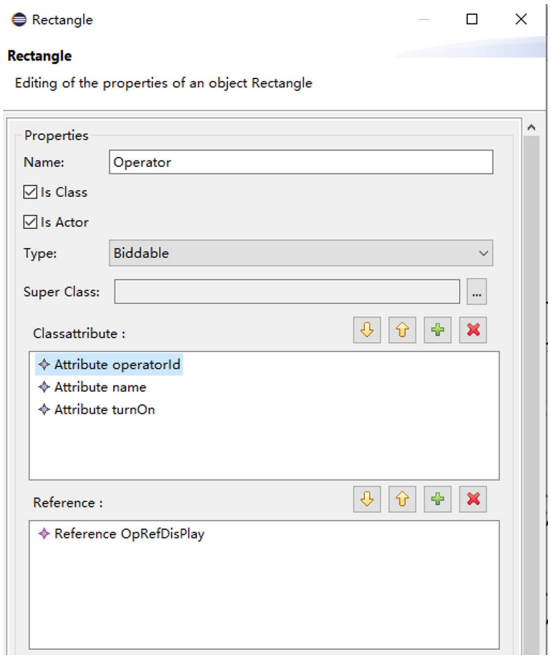

图 9　填充概念类图信息

Fig. 9　Fill in the conceptual class diagram information

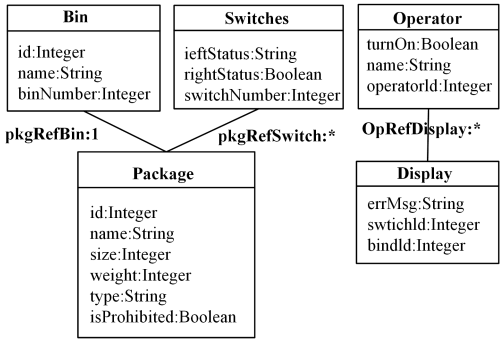

图 10　转换后的概念类图

Fig. 10　Converted conceptual class diagram

5.3.2　填充用例图信息

在问题图中，用例图中的 Actor 与问题图中的问题领域相对应，而用例与需求相对应。设计人员只需要判断问题领域是否为 Actor，如果是则将 isActor 标签设置为 true。再观察领域与需求之间是否相连，如果相连则将其作为用例。在包路由控制例子中，判断 Operator，Package，Display 和 Switches 为 Actor，与 Operator 相连的需求模块只有 ObeyOperatorsCommand。同理，添加其余 Actor 相关的需求模块，添加的相关细节如图 11 所示。其中，UseCaseRef 用于添加与该领域有关的需求。转换后的 XMI 文件的信息导入 RM2PT 工具中，如图 12 所示。

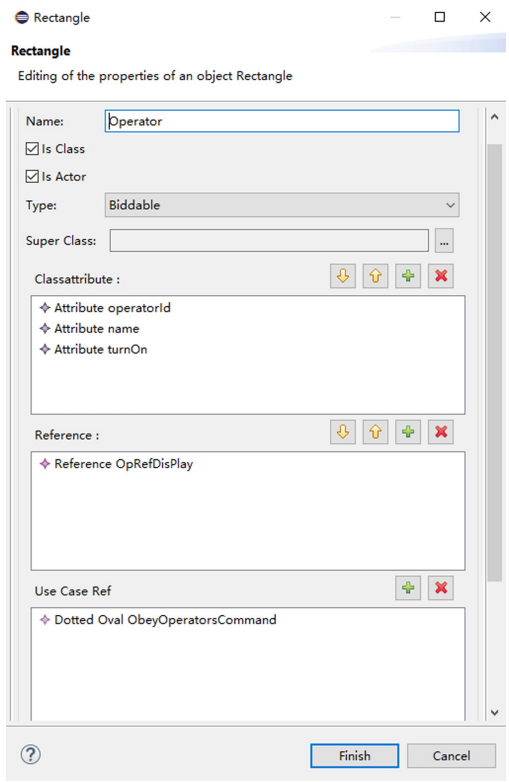

图 11　填充用例图信息

Fig. 11　Fill in the use case diagram information

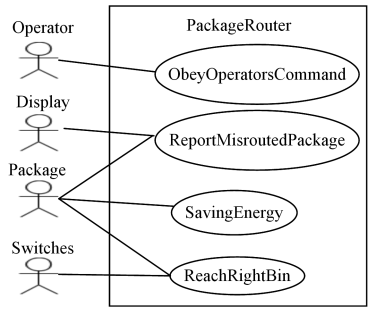

图 12　转换后的用例图

Fig. 12　Converted use case diagram

结束语　为了使问题框架建模方法过渡到软件的设计与实现，本文提出了一种从信息物理融合系统问题模型到 UML 需求模型的变换方法。首先利用模型驱动工程相关技术将问题框架、UML 概念类图和用例图相结合，构建合作建模元模型；然后使用 Eclipse 开发环境创建合作建模工具平台；最后采用模型转换 ATL 技术，将问题模型转换为 UML 需求模型。

本文采用领域涉众和开发人员合作建模的方式来构建需求模型，领域涉众建模着重于对客观事实的刻画，符合问题框架建模方法的理念，保证了软件系统在需求建模阶段的真实性和可靠性。而软件设计人员建模着重于对软件系统架构的描述和代码原型的生成。两者建模的结合提高了需求的准确性和自动生成代码的可能性，从而显著提高了开发效率。

在未来的工作中，我们将继续研究如何将问题模型转换到 UML 需求模型中的顺序图，以及如何提高生成系统操作合约的效率，最终达到从问题模型自动生成软件原型的目标。

参 考 文 献

[1] JACKSON M. Problem Frames: Analyzing and Structuring Software Development Problems [M]. New York, Oxford: Addison-Wesley, 2001.

[2] JIN Z. Environment modeling-based requirements engineering for software intensive systems [M]. Cambridge, MA: Morgan Kaufmann, 2018.

[3] JACKSON M. Software requirements & specifications: a lexicon of practice, principles and prejudices [M]. ACM Press/Addison-Wesley Publishing Co., 1995.

[4] HALL J, RAPANOTTI L, JACKSON M. Problem Oriented Software Engineering: Solving the Package Router Control Problem [J]. IEEE Transactions on Software Engineering, 2008, 34(2): 226-241.

[5] YANG Y, LI X, KE W, et al. Automated Prototype Generation from Formal Requirements Model [J]. IEEE Transactions on Reliability, 2019, 69(2): 632-656.

[6] YANG Y, LI X, LIU Z, et al. RM2PT: A tool for automated prototype generation from requirements model [C]// 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion). IEEE, 2019: 59-62.

[7] YANG Y, KE W, LI X. RM2PT: Requirements Validation through Automatic Prototyping [C]// 2019 IEEE 27th International Requirements Engineering Conference (RE). IEEE, 2019: 484-485.

[8] LI Z, JIN Z. From user requirements to software specifications: An approach based on problem transformation [J]. Journal of Software, 2013(5): 961-976.

[9] LIU G Y, WANG G H, PANG L, et al. Research and Development of Computer-aided Requirements Engineering Tool Based on Problem Frames [J]. Computer Science, 2014, 41(11): 137-140.

[10] ZHU S Y. Overview of Software Engineering Technology [M]. Science Press, 2002.

[11] STEINBERG D, BUDINSKY F, PATERBOSTRO M, et al. EMF: Eclipse Modeling Framework 2.0 [M]. Addison-Wesley Professional, 2009.

[12] OUBELLI L A, OUSSALAH M. Design and Development of Business Rules Management System (Brms) Using Atland Eclipse Sirius [J]. International Journal of Computer Science & Information Technology, 2016, 8(5): 93-105.

[13] JOUAULT F, ALLILAIRE F, BEZIVIN J, et al. ATL: A Model Transformation Tool [J]. Science of Computer Programming, 2008, 72(1/2): 31-39.

[14] KURTEV I, VAND B K. A synthesis-based approach to transformations in an MDA software development process [C]// Model Driven Ar hitecture: Foundations and Applications. 2003: 121.

[15] MAYERHOFER T, LANGER P, WIMMER M, et al. xMOF: Executable DSMLs based on fUML [C]// International Conference on Software Language Engineering. Springer, Cham, 2013: 56-75.

[16] TATIBOUËT J, CUCCURU A, GÉRARD S, et al. Formalizing execution semantics of UML profiles with fUML models [C]// International Conference on Model Driven Engineering Languages and Systems. Springer, Cham, 2014: 133-148.

[17] HUDAK P. Modular domain specific languages and tools [C]// Proceedings. Fifth International Conference on Software Reuse (Cat. No. 98TB100203). IEEE, 1998: 134-142.

[18] VIYOVI V, MAKSIMOVI M, PERISI B. Sirius: A rapid development of DSM graphical editor [C]// IEEE 18th International Conference on Intelligent Engineering Systems INES 2014. IEEE, 2014: 233-238.

[19] VUJOVIC V, MAKSIMOVIC M, PERISIC B. Comparative analysis of DSM Graphical Editor frameworks. Graphiti vs. Sirius [C]// Proc of ERK'2014. Portoroz, 2014: 7-10.

[20] ATL: Atlas Transformation Language, version 0.1 [OL]. [2020-06-26]. http://www.eclipse.org/atl/documentation/old/ATL_Installation_Guide[v0.1].pdf.

LI Zhi, born in 1969, Ph.D, professor, Ph.D supervisor, is a distinguished member of China Computer Federation. His main research interests include problem-oriented requirements engineering for big data analytics, modeling, verifying and testing cyber-physical systems (CPSs) based on problem frames, and human-computer interaction.

YANG Yi-long, born in 1988, Ph.D, assistant professor. His main research interests include automated and intelligent software engineering.