软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(2):177−194 [doi: 10.3724/SP.J.1001.2011.03755]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于问题框架的需求建模:一种本体制导的方法*

陈小红[1,3], 尹 斌[1,3], 金 芝[1,2+]

[1](中国科学院 数学与系统科学研究院,北京 100190)

[2](高可信软件技术教育部重点实验室(北京大学),北京 100871)

[3](中国科学院 研究生院,北京 100049)

Ontology-Guided Requirements Modeling Based on Problem Frames Approach

CHEN Xiao-Hong[1,3], YIN Bin[1,3], JIN Zhi[1,2+]

[1](Academy of Mathematics and Systems Science, The Chinese Academy of Sciences, Beijing 100190, China)

[2](Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, Beijing 100871, China)

[3](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

+ Corresponding author: E-mail: zhijin@amss.ac.cn

Chen XH, Yin B, Jin Z. Ontology-Guided requirements modeling based on problem frames approach. *Journal of Software*, 2011,22(2):177−194. http://www.jos.org.cn/1000-9825/3755.htm

Abstract: On the basis of intensive study of the problem frames (PF) approach, this paper extracts and develops an ontology for conceptualizing the requirements model of the PF approach and designs an ontology-guided modeling process. The aim is to present the analysts with a modeling guide and with standard the modeling activities. Finally, a case study has been given to illustrate the whole process.

Key words: problem frame approach; requirement modeling; problem frame ontology; modeling process

摘 要: 在深入研究问题框架方法的基础上,抽取并构建了一个问题框架本体,以概念化基于问题框架的需求建模,设计了一个本体制导的基于问题框架方法的需求建模过程,为需求分析员提供建模指导并规范其建模活动.最后,通过案例展示整个本体制导的建模过程.

关键词: 问题框架方法;需求建模;问题框架本体;建模过程

中图法分类号: TP311 **文献标识码**: A

需求建模是软件需求工程过程的重要阶段.不同的需求建模方法蕴含了不同的建模理念,代表了看待软件系统的不同视角.目前,学术界关注较多的需求建模方法包括面向目标的方法[1]和面向主体和意图的方法[2].其中,面向目标的方法将目标看作是软件需求的来源和依据,以目标为需求获取的基本线索,诱导需求提供者按照目标的分解、精化和抽象关系,逐步构建系统目标与/或树.与需求相关的其他概念都是围绕系统的目标树这个主结构伸展出去,包括:目标操作化为约束,约束由活动和活动所操作的对象来保证,对象被区分为事件、实体、关系和主体这 4 类,约束由主体负责完成.其代表性工作有 KAOS[3].面向主体和意图的方法则提供一种基于组织层次上下文的需求获取和建模的思路,其建模理念为刻画“有目的的参与者”.它以参与者为主要线索去识别

* 基金项目: 国家自然科学基金(90818026); 国家杰出青年基金(60625204); 国家重点基础研究发展计划(973)(2009CB320701)
收稿时间: 2008-10-24; 修改时间: 2009-07-03; 定稿时间: 2009-10-22

需求,它认为系统的参与者是有目标、有信念、有能力和有承诺的自治或者半自治主体,这些参与者之间存在各种各样的关系,主要包括目标依赖关系、任务依赖关系、资源依赖关系和软目标依赖关系这 4 种.需求获取和建模过程的主要活动有识别参与者、识别各参与者的目标和任务、对目标和任务进行精化、分析目标的实现策略、识别各参与者之间的依赖关系,其主要代表性工作包括基于 i^*建模框架[4]及其形式化扩展 Tropos 方法[5].

与上述需求建模方法不同,问题框架方法[6,7]认为,软件系统对现实世界的作用是软件问题的来源,对软件系统将与现实世界发生的作用进行结构化分析是需求分析的切入点.问题框架方法强调需要对软件系统将要作用的现实世界进行刻画,并将需求的含义指称到对现实世界相关领域的描述上.其建模的基本概念是现实世界中的领域以及未来软件系统与领域的交互.问题框架方法定义了一些常见的软件问题类型,称为问题框架.问题框架方法的基本思想就是在问题分析中使用问题框架,将复杂的现实世界软件问题结构化为相互作用的可以匹配到问题框架的子问题的集合.

近年来,问题框架方法得到了广泛关注,并取得了一些进展.在问题框架方面,除了最早提出的 5 个基本问题框架[7]以外,还出现了一些新的问题框架,如环境效应框架[8]、简单控制框架和简单查询框架[9]、简单信息回答框架[10]、四变量框架[11]、用户交互框架[12]等.在问题转换方面,研究者们提出了两种形式的问题转换方式:一种是问题分解,包括基于结构的分解[13]、用目标分解来补充问题分解[14]、利用本体进行启发式分解[15]等;另一种是问题级数,通过移除 1 个或多个领域来简化问题的上下文,通过因果推理进行形式化[16],为将需求转换为规格说明提供理论依据.另外,还有关于问题框架语义的研究[17],希望为问题框架方法建立语义基础.

同时,问题框架方法的应用也越来越广泛.如,面向问题的软件工程(POSE)[18];用问题框架方法为极限编程[19]提供需求/问题描述[20];将问题分析过程作为敏捷开发[21]过程的一部分,并将其扩展到面向对象的解决方案[22]等.还有一些工作试图将问题框架方法与其他需求工程方法,如场景[23]、面向目标的方法[24]等相结合.问题框架方法还被用于 Web service[25]描述、基于 Web 的软件开发[26]和分布式系统[27]的建模,并应用于商业模型[28]、保险业[29]、e-commerce[30]等领域中.需要指出的是,由于问题框架方法特别关注软件系统将与之交互的现实世界领域(即软件系统的环境),强调对软件系统环境的建模以及对软件系统与环境交互的建模,因此可以用于帮助软件系统边界的确定,并适合用于进行人机交互系统、嵌入式系统等的需求识别、建模与分析.特别是,由于问题框架方法具有支持环境建模和交互建模的特点,将有利于刻画软件系统的环境感知能力.可以预计,它将在 CPS (cyber-physical system)和 Internetware 的应用建模中发挥重要作用.

但目前,问题框架方法仍缺少良定的规范和有效的需求建模工具,这在很大程度上妨碍了问题框架方法从研究走向实用,建模的质量也主要依赖于需求分析员的经验.为基于问题框架方法的需求建模提供规范和指南并建立相应的建模引导工具,对问题框架方法的发展和成熟具有重要意义.

本文首先抽取并规范问题框架概念体系,构建问题框架本体,建立基于问题框架方法进行需求建模的规范的、可共享的结构模型.进而提出一个本体制导的基于问题框架的需求建模过程.该过程在问题框架本体的制导下,引导并帮助需求分析员逐步抽取和描述软件开发问题,为需求分析员提供了一个建模过程指导.

本文第 1 节描述所构建的问题框架本体.第 2 节定义本体制导的基于问题框架的需求建模过程.第 3 节通过具体案例进行过程展示.第 4 节比较相关工作,总结全文并提出进一步的工作.

1 问题框架本体

问题框架本体包含 3 部分内容:问题框架方法的概念层次、问题框架概念之间的关联以及关于问题框架概念和关联的约束.

1.1 问题框架本体的概念层次

采用问题框架方法进行需求建模首先需要描述问题.问题(problem)是要由软件开发完成的一个任务.问题分为两类.一类是基本问题(basic problem),它通常存在已知的问题求解模式,存在基本问题框架或者其变体与之对应.目前人们总结出 11 种基本问题框架,它们是需求式行为框架、命令式行为框架、信息显示框架、简单

工件框架、变换问题框架、环境效应框架、简单控制框架、简单查询框架、简单信息回答框架、四变量框架、用户交互框架.另一类是组合问题(composite problem),这类问题目前还不存在标准的问题求解模式,需要分解为子问题(subproblem)进行问题求解.

基于问题框架方法进行需求建模经历了 3 个阶段,不同阶段采用不同的问题模型(problem model),表示为由结点和边组成的图.其中,结点包括领域结点和需求结点,边则为这些结点之间的连接.

领域(domain)是基于问题框架方法进行需求建模的重要概念.它涵盖了软件开发问题中涉及的所有现实世界的实体和将要构造的实体,建模过程中主要区分两类领域:机器领域(machine,用带双竖线的矩形结点表示)和问题领域(problem domain,用矩形表示).问题领域又进一步分为设计领域(designed domain,用带竖线的矩形结点表示)和给定领域(given domain).机器领域表示待开发的软件系统,给定领域指机器将作用于的现实世界实体,设计领域指开发者设计的信息描述或信息模型的物理实现.

需求(requirement,用虚边椭圆表示)表示需求提供者对需要机器领域作用到问题领域并使其发生变化的期望以及对这些变化的约束.

问题框架方法的一个重要原理是,机器领域将通过与问题领域之间的交互来满足需求.因此,交互(interaction)是需求建模的重要元素,表示为机器领域和问题领域之间以及问题领域和需求之间的连接.根据连接的对象不同,交互分为 3 类:机器领域与问题领域之间的接口交互(interface,用实线段—表示),表示机器领域与这个问题领域可以共享一些现象;需求与问题领域之间的需求引用(reference,用虚线段---表示),表示这个需求涉及该问题领域的现象;需求与问题领域之间的需求约束(constraint,用虚箭头···▶表示),表示该需求要求机器领域应该保证这个问题领域出现所涉及的现象.

采用问题框架方法进行需求建模分 3 个阶段,分别建立上下文图(context diagram)、问题图(problem diagram)和问题框架图(PF diagram).上下文图识别机器领域和问题领域以及它们之间的连接,从而定位软件开发问题,明确表达软件开发问题的边界.问题图用来定义这个软件开发问题,它通过引入需求概念,明确表示需求对问题领域以及机器领域将对问题领域产生的作用,来表达需求相关者对待开发软件的期望.问题框架图则描述一些特定的问题类,这些问题类存在标准的需求规格说明规范.

在需求建模过程中,还涉及一些其他基础性概念,包括描述交互内容的现象(phenomena)和描述问题领域类型的领域类型(domType).包含 3 种现象,即事件(event)、状态(state)和值(value).事件是在某个时间点上出现的个体,它将被看成是原子的和瞬间发生的,例如敲一次键盘.状态是两个或多个个体之间存在的关系,它可以在一个时刻为真,在另一个时刻为假.值是一个无形的个体,它存在于时间和空间之外,是不会改变的,比如整数和字符串.有 3 类不同的领域,即因果领域(causal domain)、服从式领域(biddable domain)和词法领域(lexical domain).因果领域表示该领域的可共享现象之间存在可预测的因果关系.服从式领域通常由人组成,其主要特征是,它是物理的,但没有可预测的内部因果性.词法领域是数据的物理表示.

图 1 展示了问题框架本体的概念层次结构,其中,T 表示全域概念.

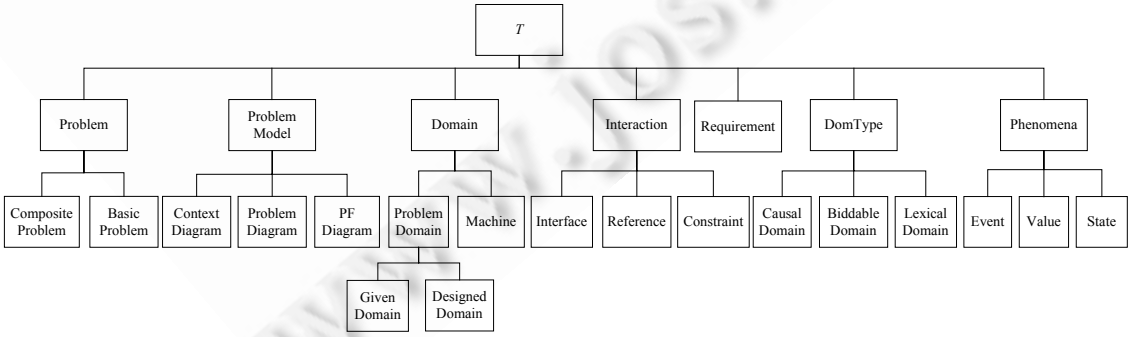

Fig.1 Concept categories of PF ontology

图 1 问题框架本体概念层次

1.2 问题框架本体的概念关联

问题框架本体中的概念不是孤立存在的,它们之间存在联系,这些概念以及概念之间的联系构成了问题框架方法的概念体系.图 2 展示了问题框架概念之间的关联,其含义在表 1 中给出.

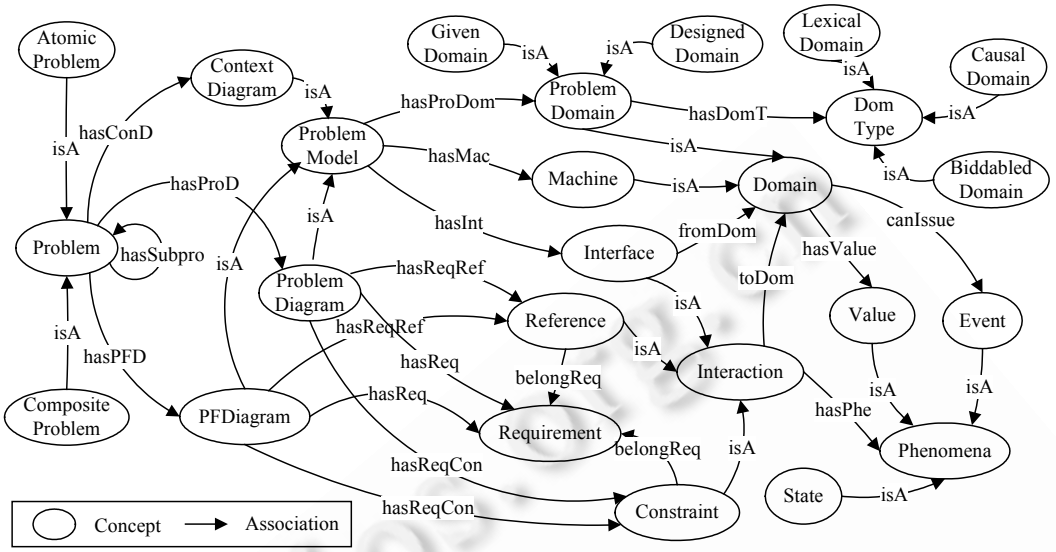

Fig.2 Associations in PF ontology

图 2 问题框架本体概念之间的关联

Table 1 Meanings of the associations in PF ontology

表 1 问题框架本体概念关联的含义

Association	Formation	Meaning	Multiplicity
hasSubpro	Problem→Problem	A problem contains a set of subproblems	1:*N*
hasConD	Problem→ContextDiagram	A problem has a context diagram	1:1
hasProD	Problem→ProblemDiagram	A problem has a problem diagram	1:1
hasPFD	Problem→PFDiagram	A problem has a set of PF diagrams	1:*N*
hasMachine	ProblemModel→Machine	A problem model has a machine	1:1
hasProDom	ProblemModel→ProblemDomain	A problem model has many problem domains	1:*N*
hasReq	ProblemDiagram→Requirement	A problem diagram has a set of requirements	1:*N*
hasReq	PFDiagram→Requirement	A PF diagram has a set of requirements	1:*N*
hasInterface	ProblemModel→Interface	A problem model has a set of interfaces	1:*N*
hasReqRef	ProblemDiagram→Reference	A problem diagram has a set of references	1:*N*
hasReqRef	PFDiagram→Reference	A PF diagram has a set of references	1:*N*
hasReqCon	ProblemDiagram→Constraint	A problem diagram has a set of constraints	1:*N*
hasReqCon	PFDiagram→Constraint	A PF diagram has a set of constraints	1:*N*
belongReq	Reference→Requirement	A references is initiated by a requirement	1:1
belongReq	Constraint→Requirement	A constraint is initiated by a requirement	1:1
fromDom	Interface→Domain	An interface is initiated by a domain	1:1
toDom	Interaction→Domain	An interaction is received by a domain	1:1
hasPhe	Interaction→Phenomena	An interaction observes a set of phenomena	1:*N*
hasValue	Domain→Value	A domain has a set of values	1:*N*
canIssue	Domain→Event	A domain issues a set of events	1:*N*
hasDomT	ProblemDomain→DomType	A problem domain may have many DomTypes	1:*N*

1.3 问题框架本体的约束

问题框架本体的概念实例和概念实例之间关联的出现必须满足一些约束,作为问题框架本体的约束.在用问题框架方法进行需求建模时,这些约束用来检测应用需求描述的一致性和完整性.

问题框架本体中的约束分为两类:

- 第 1 类是完整性约束,主要包括:
  (1) 对概念实例出现个数的限定,如上下文图必须包含 1 个机器领域、多个问题领域和多个接口;
  (2) 指明不可缺少的概念实例属性,如问题图中的问题领域必须带有领域类型;
  (3) 指明概念实例属性之间的性质,如接口交互必须连接两个不同的领域;
- 第 2 类是一致性约束,保证对同一个软件开发问题,其不同阶段的问题模型之间需要满足的内容和结构上的一致性.有两种一致性约束:
  (1) 一种是上下文图和问题图之间的一致性,即一个上下文图和它对应的问题图必须含有相同的机器领域、相同的问题领域和相同的接口;
  (2) 另一种是问题图和问题框架图之间的一致性,即问题图和它对应的问题框架图必须含有相同的机器、问题领域、接口、需求、需求引用和需求约束.

表 2 给出这些约束.采用 OWL[31]和 SWRL[32]中构造子作为表示法,包括:≥表示 owl:minCardinality,≤表示 owl:maxCardinality,∩表示 owl:intersectionOf,∀表示 owl:allValuesFrom,∃表示 owl:someValuesFrom,⊆表示 rdfs:subClassOf,Concept(?x)表示如果 x 是 Concept 的一个实例,则 Associate(?x,?y)表示 x Associate y,∧表示与,→表示推断.

Table 2 Integrity and consistency constraints in PF ontology

表 2 问题框架本体中的完整性和一致性约束

Constraint		Meaning
Problem⊆≥1hasConD∩≤1hasConD	(1)	A problem has only one context diagram
ProblemModel⊆≥1hasMac∩≤1hasMac	(2)	A problem model has only one machine
ProblemModel⊆≥1hasProDom	(3)	A problem model has at least one problem domain
ProblemModel⊆≥1hasInt	(4)	A problem model has at least one interaction
Interaction⊆≥1hasPhe	(5)	An interaction has at least one phenomenon
Interface⊆≥1fromDom∩≤1fromDom	(6)	An interaction is initiated by only one phenomenon
Interaction⊆≥1toDom∩≤1toDom	(7)	An interaction is received by only one phenomenon
Interface(?x)∧fromDom(?x,?y)∧toDom(?x,?z)→ differentFrom(?y,?z)	(8)	An interface must connect two different domains
Reference⊆≥1belongReq∩≤1belongReq	(9)	A reference must connect one requirement
Constraint⊆≥1belongReq∩≤1belongReq	(10)	A constraint must connect one requirement
ProblemDiagram⊆∀hasProDom(ProblemDomain∩∃hasDomT)	(11)	Each problem domain in problem diagram has domType
PFDiagram⊆∀hasProDom(ProblemDomain∩∃hasDomT)	(12)	Each problem domain in PFdiagram has domType
hasConD(?x,?y)∧hasProD(?x,?z)∧hasMac(?y,?a)→ hasMac(?z,?a)	(13)	Context diagram and corresponding problem diagram have the same problem domains
hasConD(?x,?y)∧hasProD(?x,?z)∧hasProDom(?y,?a)→ hasProDom(?z,?a)	(14)	Context diagram and corresponding problem diagram have the same interactions
hasConD(?x,?y)∧hasProD(?x,?z)∧hasInt (?y,?a)→ hasInt(?z,?a)	(15)	Context diagram and corresponding PF di ıgram have the same machine
hasProD(?x,?y)∧hasPFD(?x,?z)∧hasMac(?y,?a)→ hasMac(?z,?a)	(16)	Context diagram and corresponding PF di ıgram have the same machine
hasProD(?x,?y)∧hasPFD(?x,?z)∧hasProDom(?y,?a)→ hasProDom(?z,?a)	(17)	Context diagram and corresponding PF di ıgram have the same machine
hasProD(?x,?y)∧hasPFD(?x,?z)∧hasInt(?y,?a)→ hasInt(?z,?a)	(18)	Context diagram and corresponding PF di ıgram have the same machine
hasProD(?x,?y)∧hasPFD(?x,?z)∧hasReq(?y,?a)→ hasReq(?z,?a)	(19)	Context diagram and corresponding PF di ıgram have the same requirements
hasProD(?x,?y)∧hasPFD(?x,?z)∧hasReqRef(?y,?a)→ hasReqRef(?z,?a)	(20)	Context diagram and corresponding PF di ıgram have the same references
hasProD(?x,?y)∧hasPFD(?x,?z)∧hasReqCon(?y,?a)→ hasReqCon(?z,?a)	(21)	Context diagram and corresponding PF di ıgram have the same constraints

2 本体制导的需求建模过程

本体制导的基于问题框架的需求建模过程如图 3 所示.它包含 3 个步骤,即创建上下文图、创建问题图和确定(子)问题框架图.这 3 个步骤都是在问题框架本体的引导下完成的,基本问题框架库包含基本问题框架的标准解决方案.

在详细叙述建模步骤之前,需要先介绍将要使用的符号和图元.?表示请求需求分析员输入信息.↑表示信息将通过继承获得,需求分析员可以对其进行修改. $A\overset{X}{\Rightarrow}B$ 表示根据 X 的不同,$A\Rightarrow B$ 有不同的含义:若 X 是 correspond,则 $A\Rightarrow B$ 表示 B 对应于 A;若 X 是 instance,则 $A\Rightarrow B$ 表示 B 是 A 的一个实例;若 X 是 add,则 $A\Rightarrow B$ 表示通过在 A 上增加一些信息得到 B;若 X 是 part of,则 $A\Rightarrow B$ 表示 B 是 A 的一部分;若 X 是 replace,则 $A\Rightarrow B$ 表示 A 替代 B;若 X 是 generate,则 $A\Rightarrow B$ 表示 A 生成 B.图 4 给出了 3 种问题模型中用到的图元.

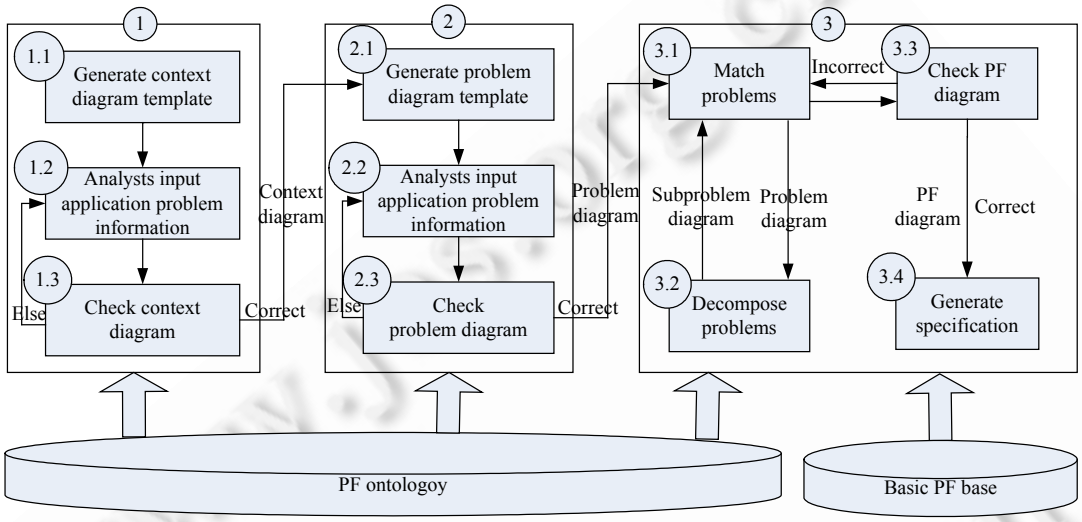

Fig.3 Ontology-Guided requirements modeling process based on PF

图 3 本体制导的基于问题框架的需求建模过程

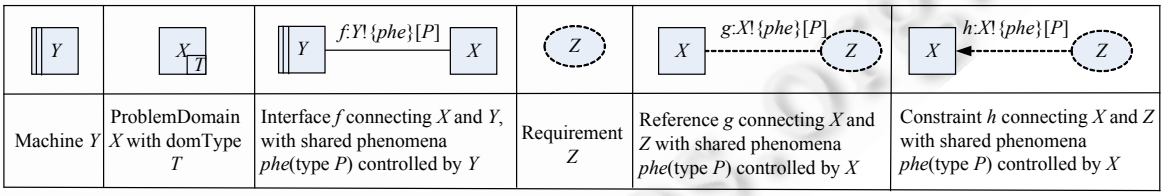

Fig.4 Diagram elements illustration in problem model

图 4 问题模型图元解释

2.1 创建上下文图

上下文图主要帮助需求分析员识别待开发的软件将要与其发生交互并产生作用的现实世界实体,由此确定软件开发问题的边界,创建上下文图的活动将获得待开发软件问题的上下文图.它有 3 个子步骤:

(1) 根据问题框架本体,生成上下文图模板.按照图 5 上部所示的问题框架本体中关于上下文图的图示片段,通过概念和关联的实例化,并按照上下文图约束(1)~约束(3)和接口约束(5)~约束(8)来约束这些概念实例的个数,生成如图 5 下部所示的上下文图模板.

(2) 需求分析员填充上下文图模板.根据上下文图模板的提示,需求分析员:

(a) 确定待开发软件问题的机器领域,填入机器领域框中;

(b) 识别待开发软件问题的问题领域,填入问题领域框中;

(c) 确定机器领域将与每个问题领域发生的接口交互,填入相应的接口位置;

(d) 确定每个接口的共享现象及现象类型;

(e) 识别每个接口中共享现象的控制领域.

(3) 检测上下文图.上下文图展示了待开发软件问题的机器领域、问题领域和接口,为了保证其完整性,需要用完整性检测器进行检测.若不完整,则需要返回步骤(2),提示需求分析员提供进一步的信息.

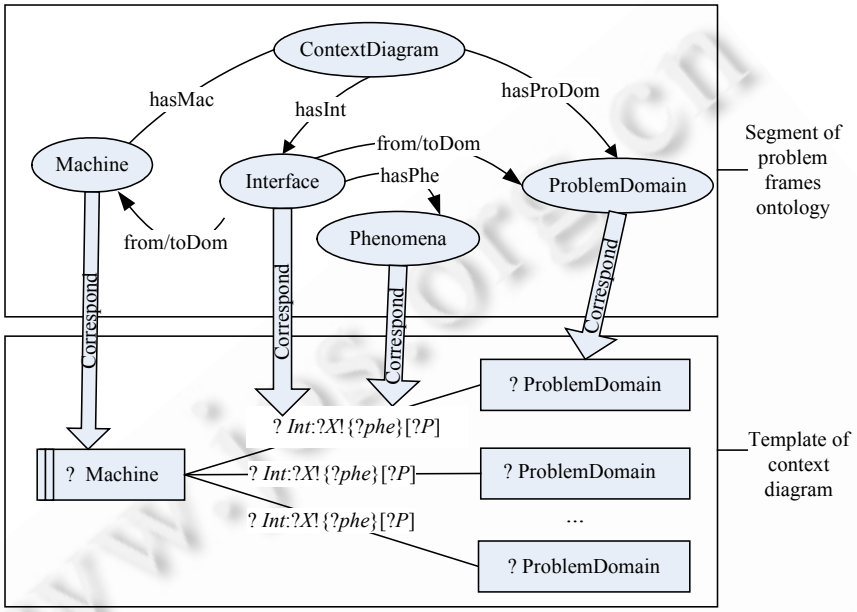

Fig.5 Generation of context diagram template

图 5 上下文图模板的生成

2.2 创建问题图

问题图主要帮助需求分析员识别待开发的软件通过交互将要给现实世界实体带来的作用效果,由此确定问题.完成创建问题图的活动之后,将获得待开发软件问题的问题图.它有 3 个子步骤:

(1) 根据问题框架本体和待开发软件问题上下文图生成待开发软件问题图模板.待开发软件问题的问题图模板的生成分两步:首先,根据如图 6 的上部和下部所示问题框架本体中关于问题图及其相关概念的图示片段,通过概念、关联的实例化,按照问题图约束(2)~约束(4)、约束(11)和交互约束(5)~约束(10)生成如图 6 中部所示问题图模板;其次,根据问题框架本体中的问题图与上下文图的一致性约束(13)~约束(15),用待开发软件问题的上下文图中的机器领域、问题领域和接口实例来替代问题图模板中的机器领域、问题领域和接口,得到待开发软件问题的问题图模板.

(2) 需求分析员填充待开发软件问题图模板.根据待开发软件问题图模板的提示,需求分析员:

(a) 识别并确定每个问题领域的领域类型,填入领域类型框中;

(b) 识别需求,填入需求框中;

(c) 识别该需求对每个问题领域的需求引用或约束,填入相应的需求引用或约束的位置;

(d) 确定并定义每个需求引用或约束的共享现象和现象类型;

(e) 识别每个需求引用或约束现象的控制领域.

(3) 检测问题图.问题图在上下文图的基础上进一步展示了待开发软件问题的需求、需求引用和需求约束.

为保证问题图的完整性,需要用完整性检测器进行检测.若不完整,则返回步骤(2),提示需求分析员提供进一步的信息.另外,为保证问题图与上下文图的一致性,还需要用一致性检测器检测问题图与上下文图的一致性.若不一致,则返回步骤(2),向需求分析员提交不一致信息,提示需求分析员进行修正.

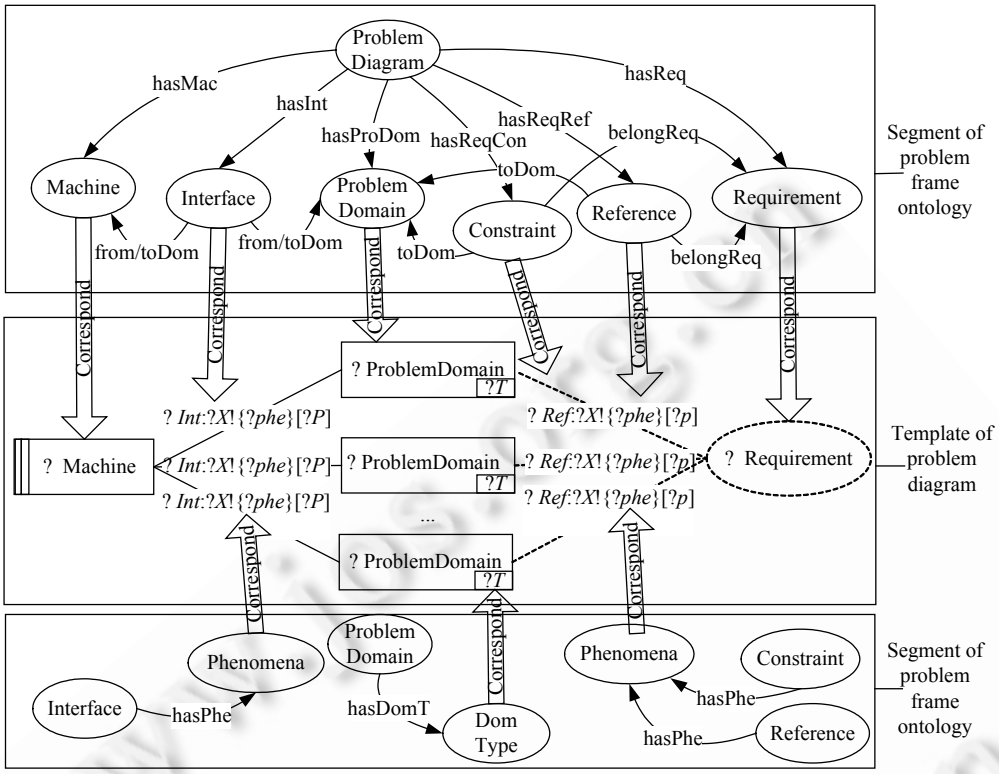

Fig.6 Problem diagram template generation

图 6 问题图模板的生成

2.3 确定(子)问题框架图

确定(子)问题框架的目的是为待开发软件问题或其子问题找到所属的问题框架,确定待开发软件(子)问题的问题类型,从而帮助需求分析员开发该软件开发(子)问题的需求规格说明.完成这个过程活动之后,将获得待开发软件(子)问题的需求规格说明规范.它包括 4 个子步骤:

(1) (子)问题匹配.若要判断一个问题是否为已知的问题框架,就要判断该问题的描述(问题图)是否与问题框架的问题描述(问题框架图)相匹配.问题图必须具有:

(a) 与问题框架图中的问题相匹配的拓扑结构;

(b) 与问题框架图具有相匹配的问题领域和领域特征;

(c) 与问题框架图具有相匹配的共享现象和类型.

从原理上说,如果不存在问题框架可以与当前的问题匹配,则说明该问题是一个组合问题,需要进入步骤(2)进行问题分解.但是,由于我们目前并不能保证现有的基本问题框架完全涵盖了目前能解决的所有基本软件问题,对未匹配的问题是进行问题分解还是创建新的基本问题框架,需要需求分析员进行判断.若可以与某个基本问题框架匹配,则得到待开发软件问题的问题框架图,进入步骤(3)进行问题框架图检测.

(2) 问题分解[15].这一步将待开发软件问题分解为一组子问题,这是问题框架方法的另一个值得深入研究

的重要问题,不属于本文讨论的范围.

(3) 问题框架图检测.待开发软件问题的问题框架图确定了待开发软件问题的问题类型,它展示了应用问题的领域、领域类型、交互等.问题框架图检测的作用是:

(a) 一是保证它的完整性.用完整性检测器进行检测,若不完整,则返回步骤(1),提示需求分析员提供进一步的信息.

(b) 二是保证它与其对应的问题图之间的一致性.用一致性检测器进行检测,若不一致,则返回步骤(1),提交错误信息,提示需求分析员进行修正.

(4) 需求规格说明生成.根据基本问题框架库中该问题框架的需求规格说明规范,引导需求分析员对待开发软件(子)问题进行分析,获得需求规格说明,建模过程结束.

在本体制导的需求建模中,需要对问题模型进行检测.为此,我们构建了完整性检测器和一致性检测器.其中,完整性检测器根据本体中关于该问题模型和相关概念的完整性约束检测问题模型的实例,具体检测算法如算法 1 所示;一致性检测器根据本体中关于问题模型的一致性约束判断问题模型实例的继承一致性关系,具体算法如算法 2 所示.

算法 1. 完整性检测器.

输入:问题模型 *pm* 及问题框架本体 PFO.

输出:若 *pm* 完整,则返回 *pm* 完整;否则,返回违反的约束.

```
1.  begin
2.  用 OWL API Jena 载入 PFO;
3.  switch(pm)
4.      case “context diagram”:构造约束集 Constraints={(2),(3),…,(8)};
5.      case “problem diagram”:构造约束集 Constraints={(2),(3),…,(11)};
6.      case “pf diagram”:构造约束集 Constraints={(2),(3),…,(9),(12)};
7.  for Constraints 中每一个约束 x
8.      switch(x)
9.          case x 形如 M⊑≥nR:
10.             计算 pm 中关联 R 的概念实例个数 m;
11.             if m<n,则返回违反约束 x;
12.         case x 形如 M⊑≥nR∩≤nR:
13.             计算 pm 中关联 R 的目标概念的实例个数 m;
14.             if m≠n,则返回违反约束 x;
15.         case x 形如 A(?x)∧B(?x,?y)∧C(?x,?z)→differentFrom(?y,?z)
16.             在 pm 中查找 A 的实例集合 I;
17.             for I 中的每一实例 i
18.                 查找 i 关联 B 的实例 m 和 C 的实例 n;
19.                 if m=n 或 m 和 n 一方不存在,则返回违反约束 x;
20.         case x 形如 M⊑∀hasP(PD∩hasD)
21.             在 pm 中查找关联 hasP 关联到的概念实例集合 I;
22.             for I 中的每一实例 i
23.                 if i 不是 PD 的一个实例,则返回违反约束 x;
24.                 else
25.                     if i 的关联 hasD 关联的实例不存在,则返回违反约束 x;
26.  返回 pm 完整;
```

27. end

算法 2. 一致性检测器.

输入:问题模型 $p1$ 和 $p2$,$p2$ 继承 $p1$ 的继承关系 $h(p1,p2)$,问题框架本体 PFO.

输出:若 $p1$ 和 $p2$ 一致,则返回 $p1$ 和 $p2$ 一致;否则,返回违反的约束.

1. begin
2. 用 OWL API Jena 载入 PFO;
3. *switch*($p1$)
4. case “context diagram”:构造约束集 *cont*={(13),(14),(15)};
5. case “problem diagram”:构造约束集 *cont*={(16),(17),(18),(19),(20),(21)};
6. for *cont* 中的每个约束 x
7. if x 形如 $A(?x,?y)\wedge B(?x,?z)\wedge C(?y,?a)\rightarrow C(?z,?a)$
8. 在 $p1$ 中的寻找 C 对应的实例 a;
9. if $p2$ 中不存在 a,返回违反约束 x;
10. else 返回约束 x 错误;
11. 返回 $p1$ 和 $p2$ 一致;
12. end

3 案例研究

本节用一个电话销售问题案例[15]来展示本体制导的需求建模过程.该案例的问题陈述如下:

一个慈善协会需要一个电话销售系统(telemarketing system)来卖彩票给它的支持者(supporter).该协会有很多种不同的彩票活动(lottery campaign).现有的活动可以由活动设计者(campaign designer)更新.电话员(telemaker)与支持者通过电话(phone)联系.所有的这些连接都是通过系统做的.若一个支持者愿意去购买一种彩票,则系统应该记录彩票订单,并产生一个彩票预订,该预订将被送到预订处理器(order processor).

3.1 创建电话销售问题的上下文图

(1) 生成上下文图模板.

(2) 根据模板创建上下文图描述.根据如图 7 上部所示上下文图模板,我们

(a) 命名要创建的机器领域为 TelmarketingSystem(*TS*).

(b) 识别出将要与机器领域发生交互的问题领域有 Supporter(*S*),Phone(*P*),Campaign Designer(*CD*),Order Processor(*OP*)和 Telemaker(*T*).

(c) 识别机器和每个问题领域的接口.如,问题领域 *S* 与机器 *TS* 领域存在接口 *a*,接口 *a* 中的共享现象有 Connect Supporter,Record Supporter Detail 和 Record Ticket Placement,现象类型都是事件(*E*).其中,现象 Connect Supporter 由领域 *TS* 控制,现象 Record Supporter Detail 和 Record Ticket Placement 由领域 *S* 控制.类似地,可以识别接口 *b*,*c*,*d*,*e*.在模板相应位置输入相应信息,得到如图 7 下部所示的电话销售问题的上下文图.

(3) 上下文图的完整性检测.用完整性检测器检测如图 7 下部所示的电话销售上下文图,结果显示为“电话销售问题上下文图完整”,进入建模过程的下一步.

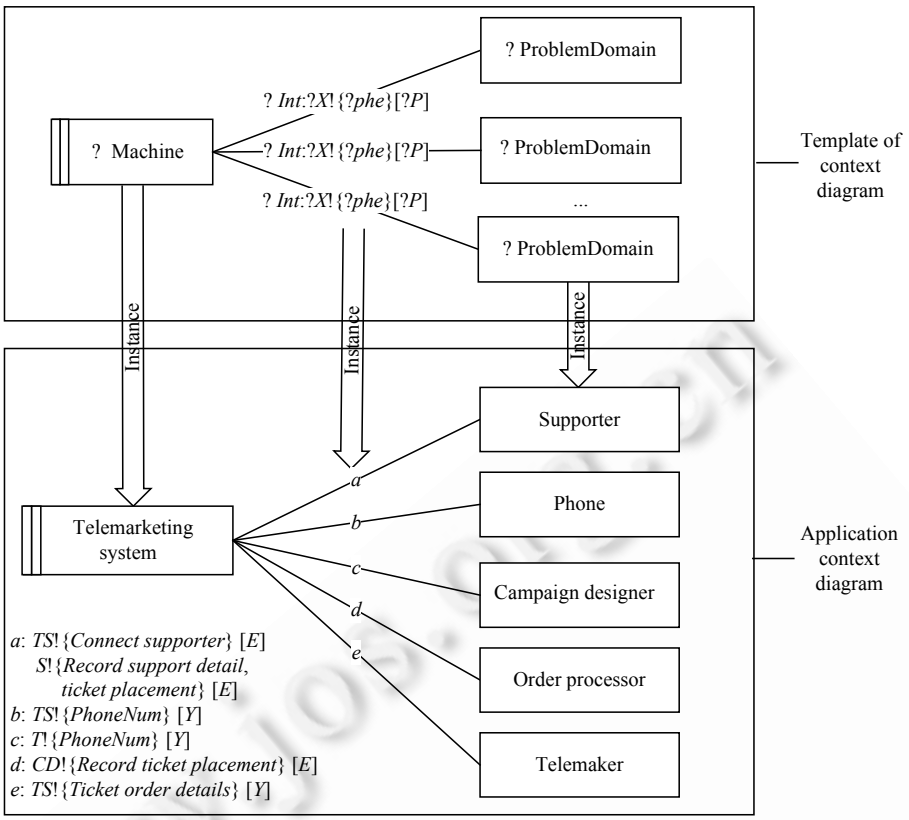

Fig.7 Context diagram of telemarketing problem

图 7 电话销售问题上下文图

3.2 创建电话销售问题图

(1) 生成电话销售问题图模板.由如图 8 上部所示的电话销售上下文图中的机器、领域和接口替代图 8 中部所示的问题图模板中的机器、领域和接口,就可以生成如图 8 下部所示的电话销售问题的问题图模板.

(2) 根据电话销售问题图模板描述问题图.由如图 9 上部所示问题图模板的提示,我们

(a) 识别各个问题领域的类型,例如,领域 Supporter 是服从式领域(*B*).

(b) 识别需求 Update campaigns,record supporter details and ticket placement,produce ticket order, send ticket order to Order Processor.

(c) 识别该需求对各个问题领域的需求引用或约束.例如,对领域 Phone 有需求引用 *h*,*h* 引用的现象是 Connect Supporter,现象的类型是事件(*E*),该现象是由机器 *TS* 控制.类似地,识别需求引用 *a*,*c*,*e*(与接口 *a*,*c*,*e* 相同)和需求约束 *j*.在模板相应位置输入相应信息之后,可得到如图 9 下部所示的电话销售问题图.

(3) 电话销售问题图完整性和一致性的检测.用完整性检测器检测如图 9 下部所示电话销售问题图的完整性,用一致性检测器检测它和图 7 下部所示电话销售问题上下文图的一致性.结果分别是,“电话销售问题图完整”和“电话销售上下文图和电话销售问题图一致”,进入建模过程的下一步.

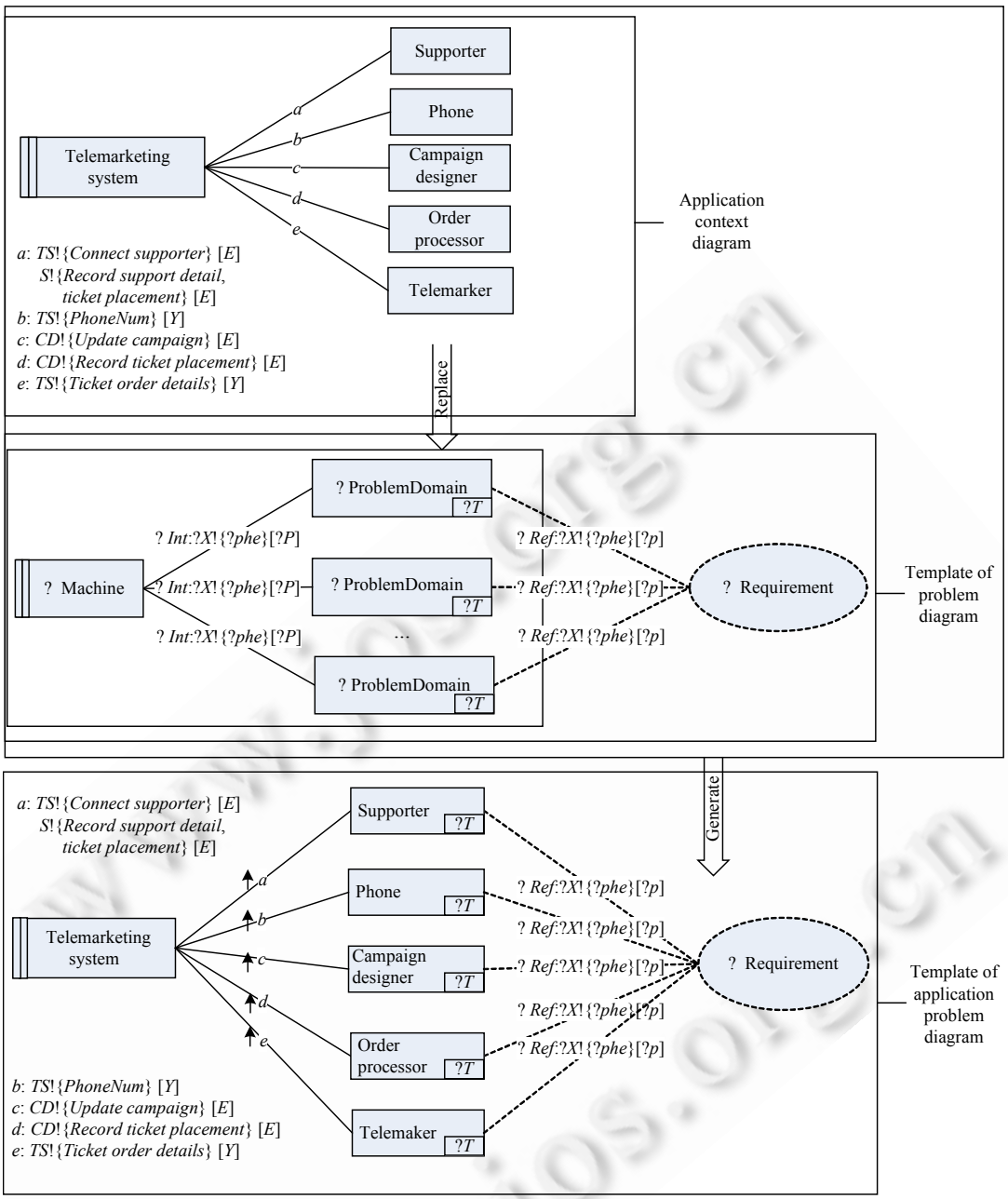

Fig.8 Problem diagram template of telemarketing problem

图 8 电话销售问题图模板

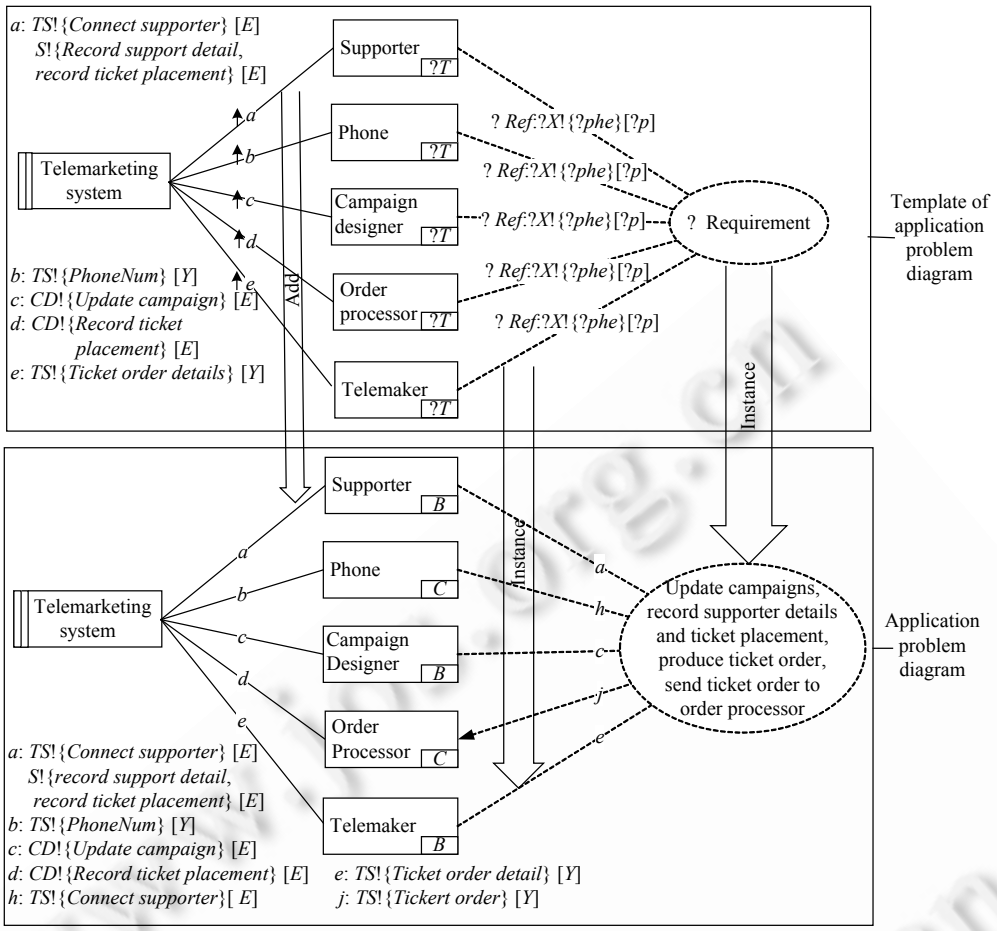

Fig.9 Problem diagram of telemarketing problem

图 9 电话销售问题图

3.3 确定电话销售子问题框架图

(1) 匹配电话销售问题

按照问题匹配过程,电话销售问题不能匹配到一个基本的框架或变体,这个问题就是一个组合问题,需要先分解.

(2) 分解电话销售组合问题

我们根据文献[15]给出如图 10 所示的电话销售问题的分解结果.这个组合问题分解成了 3 个子问题,即 Supporter Details Editor(*SDE*),Telemarketing Scheduler(*TSR*)和 Campaign Details Editor(*CDE*).

对每个子问题的处理,我们以子问题 *CDE* 为例:

(a) 生成 *CDE* 的子问题模板.其做法如下:对来自原电话销售问题中问题领域 *CD*(图 11 中灰色结点),继承 *CD* 对应的接口 *c*、需求、需求引用 *c*,而对新的领域 Campaign DataBase(*CDB*,如图 10 所示),实例化它的接口和需求引用或需求引用,于是得到如图 11 下部所示的子问题 *CDE* 的问题图模板.

(b) 在该问题模板的提示下,重新识别接口 *c*、需求和需求引用 *c*,修正需求为 Update campaign,识别新领域 *CDB* 的接口 1、需求对这个新领域的需求约束 *m* 及其现象 Campaign Info,现象类型为值(*Y*),由此得到如图 12 下部所示子问题 *CDE* 的问题图.

(c) 为保证 *CDE* 问题图的完整性,用完整性检测器判断它的完整性,结果为"*CDE* 问题图完整".类似地处理

子问题 *SDE* 和 *TSR*,进入建模过程的下一步.

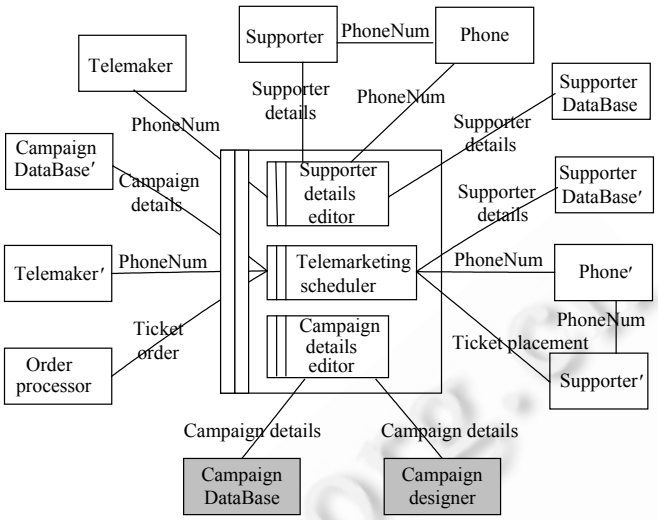

Fig.10 Decomposition result of telemarketing problem

图 10 电话销售问题分解结果

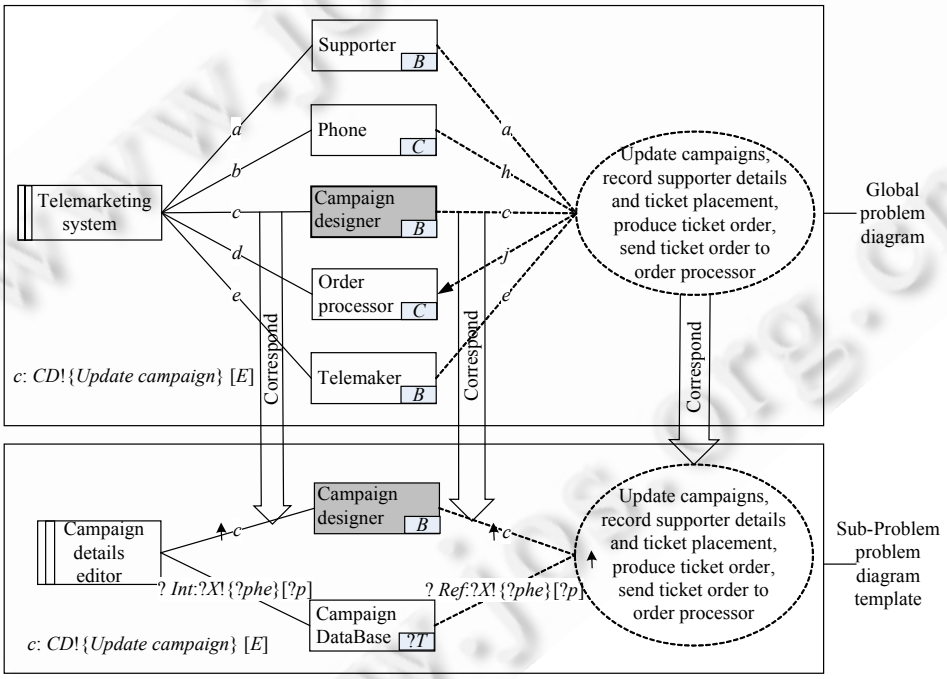

Fig.11 Sub-Problem diagram template of *CDE* generation

图 11 *CDE* 的子问题图模板生成

(3) 匹配电话销售子问题

对于电话销售子问题,根据匹配过程,*CDE* 和 *SDE* 匹配到简单工件框架,其问题框架图分别如图 12 下部和图 13 所示.*TSR* 可以匹配到命令式行为框架的变体,基本的框架如图 14 所示.

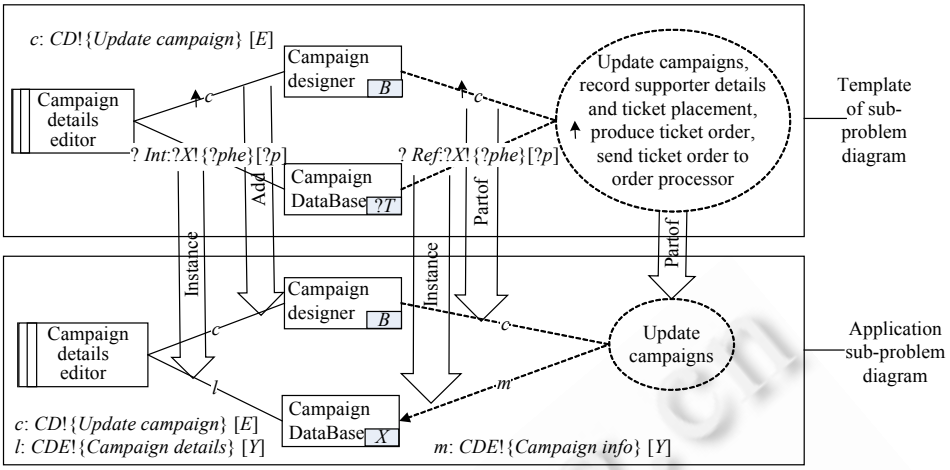

Fig.12 Problem diagram of *CDE* generation

图 12 *CDE* 问题图生成

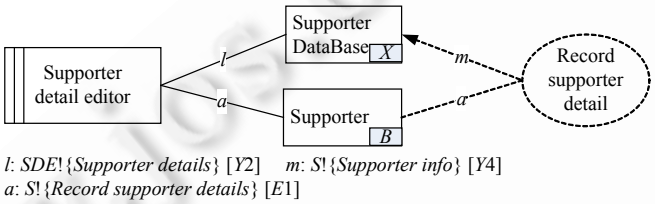

Fig.13 PF diagram of *SDE*: Fitted to simple workpiece frame

图 13 *SDE* 问题框架图:匹配到简单工件框架

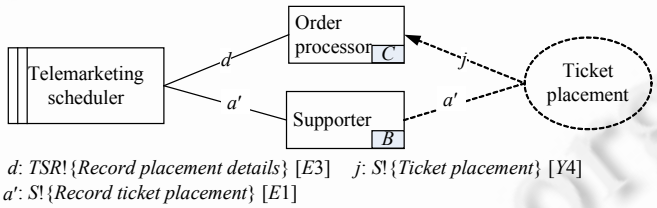

Fig.14 PF diagram of *TSR*: Fitted to command behavior frame

图 14 *TSR* 问题框架图:匹配到命令式行为框架

(4) 检测电话销售子问题框架图的完整性和一致性

用完整性和一致性检测器对电话销售问题的 3 个子问题的框架图和问题图进行检测.这 3 个子问题的问题框架图全部通过检测,进入需求规格说明阶段.

(5) 生成电话销售问题需求规格说明

由基本框架库中简单工件框架的需求规格说明标准规范,*CDE*(*SDE*)的规格说明可以简单地表示为:

- 机器能够识别 Supporter(campaign designer)的命令 Record Supporter Details(update campaign);
- 机器发出信息 Supporter Detail(campaign details),通过开拓 SupporterDatabase(campaign database)领域的已知特性,引起将保证产生所需要的 Supporter Info(campaign info)的操作.

由基本框架库中命令式行为框架的需求规格说明规范,*TSR* 子问题的规格说明可以简单地表示为:

- 机器能够检测到操作者 Supporter 发出的命令 Record Ticket Placement;
- 机器发出 Record Placement Detail 事件,通过 OrderProcessor 的领域特性,实现所需求的 Ticket Placement 状态.

4 相关工作和结束语

各种需求建模方法都有自己的建模理念,每种理念都是一种看待软件系统的观点.自 20 世纪 70 年代中期以来,结构化的需求建模方法一直是比较流行和普及的需求建模技术之一.它认为系统的功能就是“数据”流经系统时发生变迁的能力,同时需要外部事件触发进行完成变迁的过程.

面向对象的需求建模方法是当今工业界的主流方法,它认为现实系统是由各种各样的现实“对象”组成,对象可以被分类、被描述、被组织、被操作、被创建,系统是要实现对现实世界实体(对象)的计算,需要在系统中建立这些实体的映像,这些实体的个体操作模型和交互模型就是系统的功能模型.面向对象的需求建模方法的关键是从获取的需求信息中识别出问题域中的类与对象,并分析它们之间的关系,最终建立起简洁、精确和易理解的需求模型.UML[33]是随着面向对象方法发展起来的统一建模语言,包括用来表示系统静态结构的用例图、类图等,以及表示系统动态结构的状态图、活动图、序列图、协作图和配置图等.

上述方法都只是针对软件系统本身的建模方法,并没有涉及软件需求从哪里来、客户存在什么问题需要解决、为什么客户会期望或者需要软件来帮助它们解决这些问题、他们需要软件帮他们做什么等问题.20 世纪 90 年代之后,学术界提出需要对软件需求工程进行专门的研究,提出在进行软件系统建模之前,需要对软件将处于的环境,即软件将要解决的现实世界的问题进行建模,需要对包含软件及其环境的软件加强型系统进行建模,这样才能识别出或者推导出人们对软件的真正需求.

这个学科的发展产生了一些有代表性的关于软件加强型系统的建模方法.其中,面向目标的方法认为开发软件是为了实现现实系统的目标,用目标作为第 1 类概念,以目标为导引构造与/或树结构的目标层次,由抽象到具体的展开求精目标,将把较细粒度的目标分配给单个 Agent 作为其职责.面向主体和意图方法则认为,现实系统是由有意图的主体及其之间的依赖关系构成的,现实世界需要软件构成软件加强型系统,是因为需要更好地实现现实系统中主体之间的各种依赖关系,其建模过程包含两个模型,即策略依赖模型(SD)和策略推理模型(SR).策略依赖模型刻画根据有意图的主体间的依赖关系;策略推理模型描述主体的各种依赖实现方式,通过分析和评价可以选择目前最好的解决方案.

问题框架方法也是一种面向软件加强型系统的方法,但与面向目标的方法和面向主体和意图的方法侧重于人的主观意图不同的是,问题框架方法更侧重于软件将处于的客观现实世界.基于问题框架方法进行需求建模,其第 1 类概念是现实世界中的领域和未来软件系统与领域的交互.它认为,系统的功能体现在未来软件系统与现实世界领域的交互下产生的对现实世界领域的作用效果.在问题框架方法中,用机器领域显式地表示了要创建的软件系统.用问题领域建模现实世界领域,严格区分了问题领域和机器领域,由此确定了问题的边界,却又不涉及任何关于机器领域的细节描述.由此避免过早进入问题的解决方案.它强调在关注解决方案之前关注问题本身,尽可能地识别出关键的困难并尽早地加以解决.这是它与其他需求工程方法的根本区别.

目前,问题框架方法正逐步受到需求工程研究者和实践者的关注.但是,要进一步促进问题框架方法走向成熟,制定切实可行的建模规范和建模过程十分重要.本文提出并构建问题框架本体,作为基于问题框架的需求建模的建模规范,并在此基础上设计了一个本体制导的需求建模过程,逐步引导需求分析员构建需求阶段不同模型,刻画软件开发问题.本文的主要贡献包括:

(1) 构建了问题框架本体.在深入研究问题框架方法的基础上,本文抽取了问题框架方法中的基本概念和术语,总结了概念和概念之间的关联以及概念和关联之间必须满足的约束,构建了问题框架本体,建立了基于问题框架方法进行需求建模的规范的、可共享的结构模型,是对问题框架方法的一种可共享和重用的描述,为使用问题框架方法进行需求建模提供了一个规范的模型表示.这项工作为建立问题框架方法支持工具奠定了基础.

(2) 提出了一个本体制导的基于问题框架的需求建模过程.该过程在问题框架本体的制导下,引导并帮助需求分析员逐步抽取和描述软件开发问题,为需求分析员提供了一个建模过程指导.该建模过程的优点主要有:(a) 它采用图形化方式建模,方便需求分析员理解和使用.(b) 它利用问题框架本体提供的知识生成了众多模板,如上下文图模板、问题图模板.在这些模板的提示下,需求分析员逐步输入相关信息,对需求建模.(c) 它利用问题框架本体中的约束对建模过程中产生的中间和最终结果进行检测,确保需求建模的正确性.

下一步工作将继续研究如何在问题描述的基础上提供相应的策略,指导需求分析员进行问题分解,开发基于问题框架的需求建模支撑工具以帮助需求分析员构建正确的软件需求模型.此外,利用现有的推理机,如 Jess,根据问题框架本体中的 SWRL 约束对应用问题的描述结果进行检测,也是进一步的研究工作.

References:

[1] Van Lamsweerde A. Goal-Oriented requirements engineering: A guided tour. In: Titsworth FM, ed. Proc. of the 5th IEEE Int'l Symp. on Requirements Engineering (RE 2001). Washington: IEEE Computer Society, 2001. 249−263. [doi: 10.1109/ISRE.2001.948567]

[2] Yu E. Towards modeling and reasoning support for early-phase requirements engineering. In: Proc. of the 3rd IEEE Int'l Symp. on Requirements Engineering (RE'97). Washington: IEEE Computer Society, 1997. 226−235.

[3] Dardenne A, Van Lamsweerde A, Fickas S. Goal-Directed requirements acquisition. Science of Computer Programming, 1993, 20(1-2):3−50. [doi: 10.1016/0167-6423(93)90021-G]

[4] Yu E. Agent orientation as a modeling paradigm. Wirtschaftsinformatik, 2001,43(2):123−132.

[5] Bresciani P, Perini A, Giorgini P, Giunchiglia F, Mylopoulos J. Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems, 2004,8(3):203−236. [doi: 10.1023/B:AGNT.0000018806.20944.ef]

[6] Jackson M. Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices. Addison-Wesley, 1995.

[7] Jackson M. Problem Frames: Analyzing and Structuring Software Development Problems. Addison-Wesley, 2001.

[8] Jackson M. Problems, methods and specialization. Software Engineering Journal, 1994,9(6):249−255. [doi: 10.1049/sej.1994.0034]

[9] Jackson M. Problem analysis using small problem frames. South African Computer Journal, 1999,22:47−60.

[10] Jackson M. Problem analysis and structure. In: Hoare T, Broy M, Steinbruggen R, eds. Proc. of the NATO Summer School. Amsterdam: IOS Press, 2000. 3−20.

[11] Jackson M. Why software writing is difficult and will remain so. Information Processing Letters, 2003,88(1-2):13−15. [doi: 10.1016/S0020-0190(03)00391-0]

[12] Hall JG, Rapanotti L. Problem frames for socio-technical systems. In: Zaphiris P, Ang CS, eds. Human Computer Interaction: Concepts, Methodologies, Tools, and Applications. Hershey: Information Science Reference, 2009. 713−731.

[13] Rapanotti L, Hall JG, Jackson M, Nuseibeh B. Architecture driven problem decomposition. In: Proc. of the 12th IEEE Int'l Requirements Engineering Conf. (RE 2004). Washington: IEEE Computer Society, 2004. 73−82. [doi: 10.1109/ICRE.2004.1335666]

[14] Bleistein S, Cox K, Verner J. Validating strategic alignment of organizational IT requirements using goal modeling and problem diagrams. The Journal of Systems and Software, 2006,79(2):362−378. [doi: 10.1016/j.jss.2005.04.033]

[15] Jin Z, Liu L. Towards automatic problem decomposition: an ontology-based approach. In: Proc. of the 2nd Int'l Workshop on Advances and Applications of Problem Frames (IWAAPF 2006). New York: ACM Press, 2006. 41−48. [doi: 10.1145/1138670.1138678]

[16] Li Z. Progressing problems from requirements to specifications in problem frames. In: Proc. of the 3rd Int'l Workshop on Advances and Applications of Problem Frames (IWAAPF 2008). New York: ACM Press, 2008. 53−59. [doi: 10.1145/1370811.1370823]

[17] Hall JG, Rapanotti L, Jackson M. Problem frames semantics for software development. Lecture Notes in Software Systems Model, 2005,4(2):189−198. [doi: 10.1007/s10270-004-0062-1]

[18] Hall JG, Rapanotti L, Jackson M. Problem oriented software engineering: Solving the package router control problem. IEEE Trans. on Software Engineering, 2008,34(2):226−241. [doi: 10.1109/TSE.2007.70769]

[19] Beck K. Extreme Programming Explained: Embrace Change. 2nd ed., Boston: Addison Wesley, 1999.

[20] Tomayko J. Adapting problem frames to extreme programming. In: Proc. of the XP Universe Conf. 2001. http://www.xpuniverse.com/2001/pdfs/Edu02.pdf

[21] Agilealliance. 2009. http://www.agilealliance.org/

[22] Taylor P. Problem frames and object-oriented software architecture. In: Proc. of the 37th Int'l Conf. on Technology of Object-Oriented Languages and Systems (TOOLS-37 2000). Washington: IEEE Computer Society, 2000. 70−81. [doi: 10.1109/TOOLS.2000.891359]

[23] Bianco VD, Lavazza L. Enhancing problem frames with scenarios and histories: A preliminary study. In: Proc. of the 2nd Int'l Workshop on Advances and Applications of Problem Frames (IWAAPF 2006). New York: ACM Press, 2006. 25−32. [doi: 10.1145/1138670.1138676]

[24] Lin L, Jin Z. Integrating goals and problem frames in requirements analysis. In: Proc. of the 14th IEEE Int'l Requirements Engineering Conf. (RE 2006). Washington: IEEE Computer Society, 2006. 349−350. [doi: 10.1109/RE.2006.34]

[25] Cai GJ, Jin Z, Feng ZJ. A method for Web service description by using problem frames approach. In: Proc. of the 3rd Int'l Workshop on Applications and Advances of Problem Frames (IWAAPF 2008). New York: ACM Press, 2008. 23−28. [doi: 10.1145/1370811.1370816]

[26] Jeary S, Phalp K. On the applicability of problem frames to Web-based business applications. In: Cox K, Hall JG, Rapanotti L, eds. Proc. of the 1st Int'l Workshop on Applications and Advances of Problem Frames. New York: ACM Press, 2004. 35−38.

[27] Haley C. Using problem frames with distributed architectures: A case for cardinality on interfaces. In: Proc. of the 2nd Int'l Workshop from Software Requirements to Architectures (STRAW 2003). New York: ACM Press, 2003. 130−133.

[28] Cox K, Phalp K. From process model to problem frame—A position paper. In: Regnell B, Kamsties E, eds. Proc. of the 9th Int'l Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ 2003). Washington: IEEE Computer Society, 2003. 113−116.

[29] Strunk EA, Knight JC. The essential synthesis of problem frames and assurance cases. In: Proc. of the 2nd Int'l Workshop on Applications and Advances of Problem Frames (IWAAPF 2006). New York: ACM Press, 2006. 81−86. [doi: 10.1145/1138670.1138683]

[30] Weiringa R, Gordijn J, Van Eck P. Value framing: A prelude to software problem framing. In: Cox K, Hall JG, Rapanotti L, eds. Proc. of the 1st Int'l Workshop on Advances and Applications of Problem Frames (IWAAPF 2004). New York: ACM Press, 2004. 75−84. [doi: 10.1049/ic:20040228]

[31] OWL. 2004. http://www.w3.org/TR/owl-ref/#EnumeratedClass/

[32] SWRL. 2004. http://www.w3.org/Submission/SWRL/

[33] UML. 2009. http://www.uml.org/

陈小红(1982－),女,山东济宁人,博士,讲师,CCF 会员,主要研究领域为软件需求工程.

金芝(1962－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件需求工程,领域建模,基于知识的软件工程.

尹斌(1984－),男,博士生,主要研究领域为需求工程.