

基于限定中文自然语言需求的 SysML 模型自动生成方法

鲍 阳^{1,2} 杨志斌^{1,2} 杨永强^{1,2} 谢 健^{1,2} 周 勇^{1,2} 岳 涛^{1,2} 黄志球^{1,2} 郭 鹏³

¹(南京航空航天大学计算机科学与技术学院 南京 211106)
²(高安全系统的软件开发与验证技术工信部重点实验室(南京航空航天大学) 南京 211106)
³(航空工业计算所 西安 710065)
(baoyang_864@163.com)

An Automated Approach to Generate SysML Models from Restricted Natural Language Requirements in Chinese

Bao Yang^{1,2}, Yang Zhibin^{1,2}, Yang Yongqiang^{1,2}, Xie Jian^{1,2}, Zhou Yong^{1,2}, Yue Tao^{1,2}, Huang Zhiqiu^{1,2}, and Guo Peng³

¹(School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106)
²(Key Laboratory of Safety-critical Software, Ministry of Industry and Information Technology (Nanjing University of Aeronautics and Astronautics), Nanjing 211106)
³(Aviation Computing Technology Research Institute, Xi'an 710065)

Abstract Model-driven development has been gradually adopted as an important approach of designing and developing safety-critical cyber-physical systems(SC-CPSs). The requirement of SC-CPSs is often described in natural language. How to link natural language requirements and the model-driven design and development process of SC-CPSs automatically or semi-automatically is a main existing challenge. In this paper, a method named RNL2SysML is proposed for the automatic generation of SysML models from restricted natural language requirements in Chinese. Firstly, in view of the problem that glossaries need to be manually extracted, a method for extracting and recommending terms of SC-CPSs based on artificial intelligence is proposed. Secondly, in order to reduce the ambiguity of natural language requirements, a restricted natural language requirement template is proposed for requirement specification. Then, the method of transformation from natural language requirement specification to SysML model is given. Finally, based on the open source tool Papyrus, the plugin for the method proposed in this paper is implemented, and the effectiveness and practicality of the method is evaluated and proved by an industry case of the airplane air compressor system in the aviation field.

Key words safety-critical cyber-physical system; model-driven development; SysML; glossary recommendation; restricted natural language requirement; model transformation

摘 要 模型驱动开发方法逐渐成为安全关键信息物理融合系统(safety-critical cyber-physical system, SC-CPS)设计与开发的重要手段.然而,安全关键信息物理融合系统需求往往是通过自然语言描述的,如何自动化或半自动化链接自然语言需求和基于模型驱动的系统设计与开发过程是目前面临的重要挑战.

面向安全关键信息物理融合系统,提出基于限定中文自然语言需求的 SysML 模型自动生成方法 RNL2SysML.首先,为了降低自然语言需求表达的歧义性,提出一种结构化的限定自然语言需求模板进行需求规约,并通过基于人工智能的(AI)安全关键信息物理融合系统术语提取和推荐方法,对系统需求中的领域术语和数据字典加以自动提取,提高限定自然语言需求规约工作的自动化程度.然后,给出限定自然语言需求规约到 SysML 系统设计模型的转换方法.最后,基于开源工具 Papyrus 对所提方法进行了原型工具实现,并通过航空领域的飞机空气增压系统(airplane air compressor system)案例验证了方法的有效性和实用性.

关键词 安全关键信息物理融合系统;模型驱动开发方法;系统建模语言;术语推荐;限定自然语言需求;模型转换

中图法分类号 TP311

安全关键信息物理融合系统^[1-3](safety-critical cyber-physical system, SC-CPS)是指广泛应用于航空、航天、交通、能源等领域,且失效时会产生严重系统风险,从而导致财产损失、环境破坏或者人员伤亡的一类 CPS 系统.安全关键信息物理融合系统不仅集成了计算、网络和物理环境,还同时具有高可靠性、高安全性、强实时性等特点.随着功能、非功能需求的不断增加,系统规模和复杂性不断提高,这类系统出现故障的可能性及其造成的危害也日益突出.

近年来,模型驱动开发(model-driven development, MDD)越来越受到重视,并被工业界认为是保障系统安全性与可靠性切实可行的重要手段.模型驱动开发方法^[4]将模型作为系统工程生命周期的中心,在设计阶段建立系统的抽象模型,尽早对系统进行分析 and 验证,并且支持不同层级模型之间信息的传递和映射.

安全关键系统领域常用的建模语言有 UML, SysML, AADL 等.统一建模语言(unified modeling language, UML)^[5]是一种针对软件工程领域的标准化建模语言,侧重于软件的体系结构建模,但是 UML 还不足以表达系统工程领域中的所有概念.体系结构分析和设计语言(architecture analysis & design language, AADL)^[6]是一种面向安全关键嵌入式系统的建模语言,侧重于软硬件混合建模^[7-8].系统建模语言(systems modeling language, SysML)^[9]是一种针对系统工程领域的图形化建模语言,是对 UML2.0 子集的扩展,侧重于复杂系统的整体架构.相比 UML, AADL 等语言, SysML 可以整合大型复杂系统的各种视图,包括硬件、软件、数据、过程、需求,甚至包括人力和其他资源,适用于进行需求捕获、分析和初步系统设计,目前在工业界得到了广泛应用.因此,本文选择 SysML 作为系统架构建模语言.

模型驱动开发方法的生命周期通常始于系统的分析或设计阶段,较少涉及需求阶段.但是,安全关键信息物理融合系统中导致严重事故问题链的最上端原因往往是系统需求问题^[10],而这类系统的需求常常又是通过自然语言描述的.因此,如何有效链接自然语言需求和模型驱动开发方法,即实现自然语言需求到系统设计模型的自动或半自动转换是模型驱动开发方法在安全关键信息物理融合系统设计与开发过程中面临的一个主要挑战^[11-13].

已有的需求表达达到系统设计模型的转换主要包括基于自然语言处理(natural language processing, NLP)、基于限定自然语言(restricted natural language, RNL)模板^[11,14]等方法.在自然语言处理方法方面, Bajwa 等人^[15]基于自然语言处理方法提出从自然语言需求到 UML 类图中的 OCL(object constraint language)文本说明的转换,而 Letsholo 等人^[16]提出的 TRAM(tool for transforming textual requirements into analysis models)方法,主要考虑自然语言需求到 UML 类图的转换.在限定自然语言模板方法方面,岳涛等人^[17]提出的限定用例建模方法(restricted use case modeling, RUCM)方法是通过对 UML 用例图的文本说明进行自然语言限定,以此降低自然语言带来的歧义性与模糊性,基于 RUCM 给出的 aToucan 工具^[18]支持 RUCM 到 UML 活动图、顺序图等图的自动化生成;RollsRoyce 公司提出的 EARS^[19](easy approach to requirements syntax)方法,通过一些简单的需求结构来捕获和表达用户需求,从而降低自然语言需求的歧义性.王政等人提出的 SPARDL 语言^[20],通过数据字典、模块和模式图对航天软件需求进行捕获.正在制定中的 SysML V2.0 标准^[21]也将通过预定义关键字和句式对需求图的自然语言描述作出限定,从而减少 SysML 建模过程中由自然

语言带来的不精确性等问题.另外,Colombo 等人^[22]提出基于问题框架(problem frames)的需求分析并将需求规约转换到 SysML 的方法,该研究主要考虑基于问题分解的方式获得系统初始层次架构,还未考虑系统非功能等特性.

本文结合自然语言处理和限定自然语言模板两者的优点,面向安全关键信息物理融合系统,提出一种基于限定中文自然语言需求的 SysML 模型自动生成方法.为了尽量不改变国内系统工程师基于文档撰写需求的习惯,我们提出限定中文自然语言需求模板(restricted natural language requirement, RNLReq),这样使得自然语言需求表达不受限于 SysML 需求图.另外,从需求到设计模型的建立,目前大多数情况下是手动实现的,整个过程对于建模人员有着很高的要求,需要掌握来自多个学科领域的知识,所以给出限定自然语言需求到 SysML 设计模型的自动转换.本文所使用的 SysML 设计模型主要包括结构视图(即块定义图)、通信视图(即内部块图)、行为视图(即活动图)及非功能属性扩展等.

文献[23-24]中给出了基于限定中文自然语言需求的 AADL 模型自动生成方法.本文正是在现有研究的基础上,进一步考虑基于限定中文自然语言需求的安全关键信息物理融合系统 SysML 建模,并通过基于人工智能的安全关键信息物理融合系统术语提取和推荐方法,对系统需求中的领域术语和数据字典加以自动提取,提高限定自然语言需求规约工作的自动化程度.

相比已有工作,本文主要贡献有 3 个方面:

1) 提出一种结合智能化推荐的限定中文自然语言需求规约方法.为了减少中文自然语言需求表达中存在的模糊性、二义性,提出基于限定中文自然语言需求模板的需求规约方法.该方法包括领域词库、数据字典和限定自然语言模板.结构化的限定自然语言模板,对系统静态结构、内部交互、动态行为以及非功能约束在内的系统需求的各方面进行规约.为了增强需求规约方法的自动化程度,通过智能化推荐的方法,将系统需求文档中的术语加以提取并推荐到领域词库和数据字典中.

2) SysML 设计模型的自动生成方法.针对 SysML 对安全关键信息物理融合系统的安全性、可靠性、实时性等非功能需求表达能力的不足,提出 SysML 扩展子集,使其具有对非功能属性描述的能力,并给出 RNLReq 到 SysML 子集的转换规则与算法.

3) 原型工具实现和工业界案例分析.基于开源

工具 Papyrus 对本文方法进行原型工具实现,并以航空领域的飞机空气增压系统(airplane air compressor system)作为案例,对所提方法的可用性和有效性进行验证.

1 研究背景

本节主要介绍 SysML 的基本建模概念及自然语言处理技术.

1.1 SysML 概述

国际系统工程学会(International Council on Systems Engineering, INCOSE)和对象管理组织(Object Management Group, OMG)在对 UML2.0 的子集进行重用和扩展的基础上,于 2006 年提出系统建模语言 SysML.SysML 是针对系统工程领域的标准建模语言,可以构建设计良好、层次清晰且可维护的复杂系统.SysML 不仅可以整合大型复杂系统的各种视图,除了包括硬件和软件,还包括数据、过程、需求,甚至包括人力和其他资源,还可以描述系统的层次、结构、行为、属性及组件间的复杂关系.SysML 能够支持各种大型复杂系统的详细说明、分析、设计、验证和确认,适合于进行需求捕获、分析和初步系统设计,目前在主要的工业领域如航空、航天、交通、能源等领域得到了较为广泛的应用.

SysML 中定义了 3 种类别共 9 种图来刻画系统在不同视角下的特征.1)描述需求的视图:需求图(requirement diagram, REQ)提供基于文本需求的建模构件,用于描述自然语言需求、需求之间的关系,以及需求与其他模型元素之间的关系.2)描述结构的视图:模块定义图(block definition diagram, BDD)表示系统之间的组成关系;内部模块图(internal block diagram, IBD)用于描述系统的内部架构及抽象通信;参数图(parametric diagram, PAR)用于描述系统的属性信息的约束条件,表示一种或多种约束如何与系统的属性绑定;包图用于显示系统模型的组织方式.3)描述行为的视图:用例图(use case diagram, UC)从系统参与者的角度表达系统执行的用例,是一种黑盒视图;活动图(activity diagram, ACT)主要关注控制流程,以及输入通过一系列动作转换为输出的过程;序列图(sequence diagram, SD)主要关注模块的组成部分如何通过操作调用和异步信号交互;状态机图(state machine diagram, SMD)主要关注模块的一系列状态,以及响应事件时状态之间的转换.对于特定领域系统

建模, SysML 允许用户通过扩展机制对元模型进行扩展和调整, 获得原始模型不具备的描述信息存储、非功能需求以及分析的能力. 本文主要采用构造型 (stereotypes) 的方式扩展 SysML, 包括 2 种方式: 1) 扩展 (extension) UML 的元类; 2) 继承 (generalization) SysML 已有的构造型.

1.2 自然语言处理

自然语言处理 (NLP) 是指在机器语言和人类语言之间进行“翻译”, 以实现人机交流的目的. 自然语言处理的关键技术包括自动分词、词性标注、命名实体识别、句法分析、语义分析与篇章分析等. 本节主要描述分词、词性标注以及依存句法分析.

1) 分词
分词 (tokenization) 就是将句子、段落、文章这种长文本, 分解成以词语为单位的数据结构, 方便后续的处理分析工作. 同一个长词语, 通常有不同分法, 例如“飞行器管理计算机”, 可以有“飞行器管理计算机”“飞行器\管理计算机”“飞行器\管理\计算机”3 种分法.

2) 词性标注
词性标注 (part of speech tags, POS tags) 就是在给定句子中判定每个词语的语法范畴, 确定其词性并加以标注的过程. 本文中使用到的主要词性如表 1 所示.

3) 句法分析
句法分析 (syntactic parsing) 是对输入的文本句子进行分析以得到句子的句法结构的处理过程. 根据句法结构的表示形式不同, 可以分为 3 种: ①句法结构分析 (syntactic structure parsing), 作用是识别出句子中的短语结构以及短语之间的层次句法关系; ②依存句法分析 (dependency syntactic parsing), 作用是识别句子中词汇与词汇之间的相互依存

关系; ③深层文法句法分析, 即利用深层文法对句子进行深层的句法以及语义分析. 依存句法理论认为“谓语”中的动词是一个句子的中心, 其他成分与动词直接或间接地产生联系. 它将句子分析成一颗依存句法树, 描述出各个词语之间的依存关系. 一个依存关系连接 2 个词, 分别是核心词 (Head) 和依存词 (Dependent). 依存关系可以细分为不同的类型, 包括定中关系 ATT、数量关系 QUN (quantity)、动宾关系 VOB (verb-object)、主谓关系 SBV (subject-verb) 等.

Table 1 Meanings and Examples of Common POS Tags
表 1 常用词性的含义和举例

词性	含义	举例
a	形容词	紧急/a 情况/n
n	名词	计算机/n, 飞机/n
m	数词	4.0/m 千克/q
q	量词	三/m 个/q
v	动词	发送/v
vn	名动词	管理/vn, 接口/vn

2 研究框架

本文面向安全关键信息物理融合系统, 针对中文自然语言需求与模型驱动开发方法之间的鸿沟展开研究, 提出了一种基于限定中文自然语言需求模板的系统需求规约方法, 并通过基于人工智能的术语提取方法对需求规约方法中的领域术语和数据字典加以推荐; 然后, 通过转换规则和转换算法自动生成 SysML 系统设计模型. 本文的研究框架如图 1 所示, 主要分为 3 个部分:

1) 为了确保需求表达的一致性, 采用领域词库和数据字典, 实现对需求文档中的数据、对象名词的

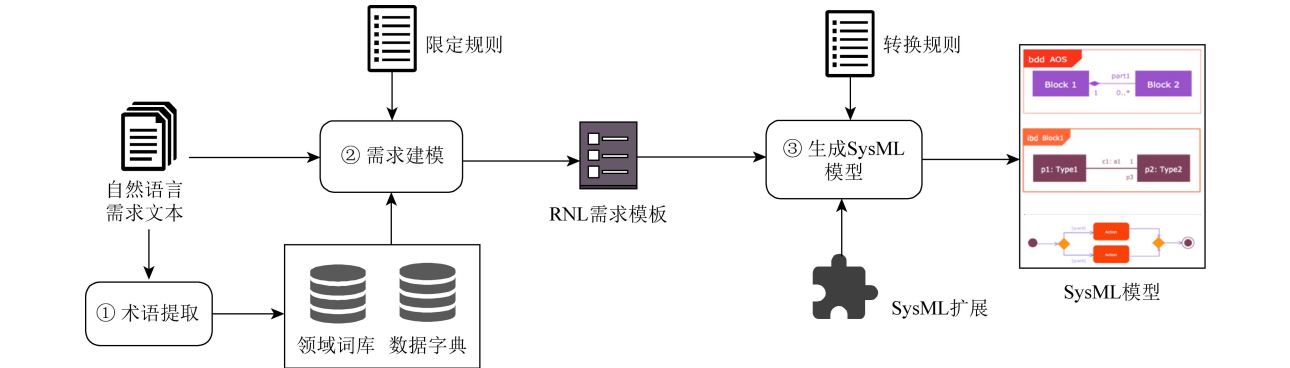


Fig. 1 The framework of the automated approach to generate SysML models from natural language requirements
图 1 基于自然语言需求的 SysML 模型自动生成方法整体框架

集中管理和规范描述.使用自然语言处理技术对系统需求文档进行分析,依据领域特征、专家意见、工程需求和文献调查等制定提取规则,完成领域词库和数据字典的推荐与制定.

2) 在数据字典和领域词库的基础上,按照设定的限定自然语言需求模板完成具体的系统需求规约,在此过程中,为了减少自然语言的二义性,对自然语言的表达方式进行适当限制,即遵循一定的限定规则给出对系统需求的描述.

3) 为了增强 SysML 对安全关键信息物理融合系统的安全性、可靠性、实时性需求的表达能力,提出 SysML 扩展子集.在通过限定自然语言需求模板完成整个系统需求规约后,根据限定自然语言需求模板到 SysML 和扩展的生成规则以及具体的转换算法,生成 SysML 系统设计模型.

3 基于限定中文自然语言模板的需求规约方法

限定自然语言需求规约方法主要由领域词库、数据字典以及需求模板 3 部分组成.

3.1 领域词库与数据字典

对象名词、数据及其属性等通常散布在需求文档各处,不利于保证同一个概念的一致表达.领域词库和数据字典可以分别对对象名词和数据进行集中管理和规范描述,方便后续的需求建模过程.

1) 领域词库

领域词库(glossary repository)是指涵盖需求文档中的各种对象名词的词库.领域词库中需要填写对象名词的中文名、英文名、名词类别、ID 等信息.图 2 给出了领域词库的 BNF 语法.针对安全关键信息物理融合系统领域的实际需求,领域词库可以包括专有的系统名、功能名、组件名、接口名、硬件设备名、模式名、状态名等.通过在领域词库中注册使用的词汇,实现对需求文档编写的限定,以减少在需求编写过程中出现人为错误的可能.

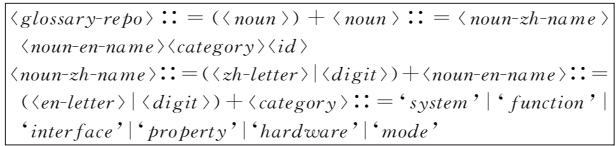


Fig. 2 BNF grammar of glossary repository
图 2 领域词库的 BNF 语法

2) 数据字典

数据字典(data dictionary)是指涵盖需求文档

中涉及的数据及其属性的字典.数据字典中需要填写数据的中文名、英文名、数据组成(例如,(角速度,角位置))、单位、范围、精度、速率、ID 等信息.图 3 给出了数据字典的 BNF 语法:

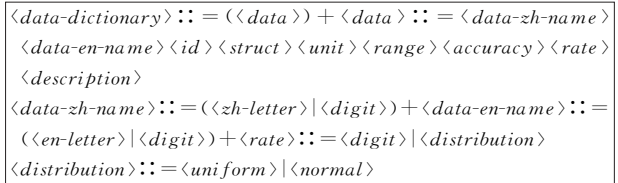


Fig. 3 BNF grammar of data dictionary
图 3 数据字典的 BNF 语法

针对领域的实际需求,数据字典中可以包括静态数据、输入数据、输出数据等^[23].其中,静态数据是指系统中的常数、预先装订的参数;输入、输出数据是指系统在运行过程中各组件的输入、输出数据.此外,针对安全关键信息物理融合系统对安全性、可靠性的要求,可以添加对故障概念进行描述的数据,如故障等级、故障类型、失效影响、失效概率等;针对 SC-CPS 结合连续和离散动态特征的特性,在数据字典中通过数据的速率特征以统一表达,速率取值为数值或者分布,当速率取值为 0 时表示连续的数据流,速率为非 0 时表示离散的数据流,速率取值为分布时表示不确定时间间隔发射的数据.对于常见的固定取值,例如分布、故障等级等,可以预先定义为静态数据,范围为枚举值:1)(distribution),如(Normal(mean, stdDev), Uniform, Basic(min, max));2)(failure level),如(NoEffect, Minor, Major, Hazard).

例如,“输入功能每隔 15 ms 将旋转变压器采集数据发送到控制功能,旋转变压器采集数据包括角速度和角位置数据”,在数据字典中注册该数据的中文名(旋转变压机采集数据/Rotary_transformer_gathered_data)、数据类型组成(例如(角速度,角位置))、单位、范围、精度、频率(10 ms)、ID 等信息,作为对该数据的解析.需要注意的是,这里的角速度和角位置不是基本数据类型,因此也需要注册到数据字典中.

3.2 基于自然语言处理的术语推荐方法

本节我们主要介绍基于自然语言处理技术提出面向安全关键信息物理融合系统的术语推荐方法,这里的术语包括领域词库和数据字典.术语推荐方法的 NLP 过程主要包括分词、词性标注、依存句法分析、规则提取、领域度过滤等步骤,如图 4 所示:

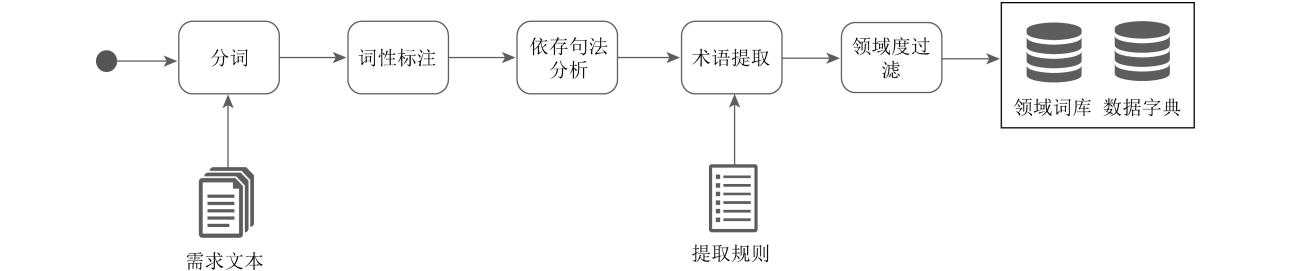


Fig. 4 Framework of glossary extraction

图 4 术语提取技术框架

详细介绍各个步骤:

1) 分词、词性标注、依存句法分析

例如,对于“飞机管理计算机通过远程接口单元,发送启动空压机的指令给空气增压机控制器”需求语句,进行分词、词性标注、依存句法分析,处理结果如图 5 所示.本文中选取的文本分析工具为 HanLP 自然语言分析工具包,由一系列模型与算法组成,因此本步骤直接使用 HanLP 来实现.

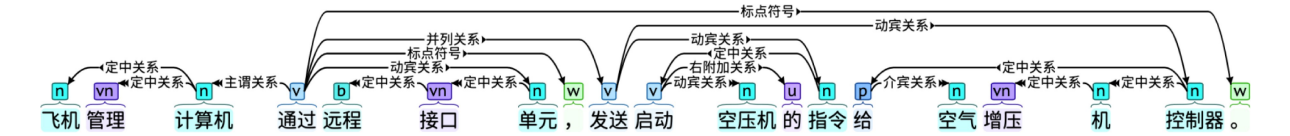


Fig. 5 Results of POS tagging and parsing

图 5 词性标注和句法分析的结果

2) 规则提取

安全关键信息物理融合系统领域术语存在语言结构特征.从外部关联来看,领域术语大多是名词性短语,其经常作为领域文本中的主语、宾语、定语等成分;从其内部语法构成来看,其组成形式包括名词+名词(“空气压缩机控制器”)、形容词+名词(“可信数据”)、动词+名词(“自检功能”)、动词(名词)+单字名词(“电磁阀”)等.

根据语言结构特征、专家意见和文献调查,结合词性标注和依存分析的结果,制定词语匹配规则,如表 2 所示.例如“飞机管理计算机发送启动空气增压机的指令”需求中符合匹配规则的候选术语有 5 个:“飞机”“飞机管理”“飞机管理计算机”“管理计算机”“计算机”,依据 R1 规则可以提取名词短语“飞机管理计算机”“空气增压机”.依据 R2 规则可以提取单独出现的名词“指令”.

Table 2 Pattern Matching Rules

表 2 匹配规则

规则	描述	说明
R ₁	连续出现的名词,且关系为修饰关系时,合并为一个名词短语	“空气/n 压缩机/n 处理器/n”组成一个名词短语,加入候选集.
R ₂	单独出现的名词,作为一个名词短语	“阀门/n”为 NP,加入候选集
R ₃	形容词和紧跟其后的名词短语,组成一个名词短语	“可信/a 数据/n”
R ₄	动词+名词的组合,且关系为修饰关系时,合并为一个名词短语	“自检/v 功能/n”、“发射/v 装置/n”
R ₅	名词+动词的组合,且关系为修饰关系时,合并为一个名词短语	“周期性/n 自检/v”
R ₆	单独出现的动词,且不是描述性动词,作为一个动词短语	“发射/v 火箭/n”
R ₇	连续出现的动词,且它们之间的关系为修饰关系时,这些动词可以被合并为一个动词短语	“启动/v 检测/v”
R ₈	存在动宾关系的谓语动词和直接名词宾语,合并为一个功能概念	“检测传感器上报的数据”中“检测数据”可作为一个功能概念.
R ₉	经过以上规则合并得到的名词短语,若为句子主语则加入领域词库中,且为其添加类型“系统”.	“控制功能主要控制电机、电磁阀和加热器”中的“控制功能”
R ₁₀	前后 2 个名词,且中间为助词“的”字结构,分别添加为“系统”类型和“属性”类型	“ACC/n 的/u 重量/n”

3) 领域度过滤

安全关键信息物理融合系统领域术语还存在特定领域特征,即该术语通常仅出现在一些特定领域的文本中.领域度过滤对存在常用词语和停用词的候选术语进行过滤.候选术语中“计算机”和“发送启动”为不存在领域特征的常用词语,因此被过滤.

经过以上步骤后,将候选词推荐给系统工程师进行人工分析筛选,用于后续的需求建模,可以有效提高建模工作的自动化程度.

3.3 限定中文自然语言需求模板

随着安全关键信息物理融合系统越来越复杂,系统中包含更多的来自不同领域的物理子系统、计算子系统、通信子系统.这类系统往往采用分布式构件设计,将一个完整的系统看成若干独立的子系统,每个独立的子系统封装为标准化的构件,对这些标准化构件的组合可以开发出具备特定功能的系统.我们正是基于系统分解结构来组织需求模板的结构.

在基于构件的系统中,系统的体系结构可以层次分解为系统、子系统、功能以及共享功能.对应的,需求模板之间的层次关系则显式地表达系统的体系结构.其中,系统(system)需求模板可以细分为多个

子系统(subsystem)需求模板,按照自顶向下分解的方法,子系统可以继续划分为多个子系统、功能(function)或共享功能(shared function)模块.

功能需求模板是只能被所属系统调用的功能.此外,共享功能模块作为独立的模块挂载在系统中,可以被其他系统或功能调用,用于描述一些经常被调用到的算法(如傅里叶变换、矩阵变换、欧拉角计算)或与外设交互(如从总线读写数据).在进行自然语言需求模板设计时,除了考虑系统构件化特征之外,还可以在需求模板中添加部分必要的设计相关元素,用于支持后期设计模型的自动化生成.

系统、子系统、功能以及共享功能均需遵循如表 3 所示的基本模板.对于每个系统或子系统而言,都有相应的输入输出信号(signal),包括数据(data)和事件(event)两种类型,同时每个(子)系统还需对其功能需求(functional requirement)、性能需求(performance requirement)、安全性需求(safety requirement)、可靠性需求(reliability requirement)进行描述.而功能需求、性能需求、安全性需求、可靠性需求还要遵循各自的模板.此外,每个(子)系统都有对应的原始需求 ID,用于记录对应的原始文本需求.下面给出具体的自然语言限定规则.

Table 3 Restricted Natural Language Requirement Template
表 3 限定自然语言需求模板

ID	唯一标识符
中文名	名词,描述系统/子系统的名称.
英文名	系统/子系统对应的英文名称或英文简称.
多重性	该系统可装配的数量.顶层系统为 1,其余为非负整数.
输入/输出	系统/子系统的输入/输出,包括数据(data)和事件(event)两种类型,可以为空.
组成部分	一个复杂的系统包含的若干个子系统及功能模块.
属性	每个系统拥有的静态属性,可以定量描述.
功能需求	包含顺序、选择、并行等执行结构.
性能需求	规定每个功能单元在性能方面的约束,如实时性、精度、最大处理能力等.
安全性需求	用于描述系统在各组件层面上的安全性要求.
可靠性需求	用于描述系统在各组件层面上的可靠性要求.
描述	对系统功能与组成进行简单的描述.
原始需求 ID	原始需求的编号,可以有多个.

1) 通用约束规则

由于自然语言表达存在着二义性,在填写需求模板时会造成歧义,针对不同类型的需求,本文规定了一系列通用的自然语言约束规则来约束需求描述的填写,如表 4 所示.

我们进一步以功能、性能、安全性、可靠性等需求为例对限定自然语言约束规则进行细化说明.

2) 功能需求约束规则

功能需求是指系统在功能层面上的需求描述,即事件、触发条件、不变量等之间的关系,还包括

多个事件序列、多个条件序列等复杂执行的情况.对于安全关键信息物理融合系统而言,功能执行主要包括顺序执行、选择执行、循环执行等控制结构,而每个控制结构由若干个简单行为句式组成.针对系

统的功能特征,制定了 5 种简单行为句式:发送(数据、信号)、赋值(数据)、调用(功能)、延时、跳转,调用语句可以描述不同模块之间的功能交互关系.具体的简单行为句式说明如表 5 所示.

Table 4 General Restriction Rules for Natural Language

表 4 通用自然语言约束规则

规则	句式描述	正例	反例
R_1	需求语句的主语必须存在,且为来自领域词库的名词.	“系统调用计算功能”.	“调用计算功能”.
R_2	只使用陈述句.	“若系统已上电,执行…”	“系统已上电吗?”
R_3	只使用现在时.	“系统打开加热器”	“系统打开加热器”
R_4	用主动语态而不用被动语态.不使用“被”字句.	“控制系统打开阀门”,“阀门接收到打开信号”	“阀门被控制系统打开”
R_5	不使用代词,如“它”、“这个”等.	“系统打开加热器”	“这个系统打开加热器”
R_6	不使用情态动词(如大概)、表示否定含义的副词/形容词.	“系统处于失效状态的概率为…”	“系统大概处于失效状态”
R_7	只使用简单句,将复杂句分解为多个简单句.	① 系统存储过滤器的寿命 ② 系统存储电磁阀开关次数 ③ 系统存储电流信息	系统存储过滤器的寿命(通过压力传感器来确定)、电磁阀开关次数,还有电流信息
R_8	准确地描述系统间的交互,如在功能句式中,主语和宾语都不能丢失.	“故障监控功能将状态参数上报到 VMC”	“故障监控功能将状态参数上报”
R_9	以一致的方式使用词语,使用固定的名词来描述某个事物.	“空压机控制系统发送上电自检信号”, “空压机控制系统发送周期自检信号”	“空压机控制系统发送上电自检信号”, “气体压缩机控制器系统发送上电自检信号”

Table 5 Simple Behavior Sentences for Restricted Natural Language

表 5 限定自然语言简单行为句式

规则	句式描述	举例说明
S_1	SimpleBehaviorSentence	将功能行为分解后的最小粒度.包括 $S_2 \sim S_7$.
S_2	Send<Data> at <Port> 表示发送数据的动作,其中<Data>和<Port>均为数据字典中定义的数据类型.	“(控制子系统)发送电机控制信号到电机控制端口”.
S_3	Assign<Data-A> to <Data-B> 表示对某一数据的赋值动作,其中<Data-A>,<Data-B>均需在数据字典中定义其类型.	“(控制子系统)将气体出口温度赋值到自检参数-温度”.
S_4	Call<Function> 表示对系统内部的一个功能模块的调用,其中<Function>为已建立的功能需求模板.	“(控制子系统)调用计算 PWM 参数”.
S_5	Call < S_1 ><SF> 表示对系统外部的一个共享功能模块的调用,其中< S_1 >为系统名词,<SF>为共享功能需求.	“(初始化子系统)调用(存储子系统)存储参数功能”.
S_6	Delay <Digit><Unit> 表示功能执行过程中的停顿,其中<Digit>为数值,<Unit>为预定义的时间单位.	“(初始化子系统)调用参数重置功能;暂停 20 ms”.
S_7	Jump <Function> “跳转到”表示转移执行功能流程中的指定功能,<Function>为 $S_2 \sim S_6$ 中的简单行为.	“(控制子系统)跳转到周期自检功能”.

上述的简单行为句式只能表示基本的功能行为,不能适用于系统需求中的一些复杂的事件序列、条件序列等执行行为,因此需要通过预定义的关键字(IF ELSE,AND,OR,LOOP 等)和句式将简单的描述语句组合成复合行为句式(compound sentence).表 6 给出限定自然语言复合控制结构的基本定义.这里以一条实际需求语句解释我们的规约思路.例如,需求语句“若加电设备运行超过 2 h,那么系统应该

Table 6 Compound Structures for Restricted Natural Language

表 6 限定自然语言复合控制结构

规则	句式描述	举例说明
R_{10}	SimpleValueCondition: $\langle P_1 \rangle \langle Comparator \rangle \langle Value \rangle \langle Unit \rangle$	“气体出口温度小于 15℃”(值条件)
R_{11}	ValueCondition: 由 AND, OR 等连接的多个 SimpleValueCondition	“气体出口温度大于 15℃”“与”“气体出口温度小于 30℃”(由“与”“或”等连接的值条件)
R_{12}	EventCondition: 当前模块接收到事件 E	“接收到加热器 1 打开信号”(事件条件)
R_{13}	TimeCondition: DURING(T_1, T_2)	“在 T_1 到 T_2 时间内”(时间段条件)
R_{14}	ConditionalSentence: IF <i>Condition</i> , THEN <i>SentenceA</i> , [ELSE <i>SentenceB</i>]	“若气体出口温度小于 15℃, 执行调用打开加热器 1, 否则执行...”(条件判断句式)
R_{15}	LoopSentence: LOOP <i>SentenceA</i> EACH T UNTIL E	“每隔 15 min 发出一次自检信号直到加电结束”(循环句式)
R_{16}	ConcurrentSentence: MEANWHILE (<i>SentenceA</i> , <i>SentenceB</i> , <i>SentenceC</i> , ...)	“同时执行电机寿命计时和寿命值存储”(并行句式)

每隔 15 min 发出一次自检信号直到加电结束”,首先需要在领域词库中添加表示“加电结束状态”的词条,在数据字典中定义“加电设备运行时间”数据,其数据组成类型为 int,单位为 h;应用 S_2, R_{14}, R_{15} 三条约束规则,可将该需求重写到需求模板:“[IF *Power-UP-Device-Run-Time* ≥ 2 h, [Loop[Send: *BIT-Signal* to *BIT-Port*]] EACH 15 min UNTIL *POWERUP-STOPPED*]]”,这里的 Send 行为为当前需求中包含的最小行为单元.如果这里在发送自检信号的同时需要进行计数,则可以应用 R_{16} 的并行语句 MEANWHILE [Send: *BIT-Signal* to *BIT-Port*, Call: *CountFunction*]来替换原来 Loop 语句中的 Send 行为.

通过多层复合结构的组合,限定自然语言需求模板可以表达复杂的行为模式.图 6 给出功能需求模板的结构示意图.从模板结构中我们可以看出,整个功能需求模板是一个复合行为,可以根据实际需求填写其中的空位,而它的系统行为的空位,可以填写的模板类型既可以是多个简单行为,也可以是层层嵌套的复合行为,使得功能需求模板具有良好的鲁棒性和可扩展性,能够表达层次性很强的功能行

为.对于复杂的功能,我们可以将其层次分解,把其中出现在多个地方的通用功能行为(如算法)抽取出来,在单独的功能需求模板中定义它,然后只需要多次调用它.

3) 非功能需求约束规则

系统的功能需求定义了期望一个系统做什么,而非功能需求(non-functional requirements, NFRs)则制定了关于系统如何运行和功能如何展示的全局限制.这里需要注意,非功能需求并不是独立于功能需求而存在的,非功能需求是对功能需求的补充.

非功能属性包括安全性(safety)、可靠性(reliability)、性能(performance)、可维护性(maintainability)、可伸缩性(scalability)和可用性(availability)等,这些非功能属性对于航空航天 CPS 而言尤为重要,因为它们无法提供服务时会造成巨大损失.其中,能够定量描述与分析的非功能属性主要有性能、可靠性、安全性.系统的时间性能要求,即实时性要求,包括周期性/非周期执行、Deadline、Timeout、Jitter、响应时间、可调度性等.可靠性要求,包括系统的故障、失效的发生概率.系统的资源性能要求,包括对功耗、CPU、存储、总线、网络带宽等资源

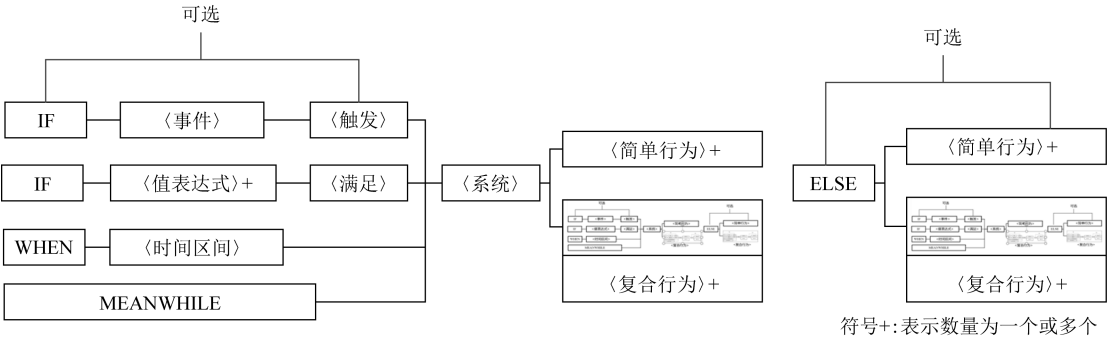


Fig. 6 Structure of the functional requirement template

图 6 功能需求模板结构

消耗和对温度、质量、压力等物理环境的约束.系统的安全性要求,一方面采用定性描述的方式来表达系统的故障行为和故障处理,在安全关键信息物理融合系统领域常用的故障或失效概念包括输入异常、设备故障、断电故障、通信故障等;另一方面采用定量描述的方式描述失效影响等级等.

依据安全关键信息物理融合系统特征,提出适用于性能、可靠性等非功能需求模板,如表 7 所示,并通过将非功能需求模板集成到对应的系统/功能模板中,使得系统工程师在设计阶段就可以全面考

虑功能设计和非功能设计.

规则 R_{17} 适用于约束系统的静态属性.例如,需求语句“周期自检功能的周期为 2 ms”,应用规则 R_{17} 可以表达,该需求语句可以重写为:“ $Period = 2\text{ ms}$ ”并添加到周期自检功能的性能需求中.而规则 R_{18} 则复用自规则 R_{15} ,用于约束系统的动态行为,对于周期性执行的行为,该需求将添加到功能需求模板之中.安全性需求中的故障处理行为,与功能需求紧密相关,可以应用规则 R_{14} 添加到功能需求中.

Table 7 Non-functional Requirement Sentences for Restricted Natural Language

表 7 限定自然语言非功能需求句式

规则	句式描述	举例说明
R_{17}	ConstraintSentence; $\langle Property \rangle \langle Comparator \rangle \langle Value \rangle \langle Unit \rangle \langle Margin\ of\ Error \rangle$	“周期自检功能的周期为 2 ms”(周期性) “最大允许失效概率为 0.05%”(失效概率) “处理器的主频不得超过最高能力的 90%”(资源约束) “ACC 的重量小于等于 4.0kg,误差识别小于等于 2g”(资源约束) “空压机控制系统丧失供电功能的影响等级为 I 级”(安全性)
R_{18}	PeriodicitySentence; LOOP SentenceA EACH T UNTIL E	“工作状态下,电机每 100 μs 接收一次驱动指令”(周期性)

3.4 限定自然语言需求模板元模型

基于 3.3 节的需求模板约束规则,给出对应的需求模板元模型 RNLReqTemplate.如图 7 所示,Model 描述的整个模型的最顶层概念,一般是指最顶层系统.System 实质上描述的是子系统的概念,一般而言,最顶层系统可以划分为若干功能相对独立的子系统,子系统依据其复杂程度又可进一步作为系统,

再次划分子系统,依此迭代,可以进行多层子系统划分.

此外,对于每个系统或子系统而言,都有相应的输入、输出信号(signal),信号一般分为数据(data)类型和事件(event)类型,同时每个(子)系统还需对其功能需求、性能需求(Performance Req)、可靠性需求(Reliability Req)、安全性需求(Safety Req)进行描述.

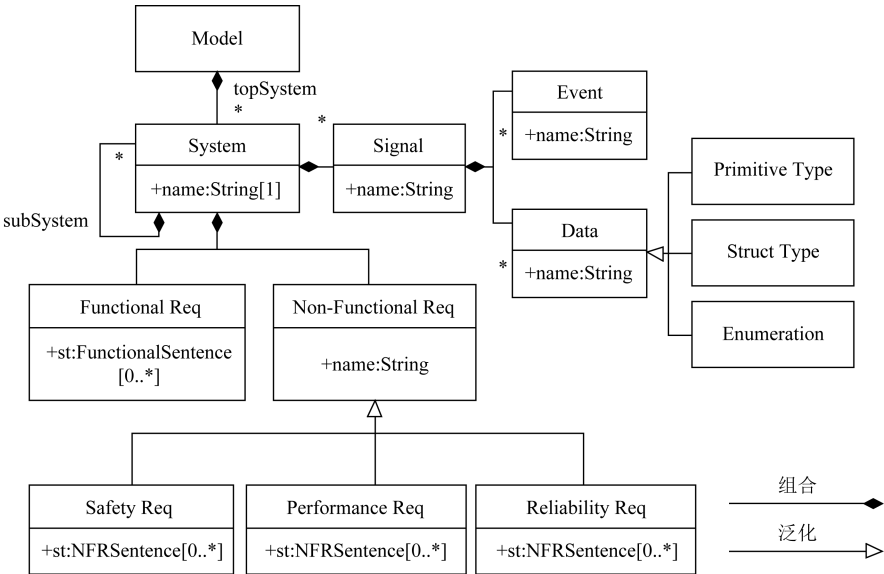


Fig. 7 Meta model of the RNLReqTemplate

图 7 RNLReqTemplate 元模型

节点可以由 Activity 定义,执行 CallBehaviorAction 时可以调用另一种行为,实现复杂功能行为的层次分解.活动分区(ActivityPartition)表示多个子系统之间的动态交互,实现结构到行为的分配.针对 SC-CPS 中连续与离散的行为特性,通过对 Control-Flow 添加 Rate 构造型以表达以一定速率发送的事件.如图 9 所示,在空气压缩机系统案例中(详见 5.2 节),自检模块接收到上电信号后,周期自检和上电自检这 2 个任务并行执行,其中周期自检信息每隔 15 ms 发送并处理直到收到下电的信号,而上电自检信息只会处理一次.输出模块和地面设备这 2 个外部设备的功能被执行,且地面设备下载数据的功能在对应的行为视图中进行了细化描述.

4) 非功能属性视图

如图 8 中的红色方框所示.基于 Constraint 的扩展以对系统中定量描述的非功能属性进行建模,Constraint 可以通过形式化的 OCL 表达式表达系统需求中的约束.这种方式可以将约束描述直接应

用到某类对象的任何实例上.图 10(a)展示了如何使用《AbstractRequirement》和《constraint》一起定义一个新的 ConstraintRequirement 约束需求构造型.因为 ConstraintRequirement 的元类是 Constraint,又是 AbstractRequirement 的子类型,所以它能够描述受约束的需求,并且 ConstraintRequirement 可以用于修饰 Block,Activitynode,Signal 等元素,将需求信息分配到这些元素中.图 9 中描述了功能行为中包含的一条性能需求“上电时间不得超过 10 ms”.图 10(b)给出了 CPS 系统中常见的温度需求,温度传感器 Sensor1 受到一个 ConstraintRequirement 构造型的约束,该约束使用 OCL 语言表达了一个限制传感器工作温度在-55~120℃的性能需求.约束的数据类型为基本类型,或由数据字典中自定义数据类型转换而来的自定义构造型.Block,Interface,Signal 等元素与 AbstractRequirement 通过《trace》关系相连接,表明需求最终在图上的某些设计元素中得到满足.

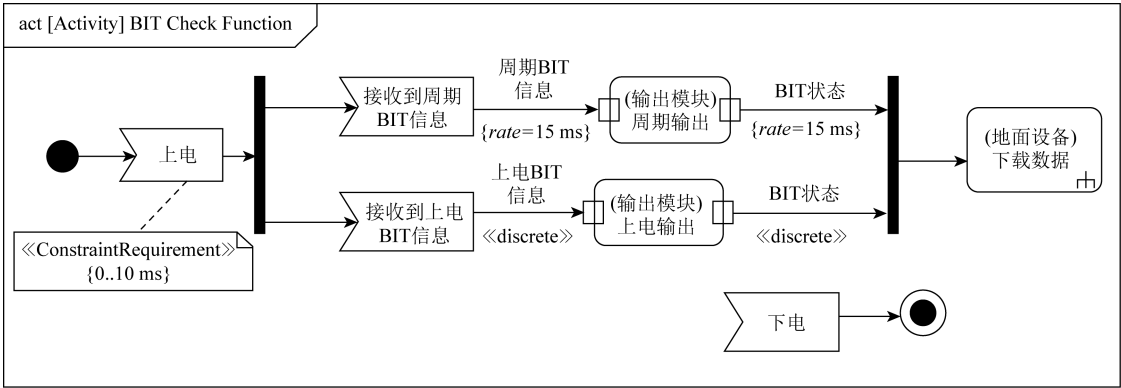


Fig. 9 Example of behavior view
图 9 行为视图举例

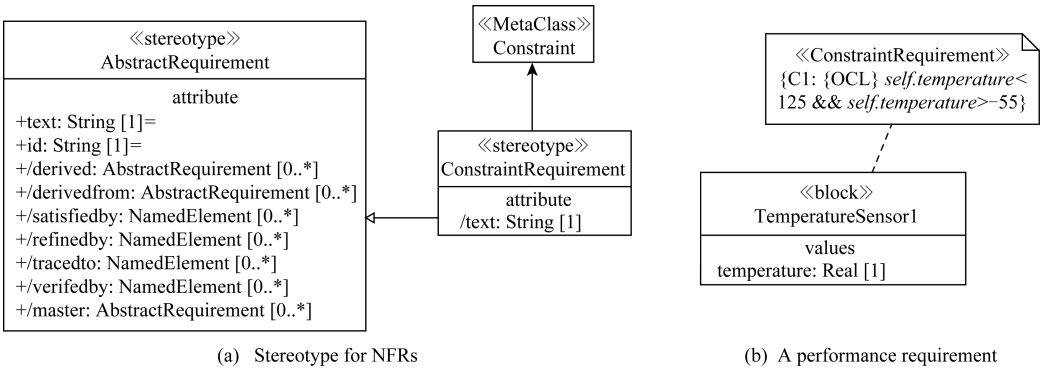


Fig. 10 Definition of the NFR extension and examples of performance requirement
图 10 非功能属性扩展定义及性能需求举例

4.2 转换规则

限定自然语言需求模板到 SysML 的转换方法

可以分为 3 个部分:需求结构到层次结构视图和内部交互视图的转换、功能需求到行为视图的转换、

非功能属性需求到非功能属性视图的转换.这里使用 System, Subsystem, Function 类型的 Component 分别表示系统、子系统、功能需求.

1) 需求结构到层次结构视图和内部交互视图的转换,具体的转换规则为:

规则 1.对 System 和 Function 类型的 Component,生成同名的 BDD 图以及 Block,系统属性转换为 Block 的 property,若属性的数据类型为数据字典中的非基本数据类型,则生成对应构造型.

规则 2.顶层 System Component 转换为 BDD 中的顶层 Block,子系统转换为下层 Block,上下层 Block 之间通过 Composition Association 连接.

规则 3.对内部端口连接信息不为空的 System Component,生成 IBD 和 Block 类型的实例.

规则 4.Subsystem Component 的 Block,添加为上一步生成 IBD 中 Block 的 Property,将端口连接转换为 Property 之间的 Port.

2) 功能需求到行为视图的转换,具体的转换规则为:

SysML 活动图是对表示系统的 Block 的功能行为的建模,主要通过活动节点、活动之间的迁移来描述系统的行为.功能需求 Functional Req 的存储形式为 Component 的 CompoundBehavior,其中由 SimpleFunction 组成功能链,将链表中的每个 SimpleFunction 转换为动作节点,并适当地添加控制节点.下面详细介绍具体的转换规则.

规则 1.对 Function 类型的 Component,生成同名的 ACT 图以及 Activity.将输入输出数据添加为 ActivityParameterNode.

规则 2.对 Component 的 CompoundBehavior 进行遍历,得到 SimpleBehavior 组成的链表.

规则 3.如果链表非空,则生成 InitialNode 和 FinalNode.

规则 4.对于条件判断类型的 Condition,转换为 DecisionNode 类型的节点.如果有 Else,则生成 2 条出边 ActivityEdge;如果没有 Else,则只生成一条出边 ActivityEdge.对 DecisionNode,生成对应的 MergeNode,两者分别作为头尾包裹住 Condition 后的 Function.

规则 5.对于 MEANWHILE 语句,生成 ForkNode 和 JoinNode,两者分别作为头尾与 MEANWHILE 后的 CompoundBehavior 生成的节点链相连.

规则 6.对于事件触发类型的 Condition,转换为

AcceptEventAction 类型的活动节点,根据输入事件生成 InputPin 添加为节点的属性,该节点只有一条出边 ActivityEdge.

规则 7.将 Call 类型的 AtomFunction 转换为 CallBehaviorAction 类型的活动节点,该节点只有一条出边 ActivityEdge.如果存在与该 AtomFunction 同名的 ACT 图以及 Activity 元素,则将同名的 Activity 元素设为节点的 behavior.输入和输出生成 InputPin 和 OutputPin,添加为节点的属性.

规则 8.将 Send 类型的 AtomFunction 转换为 SendSignalAction 类型的活动节点,该节点只有一条出边 ActivityEdge.

基于转换规则,给出从 RNLReqTemplate 到活动图转换的基本算法,如算法 1 所示.

算法 1. 功能需求到活动图的转换算法.

输入:RNLReqTemplate;

输出:SysMLModel.ActD.

```
① for each Component c in RNLReqTemplate
    where Type=SYSTEM and hasFunction do
②   Diagram actD=new ActDiagram(c.name);
③   diag.initialNode=new InitalNode();
④   diag.finalNode=new FinalNode();
⑤   for each Composite Function cf in c do
⑥     Node start=new Node();
⑦     Node end=new Node();
⑧     traverse(cf,start,end);
⑨   end for
⑩ end for
⑪ def traverse(func,start,end)
⑫   if (func.type=SimpleBehavior) do
⑬     Edge edge=new Edge();
⑭     Node n=new Node(func);
⑮     start.next=n;
⑯     end.prev=n;
⑰   else do
⑱     if (func.ifCondition≠null) do
⑲       Node dn=new DecisionNode
         (func.condition);
⑳       if (func.elseCondition≠null) do
㉑         dn.nextIfNode=
           traverse(func.ifFunction,
             start,end);
```

```

22      dn.nextElseNode =
          traverse(func.elseFunction,
                  start,end);
23  else
24      dn.nextIfNode =
          traverse(func.ifFunction,
                  start,end);
25      dn.nextElseNode = null;
26  end if
27 end if
28 end if
29 for each Function f2 in func do
30     traverse(f2,start,end);
31 end for
32 end def

```

3) 非功能属性需求到非功能属性视图的转换, 具体的转换规则为:

需求模板中的非功能需求包括对系统的属性约束和对功能行为的约束,下面详细介绍具体的转换规则.

规则 1.若类型为 SYSTEM 的 Component 组件中存在 Performance Req, Safety Req, Reliability Req,在 Block 中生成该 Req 描述的 Property 属性.

规则 2.由 Component 组件中的 Constraint-Sentence 生成 ConstraintRequirement 元素,将值约束转换为 OCL 表达式.

规则 3.若非功能需求在 System 中,并将 ConstraintRequirement 分配到 BDD 图中 Component 组件对应的 Block 上.

规则 4.若非功能需求在 System 中,并将 ConstraintRequirement 分配到活动图中 Component 组件对应的节点上.

5 工具实现与案例分析

首先介绍 RNL2SysML 原型工具的总体框架;其次介绍工业界案例飞机空气增压系统,使用限定自然语言需求模板对飞机空气增压系统建模;最后给出通过 RNL2SysML 工具生成的 SysML 初始系统模型.

5.1 RNL2SysML 原型工具实现

原型工具是在集成 Papyrus 插件的 Eclipse 平台上进行开发.开源建模工具 Papyrus 为 SysML 提供了完整的建模支持和 SysML 所需的图形编辑器.

RNL2SysML 工具框架如图 11 所示,主要包括术语推荐模块 Glossary Extractor、需求视图模块 Req Viewer、需求模板 RNL Req Tool、RNL2SysML 转换引擎、图形布局生成模块 Layout Visualizer 等.

需求视图模块 Req Viewer 基于 MVC(model-view-controller)的设计思想,将用户界面与数据模型和模型操作进行分离:1)提供了中文用户界面,主要基于领域词库和数据字典中的中文名,以及将句式中的关键字转为对应的中文描述,如 R₁₄ 句式中的 IF-THEN-ELSE 转为“若-执行-否则执行”.2)遵循表 4~7 中的约束句式进行设计,通过在界面上向用户提供选择而不是填写,以尽可能避免用户填写导致的描述不一致现象.

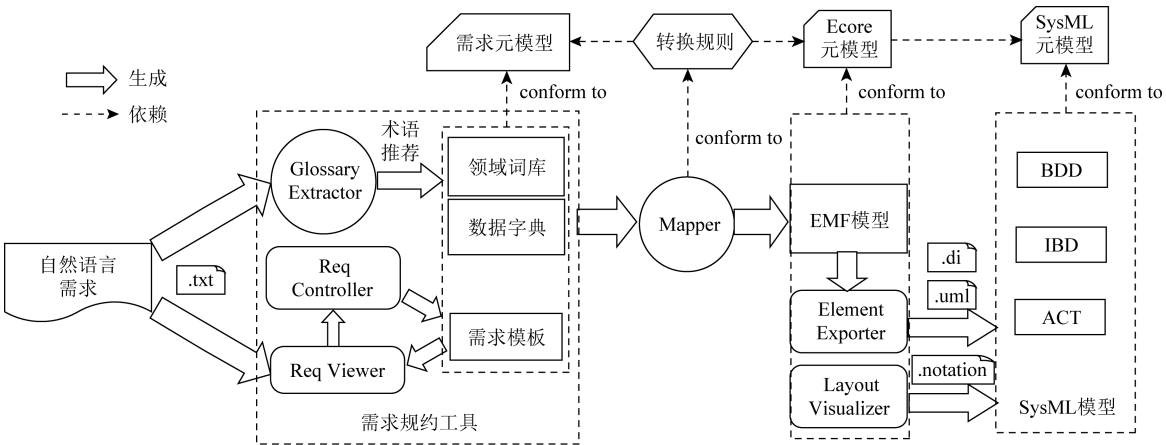


Fig. 11 Framework of prototype tool

图 11 工具框架

RNL2SysML 原型工具的主要功能代码统计如表 8 所示:

Table 8 Statistics of Functions of the Prototype Tool
表 8 原型工具主要功能的代码统计

功能模块	描述	Java 代码行数
Req Viewer	需求视图模块,用户操作的外部接口	2 000+
Req Controller	需求模型控制模块	2 000+
Glossary Extractor	术语提取模块	1 000+
RNL2SysML Mapper	RNL 到 SysML 元素的映射	1 000+
SysML Element Exportor	生成 SysML 模型元素对应的 xml 文件	1 000+
SysML Layout Visualizer	生成 SysML 图形布局对应的 xml 文件	1 500+

5.2 案例分析

本文的实验分析旨在回答 3 个研究问题:

问题 1.本文提出的术语提取方法的精确度如何? 问题 1 旨在探究本文提出的术语提取方法与通用的自动术语提取方法的比较结果.

问题 2.本文生成的 SysML 模型质量如何? 问题 2 在探究本文提出的模型生成方法生成的设计模型对系统需求的覆盖率,以及与其他工具的转换能力的比较结果.

问题 3.本文提出的模型生成方法的实用性如何? 问题 3 旨在调查本文方法的学习成本,以及在实际应用中与人工建模时间的对比.

我们通过航空工业界的飞机空气增压系统案例对问题 1~3 进行评估,受篇幅限制,这里只展示部分需求.

首先,我们对飞机空气增压系统案例进行介绍.飞机空气增压系统(airplane air compressor system, AACS),是保障飞机正常运行的核心系统,它为飞机供气增压,最核心的部件是空气增压控制器,将机上的 270 V 高压直流电源,采用无刷方式转换成方波或正弦波交流电源,用于驱动电机运转,从而带动空压机工作.如图 12 所示为 AACS 的逻辑架构,AACS 主要功能包括初始化功能、供电功能、控制功能、自检功能、模式管理功能、故障处理功能、存储功能、输入功能和输出功能,下面分别介绍.

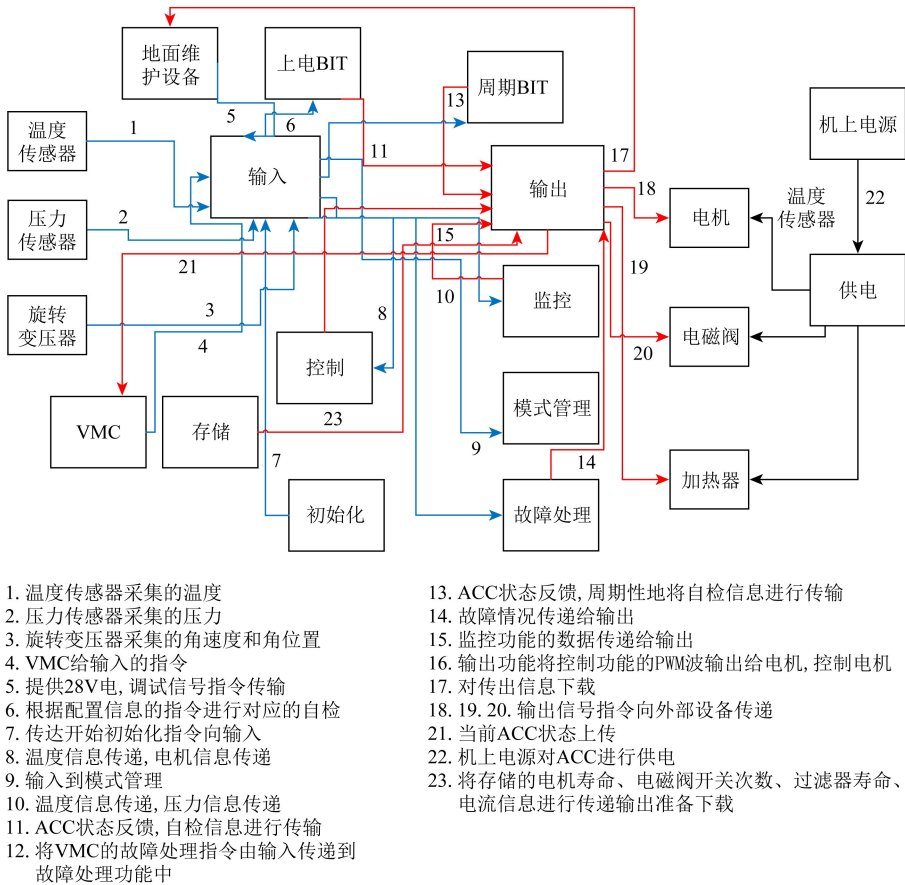


Fig. 12 Logical architecture of AACS

图 12 AACS 的逻辑架构

- 1) 初始化功能,对电路进行初始化和软件的全局变量进行初始化.
- 2) 供电功能是机上电源对供电功能提供正常机上的 270 V 电源.
- 3) 控制功能主要控制电机、电磁阀和加热器,将三者的控制指令通过输出功能传输给对应的设备.
- 4) 自检功能分为上电 BIT 和周期 BIT,上电 BIT 主要在上电时执行芯片、处理器、传感器状态等信息的自检,周期 BIT 主要在正常工作、安全保护或失效保护时执行电流、温度等信息自检.
- 5) 模式管理,根据飞行器管理计算机(VMC)的命令进行模式切换.例如,上电模式、正常工作模式、安全保护模式、失效保护模式、下电模式、地面维护模式.
- 6) 故障处理,根据不同类别的故障处理指令进行处理.
- 7) 存储功能是将电磁阀的开关次数、电机寿命、电磁阀寿命、电流等信息进行存储.
- 8) 输入功能是将 VMC 的指令、温度传感器采集的温度、压力传感器采集的压力还有旋转变压器采集的角速度和角位置等信息进行接收.
- 9) 输出功能包括:将周期自检信息或者将上电自检信息进行接收,准备传输给 VMC;将存储在存

储器中的信息下载到地面检测设备中,将控制功能中传输过来的控制指令传输到外部设备(电机、电磁阀、加热器)中从而控制电机的转速、电磁阀的开关和加热器的开关;将接收到的周期自检或上电自检信息或者故障情况传输给 VMC.

AACS 的外部设备有温度传感器、压力传感器、旋转变压器、电机、电磁阀、加热器、VMC、地面维护设备、机上电源.VMC 主要功能是发送上电指令,接受 ACC 反馈的上电和周期自检数据,决策并发送正常工作、故障清除等指令给 ACC.

我们对案例应用 RNL 需求建模.

首先,对系统需求进行层次分解,将系统功能划分为 3 层,最顶层为飞机空气增压系统,细分为二级子系统:初始化功能、供电功能、控制功能、存储功能、输入功能、输出功能等.供电功能可划分为 ACC 供电和 RIU 供电;存储功能可划分为参数记录,存储和存储下载;输入功能可划分为 VMC 指令输入、温度传感器采集、压力采集和旋转采集.飞机空气增压系统的需求模板层次结构如图 13 左侧所示.

在系统层次分解的基础上,分别按照表 5~7 中对应的需求模板完成需求规约,完善输入输出信息、功能需求、约束信息等.图 13 右侧为控制功能的 RNL 需求.首先描述系统的基本信息,如中文名、英文名

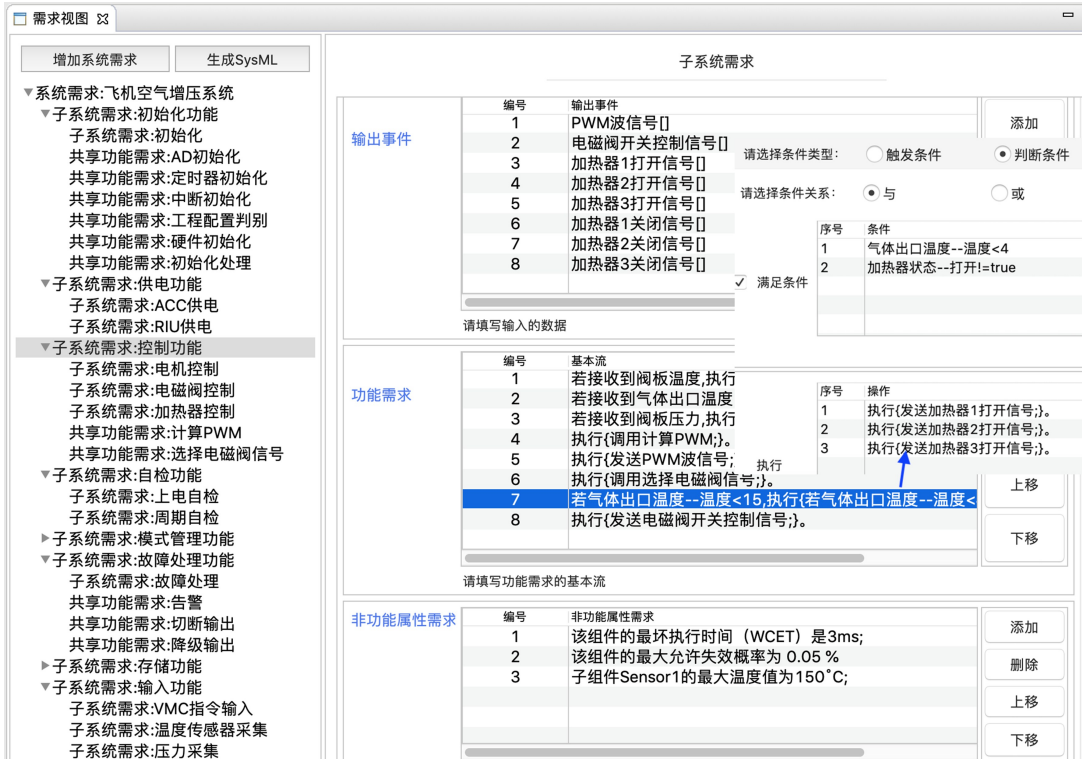


Fig. 13 RNL requirement template

图 13 RNL 需求模板

(来自领域词库和数据字典)、描述信息.然后在功能需求中详细描述功能行为的执行顺序,在接收到阀板温度、气体出口温度和阀板压力等数据后,执行计算 PWM、发送 PWM 波信号等 8 步操作.其中第 4 步中调用了共享功能需求“计算 PWM”;第 5,6 步分别为发送和调用行为句式;第 7 步中根据系统

参数值判断当前系统状态,发送信号控制对应外部设备.

使用 RNL 插件进行需求建模后,通过 RNL2SysML 转换工具,自动生成 AACs 的 SysML 初始系统模型.AACs 的层次结构视图 BDD 图如图 14 所示,正确转换了需求模板中的 3 层结构.

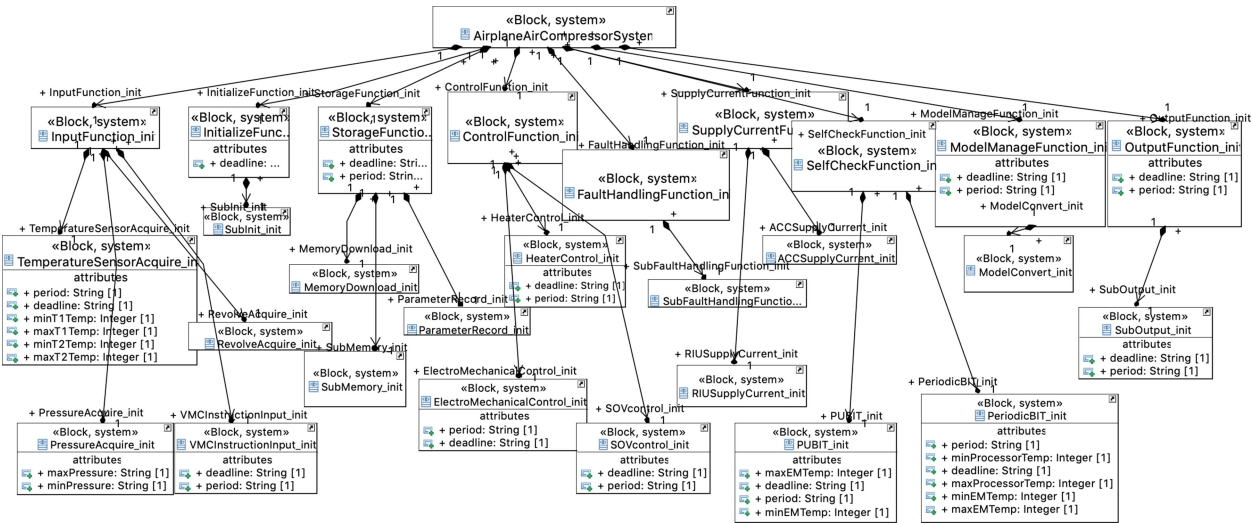


Fig. 14 Block definition diagram of AACs

图 14 飞机空气增压系统的 BDD 图

图 15 为 RNL2SysML 工具生成的 AACs 的最顶层的 IBD 图,描述了二级功能之间的交互关系.例

如,二级功能的输入功能传输 VMC 下达的自检指令,初始化指令模式切换指令,并将温度、压力、电机

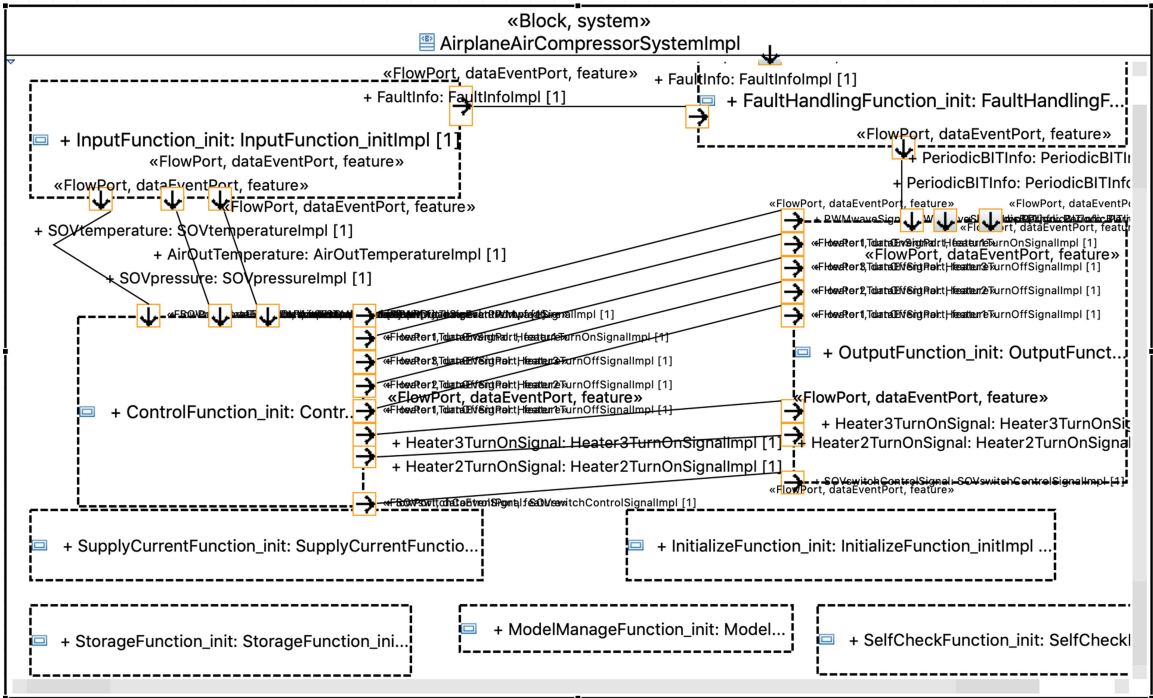


Fig. 15 Internal block diagram of AACs

图 15 飞机空气增压系统的 IBD 图

转子的角速度和角位置传递给控制功能传输,输出功能中将控制功能中传输过来的控制指令传输到外部设备、电机、电磁阀、加热器中从而控制电机的转速、电磁阀的开关和加热器的开关.顶层系统 Block 中包含了所有二级功能的实例作为 Property,数据端口建模为 Port,二级功能通过 Port 属性与其他功能进行连接.

图 16 为 RNL2SysML 工具生成的飞机空气增压系统的控制功能的活动图,控制功能是 AACS 最重要的功能,主要控制电机、电磁阀和加热器,将三者的控制指令通过输出功能传输给对应的设备.当接收到传感器采集的阀板温度、气体出口温度以及阀板压力后,调用计算 PWM,发送 PWM 波信号给 VMC,之后选择电磁阀信号,根据气体出口温度判断当前气罐温度的高低,若气体出口温度大于 15℃,则依次关闭加热器,若气体出口温度小于 15℃继续判断,若气体出口温度小于 4℃,则依次发送打开加热器的信号.该功能需求包含多重嵌套的判断逻辑,需求中的多个条件被转换为带有 guard 活动边的

DecisionNode,发送接收数据被转换为输入输出栓 pin.

问题 1.使用基础的分类指标来评估候选术语提取方法,分别为精确率(Precision)、召回率(Recall)与 F1-measure.当前任务是确定候选术语是否属于基准术语表.基准术语表先由 2 名研究者对所有案例的需求文本进行标注,然后由领域专家审查决定基准数据.属于基准术语表的候选术语为真正例(true positives, TP),不属于基准术语表的候选术语为假正例(false positives, FP),本文方法未提取出的基准术语为假负例(false negatives, FN). Precision, Recall 和 F1-measure 的计算方法为

$$Precision = TP / (TP + FP), \tag{2}$$

$$Recall = TP / (TP + FN), \tag{3}$$

$$F1\text{-measure} = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{4}$$

将本文提出方法与典型的通用候选术语提取方法 Pos-1^[25]、Pos-2^[26](如表 9 所示)、Dep^[27](仅使用基于依存句法规则提取候选术语)进行比较.

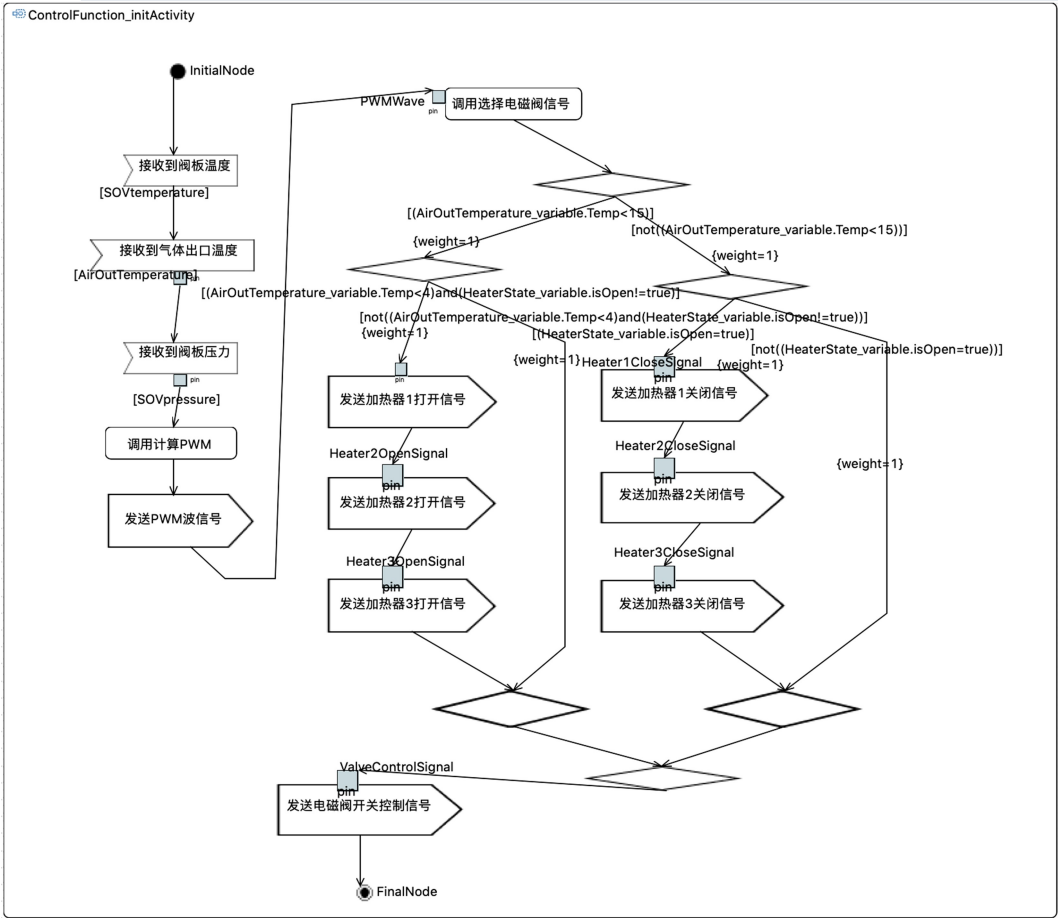


Fig. 16 Activity diagram of AACS control function

图 16 AACS 控制功能活动图

Table 9 Glossary Extraction Methods
表 9 术语提取方法

长度	Pos-1	Pos-2
2	n+n,n+v,v+n,a+n,b+n,d+n	n+n,v+n
3	n+n+n,v+n+n,n+v+n,v+v+n,b+v+n,n+m+n	n+n+n,n+b+n
4	n+n+n+n,n+n+v+n,v+n+n+n,v+n+v+n,n+v+v+n,v+v+n+n,v+n+b+n	b+m+n+n,b+n+n+n
5	v+v+n+n+n,d+v+n+n+n,m+v+m+n+n,b+v+n+v+n,n+n+v+n+n,a+n+v+n+n	n+n+n+vn+n,m+n+b+vn+n
6	n+n+c+vn+n+n,n+n+vn+c+vn+n,n+n+u+b+vn+n,vn+n+vn+c+vn+n,l+vn+k+n+vn+n,n+vn+u+n+vn+n	b+n+b+n+n+n,n+n+u+b+vn+n

本文的候选术语提取方法共产生 1 443 个候选术语.图 17 展示了术语提取方法的评估结果.实验结果表明:在保证精确率相对不变的情况下,本文方法的召回率最高,分别提高了 30.0%以及 22.9%;在精确率方面,本文方法相对于 Pos-1 以及 Pos-2 方法提高了 1.2%,0.2%,相对于 Dep 方法降低了 5.4%.

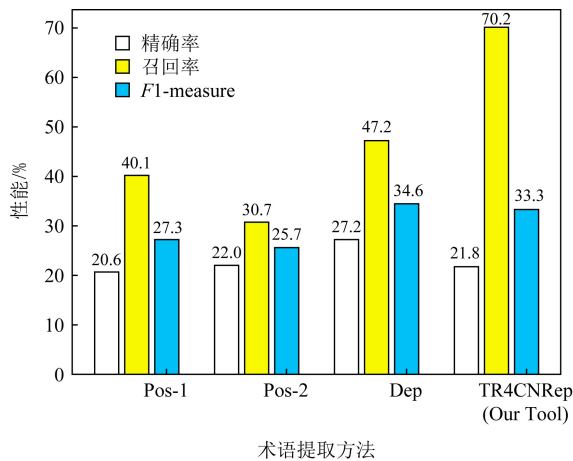


Fig. 17 Evaluation result of glossary extraction method
图 17 术语提取方法评估结果

对于安全关键信息物理融合系统中的需求术语提取工作,召回率是比精确率更为重要的指标,因为低召回率意味着遗漏了许多原本应当加入领域词库和数据字典中的术语,而低精确率仅仅意味着提取了许多不符合要求的假术语.根据领域专家的意见,工程师删去提取出的假术语的难度要远远小于再次检查需求文档、补充遗漏术语.我们的方法具有更好的召回率,因此具有更好的实践意义.

问题 2.表 10 为 AACS 系统的 SysML 模型数据统计.SysML 模型元素统计分为 3 个部分:1)描述静态功能架构的模型元素,涉及到 BDD 和 IBD 中的 Block 元素,以及没有在图上显示的自定义数据类型;2)描述系统动态行为的元素,涉及到 ACT 中的 ActivityNode 元素;3)描述非功能属性的元素,

涉及到 ConstraintRequirement 元素.由表 10 可知,案例生成的 SysML 模型对于系统需求的覆盖率较高.

Table 10 Statistics of Elements
表 10 模型元素统计信息

模型	元素	个数
Requirement	系统	1
	子系统	77
	输入/输出	52
	功能需求	80
	共享功能	20
	数据字典	92
	非功能需求	68
SysML	Block	78
	Port	52
	ActivityNode	136
	Data Type	100
	ConstraintRequirement	68

我们将 RNL2SysML 工具与现有的一些工具,如 CM-Builder^[28],DC-Builder^[29],UMLG^[30],aToucan,ABCD^[31]等进行对比,通过比较常用建模概念的覆盖程度以评估工具的有效性,如表 11 所示.从表 11 中可以看出:

- 1) 所有的工具都可以生成 UML 或 SysML 中类(Class/Block)的概念.对于组成关系,CM-Builder 和 UMLG 均不能表示.
- 2) 只有 ABCD 和 RNL2SysML 支持对功能需求的转换,而 ABCD 是通过在类图中对类之间的关系添加注解来表示行为,并不是通过活动图等行为视图来表示.
- 3) 由于其他工具均生成的是 UML 图,对于安全关键系统中常见的内部事件交互,只有 RNL2SysML 通过 IBD 图可以支持.

Table 11 Comparison with Existing Tools

表 11 RNL2SysML 与其他模型转换工具的功能对比

建模概念	CM-Builder	DC-Builder	UMLG	aToucan	ABCD	RNL2SysML
类	Y	Y	Y	Y	Y	Y
组成关系	N	Y	N	Y	Y	Y
值属性	Y	Y	N	Y	Y	Y
操作方法	N	N	Y	Y	Y	Y
多重性	Y	N	N	N	Y	Y
内部连接	N	N	N	N	N	Y
活动	N	N	N	Y	N	Y
非功能约束	N	N	N	N	N	Y
需求追踪关系	N	N	N	N	N	Y

注:“Y”表示覆盖建模概念;“N”表示不覆盖建模概念。

4) 本文所提工具通过 Profile 集成了对非功能属性的表示,以及给出从需求文本分配到图形元素的追踪关系。

问题 3.为了评估方法与工具的实用性,我们设计了人工参与的对照实验,参与对象为计算机学院不同年级的学生以及合作科研单位的系统工程师,分为系统工程师组和学生组,每组 2 人.实验步骤为:

1) 以飞机空气增压系统案例为训练案例,对所有参与者进行共计 2 h 的 RNL2SysML 原型工具的使用讲解培训.后续过程中,如果参与者遇到问题时,可以进行提问。

2) 给定 2 个工业案例的需求,要求参与者依次完成 4 项任务.任务 1,建立数据字典、领域词库;任务 2,填写 RNL 模板;任务 3,一键生成 SysML 模型;任务 4,手工建立与任务 3 中一致的 SysML 模型.组内通过分工加快进度,并交叉确认结果。

3) 如表 12 所示,对参与者的任务用时进行统计,并收集参与对象的反馈意见.其中“X”代表待定,表示本文方法结合 AI,其中的 RNL 模板填写目前仍需要人工参与。

实验结果表明:

1) 本文方法在术语推荐和建立 SysML 模型阶段的平均用时最短(<1 min),且远远低于全人工操作所需的时间(>5 h)。

2) 在所有案例中,系统工程师组的用时均小于学生组,反映出具备更多专业知识的参与者可以更快地掌握需求模板方法和原型工具的使用。

3) 根据参与者的反馈,手工完成初步设计模型对于所有参与者均有较大难度,反映出从需求到设

计之间的鸿沟.由于对 SysML 语言的理解程度不同,多个参与者手工建立的模型往往也不一致,采用本文方法可以降低不一致性,还具备学习成本低、实用性强的特点,对于实际案例具有积极意义。

Table 12 Execution Time of Case Studies

表 12 案例执行时间统计

案例	参与对象	阶段	执行时间
案例 A (姿态轨道控制系统, 420 条需求, 785 条术语, 58 个子模板)	系统工程师 SE_1, SE_2	术语推荐	2 h
		RNL 模板填写	1.5 h
		建立 SysML 模型	6 h
		总计	9.5 h
	学生 ST_1, ST_2	术语推荐	2.5 h
		RNL 模板填写	2 h
		建立 SysML 模型	11 h
		总计	15.5 h
	本文方法	术语推荐	38 s
		RNL 模板填写	X
		建立 SysML 模型	22 s
		总计	X+50 s
案例 B (雷达系统, 120 条需求, 334 条术语, 25 个子模板)	系统工程师 SE_1, SE_2	术语推荐	0.75 h
		RNL 模板填写	0.67 h
		建立 SysML 模型	3 h
		总计	4.42 h
	学生 ST_1, ST_2	术语推荐	0.83 h
		RNL 模板填写	0.75 h
		建立 SysML 模型	5 h
		总计	6.58 h
	本文方法	术语推荐	22 s
		RNL 模板填写	X
		建立 SysML 模型	20 s
		总计	X+42 s

注:“X”代表待定。

5.3 适用范围与局限性分析

首先,限定自然语言需求规约方法比较适用于CPS系统的需求捕获和概要设计阶段.例如,可以直接采用RNL需求模板记录需求,或者从需求文档中获得RNL需求.

其次,基于自然语言处理的术语推荐方法比较适用于需求术语分散于需求文档各处的情况.例如,针对软件需求以需求文档、会议记录和谈话记录等文本方式存在时,可以使用本文方法推荐出候选术语,然后通过工程师筛选得到领域术语和数据字典.

最后,基于自然语言处理的术语推荐方法的实验评估阶段,主要针对基于语言学的最新方法进行实验对比.由于本文主要针对特定领域,存在语料库有限的不足^[32],因此不具备使用机器学习或者深度学习方法^[33-34]的条件.

6 相关工作

本文的相关工作主要包括3部分:需求工程中的人工智能(AI)方法应用、限定自然语言需求建模、设计模型的转换与生成.

6.1 需求工程中的AI方法应用

目前,AI技术主要应用在术语提取、需求优先级排序、需求分类、领域模型抽取等任务中.

Berry等人^[35]提出了基于频率的方法来识别在需求中反复出现的术语;Matsuo与Ishizuka^[36]使用单词和单词序列的共现频率来识别关键术语.

Dwarakanath等人^[37]使用句法解析来提取需求文档的短语,然后基于启发式方法和基于频率的统计信息过滤结果.

Perini等人^[38]提出一种需求优先级排序方法(case-based ranking, CBRank),该方法将项目参与者的偏好与通过机器学习算法计算出的需求排序近似值相结合,效果较好.

Winkler等人^[39]提出基于卷积神经网络的自动需求分类方法,将需求文档中的“需求”和“补充信息”进行区分.

Arora等人^[40]提出了一种基于主动学习的方法,不断从用户反馈中学习相关度知识,过滤NLP提取结果中的多余信息,从而提高自动提取领域模型的准确度.

此外,刘璘等人^[41]提出了自动化的非功能需求识别和分类方法,基于语义距离和相似度计算,将非

功能需求语句分为性能、可靠性、可用性、安全性、可维护性这5类.

上述工作主要应用于英文需求文本,由于中英文之间存在较大差异性,这些工作无法直接应用于中文需求文本.目前,面向中文需求文本的自动化分析工作较少.

6.2 限定自然语言需求建模

RollsRoyce公司的Mavin等人^[42]针对用户需求难以描述和表达的问题,在大量工程经验的基础上,提出了一种EARS(easy approach to requirements syntax)方法进行需求规约.该方法通过将功能需求分为6种类型,有效简化了需求的描述与表达难度,同时有效降低自然语言的二义性.类似的有Rupp等人提出的Rupp模板^[43]将功能需求分为3类.但以上模板只对功能需求进行了分类和描述,没有全面地包含嵌入式系统的静态结构、动态行为和非功能属性.

Holtmann等人^[44]针对汽车工程领域的软件系统的需求规约展开研究,提出了一种受控自然语言(controlled natural language, CNL)用于汽车领域嵌入式软件需求描述,CNL还可以实现需求的自动化验证,以及对需求的不一致性和不完整性进行检测.

岳涛等人^[17-18]提出了一种改进的受限用例建模方法(restricted use case model, RUCM),RUCM包含一个相对完善的用例文本说明模板和系列自然语言限定规则(restricted rules),使得用例描述更加易于理解和精确,减少歧义,并且可以自动化产生分析模型.目前,已有相应的研究方法和工具支持将RUCM的用例模型自动转换为初步的UML分析模型(类图、活动图、顺序图等).

针对RUCM较难描述机载嵌入式软件的安全性需求的不足,吴际等人^[45]对RUCM进行了扩展,添加相应的模板与限定规则,提出SafetyRUCM,使其支持安全性需求的规范化描述,但主要是从用例的角度进行需求描述.

王政等人^[20]面向航天型号软件,提出了航天需求描述语言SPARDL,能够清晰、无二义地描述航天软件需求.SPARDL包括数据字典、模块和模式图3个部分,其中核心为模式图.经实践证明,SPARDL能在航天型号研究过程中有效地提高航天嵌入式软件的质量,但SPARDL采用形式化需求描述方法,实践难度较大,且针对航天信号软件设计,通用性不足.

另外,面向对象管理组织 OMG 在正在制定的 SysML2.0^[21]版本中也提出了限定自然语言需求建模的思想,旨在提高需求描述的精确性和有效性。

6.3 需求表达达到设计模型的转换与生成

已有的需求表达达到系统设计模型的转换主要包括基于自由文本的方式和基于限定自然语言模板的方式。

1) 基于自由文本

Bajwa 等人^[15]提出从自然语言到 OCL 约束的自动转换方法。

Letsholo 等人^[16]提出 TRAM 方法将自由文本需求转换到 UML 类图。

Abdessalem 等人^[31]提出 ABCD (automatic builder of class diagram)方法,定义了一千多种模式并通过 NLP 技术从自由文本中抽取概念,并自动转换到 UML 类图。

刘璘等人^[46-47]提出基于依存文法的需求文本策略依赖关系抽取方法,构建出 i^* 框架中的 SD 模型。

2) 基于限定文本(restricted text)

岳涛等人^[17-18]针对传统 UML 用例规约描述存在的不足提出了基于用例建模方法 RUCM,并使用 aToucan 工具将 RUCM 的用例模型自动转换为初步的 UML 分析模型(类图、活动图、顺序图等)。

何啸等人^[48]提出了基于模板的生成方法,包括中间元模型、从模板到中间元模型的 folding 算法、从中间元模型到目标模型的 unfolding 算法,该方法提供了不针对特定语言的、通用的模型生成思路。

Ballard 等人^[49]基于包含属性、组成关系、状态描述、状态迁移等的简单需求模板转换到 SysML 块定义图和状态机图。

杨志斌等人^[23-24]给出了基于限定中文自然语言需求的 AADL 模型自动生成方法。

Colombo 等人^[22]提出基于问题框架(problem frames)的需求分析并将需求规约转换到 SysML 的方法,分别抽取 Blackboard 和 Knowledge Source 信息并不断精化,将问题框架转换到 SysML 模块定义图、活动图等,生成早期设计模型,该研究主要考虑基于问题分解的方式获得系统初始层次架构,还未考虑系统非功能特性。

相比于上述工作,本文主要针对中文自然语言需求;其次,在限定自然语言需求建模中,我们引入了基于 AI 的需求术语推荐方法;最后,本文考虑了安全关键 CPS 系统的非功能需求表达,即对 SysML 的安全性、可靠性扩展。

7 结束语

如何将自然语言需求和模型驱动开发方法进行有效链接,即实现自然语言需求到设计模型的自动或半自动转换是模型驱动开发方法在安全关键信息物理融合系统设计与开发中实际应用的一个主要挑战,本文面向安全关键信息物理融合系统,结合自然语言处理技术对需求文本进行术语抽取,通过用户的简单编辑后将术语加入限定自然语言需求模板的需求规约方法中,构建安全关键信息物理融合系统需求模型,然后自动转换到面向多视图的 SysML 模型。

未来的研究工作将包括:1)检测原始文本需求与限定自然语言需求模板之间的一致性,并根据需求文本推荐相一致的需求模板;2)对生成 SysML 模型进行形式化验证;3)探索将用户反馈作为方法的二次输入,优化需求推荐的准确度;4)增加工业界案例数量,对本文方法进行更多的实验评估;5)完善自动术语抽取的基础评价体系;6)结合基于统计与基于规则的方法,收集更多领域语料以训练模型。

参 考 文 献

- [1] Leveson N. Engineering a Safer World: Systems Thinking Applied to Safety [M]. Cambridge, MA: MIT Press, 2016: 7-15
- [2] Rierson L. Developing Safety-critical Software: A practical Guide for Aviation Software and DO-178C Compliance [M]. Boca Raton, FL: CRC, 2017: 200-220
- [3] Rawat D, Rodrigues J, Stojmenovic I. Cyber-Physical Systems from Theory to Practice [M]. Boca Raton, FL: CRC, 2015: 110-115
- [4] France R, Rumpe B. Model-driven development of complex software: A research roadmap [C] //Proc of the 29th Int Conf on Software Engineering, Future of Software Engineering. Los Alamitos, CA: IEEE Computer Society, 2017: 37-54
- [5] OMG. UML Specification, Version 2.5.1 [OL]. [2017-12-05]. <http://www.omg.org/spec/UML/>
- [6] SAE. Architecture Analysis & Design Language (standard SAE AS5506)[OL]. [2017-01-01]. <http://www.sae.org>
- [7] Feiler P, Gluch D. Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language [M]. Reading, MA: Addison-Wesley, 2012
- [8] Yang Zhibin, Pi Lei, Hu Kai, et al. AADL: An architecture design and analysis language for complex embedded real-time systems [J]. Journal of Software, 2010, 21(5): 899-915 (in Chinese)

- (杨志斌, 皮磊, 胡凯, 等. 复杂嵌入式实时系统体系结构设计与分析 语言: AADL [J]. 软件学报, 2010, 21(5): 899-915)
- [9] OMG. SysML, Version 1.4 [OL]. [2017-05-08]. <http://www.omg.sysml.org>
- [10] Vilela J, Castro J, Martins L, et al. Integration between requirements engineering and safety analysis: A systematic literature review [J]. Journal of Systems and Software, 2017, 125(C): 68-92
- [11] Yue Tao, Briand L, Labiche Y. A systematic review of transformation approaches between user requirements and analysis models [J]. Requirements Engineering, 2011, 16(2): 75-99
- [12] Martins L, Gorschek T. Requirements engineering for safety-critical systems: A systematic literature review [J]. Information and Software Technology, 2016, 75: 71-89
- [13] Martins L, Gorschek T. Requirements engineering for safety-critical systems: Overview and challenges [J]. IEEE Software, 2017, 34(4): 49-57
- [14] Zhang Gong, Yue Tao, Wu Ji, et al. Zen-RUCM: A tool for supporting a comprehensive and extensible use case modeling framework.[J]. CEUR Workshop Proceedings, 2013, 1115: 41-45
- [15] Bajwa I, Lee M, Bordbar B. Translating natural language constraints to OCL [J]. Journal of King Saud University-Computer and Information Sciences, 2012, 24(2): 117-128
- [16] Letsholo K, Zhao Liping, Chioasca E V. TRAM: A tool for transforming textual requirements into analysis models [C] // Proc of the 28th IEEE/ACM Int Conf on Automated Software Engineering(ASE). Berlin: Springer, 2013: 738-741
- [17] Yue Tao, Briand L, Labiche Y. Facilitating the transition from use case models to analysis models: Approach and experiments [J]. ACM Transactions on Software Engineering and Methodology, 2013, 22(1): 1-38
- [18] Yue Tao, Briand L, Labiche Y. aToucan: An automated framework to derive UML analysis models from use case models [J]. ACM Transactions on Software Engineering and Methodology, 2015, 24(3): 1-52
- [19] Gregory S. Easy EARS: Rapid application of the easy approach to requirements syntax [C] //Proc of the 19th IEEE Int Requirements Engineering Conf(RE 2011). Piscataway, NJ: IEEE, 2011: No.10
- [20] Wang Zheng, Li Jianwen, Zhao Yongxin, et al. SPARDL: A requirement modeling language for periodic control system [C] //Proc of Int Symp on Leveraging Applications of Formal Methods, Verification and Validation. Berlin: Springer, 2010: 594-608
- [21] OMG. Systems Modeling Language (SysML). Version 2.0. RFP [OL]. [2018-06-03]. <https://www.omg.org/cgi-bin/doc? ad/18-06-03.pdf>
- [22] Colombo P, Khendek F, Lavazza L. Bridging the gap between requirements and design: An approach based on problem frames and SysML [J]. Journal of Systems and Software, 2012, 85(3): 717-745
- [23] Wang Fei, Yang Zhibin, Huang Zhiqiu, et al. Generating the AADL model based on restricted natural language requirement template [J]. Journal of Software, 2018, 29(8): 2350-2370 (in Chinese)
(王飞, 杨志斌, 黄志球, 等. 基于限定自然语言需求模板的AADL模型生成方法 [J]. 软件学报, 2018, 29(8): 2350-2370)
- [24] Wang Fei, Yang Zhibin, Huang Zhiqiu, et al. An approach to generate the traceability between restricted natural language requirements and AADL models [J]. IEEE Transactions on Reliability, 2020, 69(1): 154-173
- [25] Han Hongqi, Zhu Donghua, Wang Xuefeng. Patent technology term extraction method [J]. Journal of Information, 2011, 30(12): 1280-1285 (in Chinese)
(韩红旗, 朱东华, 汪雪峰. 专利技术术语的抽取方法 [J]. 情报学报, 2011, 30(12): 1280-1285)
- [26] Zeng Zhen, Lü Xueqiang, Li Zhuo, et al. A patent abstract-oriented domain term extraction method [J]. Computer Application and Software, 2016, 33(3): 48-51 (in Chinese)
(曾镇, 吕学强, 李卓, 等. 一种面向专利摘要的领域术语抽取方法 [J]. 计算机应用与软件, 2016, 33(3): 48-51)
- [27] Yu Yan, Chen Lei, Jiang Jinde, et al. Research on the selection of Chinese patent candidate term based on dependency syntax parsing [J]. Library and Information Service, 2019, 63(18): 109-118 (in Chinese)
(俞琰, 陈磊, 姜金德, 等. 基于依存句法分析的中文专利候选术语选取研究 [J]. 图书情报工作, 2019, 63(18): 109-118)
- [28] Harmain H M, Gaizauskas R. CMbuilder: A natural language-based case tool for object-oriented analysis [J]. Automated Software Engineering, 2003, 10: 157-181
- [29] Herchi H, Abdessalem W. From user requirements to UML class diagram [J]. arXiv preprint, arXiv:1211.0713, 2012
- [30] Bajwa I, Samad A, Mumtaz S. Object oriented software modeling using NLP based knowledge extraction [J]. European Journal of Scientific Research, 2009, 35(1): 22-33
- [31] Abdessalem W, Azzouz Z, Singh A, et al. Automatic builder of class diagram(ABCD): An application of UML generation from functional requirements [J]. Software Practice and Experience, 2016, 46(11): 1443-1458
- [32] Zong Chengqing. Statistical Natural Language Processing [M]. Beijing: Tsinghua University Press, 2008 (in Chinese)
(宗成庆. 统计自然语言处理 [M]. 北京: 清华大学出版社, 2008)
- [33] Zhang Xue, Sun Hongyu, Xin Dongxing, et al. Survey on automatic term extraction research [J]. Journal of Software, 2020, 31(7): 2062-2094 (in Chinese)

(张雪, 孙宏宇, 辛东兴, 等. 自动术语抽取研究综述[J]. 软件学报, 2020, 31(7): 2062-2094)

[34] Astrakhantsev N. ATR4S: Toolkit with state-of-the-art automatic terms recognition methods in Scala [J]. Language Resources and Evaluation, 2018, 52(3): 853-872

[35] Aguile C, Berry D. The use of a repeated phrase finder in requirements extraction [J]. Journal of Systems and Software, 1990, 13(3): 209-230

[36] Matsuo Y, Ishizuka M. Keyword extraction from a single document using word co-occurrence statistical information [J]. International Journal on Artificial Intelligence Tools, 2004, 13(1): 157-169

[37] Dwarakanath A, Ramnani R, Sengupta S. Automatic extraction of glossary terms from natural language requirements [C] // Proc of the 21st IEEE Int Requirements Engineering Conf (RE 2013). Piscataway, NJ: IEEE, 2013: 314-319

[38] Perini A, Susi A, Avesani P. A machine learning approach to software requirements prioritization [J]. IEEE Transactions on Software Engineering, 2013, 39(4): 445-461

[39] Winkler J, Vogelsang A. Automatic classification of requirements based on convolutional neural networks [C] //Proc of the 24th IEEE Int Requirements Engineering Conf (RE 2016). Piscataway, NJ: IEEE, 2016: 39-45

[40] Arora C, Sabetzadeh M, Nejati S, et al. An active learning approach for improving the accuracy of automated domain model extraction [J]. ACM Transactions on Software Engineering Methodology, 2019, 28(1): 4:1-4:34

[41] Jia Yidi, Liu Lin. Recognition and classification of non-functional requirements in Chinese [J]. Journal of Software, 2019, 30(10): 3115-3126 (in Chinese)
(贾一获, 刘璘. 中文非功能需求描述的识别与分类方法研究 [J]. 软件学报, 2019, 30(10): 3115-3126)

[42] Mavin A, Wilkinson P, Harwood A, et al. Easy approach to requirements syntax(EARS)[C] // Proc of the 17th IEEE Int Requirements Engineering Conf. Piscataway, NJ: IEEE, 2009: 317-322

[43] Pohl K, Rupp C. Requirements Engineering Fundamentals [M]. San Rafael, CA: Rocky Nook, 2011

[44] Holtmann J, Meyer J, von Detten M. Automatic validation and correction of formalized, textual requirements [C] //Proc of the 4th Int Conf on Software Testing, Verification and Validation Workshops. Piscataway, NJ: IEEE, 2011: 486-495

[45] Wu Xue, Liu Chao, Wu Ji. Safety requirements description method based on RUCM [J]. Computer Science, 2015, 42(12): 65-70 (in Chinese)
(吴雪, 刘超, 吴际. 基于 RUCM 的软件安全性需求描述方法[J]. 计算机科学, 2015, 42(12): 65-70)

[46] Liu Lin, Li Tianying, Kou Xiaoxi. Eliciting relations from natural language requirements documents based on linguistic and statistical analysis [C] //Proc of the 38th Annual Computer

Software and Applications Conf. Los Alamitos, CA: IEEE Computer Society, 2014: 191-200

[47] Li Tianying, Liu Lin, Zhao Dewang, et al. Eliciting relations from requirements text based on dependency analysis [J]. Chinese Journal of Computers, 2013, 36(1): 54-62 (in Chinese)
(李天颖, 刘璘, 赵德旺, 等. 一种基于依存文法的需求文本策略依赖关系抽取方法[J]. 计算机学报, 2013, 36(1): 54-62)

[48] He Xiao, Zhang Tian, Pan Minxue, et al. Template-based model generation [J]. Software and Systems Modeling, 2019, 18(3): 2051-2092

[49] Ballard M, Peak R, Cimtalay R, et al. Bidirectional text-to-model element requirement transformation [C] //Proc of 2020 IEEE Aerospace Conf. Piscataway, NJ: IEEE, 2020: 1-14



Bao Yang, born in 1992. Master. Student member of CCF. His main research interests include requirements engineering, safety-critical embedded system, model-driven development, and intelligent software engineering.

鲍 阳, 1992 年生. 硕士, CCF 学生会员. 主要研究方向为需求工程、安全关键嵌入式系统、模型驱动开发、智能软件工程。



Yang Zhibin, born in 1982. PhD, associate professor. Member of CCF. His main research interests include safety-critical embedded system, formal verification, AADL, and synchronous languages.

杨志斌, 1982 年生. 博士, 副教授, CCF 会员. 主要研究方向为安全关键嵌入式系统、形式化验证、AADL、同步语言。



Yang Yongqiang, born in 1998. Master. Student member of CCF. His main research interests include intelligent software engineering.

杨永强, 1998 年生. 硕士, CCF 学生会员. 主要研究方向为智能软件工程。



Xie Jian, born in 1988. Master. Member of CCF. His main research interests include formal methods, system safety analysis, service computing.

谢 健, 1988 年生. 硕士, CCF 会员. 主要研究方向为形式化方法、系统安全性分析、服务计算。



Zhou Yong, born in 1975. PhD, associate professor. Member of CCF. His main research interests include formal methods and software engineering.
周 勇, 1975 年生. 博士, 副教授, CCF 会员. 主要研究方向为形式化方法、软件工程.



Yue Tao, born in 1974. PhD, professor, PhD supervisor. Her main research interests include software engineering, with specific interests in MBE, uncertainty modeling, search-based software engineering, empirical software engineering.
岳 涛, 1974 年生. 博士, 教授, 博士生导师. 主要研究方向为软件工程, 尤其对模型驱动工程、不确定性建模、基于搜索的软件工程、经验软件工程等.



Huang Zhiqiu, born in 1965. PhD, professor, PhD supervisor. Member of CCF. His main research interests include software safety, formal methods, requirement engineering and software engineering.
黄志球, 1965 年生. 博士, 教授, 博士生导师, CCF 会员. 主要研究方向为软件安全、形式化方法、需求工程和软件工程.



Guo Peng, born in 1987. PhD candidate. Senior engineer. Member of CCF. His main research interests include modeling and verification of embedded system.
郭 鹏, 1987 年生. 博士研究生, 高级工程师, CCF 会员. 主要研究方向嵌入式系统建模与仿真.