软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2015,26(10):2545−2566 [doi: 10.13328/j.cnki.jos.004813]

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

可信软件非功能需求形式化表示与可满足分析*

张 璇[1,2], 李 彤[1,2], 王 旭[3], 于 倩[1,2], 郁 湧[1,2], 朱 锐[1]

[1](云南大学 软件学院,云南 昆明 650091)

[2](云南省软件工程重点实验室,云南 昆明 650091)

[3](云南大学 经济学院,云南 昆明 650091)

通讯作者: 张璇, E-mail: zhxuan@ynu.edu.cn, http://www.ynu.edu.cn

摘 要: 可信软件的可信性由其功能需求和非功能需求共同来体现,其中,非功能需求的实现是可信软件获得用户对其行为实现预期目标能力的信任程度的客观依据.针对可信软件的重要性以及对可信软件的迫切需求,在可信软件的早期需求工程阶段,提出可信软件非功能需求驱动的过程策略选取方法.首先,对可信软件需求进行定义,提出由功能需求和非功能需求中的可信关注点构成可信需求,非可信关注点的非功能需求则定义为软目标,用于表达质量需求,基于模糊集合论和信息熵对可信软件非功能需求进行排序并获取可信关注点和软目标.在此基础上,提出可信软件非功能需求驱动的过程策略选取方法.传统的软件早期需求工程阶段的目标是为了获取满足需求的技术及设计决策,与此不同,本文对可信软件非功能需求进行分析的目标是获取过程策略,从过程角度解决可信软件生产问题.由于非功能需求间复杂的相关关系,尤其是因为存在冲突关系,故提出了基于可满足性问题求解方法推理过程策略的方法,选取满足可信软件非功能需求的过程策略.最后,通过第三方可信认证中心软件的案例,说明所提出方法的可行性.

关键词: 可信软件;早期需求工程;非功能需求;软件过程;可满足性问题

中图法分类号: TP311

中文引用格式: 张璇,李彤,王旭,于倩,郁湧,朱锐.可信软件非功能需求形式化表示与可满足分析.软件学报,2015,26(10):2545−2566. http://www.jos.org.cn/1000-9825/4813.htm

英文引用格式: Zhang X, Li T, Wang X, Yu Q, Yu Y, Zhu R. Formal analysis to non-functional requirements of trustworthy software. Ruan Jian Xue Bao/Journal of Software, 2015,26(10):2545−2566 (in Chinese). http://www.jos.org.cn/1000-9825/4813.htm

Formal Analysis to Non-Functional Requirements of Trustworthy Software

ZHANG Xuan[1,2], LI Tong[1,2], WANG Xu[3], YU Qian[1,2], YU Yong[1,2], ZHU Rui[1]

[1](School of Software, Yunnan University, Kunming 650091, China)

[2](Key Laboratory of Software Engineering of Yunnan, Kunming 650091, China)

[3](School of Economics, Yunnan University, Kunming 650091, China)

Abstract: The trustworthiness of software is determined by both its functional requirements and non-functional requirements. Especially, the non-functional requirements are the determinants of the trustworthy software that show how it achieves the users' desired goals. Considering the importance of trustworthy software and the urgent needs for it, an approach to obtaining process strategies for trustworthy software in the early phase of requirements engineering is proposed. Firstly, the definition of trustworthy software requirements is defined as the combination of the trustworthiness requirements and the quality requirements. Trustworthiness requirements are defined as both

* 基金项目: 国家自然科学基金(61262025, 61502413, 61379032, 61262024); 云南省自然科学基金(2012FB118, 2012FB119); 云南省教育厅科学研究基金(2015Z020); 云南省软件工程重点实验室开放基金(2015SE202, 2012SE308); 云南大学"中青年骨干教师培养计划"专项经费; 云南大学高水平创新团队"软件工程创新团队"专项经费

收稿时间: 2014-02-14; 修改时间: 2014-09-29, 2014-11-24; 定稿时间: 2015-01-07

functional requirements and trustworthiness concerns. Quality requirements are defined as soft goals. Then, based on fuzzy set theory and information entropy, acquisition method of trustworthiness concerns and soft goals is proposed. On this basis, process strategies for obtaining framework are proposed. Unlike the traditional early-phase requirements engineering which focuses on technical and design decisions, the aim of this study is to make process decisions to support trustworthy software development. In addition, to address the conflict relationships of the non-functional requirements, a reasoning method is developed for solving satisfiability problems of non-functional requirements in trustworthy software. Finally, through analyzing a trustworthy third-party certificate authority software case, feasibility of the proposed approach is described.

Key words: trustworthy software; early-phase requirements engineering; non-functional requirements; software process; satisfiability problem

软件的“可信”是指软件系统的行为及其结果总是符合人们的预期,在受到干扰时仍能提供连续的服务,这里的“可信”强调行为和结果的可预测性和可控制性[1].从需求角度而言,可信软件的行为和结果符合人们预期可以通过可信软件功能需求和非功能需求的满足来客观反映,可以理解为功能和非功能相关行为和结果符合人们预期,其中,功能需求的实现是软件的硬性目标,而非功能需求的实现则是软件获得用户对其行为实现预期目标能力的信任程度的客观依据.用户通过软件所具有一组表达其可信属性的客观能力这一事实,从而信任软件的行为能够实现其设定的目标[2].这也符合 TSM(trusted software methodology)有关软件可信性的定义,即“软件满足一系列需求的信任程度”[3].因此,可信软件的功能需求严格、非功能需求复杂是可信软件不同于普通软件的根本原因,即:可信软件的实现必须既要严格完成功能需求,又要满足其特定的非功能需求.

然而,针对可信软件如何在早期需求工程阶段确定软件利益相关者的非功能需求以及如何找出满足这些需求的解决策略等问题,我们面临着一些固有的难点:首先,软件利益相关者人数众多,依赖于其所处环境和角色的不同,利益相关者对可信软件的需求存在着差异.同时,需求中的非功能需求又常常难以精确表达,这加大了我们通过利益相关者获取可信软件需求的难度;其次,可信软件需求中的非功能需求之间必然存在着相关关系(促进关系或者抑制关系),这意味着,当我们努力通过改善技术或者过程控制等手段实现某一非功能需求时,另外的非功能需求却有可能因此而受到负面的影响,从而导致生产出来的软件的总体可信性降低.

为解决这些难点,本文首先在第 2 节分析了可信软件与普通软件在需求上的不同,提出可信软件的可信需求概念及其分解模型.在此基础上,提出可信软件非功能需求驱动的过程策略选取框架 TRPF(trustworthiness requirements-processes framework).TRPF 由 3 个流程步骤构成,分别在本文的第 3.1 节~第 3.3 节中加以阐述.首先,针对可信软件非功能需求难以精确表达的特点,在第 3.1 节提出基于梯形模糊数描述可信软件非功能需求,并向软件利益相关者收集他们评估可信软件非功能需求重要程度的评估数据(以下简称为评估数据),然后使用信息熵检验数据的有效性,在获得有效评估数据的基础上通过模糊排序确定可信软件非功能需求中的可信关注点及软目标,并根据评估数据的排序值设定可信关注点和软目标所需要满足的状态.基于获取的可信关注点及软目标,本文在第 3.2 节定义了可信软件非功能需求与过程策略关系的元模型 TRMM(trustworthiness requirements meta-model),并提出基于知识库的非功能需求与过程策略关系建模方法.由于非功能需求之间复杂的相关关系,满足某可信软件非功能需求的过程策略有可能对其他非功能需求有抑制作用(也称为冲突关系),因此在第 3.3 节,我们根据第 3.1 节得到的可信关注点和软目标的满足状态对过程策略的选取进行可满足性推理,如果推理结果是满足,则找到了满足可信软件非功能需求的过程策略;如果推理结果是不满足,则由建模者决定导致不满足的矛盾的解决方法,解决后继续推理,直至获得满足可信软件非功能需求的过程策略.

1 相关工作

根据软件需求的分类,软件需求包括功能需求和非功能需求.目前,大部分比较成熟的需求工程方法都侧重于对功能需求进行分析,而对非功能需求的处理缺乏特别合适的解决策略,这主要是因为对非功能需求进行分析非常困难,涉及面广,而且通常需要采用领域特有的方式[4].

在软件非功能需求相关关系分析方面,Boehm 和 In 对非功能需求间冲突的识别和诊断进行了研究,并设计

实现了一个辅助非功能需求分析、识别、诊断冲突的工具 QARCC(quality attribute risk and conflict consultant)[5].陈仪香教授指导的陶红伟博士在其博士毕业论文中基于 McCall 和 Deutsch 给出的属性间关系模型,从集合论的角度给出影响软件可信性的关键属性(功能性、可维护性、可靠性和可生存性)间的量化关系[6].Moser 及其同事基于语义技术实现需求任务的自动管理,包括需求分类、需求冲突分析和需求追踪[7].除此以外,需求交互管理 RIM(requirement interaction management)也研究需求间的相关性.Mairiza 和 Zowghi 基于软件应用场景研究非功能需求之间的冲突问题,给出包含精确性等 20 个非功能需求之间的冲突关系[8].

基于非功能需求间相关关系分析的结果,我们还需要给出权衡后的需求解决方案,在该领域很多学者提出了不同的模型和技术.其中,早期由 Boehm 提出的基于共赢(win-win)需求协商来寻求需求冲突解决的办法得到业界的广泛接受[9−11].Win-Win 方法的关键出发点在于通过流程化步骤,系统地组织软件项目利益有关各方(stakeholders),根据项目具体情况来协调他们之间的需求.基于 Win-Win 方法,Robinson 和 Volkov 提出面向冲突需求的重构[12],In 及其同事提出基于多标准偏好分析的需求协商[13,14].另外,基于非功能需求间相关关系对非功能需求进行分析的方法主要以面向目标方法为主.面向目标需求工程(goal-oriented requirements engineering)方法认为:需求阶段的主要任务是要确定软件需求相关者想要实现的各项目标,目标表达了期望软件所体现出的行为以及要满足的约束.目标按照所描述的具体内容可分为功能性目标和非功能性目标.以目标为需求获取的基本线索,面向目标需求工程方法通过将目标组织成与或树的结构来表示目标的分解、精化和抽象关系,构成目标模型的主框架.按目标描述的抽象层次,可以分为高层目标和低层目标,高层目标通常是粗粒度、策略性地作用于组织范围的抽象目标,低层目标通常是细粒度、技术性地作用于软件设计层面的具体目标[4].KAOS(knowledge acquisition in automated specification)[15,16]、NFR 框架[17−20]和 *i**家族(包括 *i**模型、Tropos 和 GRL)[21−23]是最具代表性的面向目标需求工程方法.

在可信软件非功能需求获取及权衡方面,Zhu 等人提出了用模糊数学和软目标依赖图 SIG(softgoal interdependency graph)构建 FQQSIG(fuzzy qualitative and quantitative software interdependency graph)模型,进行非功能需求的相关性分析以提供权衡决策[24].

在非功能需求推理方面,KAOS、NFR 框架和 *i**家族的方法都提出了相应的非功能需求推理方法.另外,Jin 等人提出了Σ形式化描述语言用于描述目标树模型,以方便设计方案选择的自动向前推理[25].Sebastiani 等人于 2004 年首次提出用 SAT 求解方法实现目标模型的向后推理[26].2005 年,Giorgini 等人将可满足性问题求解方法应用于 Tropos 中,实现了向前及向后推理[27].2010 年,Horkoff 和 Yu 使用合取范式 CNF(conjunctive normal form)定义 *i**框架的标记传播规则,提出主体-目标需求模型的向后推理方法[28,29].

2 可信软件需求

基于面向目标需求工程的思想,以目标为需求获取的基本线索,本文将可信软件的需求定义为软件利益相关者需要可信软件具备的可信状态或条件,隐含表达了期望可信软件所体现出的行为以及要满足的约束.根据可信软件具体内容的不同,将其需求分为功能需求和非功能需求,其中,将功能需求表达为可信软件的硬目标,而将非功能需求分解表达为可信关注点和软目标.可信关注点(这里的关注点使用了面向方面方法中关注点分离的思想)是可信软件获得用户对其行为实现预期目标能力的信任程度的客观依据,根据可信软件的不同,可信关注点由不同的非功能需求集合构成,与可信关注点产生相关关系的其他非功能需求描述为软目标,软目标不是可信软件的可信依据,但对可信软件的质量有一定的影响,图 1 描述了可信软件需求的构成.本文将可信软件硬目标与可信关注点的集合定义为可信软件的可信需求,将软目标定义为可信软件的质量需求.

- 硬目标是可信软件的功能需求,可信软件的所有功能需求都必须严格实现,这是可信软件的基础,因而用硬目标描述是可信软件实现的硬性目标.由于功能需求一直以来都受到足够的重视,相关研究相对充分,因此,本文仅关注功能需求实现的完整性及正确性,为了与传统功能需求概念区分开,我们将功能完整性及正确性归入非功能需求;
- 可信关注点是可信软件非功能需求集合的一个子集,在这个子集中的非功能需求是由利益相关者共

同决定的攸关软件可信性的非功能需求,用户通过软件满足这些可信关注点的客观能力事实,从而信任软件的行为能够实现其设定的目标.因此,软件的可信意味着软件满足一系列可信关注点;

- 软目标是可信软件非功能需求中非可信关注点的非功能需求集合,用于描述软件的质量需求(这里借鉴了面向目标需求工程方法用软目标(softgoal)表达软件的质量需求).由于软件非功能需求间存在复杂的相关关系,可信关注点的实现会影响软目标.如果仅考虑可信关注点的实现,有可能损害软目标,从而导致软件质量的下降,质量低下的软件绝对不可能是可信软件.然而,需要注意的是,高质量软件又不等同于可信软件,软件质量定义为软件满足顾客、用户需求或期望的程度,软件质量的定义并不明确,大致涵盖可靠性、安全性、可维护性、性能和易用性等几个要素,这些要素虽然与可信软件基本要素一致,但可信软件明确强调获得用户信任的能力,其可信需求是明确定制的.因此,本文用软目标描述可信软件质量需求,而用可信关注点明确描述用户的可信非功能需求.

基于上述对可信软件需求的相关分析,由可信关注点和软目标构成的可信软件非功能需求是本文研究的核心.然而至今,可信软件的非功能需求仍然没有一个公认的标准,甚至没有一个完整的列表.因此,本文基于相关文献资料,总结自 1985 年可信计算系统评估标准 TCSEC(trusted computer system evaluation criteria)产生以来相关学者和机构提出的可信软件非功能需求[3,30−42],结合 Trustie 软件可信分级规范 Software Trustworthiness Classification Specification(STC 1.0)[43]中给出的软件可信属性模型及 ISO/IEC 25010 定义的软件质量需求[44],提出如图 2 所示的可信软件非功能需求分解模型.可信软件非功能需求分解模型并非一成不变,随着时代的发展、技术的进步以及不同项目需要和不同专家的建议,分解模型应根据可信软件的具体需要动态地加以调整.

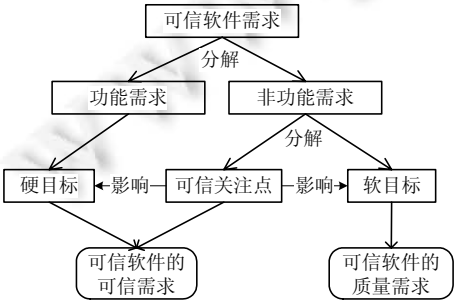


Fig.1 Goal-Oriented trustworthy software requirements

图 1 面向目标的可信软件需求

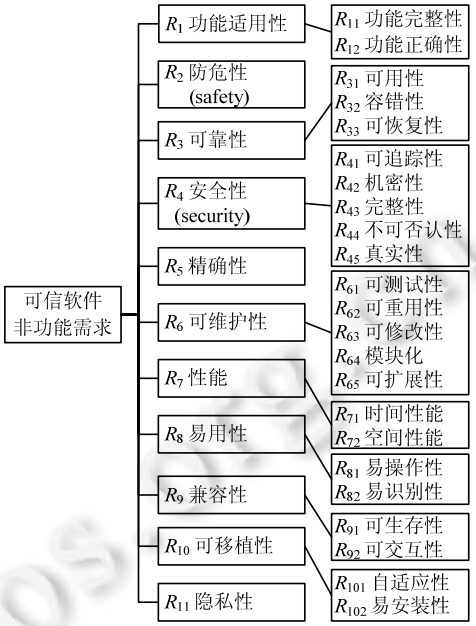


Fig.2 Decomposition model of trustworthy software non-functional requirements

图 2 可信软件非功能需求分解模型

本节分析并定义可信软件需求,提出可信软件非功能需求分解模型,目的是为了获取可信软件非功能需求中的可信关注点和软目标,进而获取满足非功能需求的过程策略.为了实现这一目标,本文提出可信软件非功能需求驱动的过程策略选取框架 TRPF,框架流程如图 3 所示.TRPF 由 3 个流程步骤构成:首先,获取可信软件非功能需求;然后,建立非功能需求与过程策略间的关系;最后,通过非功能需求选取过程策略.具体流程细节分别在本文第 3.1 节~第 3.3 节中加以阐述.

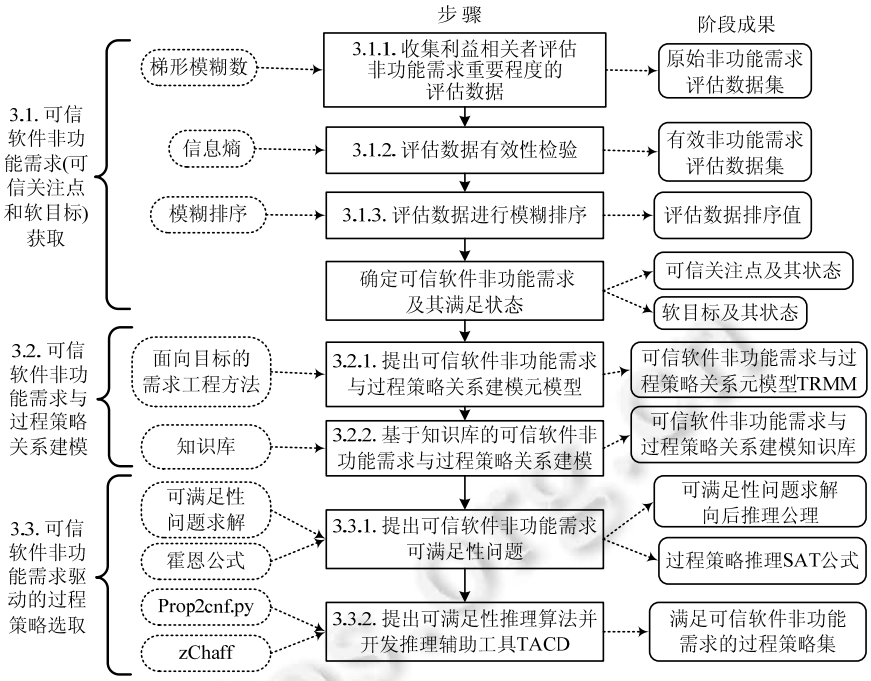


Fig.3 Trustworthiness requirements-processes framework (TRPF)

图 3 可信软件非功能需求驱动过程策略选取框架 TRPF

3 可信软件非功能需求驱动的过程策略选取框架 TRPF

3.1 可信软件非功能需求获取

可信软件的利益相关者是我们获取可信软件需求的来源,其中,非功能需求会因软件利益相关者的不同而以不同的形式表达;并且,基于利益相关者的经验以及所处立场的不同,对同一非功能需求的期望值也会有所不同.很多软件项目就因为忽视了利益相关者之间的需求平衡而导致项目失败,而当软件利益相关者众多且表达出不同期望时,相互协商并在需求上达成一致是一大挑战[45].针对可信软件非功能需求难以精确表达的特点,下面我们使用模糊集合论中的梯形模糊数量化非功能需求,通过收集利益相关者对可信软件非功能需求的评估数据,使用信息熵对评估数据进行有效性检查后,使用模糊排序方法实现可信软件非功能需求中可信关注点和软目标的获取.

3.1.1 可信软件非功能需求描述

1965 年,Zadeh 提出模糊集合论,创建了研究模糊性或不确定性问题的理论方法[46].一个经典集合中的任一元素要么属于集合,要么不属于集合,二者必居其一.模糊集合论将经典集合只取 0,1 两值推广到区间[0,1].模糊理论与技术的一个突出优点是能够较好地描述与效仿人的思维方式,总结和反映人的体会与经验,对于复杂事物和系统可以进行模糊度量、模糊识别、模糊推理、模糊控制和模糊决策[47].

可信软件的非功能需求具有无法精确表达、含糊不清以及不确定等特点,非常适合用模糊集合论的方法来描述,所有非功能需求都可以使用统一的模糊数来描述其重要程度.当利益相关者对非功能需求进行评估时,指定每项非功能需求的隶属度,隶属度描述了区间[0,1]中的值.当集合中一个元素的隶属度为 1 时,表示这个元素在集合中;相反地,当元素的隶属度为 0 时,表示元素不在集合中,而模糊元素的隶属度则介于 0 和 1 之间.

本文参照 Zhu 等人描述可信软件非功能需求的方法[24],使用通用梯形模糊数描述可信软件非功能需求的量化数值.梯形模糊数也称为模糊区间,一个通用梯形模糊数 $\tilde{A}$ 可以表示为 $\tilde{A}=(a_1,a_2,a_3,a_4;e)$, 其中,$0<e<1$,并且 a_1,a_2,a_3,a_4 是实数.隶属函数定义为

$$\mu_{\tilde{A}}(x)=\begin{cases}0, & x\leqslant a_1\\ \dfrac{x-a_1}{a_2-a_1}, & a_1<x\leqslant a_2\\ w, & a_2\leqslant x<a_3\\ \dfrac{a_4-x}{a_4-a_3}, & a_3\leqslant x<a_4\\ 0, & x\geqslant a_4\end{cases} \tag{1}$$

图 4 给出了通用梯形模糊数 $\tilde{A}$ 的隶属函数曲线(当 e=1 时, $\tilde{A}$ 成为标准梯形模糊数).

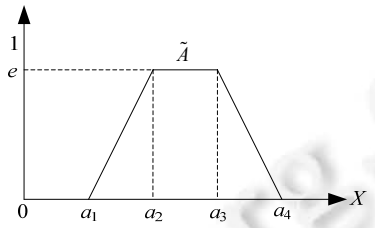


Fig.4 General trapezoidal fuzzy number $\tilde{A}$

图 4 通用梯形模糊数 $\tilde{A}$

其中,a_1,a_2,a_3 和 a_4 反映了数据的模糊性:当 a_2=a_3 时, $\tilde{A}$ 成为三角模糊数;当 a_1=a_2 且 a_3=a_4 时, $\tilde{A}$ 成为区间数;而当 a_1=a_2=a_3=a_4 且 w=1 时,模糊数退化为普通实数.

另外,可以对模糊数实施扩张运算,包括一元模糊数的补、数乘、次方以及二元模糊数的加、减、乘、除运算.本文后续内容使用到二元模糊数的加法、乘法和除法运算,假设 $\tilde{A}=(a_1,a_2,a_3,a_4,e_{\tilde{A}})$ 和 $\tilde{B}=(b_1,b_2,b_3,b_4,e_{\tilde{B}})$ 是两个通用梯形模糊数,它们的加法、乘法及除法运算[48]为:

- 模糊数加法⊕

$$\tilde{A}\oplus\tilde{B}=(a_1,a_2,a_3,a_4,e_{\tilde{A}})\oplus(b_1,b_2,b_3,b_4,e_{\tilde{B}})=(a_1+b_1,a_2+b_2,a_3+b_3,a_4+b_4;\min(e_{\tilde{A}},e_{\tilde{B}})) \tag{2}$$

- 模糊数乘法⊗

$$\tilde{A}\otimes\tilde{B}=(a_1,a_2,a_3,a_4,e_{\tilde{A}})\otimes(b_1,b_2,b_3,b_4,e_{\tilde{B}})=(a_1\times b_1,a_2\times b_2,a_3\times b_3,a_4\times b_4;\min(e_{\tilde{A}},e_{\tilde{B}})) \tag{3}$$

- 模糊数除法∅

$$\tilde{A}\varnothing\tilde{B}=(a_1,a_2,a_3,a_4,e_{\tilde{A}})\varnothing(b_1,b_2,b_3,b_4,e_{\tilde{B}})=(a_1/b_4,a_2/b_3,a_3/b_2,a_4/b_1;\min(e_{\tilde{A}},e_{\tilde{B}})) \tag{4}$$

其中,$b_1\neq0$,$b_2\neq0$,$b_3\neq0$,$b_4\neq0$.

在获取软件利益相关者的可信软件非功能需求数据时,我们使用表 1 列出的语言变量:完全不重要(AL, absolutely low)、不重要(L,low)、较不重要(FL,fairly low)、中立(M,medium)、较重要(FH,fairly high)、重要(H, high)、非常重要(AH,absolutely high),这些用自然语言表达的语言变量方便利益相关者描述他们评估各项可信软件非功能需求的重要程度.

Table 1 Linguistic variables and corresponding fuzzy numbers for trustworthy software non-functional requirements

表 1 可信软件非功能需求语言变量及对应梯形模糊数

需求评估语言	梯形模糊数
完全不重要(AL)	(0,0,0.077,0.154;1)
不重要(L)	(0.077,0.154,0.231,0.308;1)
较不重要(FL)	(0.231,0.308,0.385,0.462;1)
中立(M)	(0.385,0.462,0.538,0.615;1)
较重要(FH)	(0.538,0.615,0.692,0.769;1)
重要(H)	(0.692,0.769,0.846,0.923;1)
非常重要(AH)	(0.846,0.923,1,1;1)

重要程度的高低,将决定非功能需求中哪些是可信关注点、哪些是软目标,而且将为后面推理过程策略设定非功能需求的满足状态提供依据.

对应每一个语言变量,我们给出等间距的梯形模糊数表示,使用相等间距的原因是利益相关者应该以相同的概率选择重要程度.由于重要程度分为 7 个等级,在区间[0,1]间构造的等间距梯形模糊数隶属函数如图 5 所示.

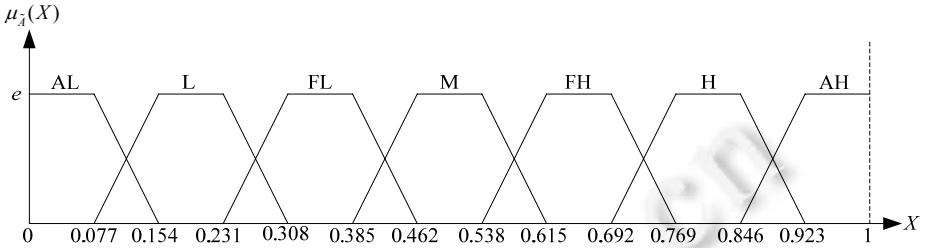


Fig.5 Membership function for linguistic variables

图 5 语言变量的隶属函数

3.1.2 可信软件非功能需求评估数据有效性检验

可信软件利益相关者对可信软件非功能需求的重要程度使用梯形模糊数给出评估数据,这些评估数据是利益相关者依据其知识背景、经验以及对项目的熟悉程度而主观给出的,在综合权衡所有评估数据之前,需要对评估数据的有效性及客观性进行检查.我们之前曾使用信息熵的方法有效地完成过类似的工作[49],因此在本文中,我们提出基于信息熵的评估数据有效性检验及综合权衡方法.首先,构造评估数据矩阵 A:

$$A=\begin{pmatrix}\tilde{A}_{11} & \cdots & \tilde{A}_{1m}\\ \vdots & \ddots & \vdots\\ \tilde{A}_{n1} & \cdots & \tilde{A}_{nm}\end{pmatrix}=(\tilde{A}_1\ \tilde{A}_2\ \ldots\ \tilde{A}_m)=\begin{pmatrix}\tilde{A}^1\\ \tilde{A}^2\\ \vdots\\ \tilde{A}^n\end{pmatrix} \tag{5}$$

其中, $\tilde{A}_{ij}$ 为第 j 个利益相关者($1\leqslant j\leqslant m$)对第 i 项非功能需求(这里的非功能需求是可信软件非功能需求分解模型中的所有非功能需求,$1\leqslant i\leqslant n$)的评估数据,即,利益相关者根据其具备的专业知识、经验以及对项目的了解对非功能需求的重要程度给出评估数据.基于矩阵 A,下面分 3 个步骤来完成评估数据的有效性检验.

Step 1. 评估利益相关者贡献.

对于矩阵 A,若 $\tilde{A}_x$ ($1\leqslant x\leqslant m$)中各项评估数据的数值相差太小,则说明提供该评估数据的利益相关者对所评估的可信软件需求不明确,其提供的数据会破坏综合评估数据的有效性,需要考虑去除这个利益相关者提供的数据或者重新获取;相反,如果评估数据数值相差较大,说明评定结果分散,该专家对所评估的可信软件需求较为明确且提供的评估数据有一定的针对性,具有较高的说服力,应该在综合评估中起关键作用.当然,如果评估数据数值相差过大,则说明利益相关者的评估赋值过于偏激或者过于随意,也会破坏综合评估的有效性,同样需要考虑去除或者重新获取.根据前面的分析,本文用如公式(6)所示的信息熵来对利益相关者所提供的评估数据的有效性进行检验.

$$H_j=-\sum_{i=1}^{n}\tilde{A}_{ij}\log_2\tilde{A}_{ij}=-(\tilde{A}_{1j}\otimes\log_2\tilde{A}_{1j}\oplus\tilde{A}_{2j}\otimes\log_2\tilde{A}_{2j}\oplus\ldots\oplus\tilde{A}_{nj}\otimes\log_2\tilde{A}_{nj}) \tag{6}$$

由熵的极值性可知 $H_j\leqslant\log_2 n$,对 H_j 作归一化处理:

$$\theta_j=\frac{H_j}{\log_2 n}=-\frac{1}{\log_2 n}\sum_{i=1}^{n}\tilde{A}_{ij}\log_2\tilde{A}_{ij} \tag{7}$$

其中,$0\leqslant\theta_j\leqslant 1$,$\theta_j$ 越大,表明利益相关者 j 对非功能需求的评估贡献越小,对综合评估数据有效性产生负面影响;相反地,θ_j 越小,表明利益相关者 j 对非功能需求的评估贡献越大,有利于保证综合评估数据的有效性.为了使计

算值能够正向地反映利益相关者提供有效评估数据的贡献度,我们使用 $1-\theta_j$ 表示其贡献度,再次对其进行归一化处理后得到各个利益相关者的贡献度.

$$E_j=\frac{1-\theta_j}{m-\sum_{j=1}^{m}\theta_j} \tag{8}$$

其中,$0\leqslant E_j\leqslant 1$.$E_j$ 越大,则表明利益相关者 j 相对重要且贡献越大,贡献大的专家提供的评估数据才是有意义的.

Step 2. 解决贡献小的利益相关者的分歧意见.

由于在评估可信软件各项非功能需求的重要程度时,需要保证评估数据的有效性,并减少主观性,因而需要考虑利益相关者之间的意见分歧,需要注意评估意见分歧大的专家.因此,根据实际情况对专家的贡献度设置最小阈值和最大阈值,对于小于最小阈值和大于最大阈值的利益相关者,考虑重新获取评估数据或者删除其提供的评估数据,要保证所获取的评估数据都维持在一个相对有效的范围内,这样可以避免贡献小的利益相关者引入不必要的分歧,致使评估数据变得不合理而获取到错误的非功能需求.

Step 3. 综合可信软件非功能需求评估数据.

保证可信软件非功能需求评估数据有效后,就可以计算综合评估数据了.对于上述矩阵,取向量 $\tilde{A}^y$ ($1\leqslant y\leqslant n$)元素值的平均值:

$$W_i=\left(\frac{1}{m}\right)\otimes(\tilde{A}_{i1}\oplus\tilde{A}_{i2}\oplus\ldots\oplus\tilde{A}_{im})=\frac{1}{m}\sum_{j=1}^{m}\tilde{A}_{ij} \tag{9}$$

其中,W_i 是第 i 项非功能需求的重要程度的平均值.平均值越大,表明利益相关者认定这项非功能需求的重要程度越高,应将该非功能需求确定为可信关注点,并且在推理时将其设定为较高的满足状态;相反地,平均值越小,则说明各个专家认为该非功能需求的重要程度越低,则应将其确定为软目标,并且在推理时将其设定为较低的满足状态.然而,由于模糊数无法直接比较大小,因此下面我们使用模糊排序方法对综合后的可信评估数据进行排序,以辅助确定可信软件非功能需求中的可信关注点和软目标以及需要设定的满足状态.

3.1.3 可信软件非功能需求数据排序

在众多模糊排序方法中,Chen 和 Sanguansat's 的方法同时考虑了正、负区域以及模糊数的高度,有效解决了其他排序方法中无法处理实数值、不区分高度不同的模糊数、无法区分以不同方式表达的模糊数以及倾向于悲观决策排序等问题.故,本文基于 Chen 和 Sanguansat's 的模糊排序方法[48]对获取后的非功能需求进行排序.

对于一组 m 个可信软件利益相关者($S_1,S_2,\ldots,S_m$),他们分别评估了 n 个可信软件非功能需求($C_1,C_2,\ldots,C_n$)的重要程度,得到各项非功能需求的重要程度值为

$$W_i=(w_{i1},w_{i2},w_{i3},w_{i4};e_{W_i}),$$

其中,$-\infty<w_{i1}\leqslant w_{i2}\leqslant w_{i3}\leqslant w_{i4}<\infty, e_{W_i}\in(0,1]$,且 $1\leqslant i\leqslant n$.W_i 的排序过程如下:

Step 1. 转换每一个通用模糊数 $W_i=(w_{i1},w_{i2},w_{i3},w_{i4};e_{W_i})$ 为标准模糊数 W_i^*:

$$W_i^*=\left(\frac{w_{i1}}{l},\frac{w_{i2}}{l},\frac{w_{i3}}{l},\frac{w_{i4}}{l};e_{W_i}\right)=(w_{i1}^*,w_{i2}^*,w_{i3}^*,w_{i4}^*;e_{W_i}) \tag{10}$$

其中,$l=\max_{ij}(\lceil|w_{ij}|\rceil,1)$,$1\leqslant i\leqslant n$ 且 $1\leqslant j\leqslant 4$.

Step 2. 计算正、负区域 $Area_{iL}^-, Area_{iR}^-, Area_{iL}^+$ 和 $Area_{iL}^+$,它们是隶属函数 $f_{W_i^*}^L$ 和 $f_{W_i^*}^R$ 的梯形区域.

$$f_{W_i^*}^L=e_{W_i}\times\frac{(x-w_{i1}^*)}{(w_{i2}^*-w_{i1}^*)}, w_{i1}^*\leqslant x<w_{i2}^* \tag{11}$$

$$f_{W_i^*}^R=e_{W_i}\times\frac{(x-w_{i4}^*)}{(w_{i3}^*-w_{i4}^*)}, w_{i3}^*\leqslant x<w_{i4}^* \tag{12}$$

$$Area_{iL}^-=e_{W_i}\times\frac{(w_{i1}^*+1)+(w_{i2}^*+1)}{2} \tag{13}$$

$$Area_{iR}^{-}=e_{W_i}\times\frac{(w_{i3}^{*}+1)+(w_{i4}^{*}+1)}{2} \tag{14}$$

$$Area_{iL}^{+}=e_{W_i}\times\frac{(1-w_{i1}^{*})+(1-w_{i2}^{*})}{2} \tag{15}$$

$$Area_{iR}^{+}=e_{W_i}\times\frac{(1-w_{i3}^{*})+(1-w_{i4}^{*})}{2} \tag{16}$$

Step 3. 计算每一个非功能需求 W_i^* 的 $XI_{A_i^*}$ 及 $XD_{A_i^*}$ 值,它们分别表示正向及负向影响.

$$XI_{A_i^*}=Area_{iL}^{-}+Area_{iR}^{-} \tag{17}$$

$$XD_{A_i^*}=Area_{iL}^{+}+Area_{iR}^{+} \tag{18}$$

Step 4. 计算每一个 W_i^* 的排序值 $Score(W_i^*)$.

$$Score(W_i^*)=\frac{1\times XI_{W_i^*}+(-1)\times XD_{W_i^*}}{XI_{W_i^*}+XD_{W_i^*}+(1-e_{W_i^*})}=\frac{XI_{W_i^*}-XD_{W_i^*}}{XI_{W_i^*}+XD_{W_i^*}+(1-e_{W_i^*})} \tag{19}$$

其中, $Score(W_i^*)\in[-1,1]$ 且 $1\leqslant i\leqslant k$.

排序值 $Score(W_i^*)$ 越大,表明对应的非功能需求越重要,应将其确定为可信关注点;而对于排序值 $Score(W_i^*)$ 小的非功能需求,则应将其确定为软目标.另外,在可信关注点集合中,排序值越大,表明软件利益相关者认为其重要程度越高,其满足状态就应设定得较高;反之,如果排序值低,表明相对不重要,则满足状态可以设定得较低.软目标满足状态的设定与此类似.

至此,我们已经获得了可信软件非功能需求中的可信关注点和软目标以及它们需要达到的满足状态.下面,我们提出基于可信软件非功能需求(包括可信关注点和软目标)的过程策略选取方法.首先,基于获取的可信关注点和软目标,将它们与实现可信软件非功能需求的过程策略之间的关系进行建模;然后,基于可信关注点及软目标的满足状态提出过程策略选取方法.

3.2 可信软件非功能需求与过程策略关系建模

传统的实现软件非功能需求的方法主要从技术或者设计策略角度加以研究,当然,技术的改进、策略的制定是非常重要的,然而,软件的质量很大程度上依赖于软件开发时所使用的过程,而开发可信软件、实现软件可信演化,其本质是在开发软件以及演化软件的过程中满足软件的可信质量,这必然依赖于软件过程.因此,本文从过程控制角度提出实现可信软件非功能需求的过程策略.

考虑到非功能需求之间复杂的相关关系,本文首先对可信软件非功能需求与过程策略间的关系进行建模(建模得到的模型以下简称为关系模型),然后,基于可满足性问题求解方法提出过程策略的选取方法.

3.2.1 可信软件非功能需求与过程策略关系元模型 TRMM

参照 NFR 框架[17−20]和 *i**模型[11]可以明确表达非功能需求和设计决策的能力,借鉴 Sebastiani 等人提出的目标模型向后推理方法[26],使用面向方面方法中关注点和方面的概念,本文提出可信软件非功能需求与过程策略关系元模型 TRMM,该模型针对可信软件,提出以可信关注点作为目标、并用过程策略实现可信关注点的方法实现可信软件非功能需求.在可信软件整个生命周期过程中,过程策略将按照软件过程的不同粒度建模为可信过程方面、可信活动方面或者可信任务方面,通过将方面编织到软件过程中,以提升生产出软件的可信性.

图 6 用图形化的方式直观地描述了 TRMM.

下面对 TRMM 中各个结点之间的关系进行解释,之后给出 TRMM 的形式化定义.

- 实现

连接可信关注点和过程策略,表示实现可信关注点的过程策略.实现可信关注点的多个过程策略之间是 OR 关系,如果过程策略之间存在依赖关系,则将其定义为一个存在分解关系的过程策略,即:一个过程策略可以分解为多个过程策略(存在依赖关系的多个过程策略),而分解关系是 AND 关系.实现也用于连接硬目标和基本

模型(即软件过程),表示软件过程实现了软件的硬目标.由于本文仅研究可信软件非功能需求,因此和功能需求有关的元素,包括硬目标和基本模型,都用虚线表示.

- 贡献

描述过程策略对可信关注点和软目标的贡献,贡献定性地表示为促进或者损害,其中,促进表示过程策略的正向促进作用,而损害表示过程策略的负向抑制作用.在 TRMM 中,贡献关系中的损害关系实际上描述了非功能需求(包括可信关注点和软目标)间的冲突关系,即:当实现一个非功能需求 a 的过程策略与另一个非功能需求 b 之间有抑制作用时,说明非功能需求 a 和 b 在使用此项过程策略时存在冲突.冲突关系如果损害到软目标,则意味着软件质量降低,损害到其他可信关注点,则意味着可信性降低,这些都使得软件可信性降低.因此,过程策略的选择需要权衡所有可能存在的冲突.

- 合成

这里使用了面向方面思想中方面与基本模型之间的合成概念,即:将方面合成到基本模型中,以实现可信的注入.在 TRMM 中,方面定义为可信方面,分为可信过程方面、可信活动方面或者可信任务方面,它们是过程策略按照软件过程不同粒度模块化的结果.由于本文不研究软件过程建模,有关将可信过程融入到软件过程模型中的相关研究,我们已在文献[50]中给出了类似的融合方法,因此,本文在 TRMM 中虽然表示出合成,但使用 TRMM 建模时并不涉及合成操作,只起到表达过程策略的作用.

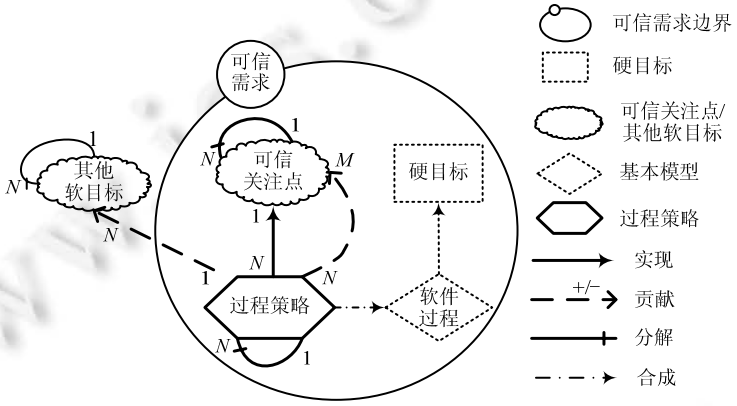


Fig.6 Trustworthiness requirements meta-model (TRMM)

图 6 可信软件非功能需求与过程策略关系元模型(TRMM)

使用 TRMM 建模可信软件非功能需求与过程策略间的关系,不仅可以展示实现可信关注点的过程策略,更重要的是描述出非功能需求之间的冲突关系.建模的目的是为了找出整体满足非功能需求的过程策略,为了解决冲突,本文使用可满足性问题求解方法,即:找出整体满足非功能需求的过程策略集合,或在项目组能够接受一定程度的矛盾的情况下找到可满足的过程策略集合.为了使用可满足性问题求解方法,本文对 TRMM 做出如下形式化定义.

定义 1(可信软件非功能需求与过程策略关系元模型 TRMM). 一个可信软件非功能需求与过程策略关系的元模型是一个二元组 $M=(N,R)$,其中,

(1) $N=T\cup S\cup TA$ 是结点的集合,其中,

- T 是可信关注点的集合,$\forall t\in T$ 是一个可信关注点;
- S 是软目标的集合,$\forall s\in S$ 是一个软目标;
- TA 是过程策略的集合,$\forall ta\in TA$ 是一个过程策略;

(2) $R=R^{dec}\cup R^{imp}\cup R^{ctr}$ 是元素之间二元偏序关系的集合,其中,

- $R^{dec}\subseteq(T\times T)\cup(S\times S)\cup(TA\times TA)$是可信关注点 T、软目标 S、过程策略 TA 的分解关系:

- $T\times T=\{(t',t)|t,t'\in T\wedge t$ 分解出 $t'\}$,称 t'为 t 的子可信关注点;
- $S\times S=\{(s',s)|s,s'\in S\wedge s$ 分解出 $s'\}$,称 s'为 s 的子软目标;
- $TA\times TA=\{(ta',ta)|ta,ta'\in TA\wedge ta$ 分解出 $ta'\}$,称 ta'为 ta 的子过程策略;

- $R^{imp}\subseteq T\times TA$ 描述可信关注点与过程策略之间的实现关系,$R^{imp}=\{(ta,t)|ta\in TA\wedge t\in T\wedge$过程策略 ta 实现可信关注点 $t\}$;
- $R^{ctr}\subseteq(TA\times T)\cup(TA\times S)$是过程策略对可信关注点和软目标的贡献关系,$\forall r\in R^{ctr}$ 是一个贡献关系,$F:r\mapsto\{+,-\}$($\mapsto$是映射关系),+为促进关系,–为抑制关系:
  - $TA\times T=\{(ta,t)|ta\in TA\wedge t\in T\wedge(ta,t)\mapsto\{+,-\}\}$,称过程策略集合 TA 对可信关注点集合 T 有促进或者抑制的贡献关系;
  - $TA\times S=\{(ta,s)|ta\in TA\wedge s\in S\wedge(ta,s)\mapsto\{+,-\}\}$,称过程策略集合 TA 对软目标集合 S 有促进或者抑制的贡献关系.

基于映射(也称为函数)的概念,我们定义 TRMM 中的贡献、分解和实现关系基数.

定义 2(关系基数). TRMM 中的关系 R 分为多对多关系和一对多关系,其中,贡献关系是多对多关系,即 R^{ctr}: $N\times\ldots\times N\mapsto M\times\ldots\times M$;分解和实现关系是一对多关系,即 R^{dec},R^{imp}:$N\mapsto N\times\ldots\times N$.

为了保证使用 TRMM 建模得到的关系模型能够转化为 SAT 求解机可接受的 CNF 形式,在使用 TRMM 建模时,需要满足如下约束.

约束 1(结点约束). 若 $dom(r)=\{x\in N|\exists y\in N:(x,y)\in r,r\in R\}$,$cod(r)=\{x\in N|\exists y\in N:(y,x)\in r,r\in R\}$,则 $dom(r)\cup cod(r)=N$,即模型无孤立结点.

约束 1 要求模型中无孤立结点,如果只有孤立的可信关注点,没有过程策略来实现,则可信关注点本质上并未得到满足,说明建模还没有完成;如果孤立存在一个过程策略,则是错误的,因为过程策略必须是为了实现可信关注点而存在的;而对于孤立的软目标,则可从模型中去除,其孤立状态表明没有任何过程策略会抑制这个软目标的满足,因此,假设这个软目标的状态可以满足.

另外,SAT 求解机的输入是表示待检验公式的有向无环图 DAG,如果模型有环,则无法转化为 DAG.因此,定义如约束 2 所示.

约束 2(贡献关系约束). 实现一个可信关注点的所有过程策略都不会和可信关注点本身有贡献关系,即:如果$(ta,t)\in R^{imp}$,则不存在$(ta,t)\in R^{ctr}$.

3.2.2 可信软件非功能需求与过程策略关系建模知识库

基于 TRMM 对可信软件非功能需求与过程策略之间的关系进行建模是一种创造性的工作,而且一个过程策略可以在多个相似建模项目中反复使用,非功能需求和过程策略之间的关系也可以反复使用,因此,有必要构建一个知识库辅助建模者完成建模.这个知识库由非功能需求库和过程策略库构成,如图 7 所示.非功能需求库包含了利益相关者及对应的非功能需求以及这些非功能需求的分解,把利益相关者加入其中可以在完成建模和推理后积累和利益相关者有关的非功能需求的冲突及决策结果.使用过程策略库,可以找出实现非功能需求的过程策略以及过程策略对其他非功能需求的贡献.

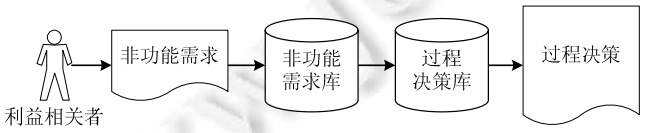


Fig.7 Knowledge base structure of TRMM

图 7 TRMM 建模知识库结构

1) 非功能需求库

非功能需求库是利益相关者关注的非功能需求的分层结构,如前所述,不同利益相关者对软件有不同的可

信期望,所有利益相关者的可信期望综合后得到可信软件的可信需求.正是基于利益相关者的可信需求来开发软件,才能开发出行为符合用户预期的可信软件.非功能需求库存储利益相关者最关注的非功能需求以及非功能需求的分解,不仅有利于快速帮助利益相关者获取可信需求,还可以了解利益相关者的潜在需求,并提供有针对性的建议及决策.另外,利益相关者通常不会关注非功能需求的所有子需求,而且有时冲突仅仅体现在子需求之间,因此,非功能需求库还需要存储分解结构.

非功能需求库的结构是由扩展的巴科斯范式 EBNF(extended Backus-Naur form)定义的,定义中的"〈 〉"表示非功能需求的结构成分,"[ ]"表示可选结构成分,可选意味着可以出现 0 次到 1 次.为了简洁,忽略了不重要的结构成分.

非功能需求的结构定义如下:

- 〈非功能需求〉::=NFR〈非功能需求名〉〈非功能需求描述〉〈利益相关者列表〉〈子需求列表〉
- 〈非功能需求名〉::=功能适用性|可靠性|防危性|安全性|精确性|易用性|性能|可维护性|可移植性|兼容性|隐私性
- 〈利益相关者列表〉::=〈利益相关者名〉|〈利益相关者名〉;〈利益相关者列表〉
- 〈利益相关者名〉::=管理方|工程方|安全工程师|可靠性工程师|市场推广工程师|法律部门|标准机构|政府部门|投资者|使用者|专家
- 〈子需求列表〉::=〈子需求名〉|〈子需求名〉;〈子需求列表〉
- 〈子需求名〉::=功能完整性|功能正确性|可用性|容错性|可恢复性|可追踪性|机密性|完整性|不可否认性|真实性|时间性能|空间性能|易操作性|易识别性|可测试性|可重用性|可修改性|模块化|可扩展性|可生存性|可交互性|自适应性|易安装性

2) 过程策略库

过程策略库存储了过程策略及其对非功能需求的贡献,根据贡献选择过程策略即是选择合适的过程策略,表 2 给出一个过程策略以及过程策略贡献的示例.

Table 2 Examples of process strategies' contributions to the non-functional requirements

表 2 过程策略对非功能需求影响的示例

非功能需求	过程策略	抑制的非功能需求	促进的非功能需求
可靠性	容错设计 冗余设计	安全性,精确性,功能适用性 安全性,可维护性,性能	易用性 防危性
安全性	分析攻击面 实现最小密码设计 定义最小安全标准	 易用性,性能 性能,可维护性,易用性	可靠性,防危性
可信性	可信性监控	性能	
兼容性	接口设计	安全性,可维护性	

同样,过程策略的结构也用扩展的巴科斯范式 EBNF 定义如下:

- 〈过程策略库〉::=〈非功能需求名〉:〈过程策略列表〉
- 〈过程策略列表〉::=〈过程策略〉|〈过程策略〉;〈过程策略列表〉

过程策略的结构定义如下:

- 〈过程策略〉::=STRATEGY〈过程策略名〉〈策略描述〉[〈贡献〉[〈冲突控制〉]]

一个过程策略对其他非功能需求的贡献定义如下:

- 〈贡献〉::=CONTRIBUTION〈贡献声明列表〉
- 〈贡献声明列表〉::=〈贡献声明〉|〈贡献声明〉〈贡献声明列表〉
- 〈贡献声明〉::=●〈非功能需求〉(+/–)〈贡献描述〉
- 〈非功能需求〉::=〈非功能需求库〉.〈非功能需求名〉|〈非功能需求库〉.〈子需求名〉

当过程策略对其他非功能需求的贡献是抑制贡献关系时,采取如下所示的冲突控制定义:

- 〈冲突控制〉::=CONTROL〈控制声明列表〉
- 〈控制声明列表〉::=〈控制声明〉|〈控制声明〉〈控制声明列表〉
- 〈控制声明〉::=●〈非功能需求〉:〈控制公式〉(〈贡献描述〉)
- 〈非功能需求〉::=〈非功能需求库〉.〈非功能需求名〉|〈非功能需求库〉.〈子需求名〉
- 〈控制公式〉::=〈可信方面标识〉〈控制符号〉〈可信方面标识〉
- 〈控制符号〉::=〈数据依赖〉|〈控制依赖〉

其中,数据依赖是指一个过程策略的执行需要另外一个过程策略的数据,而控制依赖是指一个过程策略的执行由另外一个过程策略决定.

3.3 可信软件非功能需求驱动的过程策略选取

由于非功能需求之间复杂的相关关系,满足某可信关注点的过程策略有可能对其他可信关注点或者软目标有抑制作用(也称为冲突关系),因此,我们需要对过程策略的选取进行推理.推理本质上是对非功能需求进行可满足性问题求解,如果求解结果是满足,则找到了满足非功能需求的过程策略;如果求解结果是不满足,则由建模者决定导致不满足的矛盾的解决方法,解决后继续推理,直至得到满足可信软件非功能需求的过程策略.

Nuseibel 提出了软件需求的分析应使用形式化推理,而逻辑正是实施这种分析的有效工具,使用逻辑的一大优势在于可以进行自动的推理和分析[51].基于这一思想,本文借鉴 Sebastiani 等人提出的目标模型向后推理方法[26]提出满足可信软件非功能需求的过程策略推理方法.

推理分向前推理和向后推理:向前推理是根据过程策略来判断通过实现这些过程策略后,可信软件的每一个可信关注点和软目标能够达到的满足状态;向后推理是根据软件利益相关者定义的可信关注点和软目标的满足状态来寻找可行的过程策略.由于向前推理是通过过程策略的满足状态来推断可信关注点和软目标的状态,如果推断出的状态和软件利益相关者可信预期不一致,向前推理不能回溯,则不能提供让软件利益相关者满意的过程策略方案.如果穷举过程策略组合实施向前推理,假设模型中有 n 个过程策略,则有 2^n 个可能的组合方案,如此多的组合方案都输入推理过程是不合理的,也是不必要的.因此,向前推理无法有效判断过程策略应该满足什么状态可以让可信关注点和软目标的状态符合软件利益相关者的可信预期.

为了找到让软件利益相关者满足的过程策略方案,基于可满足性问题求解方法,实现向后推理是寻找符合软件利益相关者可信需求且可以权衡冲突的合适方法.下面基于可信软件非功能需求与过程策略的关系模型,本文提出基于可满足性问题求解方法的过程策略选取方法.

3.3.1 可信软件非功能需求的可满足性问题

寻找满足可信软件非功能需求的过程策略,实际上是在求解非功能需求的可满足性问题.为了表达关系模型中各个结点的满足状态,我们用一阶谓词描述方法[26]来定义状态标记.

定义 3(状态标记 $L(n)$). 一个状态标记是一个一阶谓词 $L(n)::=SA(n)|PS(n)|PD(n)|DE(n)$,其中,$n\in N$($N$ 在定义 1 中定义为结点集合),$SA(n)$表示结点 n 的状态为满足,$PS(n)$表示结点 n 的状态为部分满足,$PD(n)$表示结点 n 的状态为部分不满足,$DE(n)$表示结点 n 的状态为不满足.

另外,根据关系模型中结点的实际含义,需要对不同结点的满足状态指定不同的约束,其中,可信关注点和软目标的状态可以是上述定义 3 中所有状态标记中的任何一种状态;而过程策略只有过程执行和不执行两种状态,对应着状态标记就只有满足状态和不满足状态.因此,我们的定义如约束 3 所示.

约束 3(过程策略状态标记). 对于过程策略 $ta\in TA$,其状态标记只能取满足状态或者不满足状态,即:

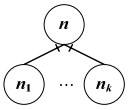
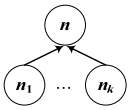
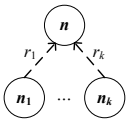
$$L(ta)::=SA(ta)|DE(ta).$$

在向后推理过程中,结点的状态标记会根据结点间不同的关系而产生不同的状态,按照分解、实现和贡献关系的不同,向后推理公理见表 3.

在应用可满足性问题求解时,会有两种结果:发现矛盾,则模型是不可满足的;反之,模型在各个结点约束了状态标记的情况下是满足的.在本文中,矛盾是指一个结点 n 的状态标记同时标记两种状态为真(true).例如:一个结点的满足和不满足状态都为真,则不知道这个结点到底是满足状态还是不满足状态,这就是矛盾.

Table 3 Axioms of backward reasoning

表 3 向后推理公理

结点关系	状态	向后状态传播公理		
分解关系 R^{dec} 	SA	$SA(n)\rightarrow\wedge_{i=1}^{k}SA(n_i)$		$DB1$
	PS	$PS(n)\rightarrow\wedge_{i=1}^{k}PS(n_i)$		$DB2$
	PD	$PD(n)\rightarrow\vee_{i=1}^{k}PD(n_i)$		$DB3$
	DE	$DE(n)\rightarrow\vee_{i=1}^{k}DE(n_i)$		$DB4$
实现关系 R^{imp} 	SA	$SA(n)\rightarrow\wedge_{i=1}^{k}SA(n_i)$		$MB1$
	PS	$PS(n)\rightarrow\vee_{i=1}^{k}SA(n_i)$		$MB2$
	PD	$PD(n)\rightarrow\vee_{i=1}^{k}DE(n_i)$		$MB3$
	DE	$DE(n)\rightarrow\wedge_{i=1}^{k}DE(n_i)$		$MB4$
贡献关系 R^{ctr} 	SA	$r_i\mapsto\{+\}$	$SA(n)\rightarrow SA(n_i)$	$CB1$
		$r_i\mapsto\{-\}$	$SA(n)\rightarrow DE(n_i)$	$CB2$
	PS	$r_i\mapsto\{-\}$	$PS(n)\rightarrow DE(n_i)$	$CB3$
	PD	$r_i\mapsto\{+\}$	$PD(n)\rightarrow DE(n_i)$	$CB4$
	DE	$r_i\mapsto\{+\}$	$DE(n)\rightarrow DE(n_i)$	$CB5$
		$r_i\mapsto\{-\}$	$DE(n)\rightarrow SA(n_i)$	$CB6$

基于上述分析,当对可信软件非功能需求进行可满足性求解时,需要构造如公式(20)所示的 SAT 公式:

$$\phi::=\phi_{Model}\wedge\phi_{Initial}\wedge\phi_{Constraint}\wedge\phi_{Inconsistency} \tag{20}$$

其中,ϕ_{Model} 是关系模型公式,$\phi_{Initial}$ 是初始状态公式,$\phi_{Constraint}$ 是结点状态约束公式,$\phi_{Inconsistency}$ 是求解过程中可接受矛盾等级公式.

1) 关系模型公式

关系模型描述了所有结点以及结点之间的关系,基于向后推理公理(见表 4),关系模型公式定义为

$$\phi_{Model}::=\wedge_{n,n_i\in N}(L(n)\rightarrow\vee_{(n,n_i)\in R}L(n_i)) \tag{21}$$

2) 初始状态公式

初始状态由可信关注点状态和软目标状态组成,是使用第 3.1 节所述方法获取可信关注点和软目标后,按照它们的重要程度而定义的初始状态,初始状态公式定义为

$$\phi_{Initial}::=\wedge_{n\in N}L(n) \tag{22}$$

初始状态公式定义的初始状态属于最低要求,即:若某结点的初始状态设置为部分满足状态 PS,则其达到完全满足状态 SA 是允许的.

3) 状态约束公式

在某些特殊情况下,没有初始状态的结点也有可能需要约束其状态,此时,用状态约束公式定义这些结点的约束状态.状态约束公式定义为

$$\phi_{Constraint}::=\wedge_{n\in N}(LL(n)|\vee_{n\in N}LL(n)) \tag{23}$$

其中,$LL(n)::=L(n)|\neg L(n),\neg L(n)\mapsto\{\neg SA(n),\neg PS(n),\neg PD(n),\neg DE(n)\}$.

4) 可接受矛盾等级公式

在进行可满足性求解过程中,完全无矛盾推出可满足结论是最佳结果,但这属于理想情况.现实情况是:在进行可满足性求解过程中,往往不可避免地会出现矛盾.此时,在 SAT 公式中运用矛盾等级公式[26]定义 3 种不同等级的矛盾.如果建模者不允许任何矛盾存在,则公式定义为

$$\phi_{Inconsistency}::=\wedge_{n\in N}(\neg(PS(n)\wedge PD(n))) \tag{24}$$

如果建模者仅允许存在弱矛盾,不允许中等矛盾和强矛盾存在,则公式定义为

$$\phi_{Inconsistency}::=\wedge_{n\in N}(\neg(SA(n)\wedge PD(n))\wedge\neg(PS(n)\wedge DE(n))) \tag{25}$$

如果建模者仅允许存在弱矛盾和中等矛盾,不允许存在强矛盾,则公式定义为

$$\phi_{Inconsistency}::=\wedge_{n\in N}(\neg(SA(n)\wedge DE(n))) \quad (26)$$

3.3.2 可信软件非功能需求驱动的过程策略选取

为了实现上述可满足性问题求解,本文基于最常用的可满足性问题求解开源工具 zChaff[52],提出可满足性推理算法,并开发工具 TACD(trustworthiness attributes collaboration diagnosis)辅助推理.

可满足性推理算法首先将输入的关系模型、初始状态、约束状态以及可接受矛盾等级生成 SAT 公式,然后使用 Choe 提供的 prop2cnf.py[53]脚本将 SAT 公式转换为 CNF,输入 zChaff 进行求解.

- 如果求解结果满足,则将所有过程策略对应的结点状态标记为真;
- 如果求解结果为不满足,则将出现矛盾的子句输出,由建模者判断矛盾是否可以接受:如果可以接受,则修改可接受矛盾等级公式或状态约束公式,重新生成 SAT 公式输入 zChaff 求解;如果不允许矛盾存在,则去除相应过程策略,重新生成 SAT 公式输入 zChaff 求解.该交互过程将一直进行,直至所有矛盾都解决,最终得到可满足的结果.

算法 1. 可信软件非功能需求可满足性求解算法 Satisfiability.

找出满足可信软件非功能需求的过程策略的状态标记 $L(n_i),(i=1,\ldots,k)$

输入:$M=(N,R)$;

输出:结点初始状态 $Initial\_Status(n_i)$、结点求解后状态 $SAT\_Status(n_i)$、结点矛盾 $Inconsistency(n_i)$.

```
BEGIN
  k_T=|T|; k_TA=|TA|; k_S=|S|;
  FOR i:=1 TO k_T+k_TA+k_S DO   /*初始状态*/
    设置结点初始状态 Initial_Status(n_i);
  φ:=MakeSAT(M);   /*生成 SAT 公式*/
  CNF:=prop2cnf.py(φ);   /*调用 prop2cnf.py,将 SAT 公式转换为 CNF*/
  satisfy:=zCHaff(CNF);   /*调用 zChaff */
  WHILE satisfy:=UNSAT DO   /*公式不满足*/
    BEGIN
      输出矛盾子句;
      建模者解决矛盾;
      IF 允许矛盾存在 THEN
        BEGIN
          设置求解后状态 SAT_Status(n_i);
          输出结点矛盾;
          修改 SAT 公式中的可接受矛盾等级公式或状态约束公式;
          φ:=MakeSAT(M);
          CNF:=prop2cnf.py(φ);
          satisfy:=zCHaff(CNF);
        END
      ELSE
        BEGIN
          设置求解后状态 SAT_Status(n_i);
          修改初始状态公式或者删除产生矛盾的结点 n_i;
          φ:=MakeSAT(M);
          CNF:=prop2cnf.py(φ);
```

satisfy:=*zCHaff*(*CNF*);

END

END;

以表格形式列出所有结点的状态标记(包括推理前和推理后)和矛盾;

END.

基于上述可满足性求解算法,我们开发了一个推理辅助工具 TACD 以辅助完成过程策略的选取,有关工具实现的效果见下一节.

4 案例分析

本文以我们参与的一个可信第三方认证中心软件项目为例,应用本文提出的方法和工具进行案例分析.认证中心提供网络身份认证服务,负责签发和管理数字证书,是一个具有权威性和公正性的第三方可信机构,是所有网上安全活动的核心环节.认证中心软件作为可信第三方认证中心的核心软件,负责完成认证中心提供的所有服务.对任何个人或企业、甚至一个地区来说,一个安全和值得信赖的网络环境依赖于认证中心软件的可信性,因此,认证中心软件属于可信软件.

4.1 可信软件非功能需求获取

认证中心软件非功能需求获取阶段的第 1 步是确定利益相关者,我们选取认证中心相关 5 名利益相关者(S_1:认证中心监管部门,S_2:工程方,S_3:认证中心专家,S_4:软件操作部门,S_5:证书持有者)对认证中心软件进行非功能需求重要程度的评估,评估数据以表格的形式收集,见表 4.

Table 4 Evaluation data for non-functional requirements of certificate authority software

表 4 认证中心软件非功能需求评估数据

非功能需求	S_1	S_2	S_3	S_4	S_5	模糊排序值
R_1 功能适用性	AH	AH	AH	AH	AH	0.942
R_{11} 功能完整性	AH	AH	AH	AH	AH	0.942
R_{12} 功能正确性	AH	AH	AH	AH	AH	0.942
R_2 防危性(safety)	AL	AL	AL	AL	L	0.085
R_3 可靠性	AH	AH	AH	AH	H	0.912
R_{31} 可用性	H	AH	H	AH	H	0.898
R_{32} 容错性	FH	AH	FH	AH	H	0.815
R_{33} 可恢复性	AH	AH	H	AH	H	0.922
R_4 安全性	AH	AH	AH	AH	AH	0.942
R_{41} 可追踪性	AH	AH	AH	AH	H	0.915
R_{42} 机密性	AH	AH	AH	AH	AH	0.942
R_{43} 完整性	AH	AH	AH	AH	AH	0.942
R_{44} 不可否认性	AH	AH	AH	AH	AH	0.942
R_{45} 真实性	AH	AH	AH	AH	AH	0.942
R_5 精确性	AL	AL	AL	AL	L	0.085
R_6 可维护性	H	AH	H	H	H	0.834
R_{61} 可测试性	M	FH	L	FL	FL	0.408
R_{62} 可重用性	M	FH	M	FL	FL	0.469
R_{63} 可修改性	M	FH	L	FL	FL	0.408
R_{64} 模块化	FH	FH	H	H	FL	0.715
R_{65} 可扩展性	H	H	H	H	M	0.746
R_7 性能	M	M	M	FH	M	0.531
R_{71} 时间性能	M	M	M	FH	M	0.531
R_{72} 空间性能	M	M	M	FH	M	0.531
R_8 易用性	FH	FH	M	H	M	0.623
R_{81} 易操作性	FH	H	M	H	FH	0.684
R_{82} 易识别性	FH	H	FH	H	M	0.684

Table 4 Evaluation data for non-functional requirements of certificate authority software (continued)

表 4 认证中心软件非功能需求评估数据(续)

非功能需求	S_1	S_2	S_3	S_4	S_5	模糊排序值
$\boldsymbol{R_9}$ **兼容性**	H	H	H	H	H	0.808
R_{91} 可生存性	M	FH	FH	H	H	0.684
R_{92} 交互性	H	H	H	H	AH	0.834
$\boldsymbol{R_{10}}$ **可移植性**	AL	AL	L	L	M	0.200
R_{101} 自适应性	AL	AL	L	L	M	0.200
R_{102} 易安装性	AL	AL	L	L	M	0.200
$\boldsymbol{R_{11}}$ **隐私**	AL	AL	AL	L	M	0.173

为检验各个利益相关者所提供的评估数据是否有效,我们计算其熵值,见表 5,其中,利益相关者 S_1、S_2、S_3 和 S_4 的熵值相对一致;而 S_5 的熵值相对较大,一定程度上表明其对于可信软件的需求相对不够明确,根据项目组需要,可以要求重新提供评估数据.

Table 5 Information entropy values for stakeholders' evaluation data

表 5 利益相关者评估数据熵值

利益相关者	熵值	模糊排序值
S_1	(0.445,0.218,0.141,0.088)	0.223
S_2	(0.387,0.161,0.088,0.045)	0.17
S_3	(0.472,0.28,0.169,0.099)	0.255
S_4	(0.453,0.248,0.119,0.052)	0.218
S_5	(0.684,0.426,0.233,0.127)	0.368

在确认评估数据有效后,各项非功能需求的模糊排序值见表 5 中最后一列所示.通过该排序值,我们将模糊排序值大于 0.7 的非功能需求确定为可信关注点,余下有相关关系的非功能需求确定为软目标,并且将模糊排序值大于 0.9 的可信关注点设定初始状态为完全满足状态,而将模糊排序值小于 0.9 的可信关注点以及软目标设定为部分满足状态.

4.2 可信软件过程策略选取

基于非功能需求与过程策略库,使用 TRMM 建模得到认证中心软件的关系模型,如图 8 所示,其对应的公式为

$$\begin{aligned}\phi_{Model} ::= {} & (SA(T_1)\rightarrow SA(T_{11}))\wedge(SA(T_{11})\rightarrow SA(ta_{111}))\wedge(SA(T_{11})\rightarrow SA(ta_{112}))\wedge(SA(T_{11})\rightarrow SA(ta_{113}))\wedge\\ & (SA(T_1)\rightarrow SA(T_{12}))\wedge(SA(T_{12})\rightarrow SA(ta_{121}))\wedge(SA(T_{12})\rightarrow SA(ta_{122}))\wedge(PS(T_2)\rightarrow PS(ta_{21}))\wedge\\ & (PS(T_2)\rightarrow PS(ta_{22}))\wedge(PS(T_2)\rightarrow PS(ta_{23}))\wedge(SA(T_3)\rightarrow SA(ta_{31}))\wedge(SA(ta_{31})\rightarrow SA(ta_{313}))\wedge\\ & (SA(T_3)\rightarrow SA(ta_{32}))\wedge(SA(ta_{32})\rightarrow SA(ta_{321}))\wedge(SA(ta_{32})\rightarrow SA(ta_{322}))\wedge(SA(ta_{32})\rightarrow SA(ta_{323}))\wedge\\ & (SA(ta_{32})\rightarrow SA(ta_{324}))\wedge(SA(T_3)\rightarrow SA(ta_{33}))\wedge(SA(ta_{33})\rightarrow SA(ta_{331}))\wedge(SA(ta_{33})\rightarrow SA(ta_{332}))\wedge\\ & (SA(ta_{33})\rightarrow SA(ta_{333}))\wedge(SA(T_3)\rightarrow SA(T_{31}))\wedge(SA(T_{31})\rightarrow SA(ta_{311}))\wedge(SA(T_{31})\rightarrow SA(ta_{312}))\wedge\\ & (SA(T_3)\rightarrow SA(T_{32}))\wedge(SA(T_{32})\rightarrow SA(ta_{325}))\wedge(SA(T_{32})\rightarrow SA(ta_{326}))\wedge(SA(T_{32})\rightarrow SA(ta_{327}))\wedge\\ & (SA(T_3)\rightarrow SA(T_{33}))\wedge(SA(T_{33})\rightarrow SA(ta_{334}))\wedge(SA(T_{33})\rightarrow SA(ta_{335}))\wedge(SA(T_4)\rightarrow SA(T_{41}))\wedge\\ & (SA(T_{41})\rightarrow SA(ta_{411}))\wedge(SA(T_4)\rightarrow SA(ta_{41}))\wedge(SA(T_4)\rightarrow SA(ta_{42}))\wedge(SA(ta_{42})\rightarrow SA(ta_{421}))\wedge\\ & (SA(ta_{42})\rightarrow SA(ta_{422}))\wedge(SA(ta_{42})\rightarrow SA(ta_{423}))\wedge(SA(ta_{42})\rightarrow SA(ta_{424}))\wedge(SA(T_4)\rightarrow SA(ta_{43}))\wedge\\ & (SA(ta_{43})\rightarrow SA(ta_{431}))\wedge(SA(ta_{43})\rightarrow SA(ta_{432}))\wedge(SA(ta_{43})\rightarrow SA(ta_{433}))\wedge(SA(ta_{43})\rightarrow SA(ta_{434}))\wedge\\ & (SA(T_4)\rightarrow SA(ta_{44}))\wedge(SA(ta_{44})\rightarrow SA(ta_{441}))\wedge(SA(T_4)\rightarrow SA(ta_{45}))\wedge(PS(T_5)\rightarrow PS(ta_{51}))\wedge\\ & (PS(ta_{51})\rightarrow PS(ta_{511}))\wedge(PS(ta_{51})\rightarrow PS(ta_{512}))\wedge(PS(T_5)\rightarrow PS(ta_{52}))\wedge(SA(T_1)\rightarrow DE(ta_{327}))\wedge\\ & (SA(T_4)\rightarrow DE(ta_{335}))\wedge(SA(T_4)\rightarrow DE(ta_{327}))\wedge(SA(T_4)\rightarrow DE(ta_{21}))\wedge(SA(T_5)\rightarrow DE(ta_{335}))\wedge\\ & (SA(T_5)\rightarrow DE(ta_{41}))\wedge(SA(T_5)\rightarrow DE(ta_{21}))\wedge(PS(S_1)\rightarrow DE(ta_{41}))\wedge(PS(S_1)\rightarrow DE(ta_{411}))\wedge\\ & (PS(S_1)\rightarrow DE(ta_{335}))\wedge(PS(S_2)\rightarrow DE(ta_{41}))\wedge(PS(S_2)\rightarrow DE(ta_{411})).\end{aligned}$$

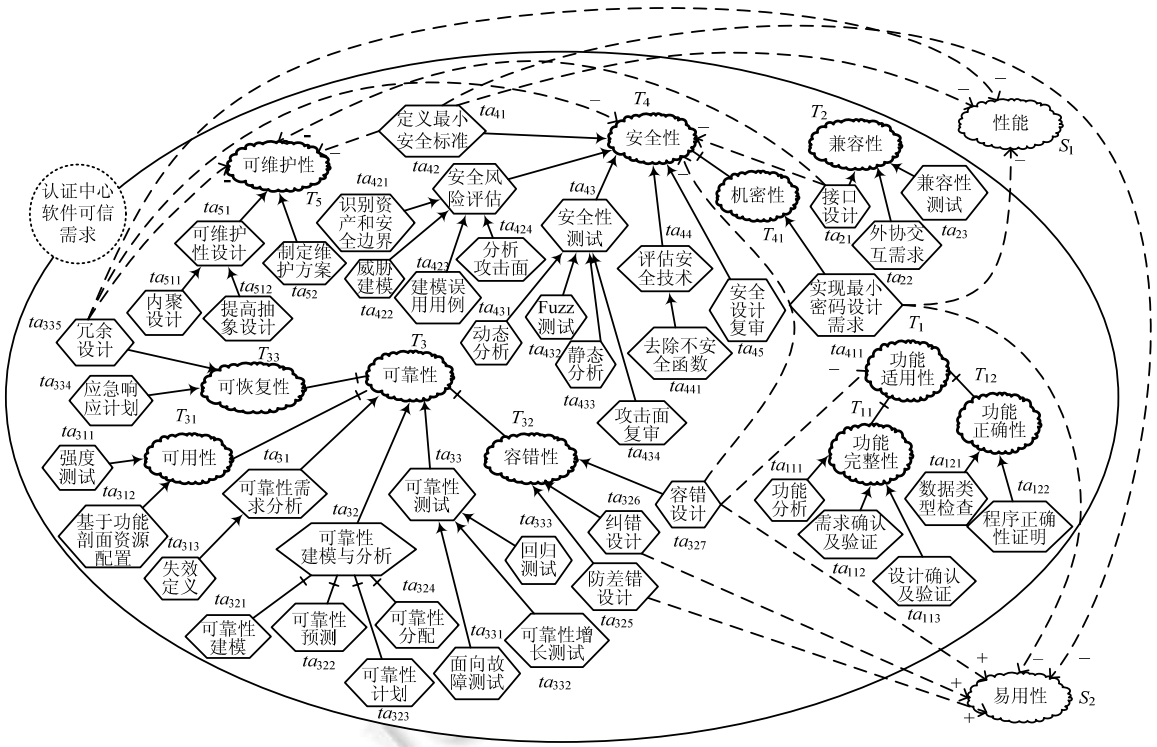


Fig.8 Relation model for certificate authority software

图 8 认证中心软件关系模型

另外,设定初始状态公式:

$$\phi_{Initial}::=SA(T_1)\wedge PS(T_2)\wedge SA(T_3)\wedge SA(T_4)\wedge PS(T_5)\wedge PS(S_1)\wedge PS(S_2).$$

状态约束公式:

$$\phi_{Constraint}::=(SA(ta_{325})\vee SA(ta_{327}))\wedge(SA(ta_{326})\vee SA(ta_{327})).$$

可接受矛盾等级公式:

$$\phi_{Inconsistency}::=\neg(SA(n)\wedge DE(n)),n \text{ 为模型中所有结点.}$$

最终得到推理公式为

$$\phi::=\phi_{Model}\wedge\phi_{Initial}\wedge\phi_{Constraint}\wedge\phi_{Inconsistency}.$$

将公式ϕ输入 TACD 工具,推理后得到如图 9 所示的结果.

推理结果显示公式ϕ不满足,矛盾子式为第 28 个子式 $SA(T_{32})\rightarrow SA(ta_{327})$,即:“容错设计”这项过程策略与我们的初始需求矛盾,在结点状态表中记录矛盾.修改公式后,继续输入 TACD 工具进行需求推理,推理结果显示:在$\neg(SA(ta_{335})\wedge DE(ta_{335}))$,$\neg(SA(ta_{41})\wedge DE(ta_{41}))$和$\neg(SA(ta_{411})\wedge DE(ta_{411}))$子句有矛盾,分别对应“冗余设计”、“定义最小安全标准”和“实现最小密码设计需求”5 项过程策略.同样,记录矛盾后再次修改公式,输入 TACD 工具进行需求推理,推理结果显示公式ϕ满足,如图 10 所示.

此时,结点状态表 6 记录了模型中结点的满足状态和矛盾状态(为描述简洁,删除了无矛盾且为满足状态的过程策略).

经过对这 5 项存在矛盾的过程策略的研究,我们将“容错设计”去除,因为“容错设计”严重损害了功能适用性和安全性,而“纠错设计”和“防差错设计”可以在一定程度上满足容错性.余下的 4 项过程策略则保留,但需要按照过程策略中定义的依赖关系织入软件过程.至此,我们得到满足非功能需求的所有过程策略.

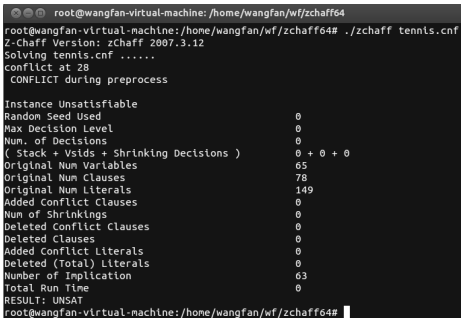


Fig.9 First reasoning result of certificate authority software

图 9 认证中心软件第 1 次推理结果

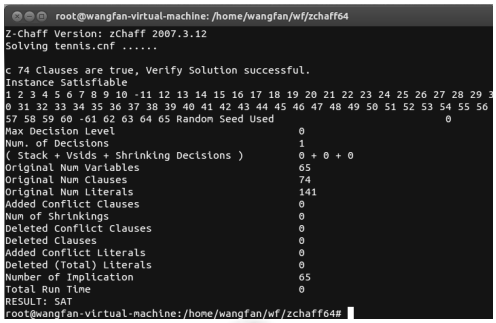


Fig.10 Final reasoning result of certificate authority software

图 10 认证中心软件最终推理结果

Table 6 Statuses of nodes

表 6 结点状态表

结点类型	结点	初始状态	推理后状态	矛盾
可信关注点	功能适用性	*SA*		
	功能完整性		*SA*	
	功能正确性		*SA*	
	兼容性	*PS*		
	可靠性	*SA*		
	可用性		*SA*	
	容错性		*SA*	
	可恢复性		*SA*	
	安全性	*SA*		
	机密性		*SA*	
	可维护性	*PS*		
软目标	性能	*PS*		
	易用性	*PS*		
过程策略	接口设计		***PS*/*DE***	非强矛盾
	冗余设计		***SA*/*DE***	可维护性,安全性,性能
	容错设计		***SA*/*DE***	功能适用性,安全性
	定义最小安全标准		***SA*/*DE***	可维护性,性能,易用性
	实现最小密码设计需求		***SA*/*DE***	性能,易用性

5 结论与展望

统计表明,非形式化的软件工程技术对软件质量的保证具有一个难以逾越的顶点,而形式化方法的实践证明形式化方法是提高软件质量的重要途径.在从高层规范至最终实现的过程中,选用适当的、以形式化方法为基础的工具进行辅助设计和验证,对提高软件的可信度是有很大帮助的.因此,本文针对可信软件,在早期需求工程阶段对可信软件非功能需求进行形式化建模与推理,找出满足可信软件非功能需求的过程策略,为可信软件的软件过程建模奠定基础.

为达到此目的,本文首先基于可信软件非功能需求已有相关研究成果,提出可信软件非功能需求分解模型;在此基础上,使用模糊集合论和信息熵方法提出可信软件非功能需求获取方法,该获取方法保证非功能需求数据的有效性;然后提出基于 TRMM 和知识库的可信软件非功能需求与过程策略关系建模方法,其中,将可信关注点和软目标分离有利于针对可信软件的可信性需求提出过程策略,同时有利于非功能需求之间复杂相关关系的研究;之后,为权衡可信关注点以及软目标,本文基于可满足性问题求解方法提出选取过程策略的推理方法,并开发推理辅助工具 TACD,找出满足可信软件非功能需求的过程策略;最后,以一个实际的可信第三方认证中心软件项目为例,应用本文提出的方法和工具进行案例分析.案例分析结果表明:使用模糊数可以有效地帮助

软件利益相关者描述其需要可信软件达到的状态,而使用信息熵能够帮助项目组检查利益相关者描述需求数据的有效性及客观性.将这两种方法相结合,可以有效地从利益相关者处获得可信软件的可信需求.另外,通过可满足性问题求解方法实现向后推理,可以有效地通过可信需求获取过程策略.在推理过程中建立推理状态表,可以清晰地展现可信需求与过程策略之间的关系,为过程策略的选取提供了有效的途径.

目前,本文的方法在两个方面还需进一步开展相关研究工作:一是本文对可信软件需求进行分析时,仅将需求分解为功能需求和非功能需求,没有考虑有些需求兼有功能和非功能需求的特性[54,55],下一步,我们将进一步细化需求描述.另外,本文的方法只能找到满足软件利益相关者的一组过程策略,而可行的过程策略组合方案可以有多个,本文未考虑选取最优过程策略组合的方法.Elahi 和 Yu 在其论文[56]中使用相等交换多标准决策分析方法来取舍需求以获得最优需求.Burgess 等人通过改进 NFR 框架定义依赖规则集,通过自动传播状态获取满足特定非功能需求集合的最优功能[57].Yin 等人使用 0-1 规划方法寻找满足非功能需求的功能实现最优方案[58].我们下一步将开展最优过程策略组合方法的研究.

References:

[1] Liu K, Shan ZG, Wang J, He JF, Zhang ZT, Qin YW. Overview on major research plan of trustworthy software. Science Foundation of China, 2008,22(3):145−151 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-8217.2008.03.005]

[2] Wang HM, Liu XD, Lang B, Xie B, Mao XG. Software trustworthiness classification specification (TRUSTIE-STC v2.0). Technical Report, School of Computer Science and Engineering, Beihang University, *et al*., 2009 (in Chinese).

[3] Amoroso E, Taylor C, Watson J, Weiss J. A process-oriented methodology for assessing and improving software trustworthiness. In: Proc. of the ACM Conf. on Computer and Communications Security (CCS). 1994. 39−50. [doi: 10.1145/191177.191188]

[4] Jin Z, Liu L, Jin Y. Software Requirements Engineering: Principles and Methods. Beijing: Science Press, 2008 (in Chinese).

[5] Boehm B, In H. Identifying quality-requirement conflicts. Software, 1996,13(2):25−35. [doi: 10.1109/52.506460]

[6] Tao HW. Research on the measurement models of software trustworthiness based on attributes [Ph.D. Thesis]. Shanghai: East China Normal University, 2011 (in Chinese with English abstract).

[7] Moser T, Winkler D, Heindl M, Biffl S. Requirements management with semantic technology: An empirical study on automated requirements categorization and conflict analysis. In: Proc. of the Advanced Information Systems Engineering. LNCS 6741, 2011. 3−17. [doi: 10.1007/978-3-642-21640-4_3]

[8] Mairiza D, Zowghi D. Constructing a catalogue of conflicts among non-functional requirements. In: Proc. of the Evaluation of Novel Approaches to Software Engineering, Communications in Computer and Information Science, Vol.230. Springer-Verlag, 2011. 31−44. [doi: 10.1007/978-3-642-23391-3_3]

[9] Boehm B, Bose P, Horowitz E, Lee M. Software requirements as negotiated win conditions. In: Proc. of the ICRE'94. IEEE Computer Society Press, 1994. 74−83. [doi: 10.1109/ICRE.1994.292400]

[10] Boehm B, Bose P, Horowitz E, Lee M. Software requirements negotiation and renegotiation aids: A theory—W based spiral approach. In: Proc. of the ICSE'95. New York: IEEE Computer Society Press, 1995. 243.

[11] Boehm B, Egyed A, Port D, Shah A, Kwan J, Madachy R. A stakeholder win-win approach to software engineering education. Annals of Software Engineering, 1998,6(1-4):295−321. [doi: 10.1023/A:1018988827405]

[12] Robinson W, Volkov S. A meta-model for restructuring stakeholder requirements. In: Proc. of the 19th Int'l Conf. on Software Engineering. Botson: IEEE Computer Society Press, 1997. 140−149. [doi: 10.1145/253228.253255]

[13] In HP, Olson D, Rodgers T. Multi-Criteria preference analysis for systematic requirements negotiation. In: Proc. of the COMPSAC 2002. 2002. 887−892. [doi: 10.1109/CMPSAC.2002.1045118]

[14] In HP, Olson D. Requirements negotiation using multi-criteria preference analysis. Journal of Universal Computer Science, 2004, 10(4):306−325.

[15] 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]

[16] van Lamsweerde A, Darimont R, Letier E. Managing conflicts in goal-driven requirements engineering. IEEE Trans. on Software Engingeering, 1998,24(1):908−926. [doi: 10.1109/32.730542]

[17] Mylopoulos J, Chung L, Nixon B. Representing and using nonfunctional requirements: A process-oriented approach. IEEE Trans. on Software Engineering, 1992,18(6):483−497. [doi: 10.1109/32.142871]

[18] Chung L, Nixon BA. Dealing with non-functional requirements: Three experimental studies of a process-oriented approach. In: Proc. of the 17th Int'l Conf. on Software Engineering (ICSE). New York: ACM, 1995. 25−25.

[19] Chung L, Nixon BA, Yu E, Mylopoulos J. Non-Functional requirements in software engineering. In: Victor RB, ed. Proc. of the Int'l Series in Software Engineering. New York: Springer-Verlag, 1999. 476.

[20] Chung L, do Prado Leite JCS. On non-functional requirements in software engineering. In: Proc. of the Conceptual Modeling: Foundations and Applications. Berlin, Heidelberg: Springer-Verlag, 2009. 363−379. [doi: 10.1007/978-3-642-02463-4_19]

[21] Yu E. Towards modeling and reasoning support for early-phase requirements engineering. In: Proc. of the 3rd IEEE Int'l Symp. on Requirements Engineering. 1997. 226−235. [doi: 10.1109/ISRE.1997.566873]

[22] Castro J, Kolp M, Mylopoulos J. Towards requirements-driven information systems engineering: The Tropos project. Information Systems, 2002,27(6):365−389. [doi: 10.1016/S0306-4379(02)00012-1]

[23] Amyot D, Mussbacher G. URN: Towards a new standard for the visual description of requirements. In: Proc. of the Telecommunications and Beyond: The Broader Applicability of SDL and MSC. Berlin, Heidelberg: Springer-Verlag, 2003. 21−37. [doi: 10.1007/3-540-36573-7_2]

[24] Zhu MX, Luo XX, Chen XH, Wu DD. A non-functional requirements tradeoff model in trustworthy software. Information Science, 2012,191:61−75. [doi: 10.1016/j.ins.2011.07.046]

[25] Wei B, Jin Z, Zowghi D, Yin B. Automated reasoning with goal tree models for software quality requirements. In: Proc. of the 2012 IEEE 36th Int'l Conf. on Computer Software and Applications Workshops (COMPSACW). 2012. 373−378. [doi: 10.1109/COMPSACW.2012.73]

[26] Sebastiani R, Giorgini P, Mylopoulos J. Simple and minimum-cost satisfiability for goal models. In: Proc. of the Advanced Information Systems Engineering. Berlin, Heidelberg: Springer-Verlag, 2004. 20−35. [doi: 10.1007/978-3-540-25975-6_4]

[27] Giorgini P, Mylopoulos J, Sebastiani R. Goal-Oriented requirements analysis and reasoning in the tropos mehtodology. Engineering Applications of Artificial Intelligence, 2005,18(2):159−171. [doi: 10.1016/j.engappai.2004.11.017]

[28] Horkoff J, Yu E. Finding solutions in goal models: an interactive backward reasoning approach. In: Proc. of the Conceptual Modeling (ER 2010). Berlin, Heidelberg: Springer-Verlag, 2010. 59−75. [doi: 10.1007/978-3-642-16373-9_5]

[29] Horkoff J. Iterative, interactive analysis of agent-goal models for early requirements engineering [Ph.D. Thesis]. University of Toronto, 2012.

[30] DoD (department of defense). Department of defense trusted computer system evaluation criteria (TCSEC). DoD 5200.28-STD. 1985. http://www.cerberussystems.com/INFOSEC/stds/d520028.htm

[31] IEC. Int'l electrotechnical vocabulary—Chapter 191: Dependability. IEC 60050-191 Ed.2.0, 1990.

[32] Howard M, Leblanc D. Writing Secure Code. Microsoft Press, 2002.

[33] Howard M, Lipner S. The Secure Development Life-cycle. Microsoft Press, 2006.

[34] Trusted Computing Group (TCG). TCG specification architecture overview. Revision 1.4. 2007. http://www.Trustedcomputing group.org

[35] Littlewood B, Strigine L. Software reliability and dependability: A roadmap. In: Proc. of the Conf. on the Future of Software Engineering. IEEE, 2000. 175−188. [doi: 10.1145/336512.336551]

[36] Schmidt H. Trustworthy components—Compositionality and prediction. The Journal of Systems and Software, 2003,65:215−225. [doi: 10.1016/S0164-1212(02)00045-6]

[37] Neumann PG. Principled assuredly trustworthy composable architectures. Project Report, Computer Science Laboratory, SRI Int'l, 2004.

[38] NSS2. Software 2015: A national software strategy to ensure U.S. security and competitiveness. 2005. http://www.cnsoftware.org/nss2report/

[39] Bernstein L, Yuhas C. Trustworthy Systems Through Quantitative Software Engineering. Vol.1, New York: Wiley-IEEE Computer Society Press, 2005.

[40] Hasselbring W, Reussner R. Toward trustworthy software systems. Computer, 2006,39(4):91−92. [doi: 10.1109/MC.2006.142]

[41] Miller A, Mclean J, Saydjari O, Voas J. Compsac panel session on trustworthy computing. In: Proc. of the 30th Annual Int'l Computer Software and Applications Conf. (COMPSAC 2006), Vol.1. IEEE Computer Society, 2006. [doi: 10.1109/COMPSAC.2006.36]

[42] Yang Y, Wang Q, Li MS. Process trustworthiness as a capability indicator for measuring and improving software trustworthiness. In: Proc. of the Int'l Conf. on Software Process (ICSP 2009). Vancouver: Springer-Verlag, 2009. 389−401. [doi: 10.1007/978-3-642-01680-6_35]

[43] Trustie. Software trustworthiness classification specification (TRUSTIE-STC v 1.0). 2009. http://www.trustie.net/

[44] ISO, IEC. ISO/IEC 25010: Systems and software engineering—Systems and software quality requirements and evaluation (SQuaRE)—System and software quality models. 2011.

[45] In HP, Olson D. Requirements negotiation using multi-criteria preference analysis. Journal of Universal Computer Science, 2004, 10(4):306−325. [doi: 10.3217/jucs-010-04-0306]

[46] Zadeh L. A fuzzy sets. Information and Control, 1965,8(3):338−353. [doi: 10.1016/S0019-9958(65)90241-X]

[47] Hu BQ. Fuzzy Theroy (V2). Wuhan: Wuhan University Press, 2010 (in Chinese).

[48] Chen SM, Sanguansat K. Analyzing fuzzy risk based on a new fuzzy ranking method between generalized fuzzy numbers. Expert Systems with Applications, 2011,38:2163−2171. [doi: 10.1016/j.eswa.2010.08.002]

[49] Zhang X, Liao HZ, Li T, Xu J, Zhang QR, Qian Y. Software security measurement based on information entropy and attack surface. Journal of Computer Applications, 2013,33(1):19−22,48 (in Chinese with English abstract). [doi: 10.3724/SP.J.1087.2013.00019]

[50] Zhang X, Li T, Xie ZW, Dai F, Liu JZ. A Petri net based model for trustworthy software process composition. In: Proc. of the 37th IEEE Computer Software and Applications Conf. on Workshop (COMPSACW). 2013. 108−114. [doi: 10.1109/COMPSACW.2013.37]

[51] Nuseibeh B, Easterbrook S. Requirements engineering: A roadmap. In: Proc. of the Conf. on the Future of Software Engineering. ACM Press, 2000. 35−46. [doi: 10.1145/336512.336523]

[52] Princeton University. zChaff 2007.3.12. 2007. http://www.princeton.edu/~chaff /zchaff.html

[53] Choe Y. prop2chf.py. CSCE 625: Introduction to machine learning. 2011. http://faculty.cs.tamu.edu/ioerger/cs625-fall11/prop2cnf.py

[54] Li FL, Horkoff J, Mylopoulos J, Guizzardi RS, Guizzardi G, Borgida A, Liu L. Non-Functional requirements as qualities, with a spice of ontology. In: Proc. of the IEEE 22nd Int'l Requirements Engineering Conf. 2014. 293−302. [doi: 10.1109/RE.2014.6912271]

[55] Guizzardi RS, Li FL, Borgida A, Uizzardi GI, Horkoff J, Mylopoulos J. An ontological interpretation of non-functional requirements. In: Proc. of the FOIS. 2014. 344−357. [doi: 10.3233/978-1-61499-438-1-344]

[56] Elahi G, Yu E. A semi-automated decision support tool for requirements trade-off analysis. In: Proc. of the 35th IEEE Annual Computer Software and Application Conf. (COMPSAC'35). 2011. 466−475. [doi: 10.1109/COMPSAC.2011.67]

[57] Burgess C, Krishna A, Jiang L. Towards optimising non-functionalrequirements. In: Proc. of the Int'l Conf. on Quality Software. 2009. 269−277. [doi: 10.1109/QSIC.2009.42]

[58] Yin B, Jin Z, Zhang W, Zhao HY, Wei B. Finding optimal solution for satisficing non-functional requirements via 0-1 programming. In: Proc. of the COMPSAC 2013. 2013. 415−424. [doi: 10.1109/COMPSAC.2013.69]

附中文参考文献:

[1] 刘克,单志广,王戟,何积丰,张兆田,秦玉文."可信软件基础研究"重大研究计划综述.中国科学基金,2008,22(3):145−151. [doi: 10.3969/j.issn.1000-8217.2008.03.005]

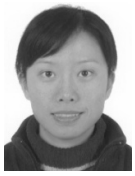
[2] 王怀民,刘旭东,郎波,谢冰,毛晓光.软件可信分级规范 v2.0.技术报告,北京航空航天大学计算机学院,等,2009.

[4] 金芝,刘璘,金英.软件需求工程:原理和方法.北京:科学出版社,2008.

[6] 陶红伟.基于属性的软件可信性度量模型研究[博士学位论文].上海:华东师范大学,2011.

[47] 胡宝清.模糊理论基础.第 2 版,武汉:武汉大学出版社,2010.

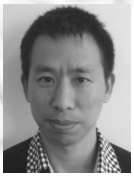
[49] 张璇,廖鸿志,李彤,徐晶,张倩茹,钱晔.基于信息熵和攻击面的软件安全度量.计算机应用,2013,33(1):19−22,48. [doi: 10.3724/SP.J.1087.2013.00019]



张璇(1978－),女,江苏南京人,博士,副教授,CCF 会员,主要研究领域为需求工程,软件过程,可信软件.

于倩(1975－),女,博士,讲师,CCF 会员,主要研究领域为软件过程,自动化技术,分布式计算.

李彤(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件过程,软件工程.



郁湧(1980－),男,博士,副教授,CCF 会员,主要研究领域为可信软件,软件工程.

王旭(1976－),男,博士,讲师,主要研究领域为金融安全,计量经济学.

朱锐(1987－),男,博士生,CCF 学生会员,主要研究领域为软件过程,数据挖掘.