软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(1):23−41 [doi: 10.13328/j.cnki.jos.005323]

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

基于模型的自适应方法综述*

赵天琪[1,2], 赵海燕[1,2], 张 伟[1,2], 金 芝[1,2]

[1](高可信软件技术教育部重点实验室(北京大学),北京 100871)

[2](北京大学 信息科学技术学院 软件研究所,北京 100871)

通讯作者: 赵海燕, E-mail: zhhy.sei@pku.edu.cn

摘 要: 自适应为管理现代软件系统的复杂性提供了有效的解决方案,被设计为自适应系统的软件能够持续地演化以应对环境中的不确定性.在现有的研究工作中,基于模型的自适应方法是一类广泛使用的方法,它将模型驱动工程技术的应用从设计时扩展到运行时,以支持自适应能力的实现.通过利用软件模型对运行时丰富和不确定的信息进行管理,这类方法避免了将自适应逻辑与程序语言交织带来的复杂性,从而简化了自适应系统的开发.对近些年来国内外学者在该研究领域取得的成果进行了系统总结.首先给出了 6 个研究问题,包括相关工作常用的需求模型、结构模型、行为模型、环境模型、模型与模型或模型与系统间的同步方式、自适应规划算法等;然后,依次总结了相关工作在这 6 个研究问题上的已有研究成果;最后,对未来研究可能面临的挑战进行了展望.

关键词: 自适应软件;模型驱动的软件工程;基于模型的自适应;自适应规划方法.

中图法分类号: TP311

中文引用格式: 赵天琪,赵海燕,张伟,金芝.基于模型的自适应方法综述.软件学报,2018,29(1):23−41. http://www.jos.org.cn/1000-9825/5323.htm

英文引用格式: Zhao TQ, Zhao HY, Zhang W, Jin Z. Survey of model-based self-adaptation methods. Ruan Jian Xue Bao/Journal of Software, 2018,29(1):23−41 (in Chinese). http://www.jos.org.cn/1000-9825/5323.htm

Survey of Model-Based Self-Adaptation Methods

ZHAO Tian-Qi[1,2], ZHAO Hai-Yan[1,2], ZHANG Wei[1,2], JIN Zhi[1,2]

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

[2](Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

Abstract: Self-Adaptation provides a promising approach to managing the complexity of modern software systems, and in particular, to enabling systems to continuously adapt themselves to uncertainty in the environment. In existing works, model-based self-adapatation is a type of widely used methods that extend the applicability of model-driven engineering techniques to the runtime environment to facilitate self-adaptation. By leveraging software models to manage the abundant information associated with runtime phenomena, model-based self-adaptation methods are able to avoid the complexity introduced by intertwining adaptation and application behaviors and therefore simplify the development of self-adaptive systems. This paper first lays out six research questions from existing studies concerning system models, environment models, relationship among models, and model-based adaptation planning methods. It then answers the research questions by investigating relevant studies. Finally, the paper offers some suggestions for future research.

Key words: self-adaptive software; model-driven software engineering; model-based self-adaptation; self-adaptation planning method

软件正在变得越来越复杂,导致软件复杂度不断上升的一个重要原因是软件运行环境及软件自身所具有

* 基金项目: 国家自然科学基金(61620106007, 61690201, 61272163); 国家重点基础研究发展计划(973)(2015CB352201)

Foundation item: National Natural Science Foundation of China (61620106007, 61690201, 61272163); National Basic Research Program of China (973) (2015CB352201)

收稿时间: 2017-02-03; 修改时间: 2017-03-23; 采用时间: 2017-05-30; jos 在线出版时间: 2017-07-12

CNKI 网络优先出版: 2017-07-12 16:17:07, http://kns.cnki.net/kcms/detail/11.2560.TP.20170712.1617.019.html

的动态性和不确定因素的持续增长.为了让软件在环境变化后仍能持续满足用户需求,一种有效的解决方案是将软件设计为自适应系统.自适应系统是指能够监测环境和系统自身的变化,并可以动态修改自身行为和结构以响应变化的系统.

在自适应软件系统的研究中,基于模型的自适应方法是一个重要的研究方向.这类方法将模型驱动工程中模型的应用从开发阶段扩展到运行阶段,利用软件模型对运行时丰富和不确定的信息进行管理.通过使用模型这类方法,使得开发者可以忽略具体的实现细节,在高层次的抽象上规约和设计自适应逻辑,从而避免了将自适应逻辑与程序语言交织带来的复杂性,简化了自适应系统的开发和设计.在基于模型的自适应方法中,一个模型是一个与系统有因果联系的系统的抽象表示,从问题空间的角度对系统的结构、行为或需求进行了刻画.这里的模型与传统的模型驱动工程中的模型共享共同的观点,又在其上进行了扩展.扩展后的模型可以为自适应系统的动态演化、在线推理、动态状态监测、系统控制、系统行为观察、制品的自动生成、增加新的设计决策等提供支持.

为了给基于模型的自适应方法的研究提供支持,本文提出了 6 个研究问题,包括相关工作常用的需求模型、结构模型、行为模型、环境模型、模型与模型或模型与系统间的同步方式、自适应规划算法等.从这 6 个研究问题入手,对现有的基于模型的自适应方法进行了分析和总结.我们首先在 IEEE、ACM、Springer 和谷歌学术等论文搜索引擎中进行检索,检索时采用的主要英文关键词为((model based) or (model driven)) and ((adaptation) or (self-adaptation) or (self-adaptive system));然后,对检索出的论文进行浏览和筛选,去掉与基于模型的自适应方法无关的论文;对于通过筛选的论文,通过阅读和检索其参考文献、搜索作者的论文列表来进一步识别出相关的论文;最终,选择出与该研究问题直接相关的高质量论文共 40 篇.根据获取到的文献,我们对提出的研究问题进行了总结和回答.在此基础上,指出现有研究中的不足,并提出解决这些不足的设想.

本文第 1 节介绍自适应系统和基于模型的自适应方法的相关知识.第 2 节阐述相关的研究问题以及获取和筛选相关工作的标准与过程.第 3 节根据对现有的相关工作的分析与研究,对提出的研究问题进行详细的回答.第 4 节对现有方法中存在的问题和可能的解决思路进行介绍.第 5 节对全文进行总结.

1 基于模型的自适应方法

本节对基于模型的自适应方法的相关内容进行介绍.首先介绍自适应系统的相关概念,然后对基于模型的自适应方法进行介绍.

1.1 自适应系统

现代软件往往被部署于动态变化的环境中,同时需要满足可能发生动态变化的用户需求.变化的发生是不可预测和不可控的,并有可能导致系统的需求被违反.为了应对这样的复杂性,将软件设计成自适应系统(self-adaptive system)是一种有效的解决方案.自适应系统[1–3]是指能监测环境和系统自身的变化,并能够动态调整自身以响应变化的系统.被设计为自适应系统的软件具有两种能力:(1) 监测环境和系统变化的能力;(2) 进行自适应的操作来改变软件,以保证需求在变化发生后仍被持续满足的能力.这里的环境是指所有能被软件系统观察到的上下文,包括终端用户输入、外部的硬件设备、程序插桩等;自适应的操作是指可由软件自主进行的操作,包括修改参数、改变算法或方法、增加或减少构件、重组体系结构等.

自适应软件通常包括被管理元素和管理元素两个部分,其中,被管理元素是指自适应软件的应用逻辑,这部分可以在运行过程中被动态地加以调整;管理元素是指自适应软件的自适应逻辑,这部分通常通过反馈回路对应用逻辑进行调控[4].一种典型的反馈回路是 MAPE-K 回路(如图 1 所示),该反馈回路包括 4 个过程,即监测(monitor)、分析(analyze)、规划(plan)和执行(execute),以及一个被各过程共享的知识库(knowledge base).其中,监测过程通过传感器从环境和软件中收集数据,建立数据的相关性,并将数据转化为行为模式;分析过程对监测到的数据进行分析,以判断是否需要做出自主的调整;规划过程做出如何调整的决定,包括决定软件的需要变化的部分以及决定对这一部分做出何种操作以获得最好的结果;执行过程负责执行规划过程做出的决定,这包括将操作映射到相应的效应器上执行,将非原始的操作按照预先定义的工作流进行执行等.

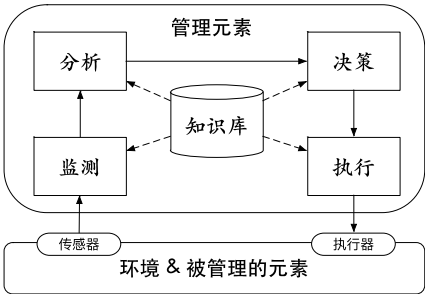

Fig.1 MAPE-K control loop of self-adaptive system

图 1 自适应系统的 MAPE-K 控制回路

1.2 基于模型的自适应方法

自适应系统的运行时环境中通常存在着大量而丰富的信息,这些信息引起的复杂性为自适应系统的开发带来了挑战.在传统的以代码为中心的自适应方法中,开发者经常使用诸如条件表达式、参数化和例外等程序语言特征来实现自适应.然而,这类以代码为中心的自适应方法将自适应逻辑与程序语言交织在一起,增加了自适应软件的复杂性.此外,这类方法也不具有良好的扩展性,无法处理多个自适应触发条件的问题.

相反地,利用软件模型来开发和管理自适应系统[5,6]能够有效地解决运行时的复杂性问题.软件模型的利用让开发者可以忽略具体的实现细节,在高层次的抽象上工作,同时能够避免将自适应逻辑与程序语言交织带来的复杂性[7,8].通常,基于模型的自适应方法利用并扩展了模型驱动工程(model-driven engineering,简称 MDE)中的模型和技术.模型驱动工程是软件工程的一个重要分支,一个传统的模型驱动工程中的模型是指为了特定目的而建立的系统抽象或系统的简化表示,例如用况模型、体系结构模型和部署模型等.这类模型传达了软件设计的含义,通常与软件设计阶段中的实体相联系.

基于模型的自适应方法需要对传统的模型驱动工程中的模型进行扩展,将模型的应用从设计阶段扩展到运行阶段,这些模型被称作运行时模型(models@run time)[9].与模型驱动工程中的模型类似,运行时模型也是与系统有因果联系的系统的抽象表示,刻画了系统的结构、行为或者目标.因为这样的因果联系,模型可以提供及时和明确的信息以驱动后续的自适应分析和规划,在模型层次做出的自适应规划也可以被追踪到系统中.与模型驱动工程中的模型从解决方案空间出发不同,运行时模型从问题空间的角度对系统进行刻画.该模型可以被视为一个在运行时仍然活动的开发模型,通常支持动态演化、在线推理、动态状态监测、系统控制、系统行为观察、制品的自动生成、新的设计决策的增加等.在基于模型的自适应方法中,MAPE-K 回路中的各过程都以模型为中心进行.监测过程会把监测到的变化在模型层次上进行体现;分析与规划过程在模型层次上进行,并做出如何改变模型的结构、元素或参数的自适应决策;执行过程除了在模型上执行做出的自适应决策之外,也会将变化即时反映到运行时的系统中去.

在基于模型的自适应方法中,常用的软件模型包括以下几类.

- 软件的结构模型.主要强调了软件的构造,例如刻画了对象、继承关系和调用路径的类图模型;刻画了构件和构件之间联系的体系结构图模型;刻画了刻面和刻面间交织关系的刻面图模型等;
- 软件的行为模型.主要强调系统执行的方式,例如刻画了系统状态和状态转移的状态图模型;刻画了系统工作流程的活动图模型;刻画了对象间交互和通信方式的交互图模型等;
- 软件的需求模型.区别于直接反映软件行为和结构的前两种模型,需求模型声明式地描述了软件的高层需求.该模型中的需求包括功能性的需求和非功能性的需求:功能性需求是指软件能提供的服务;非功能性需求是指软件所提供服务的服务质量,例如可维护性、可用性、可依赖性、高效性、安全性等.

除了对软件建模之外,基于模型的自适应方法通常会对环境建模,以管理丰富和不确定的环境信息带来的复杂性.环境模型刻画了环境的属性和可能的变化,尤其是刻画了可能导致需求违反从而触发自适应的环境变

化.在设计阶段,自适应系统无法预见所有可能的变化,因此,环境模型需要体现运行时环境的不确定性.

2 分类框架

本节首先对研究问题进行介绍,提出的 6 个研究问题分别关注建模方法、模型间关系和自适应规划算法等 3 个方面;接着对文献获取过程进行介绍,包括使用的搜索引擎、搜索关键字、搜索和阅读方法、筛选标准以及最终获取的文献数量.

2.1 研究问题

本文的主要目的是总结基于模型的自适应方法的相关工作,并给出未来的研究方向.基于这个目的,我们提出了 6 个更详细的问题.在基于模型的自适应方法中,模型通常刻画的是软件的行为、结构或目标,因此,前 3 个问题分别从行为、结构和目标的角度关注自适应系统的建模方法.除了使用软件模型之外,自适应系统经常使用环境模型来管理环境的不确定性,因此,问题 4 关注的是环境建模.问题 5 关注的是使用了多个模型的工作中如何规约模型间的关系以及如何将模型与系统进行同步.问题 6 关注自适应系统 MAPE-K 控制回路中的重要环节规划,即,如何在模型层次上以模型元素为单位进行自适应的规划.

问题 1. 相关工作通常使用哪些模型对软件的需求进行建模和管理?包括功能需求与非功能性需求.

问题 2. 相关工作通常使用哪些模型对软件的结构进行建模?

问题 3. 相关工作通常使用哪些模型对软件的行为进行建模?

问题 4. 相关工作通常使用哪些模型对环境进行建模?

问题 5. 相关工作如何规约模型与模型、模型与系统的关系?

问题 6. 相关工作使用了哪些自适应规划方法?

2.2 文献获取过程

本文第 3 节将基于 40 篇文献对上述问题做出回答,这些文献的获取过程如下.

(1) 使用 IEEE、ACM、Springer 和谷歌学术(Google scholar)等论文搜索引擎,以((model based) or (model driven)) and ((adaptation) or (self-adaptation) or (self-adaptive system))为关键词进行检索,得到包含这些关键字的论文.对得到的论文进行阅读,并按照表 1 给出的标准排除掉不符合标准的文献.

Table 1 Inclusion and exclusion criteria

表 1 包含和排除的筛选标准

包含标准	排除标准
C1:贡献领域为自适应系统或自适应软件,以及	C6:贡献领域不是自适应系统和自适应软件,或
C2:使用了基于模型的自适应方法,以及	C7:未使用基于模型的自适应方法,或
C3:超过 3 个引用量,以及	C8:低于 3 个引用量,或
C4:发表时间在 2000 年~2016 年之间,以及	C9:发表的时间早于 2000 年,或
C5:页数多于 3 页	C10:页数小于 3 页

(2) 浏览第(1)步所得论文的参考文献,对其中的可能和基于模型的自适应软件系统相关的文献进行检索,并按照表 1 的标准进行筛选.得到筛选后的文献后再循环执行本步,即:阅读文献并根据参考文献查找新的可能相关的文献,直到不再获取新的文献为止.

(3) 以第(2)步中筛选所得文献的第 1、第 2 作者为关键字,在论文搜索引擎上检索以他们为著者的论文,对所得结果进行阅读,并筛选出符合标准的文献.以文献的第 1、第 2 作者为关键字进行检索,是以作者作为线索进行相关文献的查找.一位研究者通常会在同一问题上开展一系列的研究工作,通过作者查找文献,可以了解到其系列工作的进展情况.

(4) 通过前 3 步总共筛选得到 98 篇文献,对这些文献进行精读.在阅读这篇文献的过程中,仅保留对自适应系统建模详细叙述的文献,去掉描述比较粗略的以及模型不占主要部分的文献,最终得到 40 篇文献.对这些文献进行详细的阅读和分析,以回答提出的研究问题.

3 调研结果

本节介绍根据调研和分析结果,对各个研究问题做出的回答.其中,第 3.1 节对调研结果进行概述,第 3.2 节~第 3.7 节分别对 6 个研究问题进行回答.

3.1 调研结果概述

表 2 和表 3 对调研结果进行了总结.

Table 2 Summary of modeling methods in existing studies

表 2 现有工作的建模方法总结

建模维度		采用的模型	相关工作	适用的场景
需求模型	功能性需求	(扩展的)特征模型	Ahmed, *et al*.[10], Cetina, *et al*.[11], Swanson, *et al*.[12], Alferez, *et al*.[13], Zhao, *et al*.[14], Acher, *et al*.[15], Franck, *et al*.[16], Ghezzi, *et al*.[17]	适用于建模功能性需求的变化点和变体
		正交变化性模型	Bencomo, *et al*.[18]	适用于建模功能性需求的变化点和变体
		目标模型	Chen, *et al*.[19], Qian, *et al*.[20], Lapouchnian, *et al*.[21]	适用于建模功能性需求及功能性需求间的关系
	非功能性需求	KAOS 模型	Cheng, *et al*.[22]	适用于建模非功能性需求及非功能性需求间的关系,支持形式化分析
		NFR 模型	Chen, *et al*.[19,23], Alferez, *et al*.[13], Lapouchnian, *et al*.[21], Bencomo, *et al*.[24], Zhao, *et al*.[17]	适用于建模非功能性需求及非功能性需求间的关系
		时序逻辑语言	Calinescu, *et al*.[25], Ghezzi, *et al*.[16,26], Dippolito, *et al*.[27], Sykes, *et al*.[28], Tsigkanos, *et al*.[29]	适用于建模“失败率”“成功率”“服务率”等可被形式化验证的非功能性需求
		排队模型	Vogel, *et al*.[30], Grassi, *et al*.[31], Moreno, *et al*.[32]	适用于建模“性能”类非功能性需求
		其他(建模目标与度量标准)	Ahmed, *et al*.[10], Esfahani, *et al*.[33], Qian, *et al*.[20], Filieri, *et al*.[34,35], Kim, *et al*.[36], Liliana, *et al*.[37]	适用于建模满意度可由相关度量值决定的非功能性需求
结构模型		(扩展的)体系结构模型图	Ahmed, *et al*.[10], Vogel, *et al*.[30], Tajalli, *et al*.[38], Pietschmann, *et al*.[39], Morin, *et al*.[40], Floch, *et al*.[8], Bencomo, *et al*.[18], Calinescu, *et al*.[25], Rouvoy, *et al*.[41], Cetina, *et al*.[11], Chen, *et al*.[19], Esfahani, *et al*.[33], Garlan, *et al*.[42], Liliana, *et al*.[35], Kim, *et al*.[37]	适用于建模构件及构件间的交互
		(扩展的)类图	Geihs, *et al*.[43], Salehie, *et al*.[44]	适用于建模类以及类间的结构与关系
		部署图	Grassi, *et al*.[31]	适用于建模软件在各硬件中的部署情况
		复杂因果网络	Salehie, *et al*.[44], Bencomo, *et al*.[24]	适用于建模各层次间的因果关系
行为模型		(扩展的)状态转移图	Tajalli, *et al*.[38], Calinescu, *et al*.[25], Dippolito, *et al*.[27], Sykes, *et al*.[28], Tsigkanos, *et al*.[29]	适用于建模系统演化的空间
		(扩展的)活动图	Ghezzi, *et al*.[26], Alferez, *et al*.[13]	适用于建模系统内的活动流程
		(扩展的)顺序图	Grassi, *et al*.[31], Ghezzi, *et al*.[16]	适用于建模系统的不同对象间的交互关系
		马尔可夫链	Calinescu, *et al*.[25], Ghezzi, *et al*.[13,26], Grassi, *et al*.[31], Moreno, *et al*.[32], Kim, *et al*.[37], Amoui, *et al*.[45], Ho, *et al*.[46]	适用于建模系统的概率行为和不确定行为

Table 2 Summary of modeling methods in existing studies (Continued)

表 2 现有工作的建模方法总结(续)

建模维度	采用的模型	相关工作	适用的场景
环境模型	(扩展的)特征模型	Acher, *et al.*[14], Fernandes, *et al.*[47], Zhao, *et al.*[17]	适用于建模环境的状态空间
	本体模型	Alferez, *et al.*[13]	适用于建模环境中的对象,支持推理环境对象间的关系
	概率过程	Moreno, *et al.*[32]	适用于建模环境的变化规律
	基于 KAOS 的威胁模型	Cheng, *et al.*[22], Salehie, *et al.*[44]	适用于建模环境中的威胁因素
	正交变化性模型	Bencomo, *et al.*[18]	适用于建模环境的状态空间
	状态转移图模型	Dippolito, *et al.*[27], Tsigkanos, *et al.*[29]	适用于建模环境的变化规律
	其他(建模环境参数与其取值范围)	Morin, *et al.*[7], Cetina, *et al.*[11], Franck, *et al.*[15]	适用于建模环境参数

Table 3 Summary of the unit and methods of adaptation planning in existing studies

表 3 现有工作中自适应规划单位和规划方法的总结

规划方法		相关工作	适用的场景
基于规则的推理		Cheng, *et al.*[22], Vogel, *et al.*[30], Bencomo, *et al.*[18], Cetina, *et al.*[11], Garlan, *et al.*[42], Acher, *et al.*[14], Frank, *et al.*[15], Fernandes, *et al.*[47], Zhao, *et al.*[17]	适用于利益相关者在设计阶段时具备充足的对于运行时系统和环境的知识的场景
模型检查	概率	Calinescu, *et al.*[25], Ghezzi, *et al.*[16,26], Grassi, *et al.*[31], Moreno, *et al.*[32], Sykes, *et al.*[28]	适用于自适应的目的是为了保证某一属性的满足或最优化,且该属性可以被形式化为概率逻辑进行验证的场景
	非概率	Dippolito, *et al.*[27], Alferez, *et al.*[13], Acher, *et al.*[14], Tsigkanos, *et al.*[29]	适用于自适应的目的是为了保证某一属性的满足或最优化,且该属性可以被形式化进行验证的场景
控制论		Filieri, *et al.*[34,36]	适用于自适应的目的是将指定的度量指标控制在理想值附近的场景
基于效用函数的最优化		Ahmed, *et al.*[10], Floch, *et al.*[8], Rouvoy, *et al.*[41], Geihs, *et al.*[43], Chen, *et al.*[19], Esfahani, *et al.*[33], Qian, *et al.*[20], Filieri, *et al.*[34,36], Salehie, *et al.*[44], Zhao, *et al.*[17]	适用于自适应的目的是为了最优化一个表示系统理想程度的实数的场景
基于目标的推理		Chen, *et al.*[23], Lapouchnian, *et al.*[21], Liliana, *et al.*[35]	适用于自适应的目的是为了保证系统始终处于一个理想状态的集合中
强化学习		Kim, *et al.*[37], Amoui, *et al.*[45], Ho, *et al.*[46]	适用于自适应的目的是为了将某个表示系统理想程度的奖励信号的累积值进行最大化的场景

表 2 对基于模型的自适应方法中常用的模型进行了总结,包括环境模型和软件模型,软件模型可以进一步分类为需求模型、结构模型和行为模型.这里有部分模型也是模型驱动工程中常用的种类,但是为了使用这些模型管理运行时的复杂性,大多数相关工作将这些模型进行了扩展,例如加入新的元素或赋予原有元素新的含义等.运行时的软件模型不仅能够镜像系统当前的状态与行为,还建模了需要监测的对象、需要触发自适应的条件、备选的自适应方案、备选方案对需求的贡献、自适应规划方法等,为在线推理和持续演化提供了支持.

表 3 对自适应系统的规划方法进行了总结.在基于模型的自适应方法中,自适应规划通常在模型层次上进行,做出的规划也以模型元素为基本单元.一个模型层次的自适应规划建模了理想的软件模型以及从当前模型到理想模型的迁移策略,具体的动作包括一系列模型元素的激活与删除.

3.2 自适应系统的需求建模方法

软件的需求可以分为功能性的和非功能性的需求.功能性的需求刻画了软件可提供的服务;非功能性的需求刻画了软件所提供服务的服务质量,例如可维护性、可用性、可依赖性、高效性、安全性等.其中,功能性需求的满足与否有明确的衡量标准;非功能性需求的满足程度不能用明确的“实现”和“拒绝”来衡量,而是可以被“部分实现”和“部分拒绝”的.

在基于需求模型的自适应工作中,根级别的功能性需求必须被保证始终满足,而低层的功能性需求则可以

被适度地放松和违反.这些工作通常将叶子级别的功能性需求视为备选的自适应方案.在自适应的分析和规划过程中,非功能性需求的满足程度经常会作为自适应方案理想程度的评价标准.即:在所有可行的能够实现软件功能的备选自适应方案中,使得非功能性需求的满足程度更高的自适应方案更有可能被选择.

此外,部分工作使用自适应规则模型对自适应逻辑进行了建模.这部分工作在进行分析和规划时不需要进行基于需求的推理,因此,需求模型对它们来说不是必须的.

- 非功能性需求

如表 2 所示,在基于模型的自适应方法的相关工作中,建模非功能性需求常用的模型包括 NFR(非功能性目标建模框架)模型、KAOS、时序逻辑语言、排队模型等.

NFR(non-functional requirement)框架最早由文献[48]提出,并在文献[49,50]中被进一步扩展.NFR 框架主要关注非功能性需求的建模与分析,支持对非功能性需求进行精化分解和操作化.文献[13,19,21,23,24]采用了 NFR 框架对非功能性需求进行建模,并对该框架进行了扩展,以使其更适合于建模自适应系统中的非功能性需求.例如,文献[19]对 NFR 模型进行了扩展以支持定量的分析,文献[13]在该模型固有元素的基础上增加了表示“度量标准”的元素,并将度量标准与非功能性目标对应起来.

KAOS 框架是让一切目标满意(keep all objectives satisfied)[51]的缩写,它同样支持非功能性需求的建模,但比起 NFR 框架,更能够支持形式化的分析.文献[22]采用了 KAOS 建模非功能性需求方式,并引入了 RELAX 语言[52],以给目标增加模糊的语义.扩展后的模型除了能够建模自适应系统中必须实现的需求外,还可以建模可以被适度放松和违反的需求.

时序逻辑[53]通常被用来描述和推理关于时间限定的命题的规则和符号化的任何系统,主要用于形式化验证.文献[16,25–29]采用该语言来描述与时序相关的服务质量需求.例如,文献[16]基于时序逻辑语言将“可达性”需求描述为 P>95%[F(*State*=*s*)],即,最终到达某个状态 s 的概率大于 95%.文献[25,26]采用了概率计算树逻辑来形式化地描述失败率、服务率、响应时间等需求,这里,概率计算树逻辑是一种包含了成本和概率的扩展的时序逻辑语言.文献[29]用时序逻辑语言对安全性需求进行建模.

排队模型适合于建模性能或者响应时间等需求.文献[30–32]采用了该模型建模响应时间和性能等.

文献[10,20,33–36]建模了非功能性需求与其“度量”.“度量”是一个可观测的属性值,可用于衡量对应的非功能性需求的满足程度.其中,文献[10]在建模非功能性需求时,引入了与每个需求相对应的“度量”,同时引入了效用函数来建模“度量”的观测值与非功能性需求满意程度的函数关系.例如,“快速响应”需求对应的度量为“响应时间”,监测到具体的响应时间后,可以通过效用函数计算“快速响应”需求的满意程度.文献[20]将非功能性需求的度量建模为一个四元组,包括其当前值、约束上界、约束下界和权重,其中,权重可以作为控制参数,偏离约束范围需求的权重会被适当地调高.文献[35]将非功能性需求的度量建模为关键性能指标(KPI),其属性包括名字、取值类型和误差容限,而一个非功能性需求用对应 KPI 的合法取值范围来描述.

- 功能性需求

如表 2 所示,基于模型的自适应方法的相关工作在建模功能性需求时常用的模型包括目标模型、特征模型、正交变化性模型等.

目标模型[54]描述了系统高层的功能需求和非功能性需求,其中,功能性需求被建模为目标,非功能性需求被建模为软目标.因为本小节前面的内容已经介绍了非功能性需求的建模,这里主要对功能性需求进行讨论.目标模型中,高层的目标被反复通过与/或精化关系进行分解,直到成为可被任务满足的叶子级别目标.为了满足一个被与分解的目标,其所有的子目标都必须被满足.为了满足一个被或分解的目标,至少一个子目标必须被满足.采用目标模型建模自适应系统中功能性需求的工作包括文献[19–21,23,24],它们利用目标模型建模了自适应的空间,利用目标模型中的或分解关系建模了系统的变化点,即:在一个或分解关系中,父目标的实现有多种备选的方式,每个子目标表示了一种备选的方案.在这些工作中,自适应的单位通常是叶子级别的功能性目标或者任务,一个合法的自适应方案由一组可以使得根目标被满足的叶子级别的目标或任务组成.文献[23]在目标模型的基础上加入了元素“可选目标”,被设为可选的目标在适当情况下可以被放松.

特征模型[55,56]的概念来自软件产品线社群,它描述了特征间的层次结构和约束关系,其中,一个特征表示了一组突出或独特的软件的用户可见的方面.在软件产品线工程中,特征模型的定制、特征的绑定与解绑定通常在设计阶段进行.基于模型的自适应方法的相关工作对特征模型进行了扩展,允许运行阶段动态地对特征模型进行重配置.文献[10–16,20]采用特征模型建模功能性需求,这里,特征模型建模了软件配置的空间,特征模型的一个合法定制结果表示了一个自适应时可以选择的软件配置.相应地,自适应规划包括理想的软件配置以及从当前软件配置到理想软件配置的迁移策略,自适应操作包括一系列特征的绑定和解绑定.特征模型可以通过命名约定与体系结构片段(或刻面模型)之间建立引用关系.

除了特征模型外,正交变换性模型[57]也被用来建模软件配置中的变化点和变体.文献[18]采用正交变化性模型建模了系统的构件变化点和可选的构件变体,其中的每一个变体用领域特定建模语言进行了描述.

- 小结

在选择对非功能性需求、功能性需求、自适应规则中的一个或多个进行建模时,需根据希望通过自适应达到的目的进行判断和选择:若希望系统通过自适应保证非功能性需求的满足,则需要对非功能性需求进行建模;若希望系统通过自适应保证功能性需求的满足,则需要对功能性需求进行建模;若希望系统根据预先指定的规则进行反射式的自适应,则需要对自适应规则进行建模.

在非功能性需求建模方面,以上各模型适用于不同场景下的非功能性需求的建模.NFR 和 KAOS 将非功能性目标层次式地组织起来,适用于需要对非功能性目标间的关系(包括精化关系、贡献关系等)进行推理的场景.另外 3 种建模方式,即时序模型、排队模型和度量,支持对特定非功能性需求的建模:时序模型适用于建模可被形式化的且与时序相关的非功能性需求,例如失败率、服务率、响应时间等;排队模型适用于建模“性能”类非功能性需求;“度量”适用于建模其满意度可由某一可观测属性值决定的非功能性需求.基于这 3 种建模方式,系统可以根据运行时对环境或系统的监测结果推理出非功能性需求的满意度.

在功能性需求的建模方面,以上的模型中,特征模型和正交变化性模型更多地关注共性和变化性的建模,它们建模了功能性需求中的变化点和各变化点上的变体,适合于对功能性需求进行定制的场景;目标模型更适用于满足性的推理,支持以下两种满足性推理:(1) 给定一个叶子级别的目标或者任务的集合,推理根目标是否能够被满足;(2) 给定根目标,推理能够保证根目标实现的叶子级别的目标和任务的集合.

3.3 结构模型

如表 2 所示,相关工作常用的结构模型包括体系结构模型、类图、部署图、因果网络等.基于模型的自适应方法对这些结构模型的使用与传统的软件开发过程对结构模型的使用不完全相同,通常会对其元素或含义进行扩展,以适合于对运行时的复杂性进行管理.

体系结构模型[58]建模了系统的构件以及构件的交互,其中,构件表示系统的主要计算元素和数据存储,包括服务器、客户端、数据库和用户接口等,构件间的连接子表示构件间的交互路径.体系结构元素可以用不同的属性进行注释.例如期望的吞吐量、延迟和交互协议等.体系结构模型可以使用统一建模语言(UML)、服务构件体系结构(SCA)、体系结构描述语言(ADL)等进行描述.采用体系结构模型建模系统结构的工作包括文献[8,10,11,18,19,25,30,33,35,37–42].这些文献通常对体系结构模型进行扩展,以建模出体系结构模型的哪些部分可以发生演化以及可以发生哪些演化等.其中,文献[39]使用了多个体系结构模型来建模不同的关注点,例如性能、安全、成本等.文献[40]采用扩展的体系结构模型分别建模了各个软件制品的共有部分以及可能的变体.文献[8,19,33,35,41]均对体系结构模型进行了扩展,以使其支持变化点和变体的规约,即,部分构件被定义为角色、构件类型或者抽象构件,这样的构件具有一组可能的构件实现,每个备选的构件实现上会标注其对于非功能性需求的影响.扩展的体系结构模型包括了多个变化点与备选变体,文献[8]将该扩展体系结构模型的一个定制结果命名为实例体系结构模型,它反映了系统当前的体系结构.文献[11]采用领域特定的建模语言 PervML 来描述体系结构,该语言可以图形化地描述服务、设备和通道,并能够在各元素与特征模型的特征之间建立起映射关系.

类图[59]展现了一组对象、接口、协作和它们之间的关系.文献[43]采用了 UML 的类图,该类图中的类包含了抽象的代表服务类型的类以及具体的服务,其中,代表服务类型的类可以通过接口连接到备选服务集合中的

被选定的服务.文献[44]采用类图对系统的资产进行建模,各个类与功能性需求、安全性需求之间具有关联关系.部署图[59]描述的是系统运行时的结构,展示了硬件的配置及其软件如何部署到网络结构中.文献[31]采用了UML 的部署图建模系统结构,部署图的片段可以被建模为可选的,可选的片段可以被去除或者被其他备选的片段替代.复杂因果网络是一个各层次之间具有多种因果关系的复杂网络.文献[44]采用复杂因果网络建模资产、威胁、对策之间的关系.文献[24]采用动态决策网络建模备选的任务与决策标准之间的关系.

- 小结

在对系统结构进行建模的相关工作中,大多数工作采用了体系结构模型,该类模型建模了系统构件以及构件间的交互.体系结构模型适合于作为需求层和实现层之间的中间层,可以很容易地建立起功能性需求与构件集合、构件与具体实现之间的映射关系.少量的工作采用了其他模型建模系统的结构,其中,类图可用于建模类及类间的结构与关系;部署图可用于建模软件如何部署到硬件中,多适用于分布式系统;复杂因果网络适合于建模多种层次之间的因果关系.

3.4 行为模型

如表 2 所示,相关工作常用的行为模型包括状态图、活动图、概率模型、交互图等.

状态图[59]主要用于描述一个对象在其生存期间的动态行为,表现为一个对象所经历的状态序列、引起状态转移的事件(event)以及因状态转移而伴随的动作(action).采用状态图作为分析模型的相关工作大多对状态图进行了扩展.例如,文献[25]中的每个状态表示了一个抽象的服务,该抽象服务候选的服务实现也被规约在相应的状态中.文献[38]使用状态转移图建模了软件演化的空间,其中的一个状态表示一种可能的软件体系结构,一个转移则表示一种可能的自适应操作,例如增加/移除构件、初始化/杀死构件、连接/解除构件之间的连接等.文献[29]同样用状态转移图建模了软件演化的空间,状态图中的一个转移表示一个自适应操作或者一个不可控制的环境事件.文献[18]用状态转移图建模了自适应的策略,其中,每一个状态表示一个候选的软件变体,状态之间的转移表示触发自适应的条件.文献[28]采用符号转移系统建模了软件行为,其中,各状态之间的转移概率可以通过学习进行在线更新.

活动图[59]是一种特殊的状态图,展现了系统内一个活动到另一个活动的流程.活动图有利于识别并行活动.文献[13,26]采用了扩展的 UML 活动图建模软件的行为.文献[26]允许将部分活动设为可选,被设为可选的活动可以在运行时被自适应机制自主地激活或者去除.同时,该工作允许将某两个活动之间的片段设为可选的,允许这个片段被整个去除或者替换为其他备选片段.文献[13]利用活动图中的决策节点建模了自适应的策略,决策节点表示前一个活动完成后将需要进行自适应决策,决策节点上的警戒条件用来决定执行一组备选转移中的哪一个转移.

概率模型能够以定量的方式规约不确定性和无法预测的行为,常用的概率模型包括离散时间马尔可夫链(DTMC)、连续时间马尔可夫链(CTMC)、马尔可夫奖励模型、马尔可夫决策模型(MDP)等[60].概率模型检查方法可以验证概率模型的行为,适合于验证和预测依赖性、性能和成本等非功能性需求,常用的概率模型检查工具包括 PRISM 和 MRMC[61].一个离散时间马尔可夫链可以被看作一个转移用概率进行了标注的状态机,同时,表示成本的实数也可以在转移上进行标注.在相关工作中,文献[16,25]采用了离散时间马尔可夫链作为推理模型,采用概率模型检查的方法验证当前的系统状态是否能够使需求得到满足.文献[16]采用参数化的离散时间马尔可夫模型,即:转移概率在设计时先用参数进行标注,再根据运行时的监测结果进行更新.这里,每一种备选的自适应方案对应一个马尔可夫模型,在自适应规划时,通常以各备选方案的验证结果为依据,选择出验证结果最好(即,对服务质量需求贡献最大)的备选方案.文献[26,32]采用了马尔可夫决策模型,马尔可夫决策模型在离散时间马尔可夫链的基础上加入了不确定性的动作,即,一个状态向其他状态转移时有不确定性的选择.文献[31]采用了半马尔可夫奖励模型作为分析模型,其中的状态代表可能的系统配置,转移表示了可能的系统演化,状态上的奖励建模了系统某个配置的非功能性属性,转移上的奖励建模了该重配置操作的成本.文献[37,45,46]是基于强化学习的进行软件自适应的工作,它们将软件与环境的行为建模为马尔可夫决策过程,其中,该过程的一个状态由当前的系统配置和环境状态共同组成.

交互图[59]用于描述对象间的交互关系,由一组对象和它们之间的关系组成,包含它们之间可能传递的消息.文献[16,31]将软件行为建模为扩展的时序图模型(增加了可选的模型元素以及参数),相应地,一个备选的配置方案对应于扩展时序模型的一个定制结果.时序图模型的定制结果可按预先定义的规则自动转化为参数化的离散时间马尔可夫链.

- 小结

在自适应系统的需求满足情况不仅受到系统结构的影响、也同样受到系统行为影响的情况下,需要选择恰当的行为模型对相关的系统行为进行建模.

以上的行为模型适用于不同的建模场景.状态图适用于建模系统的演化空间,这包括各个合法的系统配置(状态)以及系统在不同配置之间的转移;活动图适用于建模系统内的活动流程;马尔可夫模型建模了概率行为和/或不确定行为,其中的状态可以是一个系统配置,也可以是一个系统内的活动,当需要对系统的概率行为和不确定行为进行分析时可以采用该种模型;交互图适用于需要对对象和对象间的关系进行分析的场景.

3.5 环境建模方法

自适应系统的环境是指所有能被软件系统观察到的上下文,包括物理环境、部署环境、用户交互环境等.如表 2 所示,相关工作在对环境进行建模时,常用的模型包括变化点模型、本体模型、威胁模型、概率过程、时序语言等.

特征模型等变化点模型可以用来对环境进行建模.例如,文献[14]用特征模型对环境进行了建模,该模型是一个支持自适应规则定义的最少的环境集合,它们的值被传感器的监测结果更新.文献[47]利用扩展的特征模型对环境进行了建模,其中,特征可以分为环境实体特征和环境信息特征.环境实体特征用来表示领域中对系统有影响的环境实体;环境信息特征表示了描述一个环境实体所需要的数据,例如环境实体的名称、类型、位置、初始值和来源等.该工作还基于这些特征定义了环境表达式,环境表达式可以由多个基本环境表达式通过逻辑操作符进行组合,一个基本的环境表达式由一个环境信息特征、一个操作符和一个阈值组成.文献[18]采用正交变化性模型对环境的属性和可能的取值进行了建模.文献[7,11,15]采用了一组与自适应相关的环境变量来建模环境,其中,变量的取值可以是布尔或者枚举类型,其监测值由环境感知器给予提供.

本体模型用于描述由一套对象类型、属性以及关系类型所构成的世界,可以用来支持对该领域的属性进行推理.文献[13]采用了本体模型对环境进行建模,对领域知识进行形式化的分析.威胁模型分析建模了系统中的潜在威胁,用于分析和发现对抗的策略,以建立安全的系统.文献[22]采用威胁模型来建模环境中的不确定性,并使用概念理论模型来限定环境的范围.文献[44]采用 KAOS 的威胁模型对威胁代理、威胁目标和可能的攻击进行建模.概率过程是一连串随机事件动态关系的定量描述.文献[32]将环境建模为一个概率过程,用随机变量建模环境的状态.文献[27]用时序语言对环境中的假设进行建模,并将环境建模分为多个层次,其中,高层的环境模型是对低层环境模型的理想化表示,每一层的环境由有限状态过程表达式进行建模.文献[29]用双向图建模了环境中的拓扑结构,并用符号转移系统对该拓扑结构上可能发生的变化进行了建模.该工作的建模没有明确对环境和软件进行区分.

- 小结

以上各模型对环境的不同方面进行了建模,在建模环境时,需要以何种环境信息与系统的自适应相关为依据选择适合的环境模型.以上模型中,特征模型和正交变化性模型适合于对环境的状态空间进行建模,包括建模环境中相关的属性及属性中可能的取值等;在环境属性间的层次和约束关系与系统的自适应不相关的情况下,可以不采用变化性模型,而是直接描述相关的环境属性和它们的可能取值;威胁模型适合于建模的是环境中的威胁,适用于安全性为关键质量目标的系统;本体模型建模了环境中的对象、对象的属性、对象间的关系,适用于需要对环境中的各对象进行详细分析、推理的情况;概率过程和状态转移图适合于对环境的变化规律进行建模,其中,概率过程估计了环境在某一状态上进入其他状态的概率,这些模型适用于需要对环境的变化进行预测的情况.

3.6 模型与系统的关系

大多数的相关工作使用了超过一种的模型来建模和管理自适应系统,其中,常见的模型间的关系包括功能性需求模型、结构模型或行为模型对非功能性需求模型的影响关系,需求模型、结构模型、行为模型彼此之间的追踪关系,环境模型与软件模型之间的约束关系,软件模型与运行时系统之间的同步/镜像关系等.

功能性需求模型对非功能性需求模型的影响通常用贡献链接进行建模.一个贡献链接存在于一个目标(或特征)与一个非功能性需求之间,表示该目标(或特征)的实现/拒绝能给非功能性需求的实现/拒绝提供一定程度的证据.按照目标(或特征)对非功能性需求的影响是正还是负,可以将贡献链接的类型分为“帮助”和“伤害”,或者“++”“––”“+”和“–”.文献[21,23]即采用了这样的定性的贡献关系.为了便于基于非功能性需求的实现程度建立效用函数,文献[19,20]将定性的贡献关系进行量化,以支持定量的在线推理.文献[10,33]可以通过在线学习自动地学到特征对非功能性需求的贡献关系,学习到的结果是一个以特征值为自变量、功能性需求的满足程度为因变量的贡献函数.

结构模型对非功能性需求模型的影响通常会在各备选的构件、类或服务等上面进行标注.例如,文献[8,19,33,41]都在备选的构件上标注了其对于各非功能性需求的影响,代表了选择该构件后期望的吞吐量、延迟和交互协议等.行为模型会在状态或转移上标注其对于非功能性需求的影响.例如,文献[26]在代表服务的状态上标注了执行该服务消耗的时间、对可用性的影响等,在转移上标注的概率值可用来计算可达性需求的满足程度等.

高层的需求模型通常会与低层的结构或行为模型之间建立起映射关系,即,将一个需求模型中的元素映射到一个或一组结构/行为模型中的元素.两个模型之间的同步可以通过模型转换语言或者现有工具来实现.文献[10]在特征模型与体系结构模型之间建立了映射关系,使用模型转换语言实现两个模型的双向同步.文献[11]将一个特征映射为一个构件的集合,当该特征被绑定时,集合中的构件将被全部插入系统,当该特征被解绑定时,集合中的构件将被全部去除.文献[19]在目标模型与体系结构之间建立起了映射关系,将一个目标映射到一个设计问题,设计问题的每一个备选的设计方案被映射到了体系结构模型的一个片段.文献[23]将叶子级别的目标映射到体系结构的构件,可选目标的选择被映射到切换连接子,以实现构件的绑定和解绑定.

有的相关工作同时采用了与系统联系更紧密的设计模型以及更适合进行自适应验证或决策的分析模型.设计模型与分析模型可以通过模型转换语言进行同步,或者通过现有工具自动地根据设计模型生成分析模型.文献[39]使用双向模型转换语言,将建模了不同关注点的多个体系结构模型与镜像了系统当前结构的反映模型之间通过模型转换语言进行同步.文献[26]将一个包含可选片段的状态图转化为马尔可夫链进行分析.文献[31]采用中间模型作为从设计模型到分析模型转换的桥梁,设计模型可以通过两次模型转换生成分析模型.文献[16]采用工具将特征模型自动地转化为扩展的顺序图,再将顺序图转化为马尔可夫链模型以进行分析.

环境模型与软件模型之间的约束可以通过约束规则或自适应规则进行规约.文献[11,20,22,30,42]采用规则规约了环境模型与软件模型之间的约束关系,包括:当某个环境特征被选定时,特定的软件特征必须被选定(或不能被选定);或者当某个环境表达式被满足时,特定的软件特征必须被选定.与基于规则推理的方法不同,在线进行自适应规划的方法不会在离线阶段指定环境模型与软件模型之间的约束关系,而是由系统在在线过程中自动推理.

系统模型与系统之间需要进行即时的同步.系统的更新应当及时地在反映模型上进行体现,而反映模型被更新后,系统也应当被相应地更新.模型与系统间的同步可以通过模型-代码转换语言或者借助中间件来实现.模型与系统间的同步包括完全的和增量式的同步,其中,完全的同步是指从模型出发生成全部的应用代码,或从应用代码出发构建出全部的模型元素,这种方法在更新系统时需要停止运行和重新部署,不能支持高效的自适应;增量式的模型转换方法则会增量式地对模型或者系统进行更新,可以进行高效的更新并且不会中断系统的运行.自适应相关的工作通常选择的是增量的、在线的更新方式.文献[11]在进行重配置时,会计算出当前体系结构模型与目标体系结构模型的差别,再利用 OSGI 框架[62]执行重配置的步骤,OSGI 框架可以在不重启系统的情况下安装、启动、重启和卸载构件.文献[19]在将模型的更新同步到系统时,首先由自适应机制自动生成从当前

的体系结构模型向目标体系结构模型作增量式转换的脚本,再由能够进行体系结构管理的中间件执行系统的重配置.文献[10,33]在进行自适应时,由中间件 XTEAM 对体系结构模型进行修改,同时将变化自动地反映到运行在中间件 Prism-MW 的系统上去.文献[30,39]采用了三元图语法规则(TGG rule)来实现模型与系统的同步.文献[18]在不同的模型层次之间建立起追踪关系,当抽象层次高的模型被选定之后,抽象层次低的模型和相应的代码就会由模型-模型和模型-代码的转换自动生成.文献[43]从结构模型出发,通过两次模型转换生成中间件可以解析执行的文档,再由中间件对运行于其上的系统进行重配置操作.文献[26]将每一个备选服务追踪到代码片段,并在代码片段上标注其对应的抽象功能.文献[42]在模型和系统之间增加了翻译层,该层中的翻译资源库保存了模型元素到系统元素之间的映射关系.

- 小结

本节介绍了模型与模型、模型与系统间的关系.对于基于模型的自适应来说,如何将模型层次上的自适应同步到运行时的系统中,是非常重要的研究课题.在进行模型到系统的同步时,可以通过模型-代码转换或者通过中间件直接把变化同步到系统中,也可以借助中间模型进行同步.例如,将功能性需求模型发生的变化同步到系统中时,通常会首先将变化同步到层次较低的结构模型或行为模型上,再由结构或行为模型将变化同步到系统.不同于功能性需求建模了系统能够提供的服务,非功能性需求建模了系统提供服务的质量.因此,非功能性需求不能直接与结构模型、系统等建立映射关系,一般地,功能性需求模型、结构模型、行为模型中的部分元素与非功能性需求之间存在正向或负向的贡献关系.

3.7 基于模型的自适应规划方法

在基于模型的自适应方法中,规划单位通常是模型中的元素,例如特征、目标、构件、服务等,或者是模型中的参数和在模型基础上定义的操作.这里的单位指的是进行自适应规划时的单位,不考虑之后带来的相应的改变.例如,文献[10]以特征为单位做出规划后,会通过模型间的同步对体系结构元素做出改变,这里的规划单位将被认为是特征.也有相关工作采用了多种自适应规划单位,例如,文献[19]可以以特征为基本单位在需求层次上进行规划,也可以以构件为单位在体系结构层次上进行规划;文献[35]可以以构件或者参数为单位进行规划.

如表 3 所示,常用的自适应规划方法包括基于规则的自适应、模型检查、控制论、基于效用函数的最优化、基于目标的推理、强化学习等.也有工作将多于一种的方法结合起来进行自适应规划.

在基于规则的方法中,自适应策略由一组规则进行预先定义,每条规则指定在一个特定的环境中应该执行某种特定的操作.自适应规则的形式通常包括“事件:条件→操作”(ECA 规则)或“条件→操作”(CA 规则)两种.相关工作[11,14,15,17,18,20,22,30,42,47]对规则的“条件”和“操作”有不同的定义,例如,文献[14]的规则模型中,“条件”为环境特征,“操作”为软件特征的绑定;文献[15]的“条件”为一个基于环境信息的布尔表达式,“操作”为软件构件的插入或拔出;文献[47]的规则形式包括“前件 imply(推出)后件”和“前件 exclude(排除)后件”两种,其中,前件为环境表达式,后件为软件特征.采用基于规则的方法进行决策时,系统逐条判断各条自适应规则的触发条件是否为真:若为真,则执行这条规则指定的操作.若不同规则指定的操作存在冲突,则由系统根据指定的规则间的优先顺序做出选择,或者由利益相关者手动地在冲突的规则之间进行选择.自适应规则规约了自适应的逻辑,指定了在特定的环境变化发生时应当执行的自适应操作.该类方法的优点是简单和规划过程的高效,但缺点是因为策略被预先定义所以不够灵活,并且很可能有着过度庞大的状态空间或者规则集.

基于目标推理的方法通常会指定理想的状态,或者指定一个或多个刻画理想状态集合的标准,所有处于这个集合中的状态都是可以接受的,系统需要通过计算找到可从当前状态转移到理想状态的操作.基于目标推理的方法将自适应规划转化成了一个满足性问题或有约束的满足性问题.文献[21]采用了基于目标推理的方法,其理想状态是根目标被满足,所有使得根目标被满足的备选方案都是可行的.当在候选的方案中进行选择时,以该候选集对非功能性需求的贡献以及非功能性需求的优先级为依据进行选择.文献[23]采用了一个 PID 控制器来判断何时需要进行基于目标的推理,理想状态为关键目标被满足并且使得控制变量的“业务价值”在合法范围内的状态.文献[35]定义的理想状态为所有的非功能性需求被满足,即,所有的关键性能指标均处于合法的范围内.当有非功能性需求被违反,即,对应的关键性能指标偏离了合法范围时,系统会根据各自适应操作对关键

性能指标的影响,选择出能使关键性能指标恢复到合法范围的自适应操作集合.文献[20]结合了基于目标的推理方法和基于案例的推理方法来进行自适应规划:对于没有出现过的自适应场景,该工作采用基于目标推理的方法找到最优的自适应方案,同时会把场景与对应的方案作为一个案例保存下来;对于出现过的自适应场景,该工作将采用基于案例的推理方法,即,采用保存在案例中的该场景对应的自适应方案.

基于效用的方法将每个变体的理想程度映射为一个真值标量,在规划时,从当前可行的变体中选择出使效用函数值最大的变体.基于效用的方法将自适应规划转化成了一个最优化问题.部分相关工作将效用函数定义为系统非功能性需求(例如快速响应、高可用性、高安全性等)满足程度的加权和.在非功能性需求满意程度的衡量上,文献[10]为每个非功能性需求引入了可观测的属性作为其度量,能够基于度量的观测值,按预先定义的计算标准得到非功能性目标的满意度;文献[8]给每个候选的变体标注了在特定环境下的非功能性属性(例如,内存、响应时间)的预测函数,根据该函数和监测到的环境状态,可以计算出选择该变体时非功能性属性的预测值;文献[33]定义了根据系统配置估计非功能性需求满足度的分析模型,该分析模型是基于模糊数学和概率理论建立的,可以度量软件对非功能性需求影响的不确定性.文献[24,44]将目标模型映射到了动态决策网络上,非功能性需求的满意度被映射为效用节点,备选的自适应方案被映射为决策节点,备选方案对非功能性需求的贡献也被映射到网络中作为期望效用,从而将自适应规划转化成了基于效用函数和该复杂因果网络的最优化问题.

基于概率模型检查的自适应规划方法通常将软件的行为或环境建模为马尔可夫模型,例如离散时间马尔可夫链、连续时间马尔可夫链、马尔可夫决策过程等,同时,用概率计算树逻辑的形式表述目标.PRISM[61]是一种为概率模型检查提供支持的工具,它支持各类马尔可夫链的建模,并能够支持概率计算树逻辑.大量基于马尔可夫链的自适应工作都采用了 PRISM 来进行概率模型检查.文献[25,28,31]采用离散时间马尔可夫链建模软件的行为,在运行过程中,自适应机制会通过概率模型检查来验证软件行为是否能够满足需求.在当前的软件行为不能满足需求的情况下,系统会重新选择其他备选方案以保证需求的持续满足.文献[26,32]用马尔可夫决策过程建模了软件的行为,在进行自适应规划时,自适应机制会在各备选的不确定转移之间进行选择,采用概率模型检查的方法选出使得非功能性需求被满足的概率最大的那个备选转移.文献[32]介绍了主动自适应的工作,它能根据对环境情况的预测,在变化发生前主动地做出自适应规划.文献[28]会根据运行时的监测结果对马尔可夫链上的概率进行自动的更新,然后在更新后的马尔可夫链上,用概率模型检查的方法得到使得到达目标状态概率最大的自适应方案.

(非概率的)模型检查的方法通常将软件的行为或环境建模为形式化的状态表达式,将目标用形式化的逻辑语言进行描述,在此基础上,通过模型检查验证需求的满足情况或者选择能最优化需求的备选方案.文献[27]将系统的行为建模为有限状态表达式,将目标用线性时序逻辑进行描述,通过模型检查的方法自动合成在不同领域假设情况下的自适应策略.文献[29]用时序逻辑语言描述了安全性需求,采用符号转移系统建模了环境和系统的演化模型,该演化模型对未来所有可能出现的状态进行了预测.基于演化模型和形式化的需求,该工作采用模型检查的方法判断每个状态是否有可能违反需求,并将有可能违反需求的状态标记为不合法的,接着找出可以避免进入或者离开不合法状态的自适应策略.

基于控制论[63]的自适应规划方法根据控制模型做出自适应规划,该控制模型描述了如何根据观察值与目标值的偏差对控制参数进行调整.设计一个精确地用于自适应软件的控制模型非常困难,因为实时系统通常有着复杂的非线性的动态性.文献[34,36]通过在线学习自动地构建控制模型,可以避免设计该模型的复杂性问题.文献[34]通过在线学习得到分析模型,分析模型建模了控制参数与非功能性属性的关系.基于该模型得到控制策略,控制策略描述了如何根据属性值的偏差对控制参数的值进行调整.文献[36]采用的分析模型建模了控制信号与属性值的关系以及控制信号与软件配置的关系,基于第 1 个模型可以得到控制策略,即:如何根据当前属性值与目标属性之间的差异,选择控制信号;基于第 2 个模型得到软件配置,找到最佳软件配置被转化为了一个有约束的最优化问题.

基于强化学习的自适应规划方法在自适应场景与强化学习问题之间建立起映射关系,通过强化学习的算

法选择最优的自适应操作.将环境和软件状态映射为强化学习中的状态,将自适应操作映射为强化学习中的操作,将非功能性需求的满足度映射为强化学习中的奖励函数,通过强化学习可以得到一组使得奖励函数最大化的策略,其中,一条策略定义了一个状态与该状态下的最优操作的映射关系.文献[37,45,46]均采用了强化学习算法来进行自适应规划,其中,文献[46]采用了基于世界模型的强化学习,即:首先学习到环境模型,之后,基于该模型进行规划来获得最优策略.

- 小结

本节介绍了自适应规划方法.在选择适合的规划方法时,需要根据自适应软件的特点以及希望通过自适应达到的目的等进行考虑.若在设计阶段,开发者具备对于运行时的系统和环境的充足的知识,则可由利益相关者预先将自适应逻辑规约为自适应规则,采用基于规则的推理方式进行自适应规划.若自适应系统的动态性和不确定性较强,则不适合进行基于规则的规划,而更适合主动推理的规划方式.若自适应的目的是为了保证系统始终处于一个理想状态的集合中,则适合使用基于目标的推理方法;若自适应的目的是为了使某一表示系统状态理想程度的效用函数达到最优,则适合使用基于效用函数的最优化;若自适应的目的是为了将某一度量指标控制在理想值附近,则适合采用控制论的方法;若自适应的目的是为了保证某一属性的满足,且该属性可以被形式化并可被验证,则可采用模型检查或概率模型检查的方式进行规划;若自适应的目的是为了最优化即时奖励信号,可采用强化学习的方式进行规划.

4 总结与展望

本节对研究现状进行了分析,并在此基础上给出了未来的研究方向,主要关注 3 个方面的内容:建模方法研究、模型间的关系和自适应规划算法.

4.1 建模方法研究

对于基于模型的自适应方法来说,选择最恰当的建模方法对软件和环境进行建模是一个重要的研究课题.

在自适应软件系统的建模方面,相关工作采用了不同的模型,从某一方面上对软件的需求、软件的结构和行为、环境的状态和行为等进行了建模.但是,采用不同的模型对软件规约、环境和需求进行建模,不利于自适应系统进行基于模型的规划和验证.自适应软件需要通过自适应以保证 S(系统规约)、E(环境)、R(需求)被持续地满足,若能提出一个集成的建模框架,采用同一种模型从 3 个维度上对系统规约、环境、需求分别进行建模,则可为模型间关系的建立、基于模型的自适应系统的规划和验证过程提供更好的支持.

此外,现有的自适应系统工作对环境的建模较为简单.在环境的建模方面,相关工作大多数对环境的变化点和变体进行了简单的建模,对环境行为进行建模的工作较少,也缺乏对环境不确定性的显式表示.有少数的相关工作建模了环境的行为,例如文献[29]使用了符号转移系统,文献[32]使用了概率过程.但是文献[32]中的概率过程只考虑了一个单个的环境变量,文献[29]考虑的环境变化是确定的几个动作.

针对现有工作的问题,这里给出下面我们所建议的研究方向.

- 自适应系统的集成式建模框架.采用同一种模型(例如特征模型),从不同的维度上对系统规约、环境和需求进行建模.该集成式建模框架能够为基于模型的推理提供更好的支持;
- 环境行为和不确定性的建模.对环境的不缺定性进行建模,有利于做出考虑了不确定性的更合理的自适应规划.对环境可能的变化进行预测,可以在环境变化之前进行主动的自适应规划.

4.2 模型间的关系研究

这里主要考虑自适应方案与非功能性需求之间的关系、软件模型与环境模型之间的关系以及软件模型与系统间的同步.

4.2.1 备选自适应方案与非功能性需求之间关系的研究

在基于模型的自适应方法中,非功能性需求或软目标的实现程度通常作为评价备选自适应方案的标准.因此,更好地规约或者学习备选方案对软件目标实现程度的贡献是一个重要的研究课题.

文献[21]等采用了定性的目标模型建模备选的自适应方案(叶子级别的功能性目标)对非功能性需求的贡献.定性的目标模型是一种广泛使用的目标模型,但当被用于自适应系统的在线推理中时,往往会存在粒度太粗、无法很好地区分备选方案的问题.

文献[19,23]用实数代替了“+”“–”这样的定性关系,但并未给出系统的量化方法,而是假设这样的数字可以在设计阶段根据经验直接给出.文献[8,19,35,41]在变体上标注了备选方案对非功能目标满足度的影响,但同样是假设领域专家会在设计阶段给出影响值,并且假设这样的影响值在运行阶段不会发生变化.

文献[10,33]等可以通过在线学习学到特征对非功能性需求的影响,能够基于收集到的运行时的特征值和非功能性需求度量的观测值,通过线性回归、MD5 模型树等方法学习到特征对非功能性需求满足程度的影响.学习到的影响值会在运行时根据环境的具体情况进行即时的更新.文献[33]的学习结果是控制参数对度量值的影响函数,文献[10]的学习结果是特征对度量值的影响函数,这个函数可以是以环境变量的取值作为分段参数的分段函数.这两个工作都没有学习环境对于非功能性需求的度量值的影响.

针对现有工作的问题,这里给出下面我们所建议的研究方向.

- 将经验与学习算法相结合,得到备选方案与非功能性需求的关系.在设计阶段,可以采用基于 AHP 的量化方法等,由软件专家对备选方案和非功能性需求之间的关系进行量化;在运行阶段,可以采用在线学习算法,根据运行环境对量化值进行更新;
- 在规约或者量化备选方案对非功能性需求的影响时,将环境属性考虑进来.非功能性需求的满足程度是系统与环境相互作用的结果,软件配置对非功能性需求满足程度的影响,在不同环境情况下会有较大的差别.

4.2.2 软件模型与环境模型之间关系的研究

现有的研究工作一般通过规则来规约环境模型与软件模型之间的约束,通常使用的规则形式为“事件:条件→操作”(ECA 规则)或“条件→操作”(CA 规则).这样的规则存在很多缺点,包括:

1) 规则不能随需求的变化而动态演化.这是因为 ECA 形式的规则缺乏与需求之间的关系,因此很难以需求为依据对其进行动态的定制;
2) 很难检测冲突的发生.这是因为 ECA 形式的规则缺乏良好的结构化,即,每一条规则都是全局的,从而导致无法高效地在整个规则库上进行冲突消解.

同时,对于高可配置系统或者环境频繁变化的系统来说,由人工定义自适应规则非常困难.这是因为:(1) 在环境状态和软件配置的数量都非常巨大的情况下,无法用一组较少的人工定义的规则集来覆盖它们之间复杂的对应关系;(2) 由于规则与需求之间缺乏联系,很难对定义出的规则集的质量进行验证;(3) 当规则中蕴含的预设知识与环境动态性所反映的真实知识不相符时,这些在设计时预先定义的规则就无法在运行时带来较好的自适应结果;(4) 当需求发生变化或者环境发生非稳态的变化时,旧的规则集就会失效.

针对现有工作的问题,这里给出下面我们所建议的研究方向.

- 提出一种新的规则,同时,对需求、软件规约和环境的关系进行规约.每条规则的触发条件不仅包括环境情况,还包括其能够维护的需求.当采用这样的规则集时,处于激活状态的规则子集会因需求设定的不同而不同;
- 提出能够从目标和环境出发,自主地生成自适应规则(即环境模型与软件模型间的约束关系)的算法.

4.2.3 软件模型与系统模型之间同步关系的研究

基于模型的自适应系统是目前自适应系统主流的研究方向.为了弥合模型与实现间的鸿沟,相关工作提出了一些支持的框架与中间件,它们可以根据系统的设计模型自动地生成代码.但是现有中间件支持的模型类型和同步类型仍然很有限,尤其是缺乏对以下两个问题的支持:(1) 如何根据设计模型,将现有的非自适应的传统系统再造为自适应系统;(2) 如何根据设计模型,将自适应系统与非自适应系统结合起来.因此,研究如何弥合自适应系统的模型与实现间的鸿沟、支持更安全的模型与系统间的同步方式、处理复杂场景下的模型与系统间的同步,是一个可能的未来研究方向.

4.3 自适应规划算法

自适应系统需要具备自主进行自适应规划的能力,即:当变化发生后,能够自主地选择一组自适应操作,以使得 S(软件配置)、E(环境状态)、R(用户需求)被持续地满足.自适应规划应当能够响应以下两种变化:(1) 环境的变化;(2) 需求的变化.其中,环境的变化可能是稳态的或非稳态的,稳态的变化是指环境状态的变化,非稳态的变化是指环境状态迁移概率的变化或者环境状态对需求影响的变化;需求的变化是指需求的增加或者减少以及各需求优先级的变化.

表 3 展示了现有的规划算法.这些规划算法可以划分为反射式的算法和主动推理的算法.反射式的规划算法通常依赖于预先规约好的自适应规则,具有规划过程高效、规则易读性强且易修改的优点.但是这类方法采用的自适应规则通常是在离线阶段预先定义的,并且缺乏与高层需求的联系.因此,反射式的规划算法只能响应稳态的环境变化,而无法响应需求的变化和非稳态的环境变化.相反地,采用主动推理算法的自适应系统会根据运行时的环境情况和决策标准推理出最适合的操作.因此,主动推理算法更有可能做出有效的自适应规划.但是在有多个需求、多个控制参数或多个备选方案的情况下,主动推理算法具有较高的复杂性.在备选方案空间庞大的情况下,基于效用函数的最优化、基于目标的推理、概率模型检查、控制模型的合成等规划算法都会面临低效性的问题,有可能无法及时响应运行时的变化.

综上,如何提高现有的动态规划算法、设计出结合反射式与主动推理自适应算法优点的新算法,是一个重要的研究点.理想的规划算法应当具备以下能力:(1) 有效,即,能够自主地选择自适应操作去响应环境和需求的变化,并且操作的执行能够保证需求被持续满足;(2) 高效,即,规划过程应当高效,能够及时地对运行时的变化做出响应.

5 结 论

本文对现有的基于模型的自适应方法的相关研究工作进行了综述.首先,提出了 6 个研究问题,关注相关工作如何进行建模、分析和规划等;然后,对现有的基于模型的自适应方法的相关研究工作进行了分析,对提出的 6 个研究问题进行了回答;最后,基于对现有方法的分析,指出现有研究中存在的问题和不足,提出了解决这些问题和不足的设想,还给出了与基于模型的自适应方法相关的未来的研究方向.

致谢 在此,我们向对本文的工作给予支持和建议的老师和同学表示感谢.

References:

[1] Cheng BH, Lemos R, Giese H, Inverardi P, Magee J, Andersson J, Becker B, Bencomo N, Brun Y, Cukic B, Serugendo GM, Dustdar S, Finkelstein A, Gacek C, Geihs K, Grassi V, Karsai G, Kienle HM, Kramer J, Litoiu M, Malek S, Mirandola R, Müller HA, Park S, Shaw M, Tichy M, Tivoli M, Weyns D, Whittle J. Software engineering for self-adaptive systems: A research roadmap. In: Proc. of the Software Engineering for Self-Adaptive Systems. Springer-Verlag, 2009. 1–26. [doi: 10.1007/978-3-642-02161-9_1]

[2] de Lemos R, Giese H, Müller HA, Shaw M, Andersson J, Litoiu M, Schmerl B, Tamura G, Villegas NM, Vogel T, Weyns D, Baresi L, Becker B, Bencomo N, Brun Y. Software engineering for selfadaptive systems: A second research roadmap. In: Proc. of the Software Engineering for Self-Adaptive Systems II. Dagstuhl Castle: Int'l Seminar, 2010. [doi: 10.1007/978-3-642-35813-5_1]

[3] Wang QX, Shen JR, Mei H An introduction to self-adaptive software. Computer Science, 2004,31(10):168−171 (in Chinese with English abstract). [doi: 10.3969/j.issn.1002-137X.2004.10.045]

[4] Brun Y, Serugendo GDM, Gacek C, Giese H, Kienle H, Litoiu M, Müller H, Pezzè M, Shaw M. Engineering self-adaptive systems through feedback loops. In: Proc. of the Software Engineering for Self-Adaptive Systems. Berlin, Heidelberg: Springer-Verlag, 2009. 48−70. [doi: 10.1007/978-3-642-02161-9_3]

[5] Feng YD, Huang G, Mei H. A method for modeling and realizing self-adaptive software architecture. Acta Scientiarum Naturalium Universitatis Pekinensis, 2008,44(1):67−76 (in Chinese with English abstract). [doi: 10.3321/j.issn:0479-8023.2008.01.014]

[6] Zhao XP, Li MS, Wang Q, Chen ZC, Liang JN. An agent-based self-adaptive software process model. Ruan Jian Xue Bao/Journal of Software, 2004,15(3):348−359 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/20040304.htm

[7] Morin B, Barais O, Jezequel JM, Fleurey F, Solberg A. Models@ run. Time to support dynamic adaptation. Computer, 2009, 42(10).

[8] Floch J, Hallsteinsen S, Stav E, Eliassen F, Lund K, Gjorven E. Using architecture models for runtime adaptability. IEEE Software, 2006,23(2):62−70. [doi: 10.1109/MS.2006.61]

[9] Blair G, Bencomo N, France RB. Models@ run. Time. Computer, 2009,42(10). [doi: 10.1109/MC.2009.326]

[10] Elkhodary A, Esfahani N, Malek S. FUSION: A framework for engineering self-tuning self-adaptive software systems. In: Proc. of the 18th ACM SIGSOFT Int'l Symp. on Foundations of Software Engineering. ACM Press, 2010. 7−16. [doi: 10.1145/1882291.1882296]

[11] Cetina C, Giner P, Fons J, Pelechano V. Autonomic computing through reuse of variability models at runtime: The case of smart homes. Computer, 2009,42(10):37−43. [doi: 10.1109/MC.2009.309]

[12] Swanson J, Cohen MB, Dwyer MB, Garvin B, Firestone J. Beyond the rainbow: Self-Adaptive failure avoidance in configurable systems. In: Proc. of the 22nd ACM SIGSOFT Int'l Symp. on Foundations of Software Engineering. ACM Press, 2014. 377−388. [doi: 10.1145/2635868.2635915]

[13] Alferez GH, Pelechano V. Context-Aware autonomous Web services in software product lines. In: Proc. of the 15th Int'l Software Product Line Conf. (SPLC). IEEE, 2011. 100−109. [doi: 10.1109/SPLC.2011.21]

[14] Acher M, Collet P, Fleurey F, Lahire P, Moisan S, Rigault J. Modeling context and dynamic adaptations with feature models. In: Proc. of the 4th Int'l Workshop Models@ run. Time at Models 2009 (MRT 2009). 2009. 10. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.204.5778

[15] Fleurey F, Delhen V, Bencomo N, Morin B, Jézéquel JM. Modeling and validating dynamic adaptation. In: Proc. of the 3rd Int'l Workshop on Models@Runtime, at MoDELS 2008. Toulouse, 2008. [doi: 10.1007/978-3-642-01648-6_11]

[16] Ghezzi C, Sharifloo AM. Dealing with non-functional requirements for adaptive systems via dynamic software product-lines. In: Proc. of the Software Engineering for Self-Adaptive Systems II. Berlin, Heidelberg: Springer-Verlag, 2013. 191−213. [doi: 10.1007/978-3-642-35813-5_8]

[17] Zhao TQ, Zan T, Zhao HY, Hu ZJ, Jin Z. Integrating Goal model into rule-based adaptation. In: Proc. of the 23rd Asia-Pacific Software Engineering Conf. 2016. 289−296. [doi: 10.1109/APSEC.2016.048]

[18] Bencomo N, Sawyer P, Blair GS, Grace P. Dynamically adaptive systems are product lines too: Using model-driven techniques to capture dynamic variability of adaptive systems. In: Proc. of the Int'l Conf. Software Product Lines, Vol.2. Limerick, 2008. 23−32. https://link.springer.com/chapter/10.1007/978-3-642-35813-5_8

[19] Chen B, Peng X, Yu Y, Nuseibeh B, Zhao W. Self-Adaptation through incremental generative model transformations at runtime. In: Proc. of the 36th Int'l Conf. on Software Engineering. ACM Press, 2014. 676−687. [doi: 10.1145/2568225.2568310]

[20] Qian W, Peng X, Chen B, Mylopoulos J, Wang H, Zhao W. Rationalism with a dose of empiricism: Case-Based reasoning for requirements-driven self-adaptation. In: Proc. of the 22nd IEEE Int'l Requirements Engineering Conf. (RE). IEEE, 2014. 113−122. [doi: 10.1109/RE.2014.6912253]

[21] Lapouchnian A, Yu Y, Liaskos S, Mylopoulos J. Requirements-Driven design of autonomic application software. In: Proc. of the 2006 Conf. of the Center for Advanced Studies on Collaborative Research. IBM Corp., 2006. 7. [doi: 10.1145/1188966.1188976]

[22] Cheng BHC, Sawyer P, Bencomo N, Whittle G. A goal-based modeling approach to develop requirements of an adaptive system with environmental uncertainty. In: Proc. of the Int'l Conf. on Model Driven Engineering Languages and Systems. Berlin, Heidelberg: Springer-Verlag, 2009. 468−483. [doi: 10.1007/978-3-642-04425-0_36]

[23] Chen B, Peng X, Yu Y, Zhao W. Are your sites down? Requirements-Driven self-tuning for the survivability of Web systems. In: Proc. of the 19th IEEE Int'l Requirements Engineering Conf. (RE). IEEE, 2011. 219−228. [doi: 10.1109/RE.2011.6051650]

[24] Bencomo N, Belaggoun A. Supporting decision-making for self-adaptive systems: From goal models to dynamic decision networks. In: Proc. of the REFSQ 2013. 2013. 221−236. [doi: 10.1007/978-3-642-37422-7_16]

[25] Calinescu R, Ghezzi C, Kwiatkowska M, Mirandola R. Self-Adaptive software needs quantitative verification at runtime. Communications of the ACM, 2012,55(9):69−77. [doi: 10.1145/2330667.2330686]

[26] Ghezzi C, Pinto LS, Spoletini P, Tamburrelli G. Managing non-functional uncertainty via model-driven adaptivity. In: Proc. of the 2013 Int'l Conf. on Software Engineering. IEEE Press, 2013. 33−42. [doi: 10.1109/ICSE.2013.6606549]

[27] D'Ippolito N, Braberman V, Kramer J, Magee J, Sykes D, Uchitel S. Hope for the best, prepare for the worst: Multi-Tier control for adaptive systems. In: Proc. of the 36th Int'l Conf. on Software Engineering. ACM Press, 2014. 688−699. [doi: 10.1145/2568225.2568264]

[28] Sykes D, Corapi D, Magee J, Inoue K. Learning revised models for planning in adaptive systems. In: Proc. of the 35th Int'l Conf. on Software Engineering (ICSE). IEEE, 2013. 63−71. [doi: 10.1109/ICSE.2013.6606552]

[29] Tsigkanos C, Pasquale L, Ghezzi C, Nuseibeh B. On the interplay between cyber and physical spaces for adaptive security. IEEE Trans. on Dependable and Secure Computing, 2016. [doi: 10.1109/TDSC.2016.2599880]

[30] Vogel T, Giese H. A language for feedback loops in self-adaptive systems: Executable runtime mega models. In: Proc. of the 2012 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). IEEE, 2012. 129−138. [doi: 10.1109/SEAMS.2012.6224399]

[31] Grassi V, Mirandola R, Randazzo E. Model-Driven assessment of QoS-aware self-adaptation. In: Proc. of the Software Engineering for Self-Adaptive Systems. Berlin, Heidelberg: Springer-Verlag, 2009. 201−222. [doi: 10.1007/978-3-642-02161-9_11]

[32] Moreno GA, Cámara J, Garlan D, Schmerl B. Proactive self-adaptation under uncertainty: A probabilistic model checking approach. In: Proc. of the 10th Joint Meeting on Foundations of Software Engineering. ACM Press, 2015. 1−12. [doi: 10.1145/2786805.2786853]

[33] Esfahani N, Kouroshfar E, Malek S. Taming uncertainty in self-adaptive software. In: Proc. of the 19th ACM SIGSOFT Symp. and the 13th European Conf. on Foundations of Software Engineering. ACM Press, 2011. 234−244. [doi: 10.1145/2025113.2025147]

[34] Filieri A, Hoffmann H, Maggio M. Automated design of self-adaptive software with control-theoretical formal guarantees. In: Proc. of the 36th Int'l Conf. on Software Engineering. ACM Press, 2014. 299−310. [doi: 10.1145/2568225.2568272]

[35] Rosa L, Rodrigues LET, Lopes A. Self-Management of distributed systems using high-level Goal policies. In: Proc. of the Software Engineering for Self-Adaptive Systems. 2010. 162−190. [doi: 10.1007/978-3-642-35813-5_7]

[36] Filieri A, Hoffmann H, Maggio M. Automated multi-objective control for self-adaptive software design. In: Proc. of the 10th Joint Meeting on Foundations of Software Engineering. ACM Press, 2015. 13−24. [doi: 10.1145/2786805.2786833]

[37] Kim DS, Park SY. Reinforcement learning-based dynamic adaptation planning method for architecture-based self-managed software. In: Proc. of the SEAMS 2009. 2009. 76−85. [doi: 10.1109/SEAMS.2009.5069076]

[38] Tajalli H, Garcia J, Edwards G, Medvidovic N. PLASMA: A plan-based layered architecture for software model-driven adaptation. In: Proc. of the IEEE/ACM Int'l Conf. on Automated Software Engineering. ACM Press, 2010. 467−476. [doi: 10.1145/1858996.1859092]

[39] Pietschmann S. A model-driven development process and runtime platform for adaptive composite Web applications. Int'l Journal on Advances in Internet Technology, 2010,2:277−299.

[40] Morin B, Fleurey F, Bencomo N, Jezequel J, Solberg A, Dehlen V, Blair G. An aspect-oriented and model-driven approach for managing dynamic variability. In: Proc. of the Int'l Conf. on Model Driven Engineering Languages and Systems. Berlin, Heidelberg: Springer-Verlag, 2008. 782−796. [doi: 10.1007/978-3-540-87875-9_54]

[41] Rouvoy R, Barone P, Ding Y, Eliassen F, Hallsteinsen S, Lorenzo J, Mamelli A, Scholz U. Music: Middleware support for self-adaptation in ubiquitous and service-oriented environments. In: Proc. of the Software Engineering for Self-Adaptive Systems. Berlin, Heidelberg: Springer-Verlag, 2009. 164−182. [doi: 10.1007/978-3-642-02161-9_9]

[42] Garlan D, Cheng SW, Huang AC, Schmerl B, Steenkiste P. Rainbow: Architecture-Based self-adaptation with reusable infrastructure. Computer, 2004,37(10):46−54. [doi: 10.1109/MC.2004.175]

[43] Geihs K, Reichle R, Wagner M, Khan MU. Modeling of context-aware self-adaptive applications in ubiquitous and service-oriented environments. In: Proc. of the Software Engineering for Self-Adaptive Systems. Berlin, Heidelberg: Springer-Verlag, 2009. 146−163. [doi: 10.1007/978-3-642-02161-9_8]

[44] Salehie M, Pasquale L, Omoronyia I, Ali R, Nuseibeh B. Requirements-Driven adaptive security: Protecting variable assets at runtime. In: Proc. of the 20th IEEE Int'l Requirements Engineering Conf. (RE). IEEE, 2012. 111−120. [doi: 10.1109/RE.2012.6345794]

[45] Amoui M, Salehie M, Mirarab S, Tahvildari L. Adaptive action selection in autonomic software using reinforcement learning. In: Proc. of the 4th Int'l Conf. on Autonomic and Autonomous Systems. 2008. [doi: 10.1109/ICAS.2008.35]

[46] Ho HN, Lee E. Model-Based reinforcement learning approach for planning in self-adaptive software system. In: Proc. of the 9th Int'l Conf. on Ubiquitous Information Management and Communication (IMCOM 2015). 2015. [doi: 10.1145/2701126.2701191]

[47] Fernandes P, Werner C, Murta LGP. Feature modeling for context-aware software product lines. In: Proc. of the 20th Int'l Conf. on Software Engineering & Knowledge Engineering (SEKE 2008). World Scientific, 2008. 758−763. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.525.1191

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

[49] Chung L, Nixon B, Yu E, Mylopoulos J. Non-Functional Requirements in Software Engineering. Kluwer, 2000. http://www.springer.com/us/book/9780792386667

[50] Mylopoulos J, Chung L, Nixon B. 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]

[51] Darimont R, Delor E, Massonet P, Lamsweerde AV. GRAIL/KAOS: An environment for Goal-driven requirements engineering. In: Proc. of the 19th Int'l Conf. on Software Engineering. ACM Press, 1997. 612−613. [doi: 10.1145/253228.253499]

[52] Whittle J, Sawyer P, Bencomo N, Cheng BHC, Bruel JM. Relax: Incorporating uncertainty into the specification of self-adaptive systems. In: Proc. of the 17th IEEE Int'l Requirements Engineering Conf. 2009. (RE 2009). IEEE, 2009. 79−88. [doi: 10.1109/RE.2009.36]

[53] Vardi MY. An automata-theoretic approach to linear temporal logic. In: Proc. of the Logics for Concurrency. Berlin, Heidelberg: Springer-Verlag, 1996. 238−266. [doi: 10.1007/3-540-60915-6_6]

[54] Van Lamsweerde A. Goal-Oriented requirements engineering: A guided tour. In: Proc. of the 5th IEEE Int'l Symp. on Requirements Engineering. IEEE, 2001. 249−262. [doi: 10.1109/ISRE.2001.948567]

[55] Batory D. Feature models, grammars, and propositional formulas. In: Proc. of the Int'l Conf. on Software Product Lines. Berlin, Heidelberg: Springer-Verlag, 2005. 7−20. [doi: 10.1007/11554844_3]

[56] Kang KC, Cohen SG, Hess JA, Novak WE, Peterson AS. Feature oriented design analysis (FODA) feasibility study. Technical Report, CMU/SEI-90-TR-21-ESD-90/TR-222, Carnegie Mellon University, 1990.

[57] Roos-Frantz F, Benavides D, Ruiz-Cortés A, Heuer A, Lauenroth K. Quality-Aware analysis in product line engineering with the orthogonal variability model. Software Quality Journal, 2012,20(3-4):519−565. [doi: 10.1007/s11219-011-9156-5]

[58] Clements P, Bachmann F, Bass L, Garlan D, Ivers J, Little R, Merson P, Nord R, Stafford J. Documenting Software Architectures: Views and Beyond. Addison-Wesley, 2003. https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=30386

[59] Rumbaugh J, Jacobson I, Booch G. Unified Modeling Language Reference Manual. Pearson Higher Education, 2004. https://dl.acm.org/citation.cfm?id=993859

[60] Kemeny JG, Snell JL. Finite Markov Chains. Princeton: van Nostrand, 1960.

[61] Kwiatkowska M, Norman G, Parker D. Prism: Probabilistic model checking for performance and reliability analysis. ACM Performance Evaluation Review, 2009,36(4):40−45. [doi: 10.1145/1530873.1530882]

[62] Gu T, Pung HK, Zhang DQ. Toward an OSGi-based infrastructure for context-aware applications. IEEE Pervasive Computing, 2004,3(4):66−74. [doi: 10.1109/MPRV.2004.19]

[63] Lee EB, Markus L. Foundations of optimal control theory. Journal of the Royal Statistical Society, 1967, 589. [doi: 10.2307/2343766]

附中文参考文献:

[3] 王千祥,申峻嵘,梅宏.自适应软件初探.计算机科学,2004,31(10):168−171. [doi: 10.3969/j.issn.1002-137X.2004.10.045]

[5] 冯耀东,黄罡,梅宏.一种自适应软件体系结构建模及其实施方法.北京大学学报(自然科学版),2008,44(1):67−76. [doi: 10.3321/j.issn:0479-8023.2008.01.014]

[6] 赵欣培,李明树,王青,陈振冲,梁金能.一种基于 Agent 的自适应软件过程模型.软件学报,2004,15(3):348−359. http://www.jos.org.cn/1000-9825/20040304.htm

赵天琪(1990－),女,山西运城人,博士,主要研究领域为需求工程,自适应系统,强化学习.

张伟(1978－),男,博士,副教授,主要研究领域为群体化软件开发方法,软件需求工程.

赵海燕(1966－),女,博士,副教授,CCF 高级会员,主要研究领域为需求工程,软件复用,程序语言.

金 芝(1962－),女,博士,教授,博士生导师,CCF 会士,主要研究领域为需求工程,知识工程.