DOI:10.11992/tis.201507043
网络出版地址:http://www.cnki.net/kcms/detail/23.1538.tp.20151110.1354.022.html

基于可拓学的自适应软件形式化方法

范锐[1],彭银桥[1],陈月峰[1],雷桂斌[1],刘小辉[2]
(1.广东海洋大学 软件学院,广东 湛江 524088; 2.广东海洋大学 财务处,广东 湛江 524088)

摘 要:可拓学的核心是建立灵活变通地应对不确定变化和灵感涌现的适应性模型。讨论引入可拓理论去描述、分析和评价软件系统的自适应性质、范围和程度的可能性。用基元描述软件实体,将软件系统构造成基元网,利用拓展分析、可拓变换和优度评价等定性与定量相结合的方法揭示了自适应软件系统的动态性质,并形成了一种自适应软件形式化方法。

关键词:可拓学;可拓基元;拓展分析;可拓变换;优度评价;形式化方法;自适应软件
中图分类号: TP311 **文献标志码**:A **文章编号**:1673-4785(2015)06-0901-11

中文引用格式:范锐,彭银桥,陈月峰,等. 基于可拓学的自适应软件形式化方法[J]. 智能系统学报, 2015, 10(6): 901-911.
英文引用格式:**FAN Rui, PENG Yinqiao, CHEN Yuefeng, et al. A method for self-adaptive software formal modeling by Extenics [J]. CAAI Transactions on Intelligent Systems**, 2015, 10(6): 901-911.

A method for self-adaptive software formal modeling by Extenics

FAN Rui[1], PENG Yinqiao[1], CHEN Yuefeng[1], LEI Guibin[1], LIU Xiaohui[2]
(1. Software School, Guangdong Ocean University, Zhanjiang 524088, China; 2. Finance Department, Guangdong Ocean University, Zhanjiang 524088, China)

Abstract: The core of extenics is to set up adaptability mathematic model which can flexibly cope with uncertain change and inspiration springing up. The feasibility of introducing the theory of Extenics to describe, analyze and evaluate the self-adaptive nature, scope and extent about the software system was discussed. First, the basic-element description software entities were used to construct the software system into basic-element net, then the dynamic nature of self-adaptive software system was revealed by combining the qualitative and quantitative methods of extension analysis, extension transformation, and superiority evaluation, etc., initially establishing a formal method of self-adaptive software.

Keywords: Extenics; basic-element; extension analysis; extension transformation; superiority evaluation; formal method; self-adaptive software

软件自适应是指为了保证持续、高质量地提供服务,软件在运行时检测环境变化和自身状态,据此对自身行为进行主动调整的活动[1]。自适应软件能够评估自身的行为,当评估显示其自身并非正接近所倾向完成的目标,或其有可能提供更好的功能或性能时,软件改变自身的行为[2]。文献[1]对已有的研究和实践进行综述给出软件自适应的概念内涵,概述软件自适应活动在感知、决策、执行各环节上的特征分类,阐述面向自适应软件构造、关注程度较高的一系列使能技术,进而在分析典型研究项目现状的基础上,给出自适应软件构造领域的未来主要研究趋势。文献[3]从计算机科学视角对许多自适应软件相关论文的研究分析,总结并指出自适应系统新的观点、概念、方法和新的挑战,特别强调了

收稿日期:2015-07-28. 网络出版日期:2015-11-10.
基金项目:广东省科技计划资助项目(2014A040402010).
通信作者:范锐. E-mail:fanrui@gdou.edu.cn.

计算机软件在自适应系统已成为主要元素,也强调了反馈控制技术和人工智能技术对自适应软件系统的重要性。文献[4]从移动计算和普适计算的角度总结、评价存在的自适应软件方法,重点在资源受限器件软件动态演化的解决方法的分析、评估和进展研究,并提出其发展趋势和挑战。文献[5-9]从自适应需求的原理和概念框架、自适应系统建模和模型演化机制、运行时系统自适应调节机制、社会的角度看软件的自适应等4个方面分别回答软件系统为什么需要和如何拥有自适应能力问题。文献[10-14]分别从不同角度进行了自适应软件系统模型的研究。可拓学是研究事物的可拓性以及可拓规律与方法,并用以解决矛盾问题的原创性学科。经过30多年的发展,可拓学在基础理论、形式化方法、行业应用等方面都有长足进展,并逐步发展为可拓学[15-17]。它用形式化方法描述问题的目标和条件,建立核模型,用灵活多变的方法对问题进行拓展分析,用定量与定性结合的可拓变换推导出解决矛盾问题的思路、知识和方法,最终生成创意或创新策略。它用“变不知为知”、“变错误为正确”、“变不行为行”、“变不属于为属于”的变通思想,创造出了包括基本方法,创意生成方法,可拓数据挖掘方法和可拓思维模式等可拓创新方法体系[17]。可拓学用数学模型来探索人类创新创意规律,并指导开发出能够拓展智能、激发创新的可拓软件。应用这种灵活变通地应对不确定变化和灵感涌现的适应性模型来探索自适应软件形式化建模方法应是一条新的研究路径。

1 相关工作

当前研究自适应软件最活跃的自适应实体是构件、服务和智能体技术。构件通过对外提供接口实现软件系统的组装,服务通过组合构造更大、更松散的构件,智能体通过外界感知、智能推理和行为优化去响应外界的变化和用户的需求。建立模拟灵活变通智能的软件模型,包括独立自治、动态演化的软件实体以及由它们智能聚合的软件系统,从来就是软件学术界和工业界的研究热点。

基于构件的软件工程(component-based software engineering),目的在于借鉴其他制造工业的产品零配件组装生产方式,通过零配件的分工生产和组装来大幅度提高生产力,提高满足不同用户不同需求的灵活性。软件构件的内部封装、接口交互、动态替换等特性极大地提升了软件系统适应变化的灵活性。目前有许多的构件模型,例如,COM+、EJB、CORBA等。支撑软件构件集成的软件体系结构研究是随着描述大型、复杂系统结构的需要和开发人员及计算机科学家在大型软件系统的研制过程中对软件系统理解的逐步深入而发展起来的[18]。业界提出了Darwin、Wright、ACME、A-ADL、XYZ/ADL、Tracer和ABC/ADL等软件体系结构描述语言,并使用CRAM[19]、Z语言[20]、Pi演算[21-22]、Bigraph[23-26]等形式化方法试图准确描述软件体系结构。随着Internet的普及,网构软件、网络式软件等新的软件构件实体具有自治开放、动态演化等新的特点[27-30]。但是,软件构件建模和形式化方法重点关注软件实体和由它们构造的软件系统为适应变化如何改变结构且保证行为的一致性,没考虑软件与环境或需求不相适应而持续动态演化致使行为不一致需要解决矛盾冲突的智能化变通策略。

面向服务的软件工程(service-based software engineering),以服务为软件实体,通过服务重用和快速组合构建随需应变的松耦合分布式系统[31]。核心是SOA(service oriented architecture),它通过服务注册把服务提供者和服务请求者联系起来,形成动态松耦合的分布式系统。Web服务组合技术将平台独立、自描述、位置透明的软件模块进行组合,快速灵活地满足复杂多变的业务需求[32]。流行的服务组合语言是BPEL4WS[33],而组合分析验证的工作有:文献[34]借鉴颜色Petri网的语义和面向对象思想,提出WS-Net模型,使验证更容易。文献[35]引入基于BPEL4WS的BPE演算。使用PAC工具和CWB-NC工具来建模和验证Web服务协作。文献[36]用Pi演算作为服务组合的形式化表示。基于服务的软件系统构造成为了当前软件开发的主流。虽然,服务技术克服了构件实体面对多平台,多协议、多设备的Internet复杂环境,由于紧偶合、细粒度等缺点不能胜任分布式、异构性、非稳定的网络计算的困难。但是,服务是典型的被动实体,没有自治、智能特性,当然更没有灵活变通的解决矛盾问题的可拓智能。

软件Agent具有的自主性、反应性、能动性、社会性等特点,是解决软件构件技术不足的新途径。基于Agent的软件工程(agent-based software engineering)取得了许多研究成果[37-39]。中国科学院计算技术所智能科学课题组开展包括智能主体、机器学习、神经计算、认知科学以及相关应用研究[40];国防科技大学开展了面向主体软件开发研究[41]、基于智能

体技术的软件自适应动态演化机制研究[42]。但是,Agent 并没有成为软件开发的主流技术,也不能够构建可拓创新软件模型。

围绕研制具有灵活变通的可拓软件,提出了可拓信息-知识-智能形式化体系,提出了信息的基元表示方法、基于可拓规则的知识表示方法和策略的形式化表示方法,为用计算机进行矛盾问题智能化处理构建了形式化工具[43]。研究者开发了一些获得发明专利、软件著作权以及会议展示的可拓软件[16],研制了可拓策略生成系统,开发了一批可拓策略生成软件[44-46];研制了可拓数据挖掘软件,开发了可拓挖掘软件[47]。在可拓设计应用领域,应用可拓理论与方法对概念设计功能、原理、布局、形状、结构等上游设计知识进行形式化描述,提出一种基于多级菱形思维模型的复杂产品定性和定量相结合的设计方法,为概念设计的形式化和智能化提供了新的途径[48-50]。研究可拓学与建筑设计的结合问题,将可拓策划方法应用于景观设计和建筑设计创新与可拓思维模式,进行了可拓建筑策划与设计的系统研究[51-52]。研究智能知识管理,构建了企业自主创新的可拓创新模型[53-54]。从软件构造角度就可拓策略生成系统的实现进行了研究。结合本体和 Agent 建立不相容问题的策略生成、利用转换桥建立化对立为共存的策略生成,基于可拓集的可拓数据挖掘、利用传导推理建立矛盾问题的转化等一批可拓创新智能化方法取得了进展[55]。研究者利用知网增强策略生成[56]、用认知概念构建可拓变换结构[57]、研究过程基因基元化方法[58]、基于可拓方法构建社会网络[59]等,从不同角度为可拓软件模型构建提供了新的思路。

可拓学的目标是探索人类创新创意规律,并构建数学模型来指导开发拓展智能、激发创新的可拓软件。该文尝试把可拓学理论应用于自适应软件形式化建模中,从把自适应软件构造成基元网出发,利用可拓基元的拓展分析方法和共轭分析方法,分析自适应软件实体及他们之间关系的动态变化,利用可拓变换去探讨他们的动态演化性质、趋势和程度,并利用优度评价方法评价软件的自适应能力。通过这种新的理论与方法,可以用定性与定量相结合的方法描述、分析、演化和评估自适应软件,为形式化建模提供一种新的方法。

2　自适应软件的可拓形式化方法

自适应软件系统可以看作一组已经或有可能(直接或间接)连接的“点”(软件实体),以及“点”之间的关系构成的集合。将自适应系统中各部分(“点”)的特征与相互之间的关系(“连接”)用网络的形式表示出来,然后分析其特征和关系以及按需演化的理论方法就是自适应软件形式化方法。引入可拓理论来研究和构建自适应软件形式化模型,首先用基元表示自适应软件系统的各层次软件实体,建立基元网描述各层次软件实体间的关系;再用可拓拓展推理去扩展、挖掘和推导出各层次软件实体以及它们之间关系的丰富语义;用可拓变换去分析、演算和评估软件实体演化对自身、对相关软件实体、对构成网络的变化、趋势和程度;用关联函数定量计算软件实体演化的趋势和程度以及对环境的影响趋势和程度;最后,结合丰富多样的创新创意方法去解决环境变化或用户需求变化导致软件系统不相适应的矛盾,生成应对变化的策略,通过最优评估,选出最佳策略执行,实现软件系统的动态演化。

2.1　智能服务构件要素基元化

用智能服务构件作为自适应软件实例来说明可拓形式化方法[60]。

把智能服务构件内部各层次软件实体以及它们的关系用图 1 表示。然后用可拓基元 B 表示智能服务构件内部是由 ID、接口 I、控制 C、事件 E、目标 G、规划 P、知识库 K、服务 S 等软件实体组成。基于对其组成元素自适应性质进行分析、变换和评价的需要,进一步细化了接口基元 I、控制基元 C、事件基元 E、目标基元 G、规划基元 P、知识库基元 K 和服务基元 S 等的内部结构。

用可拓基元(事物,属性,属性值)三元组可以把事物的质和量有机地结合起来去丰富其语义。通过对基元反复细化,能够实现从分析到设计全过程形式化表示、分析、变换和控制。

$$B=\begin{bmatrix} \text{智能服务构件} & \text{标号} & \text{ID} \\ & \text{接口} & I \\ & \text{控制} & C \\ & \text{事件} & E \\ & \text{目标} & G \\ & \text{规划} & P \\ & \text{知识库} & K \\ & \text{服务} & S \end{bmatrix}$$

$$G=\begin{bmatrix} \text{目标} & \text{标号} & \text{ID} \\ & \text{子目标} & \text{SG} \end{bmatrix}$$

$$E=\begin{bmatrix} \text{事件} & \text{内部事件} & E_1 \\ & \text{外部事件} & E_2 \end{bmatrix}$$

$$I=\begin{bmatrix}\text{接口} & \text{提供接口} & I_1\\ & \text{请求接口} & I_2\\ & \text{感知} & I_3\\ & \text{效应} & I_4\\ & \text{属性} & I_5\\ & \text{设置} & I_6\end{bmatrix}$$

$$C=\begin{bmatrix}\text{控制} & \text{知识管理} & C_1\\ & \text{事件管理} & C_2\\ & \text{解释器} & C_3\\ & \text{目标管理} & C_4\\ & \text{服务管理} & C_5\\ & \text{规划管理} & C_6\\ & \text{执行} & C_7\end{bmatrix}$$

$$P=\begin{bmatrix}\text{规划} & \text{标号 ID} & \\ & \text{前置条件} & P_1\\ & \text{规划体} & P_2\\ & \text{后置条件} & P_3\end{bmatrix}$$

$$K=\begin{bmatrix}\text{知识库} & \text{规则} & K_1\\ & \text{信念} & K_2\\ & \text{本体} & K_3\\ & \text{领域} & K_4\end{bmatrix}$$

$$S=\begin{bmatrix}\text{服务} & \text{请求服务} & S_1\\ & \text{提供服物} & S_2\\ & \text{组合服务} & S_3\\ & \text{原子服务} & S_4\end{bmatrix}$$

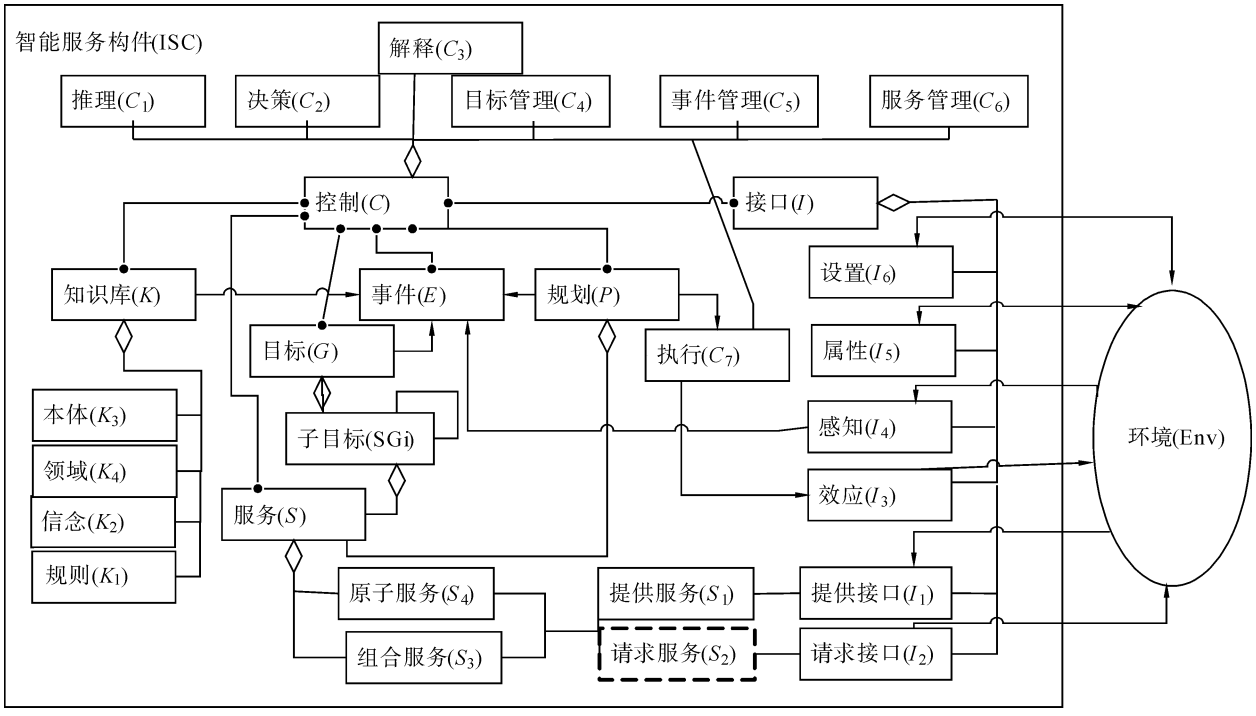


图 1 智能服务构件(ISC)内部元素关系图

Fig.1 Intelligent service component relation

2.2 智能服务构件基元网及语义分析

图论中用点之间的连线代表个体之间的关系,包括单向关系和双向关系。图 1 中 2 个基元 B_i 与 B_j 之间的关系(图中的连接线)用关系元 R_x 表示。$\vec{R}_x$ 和 $\overleftarrow{R}_x$ 表示单向关系,$\overleftrightarrow{R}_x$ 表示双向关系,x 表示关系名。关系的密切度可以用程度衡量,则图 1 的基元关系可形式化表示为图 2 的基元网。它给出了智能服务构件丰富的关系语义。

$$R_x=\begin{bmatrix}\text{关系 } x & \text{前项} & B_i\\ & \text{后项} & B_j\\ & \text{方向} & \text{Direction}\\ & \text{程度} & \text{Extent}\end{bmatrix}$$

2.2.1 静态语义分析

1)控制关系($\overleftrightarrow{R}_{\text{control}}$):控制 C_3 控制 C_1、C_2、C_4、C_5、C_6、C_7 等 6 个子控制,即,$\overleftrightarrow{R}_{\text{control}}(C_3,(C_1,C_2,C_4,C_5,C_6,C_7))$;而每一个子控制又分别控制 K、E、G、S、P 和 (P,I),即,$\overleftrightarrow{R}_{\text{control}}(C_1,K)$,$\overleftrightarrow{R}_{\text{control}}(C_2,E)$,$\overleftrightarrow{R}_{\text{control}}(C_4,G)$,$\overleftrightarrow{R}_{\text{control}}(C_5,S)$,$\overleftrightarrow{R}_{\text{control}}(C_6,P)$,

$\overleftrightarrow{R}_{\text{control}}(C_7,(P,I))$。

2)聚合关系($\overleftarrow{R}_{\text{assemble}}$):知识库 K 由 K_1、K_2、K_3、K_4 等 4 个知识元件组成,即,$\overleftarrow{R}_{\text{assemble}}(K,(K_1,K_2,K_3,K_4))$;目标 G 由若干子目标 SG_i 组成,即,$\overleftarrow{R}_{\text{assemble}}(G,(SG_i,i=1,2,\cdots,n))$;规划 P 由若干服务 S_i 组成,即,$\overleftarrow{R}_{\text{assemble}}(P,(S_i,i=1,2,\cdots,k))$;组合服务 S_3 由原子服务 $S4i$,$i=1,2,\cdots,M$ 组成,即,$\overleftarrow{R}_{\text{assemble}}(S_3,(S4i,i=1,2,\cdots,m))$;接口 I 由 I_1、I_2、I_3、I_4、I_5、I_6 等 6 个接口组成,即,$\overleftarrow{R}_{\text{assemble}}(I,(I_1,I_2,I_3,I_4,I_5,I_6))$;环境 Env 由若干 ISC_i,$i=1,2,\cdots,l$ 和许多 Env$j$,$j=1,2,\cdots,p$ 元件组成,即,$\overleftarrow{R}_{\text{assemble}}(\text{Evn},(\text{ISC}_i,\text{Evn}_j,i=1,2,\cdots,l,j=1,2,\cdots,p))$。

$$
C_3(\overleftrightarrow{R}_{\text{control}})\begin{cases}
C_1(\overleftrightarrow{R}_{\text{control}})K(\overleftarrow{R}_{\text{assembel}})\begin{bmatrix}K_1\\K_2\\K_3\\K_4\end{bmatrix}\\
C_2(\overleftrightarrow{R}_{\text{control}})E(\overleftarrow{R}_{\text{production}})\begin{bmatrix}I\\K\\G\\P\\S\end{bmatrix}\\
C_4(\overleftrightarrow{R}_{\text{control}})G(\overleftarrow{R}_{\text{assembel}})\left[\text{SG}_i(i=1,2,...,n)\overleftarrow{R}_{\text{implement}}S\right.\\
C_5(\overleftrightarrow{R}_{\text{control}})S(\overleftarrow{R}_{\text{assembel}})\begin{bmatrix}S_1(\overleftrightarrow{R}_{\text{link}})[I_1\\S_2(\overleftrightarrow{R}_{\text{link}})[I_2\\S_3(\overleftrightarrow{R}_{\text{assembel}})[S_4\\S_4\end{bmatrix}\\
C_6(\overleftrightarrow{R}_{\text{control}})P(\overleftarrow{R}_{\text{assembel}})S\\
C_7(\overleftrightarrow{R}_{\text{control}})\begin{cases}
P(\overleftarrow{R}_{\text{assembel}})[S\\
I(\overleftarrow{R}_{\text{assembel}})\begin{cases}
I_1(\overleftrightarrow{R}_{\text{pro-service}})[\text{ISC}_s\\
I_2(\overleftrightarrow{R}_{\text{Req-service}})[\text{ISC}_s\\
I_3(\overleftarrow{R}_{\text{input}})\left[\text{Env}(\overleftarrow{R}_{\text{assembel}})\begin{bmatrix}\text{ISC}_s\\\text{Env}\end{bmatrix}\right.\\
I_4(\overrightarrow{R}_{\text{outut}})\left[\text{Env}(\overleftarrow{R}_{\text{assembel}})\begin{bmatrix}\text{ISC}_s\\\text{Env}\end{bmatrix}\right.\\
I_5(\overleftrightarrow{R}_{\text{set/get}})\left[\text{Env}(\overleftarrow{R}_{\text{assembel}})\begin{bmatrix}\text{ISC}_s\\\text{Env}\end{bmatrix}\right.\\
I_6(\overleftrightarrow{R}_{\text{config}})\left[\text{Env}(\overleftarrow{R}_{\text{assembel}})\begin{bmatrix}\text{ISC}_s\\\text{Env}\end{bmatrix}\right.
\end{cases}
\end{cases}
$$

图 2　智能服务构件基元网

Fig.2　Intelligent service component basic-elements network

3)实现关系($\overleftrightarrow{R}_{\text{implement}}$,表示子目标 SG_i 由服务 S_i 来实现,即,$\overleftrightarrow{R}_{\text{implement}}(\text{SG}_i,(S_i,i=1,2,\cdots,n))$。

4)产生关系($\overleftarrow{R}_{\text{production}}$),表示事件 E 由接口 I、知识库 K、目标 G、服务 S 以及规划 P 产生,即,$\overleftarrow{R}_{\text{production}}(E,(I,K,G,S,P))$。

5)输入关系($\overleftarrow{R}_{\text{input}}$),表示环境 Env 通过感知器 I_3 传入变化信息,即,$\overleftarrow{R}_{\text{input}}(I_4,\text{Env})$。

6)输出关系($\overrightarrow{R}_{\text{output}}$),表示通过效应器 I_4 去影响环境 Env,即,$\overrightarrow{R}_{\text{output}}(I_4,\text{Env})$。

7)提供服务关系($\overrightarrow{R}_{\text{pro-service}}$)通过接口 I_1 与请求服务的 ISC_s 交互,即,$\overleftrightarrow{R}_{\text{pro-service}}(I_1,\text{ISC}_s)$。

8)请求服务关系($\overleftrightarrow{R}_{\text{req-service}}$)通过接口 I_2 与提供服务的 ISC_s 交互,即,$\overleftrightarrow{R}_{\text{req-service}}(I_2,\text{ISC}_s)$。

9)链接关系($\overleftrightarrow{R}_{\text{link}}$)表示提供或请求服务与相应接口 I 之间的联系,即,$\overleftrightarrow{R}_{\text{link}}(I_1,S_1)$ 与 $\overleftrightarrow{R}_{\text{link}}(I_2,S_2)$。

10)设置/得到关系($\overleftrightarrow{R}_{\text{set/get}}$)表示通过 I_5 接口,环境 Env 可以设置和得到 ISC 的属性值,即,$\overleftrightarrow{R}_{\text{set/get}}(I_5,\text{Env})$。

11)配置关系($\overleftrightarrow{R}_{\text{config}}$)表示通过 I_6 接口,环境 Env 可以进行 ISC 动态配置,即,$\overleftrightarrow{R}_{\text{config}}(I_6,\text{Env})$。

2.2.2　动态语义分析

为了描述动态关系,引入事元 A_x,下标 x 来表示动作名或功能名,其他属性有方法、工具、时间、地点、程度等根据需要进行取舍。用 $B_iA_xB_j$ 表示基元 B_i 施加动作或功能 x 给基元 B_j。$B_1A_1B_2A_2B_3\cdots B_{i-1}A_{i-1}B_n$ 表示一系列基元 B_i 依次施加一系列动作或功能给 B_i+1。另外,引入算子·、|、+等分别表示后续、并行和选择等语义。

$$
Ax=\begin{bmatrix}
\text{动作 } x & \text{支配象} & B_j\\
 & \text{施动对象} & B_i\\
 & \text{方法} & \text{Method}\\
 & \text{工具} & \text{Tool}\\
 & \text{时间} & \text{Time}\\
 & \text{地点} & \text{Place}\\
 & \text{程度} & \text{Extent}
\end{bmatrix}
$$

动态语义说明如下:

1)解释器 C_3 控制知识库管理器 C_1 初始化知识

库 K,C_1 读取领域本体 K_3、领域规则 K_1 和领域应用 K_4 的相关知识并写入到信念库 K_2,完成初始化任务。其可拓语义是:

$$C_3\vec{A}_{\text{control}}(C_1\overleftarrow{A}_{\text{read}}(K_1,K_3,K_4).C_2\vec{A}_{\text{write}}K_2)$$

2)环境 Env 产生变化事件 E 通过接口 I_3 输入到事件队列 E。其可拓语义是:

$$\text{Env}\,\vec{A}_{\text{producte}}E\,\vec{A}_{\text{input}}I_3\,\vec{A}_{\text{input}}E$$

3)解释器 C_3 控制事件管理器 C_2 按优先级读取事件队列 E 的首个事件 e,按照构件的当前认知 K_2,采用菱形思维推理,转化为响应该事件的目标 G。其可拓语义是:

$$C_3\vec{A}_{\text{control}}(C_2\overleftarrow{A}_{\text{read}}(E,K_2).C_2\vec{A}_{\text{rhombus}}(e,K_2).C_2\vec{A}_{\text{get}}G)$$

4)解释器 C_3 控制目标管理 C_4 读取目标库 G,按照构件的当前认知,采用菱形思维推理,分解目标为子目标 SG_i 构成的目标树。目标分解与分派也可能产生内部消息 E。其可拓语义是:

$$C_3\vec{A}_{\text{control}}(C_4\overleftarrow{A}_{\text{read}}G.C_4\vec{A}_{\text{rhombus}}(g,K_2).C_4\vec{A}_{get}\text{SG}_i$$
$$(i=1,2,\cdots,n)\mid \text{SG}_i\vec{A}_{\text{producte}}E)$$

5)解释器 C_3 控制服务管理器 C_5 读取目标树,按照构件的当前认知,发现、聚集服务 S_i 去实现目标树的每一个叶子目标。服务动态聚集也可能产生内部消息 E。其可拓语义是:

$$C_3\vec{A}_{\text{control}}(C_5\overleftarrow{A}_{\text{read}}\text{SG}_i.C_5\vec{A}_{\text{find}}S_i.C_5\vec{A}_{\text{assemble}}S_i$$
$$(i=1,2,\cdots,n)\mid S_i\vec{A}_{\text{producte}}E)$$

6)解释器 C_3 控制规划管理器 C_6 读取聚集的服务集合 S_i,按照每一个叶子目标实现的先后顺序组成规划序列 P_i,封装到规划库 P 中。规划动态排序也可能产生内部消息 E。其可拓描述语义是:

$$C_3\vec{A}_{\text{control}}(C_6\overleftarrow{A}_{\text{read}}S_i.C_6\vec{A}_{\text{order}}S_i.C_6\vec{A}_{\text{encapsulate}}(S_i,P)$$
$$(i=1,2,\cdots,n)\mid P\vec{A}_{\text{producte}}E)$$

7)解释器 C_3 控制执行器 C_7 读取规划库 P,按照规划序列执行 Pi。其可拓语义是:

$$C_3\vec{A}_{\text{control}}(C_7\overleftarrow{A}_{\text{read}}P.C_7\vec{A}_{\text{execute}}Pi,i=1,2,\cdots,n)$$

8)执行规划 P 通过输出接口 I_4 去影响环境 Env,环境 Env 将产生新的变化事件 E 通过输入接口 I_3 进入消息队列 E。其可拓语义是:

$$C_7\vec{A}_{\text{execute}}(Pi,i=1,2,\cdots,n).$$
$$C_7\vec{A}_{\text{output}}I_4\,\vec{A}_{\text{output}}\text{Env}\,\vec{A}_{\text{input}}I_3\,\vec{A}_{\text{input}}E$$

9)解释器 C_3 重新执行 3。

如此反复,形成感知监控-目标分解-服务聚集-规划封装-规划执行自适应循环链。

2.3 利用拓展分析研究自适应目标分解与服务聚集机制

由于自适应软件是由自适应软件实体构成的,可以利用对自适应软件实体的拓展分析研究自适应软件的演化。以上述基元网表示的自适应软件实体为例,利用相关分析构成基元相关网,利用发散分析构成基元发散网,利用蕴含分析构成基元的蕴含网,利用可扩分析构成基元可扩网。

例如,服务实体都具有同一个特征元(服务提供接口,S_i),根据“一征多物”的发散分析,由基元 S_i = (Services, Pro-Service, PS)可以得到一个表达“提供相同服务”的发散网:

$$S=(\text{Service},\text{Pro-Service},S_2)\ -|$$
$$S_i=(\text{Servicei},\text{Pro-Service},S_2),i=1,2,\cdots,n$$

再如,服务实体与服务接口之间存在联接相关性,构成基元相关网:$S_i\sim Ij,i,j=1,2,\cdots,n$。

自适应智能服务构件的目标分解可以构成基元蕴含网,叶目标与实现叶目标的服务之间也可以构成基元蕴含网。蕴含树表达了目标和服务之间的蕴含关系,实现目标有 3 种服务类型:组合服务(Com-Service)、提供服务(Pro-Service)和请求服务(Req-Service)。图 3 表示目标分解及服务实现的蕴含网。

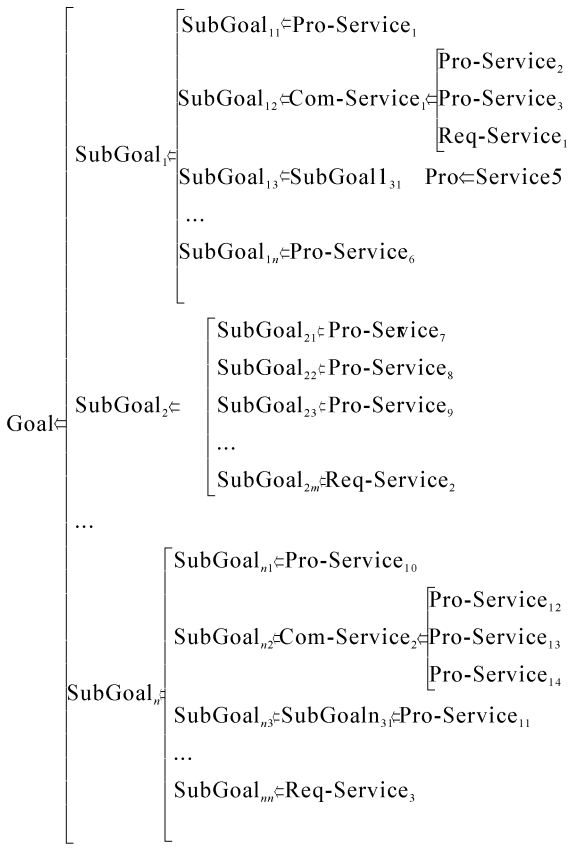


图 3 目标分解及服务实现的蕴含网

Fig.3 Goal Tree & Services Implement

图 3 的目标 Goal 分解成若干子目标 SG_i,用菱形推理优化子目标 SG_i,然后发布到目标库 G。服务容器 S 中的可提供服务 PS 主动聚集实现子目标 SG_i,不能实现的某些子目标 SG_i,向外发布请求服务 RS,发现并调用其他智能服务构件 ISC 提供的相关服务 S_i。其可拓语义是:

$$\text{Goal}\ \vec{A}_{\text{Decomposition}}\text{SG}_i\ \vec{A}_{\text{rhombus}}\text{SG}_i\ \overleftarrow{A}_{\text{implement}}$$
$$(S_i\ \overleftarrow{A}_{\text{assemble}}\text{PS}\mid S_i\ \overleftarrow{A}_{\text{requment}}\text{RS}\ \overleftarrow{A}_{\text{provide}}\text{ISCs})$$

由于自适应软件实体基元的可拓展性,导致由自适应软件实体构成的自适应软件基元网也具有可拓展性。将拓展分析得到的各个基元的发散网、相关网和蕴涵网替换图 2 的智能服务构件基元网中的基元节点,就得到了自适应软件网络的可拓展性以及更丰富的可替换、可分解、可精化的更丰富知识。

2.4 可拓变换对智能构件动态适应的作用制

任何一个自适应软件网络都会随时间的变化而变化,因而,自适应软件是一个动态网络。利用动态基元网表示自适应软件,更便于通过可拓变换对动态自适应软件进行分析和研究。

变换对自适应软件的影响也不容小视,也必须研究实施变换后自适应软件会如何变化。不同的变换,会导致自适应软件发生不同的传导变换,其连接方式、网络密度、网络集中度、甚至相互性和传递性都会发生改变。因此,必须研究变换对自适应软件的作用规律,才能更好地研究自适应软件结构。可拓变换理论与方法为研究变换对自适应软件的作用提供了工具。可拓变换包括基本变换、变换的运算、传导变换、共轭变换、复合变换等等,详情可以参阅文献[18]。

例如,对图 3 所示的基元网,如果有新的用户需求,智能服务构件感知新的事件,然后产生新的目标,经分解会产生新的子目标和叶子目标,为实现新的叶子目标,必须重新查找新的服务。例如,要增加一个新的子目标(new sub-goal)和新的服务(new service)。图 3 的结构变化如图 4。即,对 Goal 实施某变换 T, 得到新的目标 Goal'。

图 4 表达的是子目标(SubGoaln)增加了一个新的子目标(New SubGoal),要实现新目标,必须查找一个新的请求服务(New Req-Service)。其可拓语义是:

$$T\text{Goal} = \text{Goal}' =$$
$$\text{Goal} \cup (SG_{\text{New}}\ \overleftarrow{A}_{\text{implement}}S_{\text{New}}\ \overleftarrow{A}_{\text{requment}}RS\ \overleftarrow{A}_{\text{provide}}ISCs)$$

可以看到,用户需求或环境的变化(TEvn),导致新的事件产生(TEvent),推导出新的目标(T$Goal$),分解出新的子目标(TSG),查找并聚集新的服务(TService),实现新的规划(TPlan),触发新的执行(TAtion),最后,进一步影响环境(TEvn)。智能服务构件的动态演化可以用可拓传递变换来表示:

$$T\text{Env} \to T_{\text{Event}} \to T_{\text{Goal}} \to TSG \to T_{\text{Service}} \to T\ \text{Plan} \to T_{\text{Action}} \to T_{\text{Env}}$$

这个传递变换刻画了自适应软件监控-分析-规划-执行(MAPE)循环。

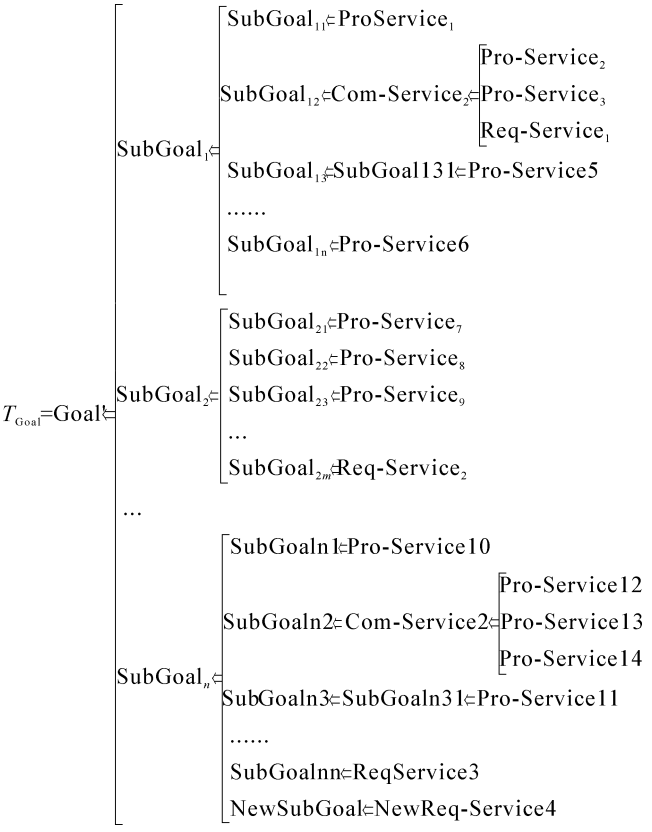


图 4　变换后的目标分解及服务实现

Fig.4　Change of goal tree & services implement

2.5 对自适应软件的评价

在可拓学中,可以用关联度表示研究对象具有某种性质的程度,包括初等关联函数、简单关联函数、离散关联函数等形式。

例如,离散关联函数根据等级分类等应用,可以设定一些离散值 $a_1, a_2 \cdots, a_k, 0, b_1, b_2, \cdots b_l$ 来计算。

$$k(x) = \begin{cases} A_1(>0), x = a_1 \\ A_2(>0), x = a_2 \\ \cdots \\ A_k(>0), x = a_k \\ 0, \\ B_1(<0), x = b_1 \\ B_2(<0), x = b_2 \\ \cdots \\ B_l(<0), x = b_l \end{cases}$$

要评价智能服务构件或智能服务系统响应环境

变化的自适应能力,核心是要判断能提供实现 n 个叶子目标的服务个数 $m = l + r$,l 为智能服务构件服务容器能提供的本地服务,r 为其他智能服务构件能提供的远程服务,有离散关联函数公式:

$$k(x) = \begin{cases} 1, n << l \\ 0.75, n < l \\ 0.5, n = l \\ 0.25, l < n < m \\ 0, n = m, l = 0 \\ -0.25, n \geqslant m, l = 0 \\ -0.5, n > m, l = 0 \\ -1, n \gg m = 0 \end{cases}$$

其可拓语义是:当能够提供实现叶子目标的本地服务非常多时,关联函数值为 1;当能够提供实现叶子目标的本地服务较多时,关联函数值为 0.75;当能够提供实现叶子目标的本地服务相等时,关联函数值为 0.5;当能够提供实现叶子目标的本地服务少于叶子目标数 n ,但发现的远程服务 r 与本地服务 l 之和大于 n 时,关联函数值为 2.5;当能够提供实现叶子目标的本地服务为 0,且远程服务 r 等于 n 时,关联函数值为 0;当能够提供实现叶子目标的本地服务为 0,且远程服务 r 小于等于 n 时,关联函数值为 -0.5;当能够提供实现叶子目标的本地服务为0,且远程服务 r 非常少时,关联函数值为 -1。

显然,关联函数值越高,其自适应能力越高。只有通过发现服务变换 r,下载服务变换 $r \rightarrow l$ 等等来提高关联函数值。即通过可拓变换 $T_{\text{find}}S$、$T_{\text{download}}S$ 来提高软件的自适应能力。

针对不同的评价特征,可以选择不同的关联函数。对于多评价特征的情况,还可建立综合关联度,用于综合评价研究对象的优劣。将其应用于自适应软件研究,可以作为自适应软件中的成员“聚类”或“分类”的衡量指标之一,对自适应软件进行综合评价。详细方法请参考文献[16]。

5 结束语

引入可拓学理论,本文研究了自适应软件的基元网方法,利用拓展分析丰富自适应软件系统的演化知识,探讨了可拓变换对软件实体、软件系统以及环境自适应机制的描述、分析和影响程度、趋势等等,用关联函数给出了自适应软件的定量评价方法。基于可拓学的自适应软件形式化方法,充分考虑自适应软件中各级个体成员的演化对软件系统演化的影响,为研究动态自适应软件系统提供定性与定量相结合的方法。

进一步的工作是通过可拓形式化方法与其他形式化方法的兼容性研究,拓宽可拓形式化方法的应用范围并研制出验证支撑工具。

参考文献:

[1]丁博, 王怀民, 史殿习. 构造具备自适应能力的软件[J]. 软件学报, 2013, 24(9): 1981-2000.
DING Bo, WANG Huaimin, SHI Dianxi. Constructing software with self-adaptability[J]. Journal of Software, 2013, 24(9): 1981-2000.

[2]LADDAGA R. Creating robust software through self-adaptation[J]. IEEE Intelligent Systems and Their Applications, 1999, 14(3): 26-29.

[3]KAKOUSIS K, PASPALLIS N PAPADOPOULOS G A. A survey of software adaptation in mobile and ubiquitous computing[J]. Enterprise Information Systems, 2010, 4(4): 355-389.

[4]F D MACIAS-ESCRIVA, R H AW, R DEL TOROV, et al. Self-adaptive systems: a survey of current approaches, research challenges and applications[J]. Expert Systems with Applications, 2013, 40(18): 7267-7279.

[5]金芝, 吕建. 软件系统需要自适应能力-为什么和如何拥有?[J]. 中国计算机学会通讯, 2013, 9(6): 6-7.
JIN Zhi, LYU Jian. Why and how to achieve self-adaptive ability for software systems?[J]. Communication of CCF, 2013, 9(6): 6-7.

[6]张伟, 赵海燕. 软件可信性与自适应软件随想[J]. 中国计算机学会通讯, 2013, 9(6): 8-13.
ZHANG Wei, ZHAO Haiyan. Some thinking on trustworthiness and self-adaptability of software[J]. Communication of CCF, 2013, 9(6): 8-13.

[7]刘磷. 软件服务的自适应与演化需求建模[J]. 中国计算机学会通讯, 2013, 9(6): 13-20.
LIU Lin. Adaptation and evolution requirements of software systems services[J]. Communication of CCF, 2013, 9(6): 13-20.

[8]毛新军, 孙跃坤. 社会技术系统的自适应技术[J]. 中国计算机学会通讯, 2013, 9(6): 20-26.
MAO Xinjun, SUN Yuekun. Self-adaptation technology for social technical system[J]. Communication of CCF, 2013, 9(6): 20-26.

[9]彭鑫, 陈碧欢, 赵文耘. 需求驱动的软件自适应方法[J]. 中国计算机学会通讯, 2013, 9(6): 27-36.
PENG Xin, CHEN Bihuan, ZHAO Wenyun. Requirement-driven software self-adaptation[J]. Communication of CCF, 2013, 9(6): 27-36.

[10]丁博, 王怀民, 史殿习, 等. 一种支持软件可信演化的构件模型[J]. 软件学报, 2011, 22(1): 17-27.
DING Bo, WANG Huaimin, SHI Dianxi, et al. Component

model supporting trustworthiness-oriented software evolution[J]. Journal of Software, 2011, 22(1): 17-27.

[11]何克清. 面向按需服务的软件方法及其标准化研究进展[J]. 中国计算机学会通讯, 2010, 6(9): 21-25.
HE Keqing. The research progress for software method and standardization based on on-demand services[J]. Communication of CCF, 2010, 6(9): 21-25.

[12]陈洪龙, 李仁发. 自适应演化软件研究进展[J]. 计算机应用研究, 2010, 27(10): 3612-3616, 3621.
CHEN Honglong, LI Renfa. Survey on self-adaptive evolution software [J]. Application Research of Computers, 2010, 27(10): 3612-3616, 3621.

[13]刘智斌, 朱晓龙, 曹宝香. 一种自适应程序设计方法[J]. 计算机工程与应用, 2011, 47(36): 80-82, 126.
LIU Zhibin, ZHU Xiaolong, CAO Baoxiang. Method of adaptive programming[J]. Computer Engineering and Applications, 2011, 47(36): 80-82, 126.

[14]王振东, 王慧强, 冯光升, 等. 自律计算系统及其关键技术研究[J]. 计算机科学, 2013, 40(7): 15-18, 53.
WANG Zhendong, WANG Huiqiang, FENG Guangsheng, et al. Research on autonomic computing system and its key technologies[J]. Computer Science, 2013, 40(7): 15-18, 53.

[15]蔡文, 杨春燕. 可拓学的基础理论与方法体系[J]. 科学通报, 2013, 58(13): 1190-1199.
CAI Wen, YANG Chunyan. Basic theory and methodology on Extenics [J]. Chinese Science Bulletin, 2013, 58(13): 1190-1199.

[16]YANG Chunyan, CAI Wen. Extenics: theory, method and application[M]. Beijing: Science Press, 2013: 1-375.

[17]杨春燕, 李兴森. 可拓创新方法及其应用研究进展[J]. 工业工程, 2012, 15(1): 131-137.
YANG Chunyan, LI Xingsen. Research progress in extension innovation method and its applications[J]. Industrial Engineering Journal, 2012, 15(1): 131-137.

[18]孙昌爱, 金茂忠, 刘超. 软件体系结构研究综述[J]. 软件学报, 2002, 13(7): 1228-1237.
SUN Chang´ai, JIN Maozhong, LIU Chao. Overviews on software architecture research [J]. Journal of Software, 2002, 13(7): 1228-1237.

[19]SHAW M. The coming-of-age of software architecture research[C]//Proceedings of the 23rd International Conference on Software Engineering. Washington, DC: IEEE Computer Society, 2001: 656-664.

[20]ALLEN R, DOUENCE R, GARLAN D. Specifying and analyzing dynamic software architectures[M]// ASTESIANO E. Fundamental Approaches to Software Engineering. Berlin Heidelberg: Springer, 1998, 1382: 21-37.

[21]任洪敏. 基于 π 演算的软件体系结构形式化研究[D]. 上海: 复旦大学, 2003: 1-22.
RAN Hongmin. Research on software architectural formalism based on π calculus[D]. Shanghai, China: Fudan University, 2003: 1-22.

[22]李长云. 基于体系结构的软件动态演化研究[D]. 杭州: 浙江大学, 2005: 1-31.
LI Changyun. Research on architecture-based software dynamic[D]. Hangzhou: Zhejiang University, 2005: 1-31.

[23]MILNER R. The space and motion of communicating Agents [M]. Cambridge: Cambridge University Press, 2009: 1-179.

[24]汪玲, 戎玫, 张广泉, 等. 基于 Bigraph 的面向方面动态软件体系结构演化研究[J]. 计算机科学, 2010, 37(9): 137-140.
WANG Ling, RONG Mei, ZHANG Guangquan, et al. Research on bigraph-based aspect-oriented dynamic software architecture evolution [J]. Computer Science, 2010, 37(9): 137-140.

[25]陈洪龙, 李仁发. 基于 Bigraph 理论的动态演化软件相关特性分析与验证方法[J]. 小型微型计算机系统, 2010, 31(12): 2305-2309.
CHEN Honglong, LI Renfa. Analyzing and verifying method for dynamic evolution software based on bigraphical theory[J]. Journal of Chinese Computer Systems, 2010, 31(12): 2305-2309.

[26]刘培培, 章勇, 莫启, 等. 一种基于 Bigraph 理论的软件演化过程模型研究[J]. 计算机应用研究, 2013, 30(5): 1423-1426.
LIU Peipei, ZHANG Yong, MO Qi, et al. Research on software evolution process model based on Bigraph theory [J]. Application Research of Computers, 2013, 30(15): 1423-1426.

[27]吕建, 马晓星, 陶先平, 等. 网构软件的研究与进展[J]. 中国科学 E 辑 信息科学, 2006, 36(10): 1037-1080.
LYU Jian, MA Xiaoxing, TAO Xianping, et al. Research and development for Internetware software[J]. Science in China Ser. E Information Sciences, 2006, 36(10): 1037-1080.

[28]何克清, 彭蓉, 刘玮, 等. 网络式软件[M]. 北京: 科学出版社, 2008: 1-30.
HE Keqing, PENG Rong, LIU Wei, et al. Netwoked Sofware[M]. Beijing: Science Press, 2008: 1-30.

[29]何克清, 马于涛, 刘婧, 等. 软件网络[M]. 北京: 科学出版社, 2008: 1-25.
HE Keqing, MA Yutao, LIU Jing, et al. Software Netwok [M]. Beijing: Science Press, 2008: 1-25.

[30]何克清, 何扬帆, 王翀, 等. 本体元建模理论与方法及其应用[M]. 北京: 科学出版社, 2008: 1-34.
HE Keqing, HE Yangfan, WANG Chong, et al. Ontological metamodeling theory, methods and application [M].

Beijing: Science Press, 2008: 1-34.

[31]JORSTAD I, DUSTDAR S, Van THANH D. A service oriented architecture framework for collaborative services [C]//Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. Washington, DC: IEEE Computer Society, 2005, 121-125

[32]CHUNG J Y. An industry view on service-oriented architecture and web services[C]//Proceedings of IEEE International Workshop on Service-Oriented System Engineering. Beijing: IEEE, 2005: 59.

[33]ANDREWS T, CURBEAR F, DHOLAKIA H, et al. Business Process Execution Language for Web Services version 1.1 [EB/OL] 2007, http://www.ibm.com/developerworks/library/specification/ws-bpel/.

[34]ZHANG Jia, CHUNG J Y, CHANG C K, et al. WS-Net: A petri-net based specification model for web services [C]// Proceedings of the 20th IEEE International Conference on Web Services. San Diego, California: IEEE Computer Society, 2004: 420-427.

[35]KOSHKINA M, VAN BREUGEL F. Modelling and verifying web service orchestration by means of the concurrency workbench [J]. ACM SIGSOFT Software Engineering Notes, 2004, 29(5): 1-10.

[36]GU Xiwu, LU Zhengdin. A pi-calculus based formal model for BPEL4 WS web service composition[J]. Computer Science, 2007, 30(3): 69-74.

[37]MULLER J P, COSSENTINO M. Agent-oriented software engineering XIII [C]//13th International Workshop, AOSE 2012, Valencia, Spain, June 4, 2012, Revised Selected Papers. Lecture Notes in Computer Science. Berlin Heidelberg: Springer, 2013, 7852: 22-41.

[38]WEYNS D, MULLER J P. Agent-oriented software engineering XII [C]//VILLATORO D, SABATER-MIR J, SICHMAN J S. The 12th International Workshop, AOSE 2011, Taipei, May 2011, Revised Selected Papers. Lecture Notes in Computer Science. Berlin Heidelberg: Springer, 2012, 6938: 18-43.

[39]WEYNS D, GLEIZES M P. Agent-oriented software engineering XI[C]//WEYNS D, GLEIZES, M P. 11th International Workshop, AOSE 2010, Toronto, Canada, May 10-11, 2010, Revised Selected Papers. Lecture Notes in Computer Science. Berlin Heidelberg: Springer, 2011, 6788: 21-45.

[40]SHI Zhongzhi, ZHANG Jianhua, YUE Jinpeng, et al. A cognitive model for multi-agent collaboration[J]. International Journal of Intelligence Science, 2014, 4(1): 1-6.

[41]常志明, 毛新军, 王戟, 等. 多 Agent 系统中软构件的动态绑定机制及其操作语义[J]. 计算机研究与发展, 2007, 44(5), 806-814.

CHANG Zhiming, MAO Xinjun, WANG Ji, et al. Dynamic binding mechanism and its operational semantics of component in multi-agent system[J]. Journal of Computer Research and Development, 2007, 44(5): 806-814.

[42]李青山, 王璐, 褚华, 等. 一种基于智能体技术的软件自适应动态演化机制[J]. 软件学报, 2015, 26(4): 760-777.

LI Qingshan, WANG Lu, CHU Hua, et al. Agent-Based software adaptive dynamic evolution mechanism[J]. Journal of Software, 2015, 26(4): 760-777.

[43]杨春燕, 蔡文. 可拓信息-知识-智能形式化体系研究[J]. 智能系统学报, 2007, 2(3): 8-11.

YANG Chunyan, CAI Wen. A formalized system of extension information-knowledge-intelligence[J]. CAAI Transactions on Intelligent Systems, 2007, 2(3): 8-11.

[44]方卓君, 李卫华, 李承晓. 自助游可拓策略生成系统的研究与实现[J]. 广东工业大学学报, 2009, 26(2): 83-89.

FANG Zhuojun, LI Weihua, LI Chengxiao. Research and realization of Extension strategy generating system for independent travel [J]. Journal of Guangdong University of Technology, 2009, 26(2): 83-89.

[45]叶广仔, 李卫华, 张希花. 防治空气污染的可拓策略生成系统研究与实现[J]. 广东工业大学学报, 2007, 24(4): 42-48.

YE Guangzai, LI Weihua, ZHANG Xihua. Research and realization of extension strategy generating system for air pollution prevention and cure[J]. Journal of Guangdong University of Technology, 2007, 24(4): 42-48.

[46]陈智斌, 彭平, 陈宇亮. 基于可拓策略生成技术的商品搜索服务改进研究[J]. 数学的实践与认识, 2009, 39(4): 160-167.

CHEN Zhibin, PENG Ring, CHEN Yuliang. Research on improvement of product search service base on extension strategy generating technique[J]. Mathematics in Practice and Theory, 2009, 39(4): 160-167.

[47]朱伶俐, 李卫华, 李小妹. 客户价值可拓知识挖掘软件研究[J]. 广东工业大学学报, 2012, 29(4): 7-13.

ZHU Lingli, LI Weihua, LI Xiaomei. Research on extension knowledge mining software for customer value [J]. Journal of Guangdong University of Technology, 2012, 29(4): 7-13.

[48]苏楠. 基于可拓逻辑的产品族配置设计方法[D]. 杭州: 浙江工业大学, 2009: 1-20.

SU Nan. Configuration design on the extension logic for product family [D]. Hangzhou: Zhejiang University of Technology, 2009: 1-20.

[49]杨国为, 王先梅, 涂序彦. 面向计算机的产品创新设计的新模型与新原理(1)[J]. 计算机工程与应用, 2003, 39 (32): 7-9, 47.

YANG Guowei, WANG Xianmei, TU Xuyan. New Models and principles on computer-oriented innovative and creative design of products (I)[J]. Computer Engineering and applications, 2003, 39(32): 7-9, 47.

[50]杨国为, 王先梅, 涂序彦. 面向计算机的产品创新设计的新模型与新原理(2)[J]. 计算机工程与应用, 2003, 39 (33): 22-24, 64.
YANG Guowei, WANG Xianmei, TU Xuyan. New models and principles on computer-oriented innovative and creative design of products (Ⅱ)[J]. Computer Engineering and applications, 2003, 39(33): 22-24, 64.

[51]邹广天. 建筑设计创新与可拓思维模式[J]. 哈尔滨工业大学学报, 2006, 38(7): 1120-1123.
ZOU Guangtian. Innovation of architectural design and extension thinking modes[J]. Journal of Harbin Institute of Technology, 2006, 38(7): 1120-1123.

[52]王涛, 邹广天. 空间元与建筑室内空间设计中的矛盾问题[J]. 哈尔滨工业大学学报, 2006, 38(7): 1139-1142, 1145.
WANG Tao, ZOU Guangtian. Space-element and contradictory problems consisting in design of architectural interior space[J]. Journal of Harbin Institute of Technology, 2006, 38(7): 1139-1142, 1145.

[53]周志丹, 李兴森. 企业自主创新的可拓创新模型构建与应用研究[J]. 科学学研究, 2010, 28(5): 769-776.
ZHOU Zhidan, LI Xingsen. Research on extenics-based innovation model construction and application of enterprise independent innovation[J]. Studies in Science of Science, 2010, 28(5): 769-776.

[54]李兴森, 刘艳彬. 可拓学与信息管理、知识管理的关系研究[J]. 当代经济管理, 2011, 33(11): 6-9.
LI Xingsen, LIU Yanbin. Study on the relationship among extenics, information management and knowledge management in the knowledge economy[J]. Contemporary Economy & Management, 2011, 33(11): 6-9.

[55]杨春燕, 李卫华, 李小妹. 矛盾问题智能化处理的理论与方法研究进展[J]. 广东工业大学学报, 2011, 28 (1): 86-94, 97.
YANG Chunyan, LI Weihua, LI Xiaomei. Recent research progress in theories and methods for the intelligent disposal of contradictory problems[J]. Journal of Guangdong University of Technology, 2011, 28(1): 86-94, 97.

[56]李卫华. 利用知网增强可拓策略生成机制研究[J]. 广东工业大学学报, 2013, 30(2): 1-6.
LI Weihua. Research on taking advantage of the hownet to enhance mechanisms of extension strategy generation[J]. Journal of Guangdong University of Technology, 2013, 30 (2): 1-6.

[57]陈智斌, 彭平, 贾西平. 基于认知科学的可拓变换构造问题研究[J]. 广东工业大学学报, 2011, 28(4): 1-6.
CHEN Zhibin, PENG Ping, JIA Xiping. Research on the construction of extension transformation based on cognitive science[J]. Journal of Guangdong University of Technology, 2011, 28(4): 1-6.

[58]李博, 同淑荣, 白晶. 设计过程基因可拓基元模型及过程基因变异[J]. 机械科学与技术, 2012, 31(1): 87-95.
LI Bo, TONG Shurong, BAI Jing. Extensible basic-element model of design process gene and mutation of design process gene[J]. Mechanical Science and Technology for Aerospace Engineering, 2012, 31(1): 87-95.

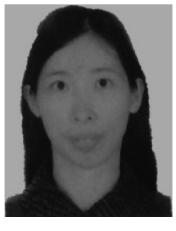
[59]杨春燕, 李志明. 基于可拓学的社会网络结构研究[J]. 广东工业大学学报, 2014, 31(1): 1-6.
YANG Chunyan, LI Zhiming. Extenics based social network structure[J]. Journal of Guangdong University of Technology, 2014, 31(1): 1-6.

[60]Fan Rui. Modelling extenics innovation software by intelligent service components[J]. The Open Cybernetics & Systemics Journal, 2014, 8: 1-7.

作者简介:

范锐,男,1958 年生,教授。主要研究方向为软件工程、软件形式化方法、自适应软件、多智能体系统、可拓工程、企业信息系统。曾主持省科技项目 1 项、校教改项目 3 项,参与国家级、省级科技项目 3 项,发表论文 32 篇,其中被 SCI、EI、ISTP 检索 12 篇。

彭银桥,男,1969 年生,副教授,博士。主要研究方向为软件工程、可拓工程、信息安全、电子信息材料与器件,传感技术等。主持广东海洋大学启动基金项目 1 项,广东海洋大学自然基金项目 2 项。发表学术论文 20 余篇,其中被 SCI 检索 2 篇,被 EI、ISTP 检索 6 篇。



陈月峰,女,1971 年生,副教授。主要研究方向为软件工程,人工智能,可拓工程。参加省级、校级科研和教改项目 3 项,发表学术论文 10 篇,其中被 SCI、EI、ISTP 检索 4 篇。