软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(9):2001−2019 [doi: 10.3724/SP.J.1001.2013.04433]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

软件产品线可变性建模技术系统综述*

聂坤明, 张 莉, 樊志强

(北京航空航天大学 软件工程研究所,北京 100191)

通讯作者: 聂坤明, E-mail: niekunming@cse.buaa.edu.cn

摘 要: 软件产品线是实现大规模的软件复用、保证高质量的新产品开发的最佳实践.软件产品线的关键问题是如何进行可变性管理,并基于可变性管理实现软件核心资产的复用.软件产品线可变性建模是可变性管理的关键技术,实现产品家族成员的共性和可变性的描述.可变性建模涉及软件开发的全生命周期,在领域工程和应用工程中,尤其是在产品构建过程中,起到重要的作用.从众多的建模技术中选择合适的建模技术是十分困难的,在软件产品线领域中开展了可变性建模技术的系统综述,按照系统综述的方法对可变性建模技术进行了系统总结,根据系统综述规则,选取了从 1990 年~2011 年发表的论文进行综述.讨论了系统综述的研究成果,从可变性建模方法分类、重要可变性建模技术对比等方面进行深入的探讨,为建模人员和研究人员对可变性建模技术的选择和研究提供支持.最后分析了可变性建模技术的研究趋势,并对可变性建模技术有待深入的研究难点和发展趋势进行了展望.

关键词: 系统文献综述;软件产品线;可变性建模;研究趋势

中图法分类号: TP311 **文献标识码**: A

中文引用格式: 聂坤明,张莉,樊志强.软件产品线可变性建模技术系统综述.软件学报,2013,24(9):2001−2019. http://www.jos.org.cn/1000-9825/4433.htm

英文引用格式: Nie KM, Zhang L, Fan ZQ. Systematic literarture review of software product line variability modeling techniques. Ruan Jian Xue Bao/Journal of Software, 2013,24(9):2001−2019 (in Chinese). http://www.jos.org.cn/1000-9825/4433.htm

Systematic Literarture Review of Software Product Line Variability Modeling Techniques

NIE Kun-Ming, ZHANG Li, FAN Zhi-Qiang

(Software Engineering Institute, BeiHang University, Beijing 100191, China)

Corresponding author: NIE Kun-Ming, E-mail: niekunming@cse.buaa.edu.cn

Abstract: The software product line is one of the most effective strategies for reuse of large-scale software and rapid development of new software products with good quality levels. The variability modeling technique used to describe the commonality and variability of software product families is one of the most important techniques in software product line. Variability modeling plays an important role in the product line domain engineering and product line application engineering. However, it is difficult for modelers and engineers to choose one or more suitable modeling techniques from various variability modeling techniques. A systematic literature review of the variability modeling techniques in software product line reported from 1990s to 2011 is carried out. In this systematic review, the existing variability modeling techniques are analyzed and compared to help engineers and/or researchers to select the most suitable variability modeling techniques. Finally, the research trend of variability modeling technique is given, and the prospects for future research and suggestions for possible extensions are also discussed.

Key words: systematic literature review; software product line; variability modeling; research trend

* 基金项目: 国家自然科学基金(61170087, 61370058); 软件开发环境国家重点实验室自主课题(SKLSDE-2012ZX-13); 中央高校基本科研业务费专项资金

收稿时间: 2012-02-22; 修改时间: 2012-04-18; 定稿时间: 2013-03-07; jos 在线出版时间: 2013-06-26

CNKI 网络优先出版: 2013-06-26 11:39, http://www.cnki.net/kcms/detail/11.2560.TP.20130626.1139.001.html

© 中国科学院软件研究所 http://www.jos.org.cn

软件产品线是实现软件复用的一种重要方式,工业企业的应用表明,软件产品线能够有效地提高产品质量及缩短产品推向市场的时间.软件产品线技术的应用,使得惠普、诺基亚和飞利浦等公司获得了经济效益,不仅缩短了产品上市时间,而且提高了用户满意度.软件产品线技术近几年得到了广泛的发展,目前已经有商业软件BigLever[1]和 Pure::systems[1]提供对软件产品线的建模支持.另外,一些领域内的标准也在建立之中,例如,OMG组织正在进行标准化的通用可变性建模语言(CVL)[2].

不同于单独描述一个软件系统,产品线工程描述了一系列产品,即产品家族.产品家族成员在构建时都表现出对相同或相似功能的支持,可变性则用于描述产品家族中产品成员之间的差异,是软件产品线工程区别于其他软件开发过程的重要特征.通过产品家族的可变性管理,可以重用产品家族系列产品的核心资产(asset),包括需求、体系结构、构件、代码和测试用例等,并实现核心资产在产品线的特定产品生产环境下的应用.

可变性的管理跨越整个产品的生命周期,包含从软件产品的分析到实现的整个过程.在分析阶段,进行系统的可变性分析;在设计阶段,可变性管理可以支持可变性的获取;在实现阶段,可变性管理可以指导产品线的实现;在产品的演化阶段,可变性管理可以提供产品演化的支持.

软件可变性建模是实现软件可变性管理的关键技术,软件产品线工程包括领域工程和应用工程.在领域工程中,可变性建模关注可变性的获取和表示.软件的可变性建模是领域分析的一项重要工作,通过对于领域中典型系统的分析,识别这些系统的共性和变化性,运用选择、抽象等方法建立领域模型.在应用工程中,基于可变性模型,并利用核心资产的重用实现产品的构建.

目前,可变性建模有各种各样的技术,并且侧重点不同,难以建立可变性建模技术的整体把握,需要通过系统化的方法对于各种可变性建模技术进行总结分类.文献[3,4]指出,当使用可变性建模技术的时候,有 3 个方面对于工程人员至关重要,即对什么进行建模,模型的创建与利用是否存在相应的工具支持,以及建模过程是否存在相应的指导.但论文只是基于前两个方面对于可变性建模技术进行了分类,不够全面.Chen 等人[5]对于可变性管理技术和可变性管理技术的评价进行了研究,但是缺少对于重要可变性建模技术的对比.

因此,利用系统文献综述方法对于可变性建模技术进行综合分析,通过按照时间顺序和不同分类对可变性建模技术进行总结,对比重要可变性建模技术并分析可变性建模技术的研究趋势,可以帮助建模和研究人员选择适合自己领域的建模技术,促进产品线建模技术的研究和在工业界的有效应用.

本文第 1 节介绍系统综述的方法,并按照系统综述的过程进行资料收集和分析.第 2 节对于可变性建模系统综述的结果进行分析,从可变性建模方法分类、重要的可变性模型对比方面进行分析.第 3 节分析可变性建模技术的研究趋势.最后对本文进行总结.

1 研究方法

系统文献综述(systematic literature review),简称系统综述,是主要在医学领域和社会学研究领域针对某一个研究性问题开展的基于文献的系统化综述方法,并于 2004 年引入软件工程领域[6].系统文献综述的过程包含 3 个主要的阶段:

- 制定综述方案:主要进行综述需求分析,明确综述的目的,设计开展综述的规程,以指导后期的文献收集等工作;
- 开展综述:根据综述方案预定的目标,按照综述步骤进行文献收集,包括确定主要的文献来源、确定收录标准、实现文献质量评价、开展数据抽取与综合;
- 形成综述报告:对于综述进行总结.

本文按照上述过程开展了可变性建模技术的系统综述.在从文献数据库[6]中检索出相关文献后,手工检索相关会议论文集,并删除重复文献,总共有 96 篇相关文献.这些文献遵循文献收录和排除的原则进行筛选.在数据抽取的过程,按照文献的选取规则选取相关文献.最终,54 篇文献被确定用于系统综述分析,见表 1.其中, 2008 年和 2009 年的文献最多,并且呈现逐渐上升的趋势;2011 年的文献由于没有完全统计,所以偏少.

© 中国科学院软件研究所 http://www.jos.org.cn

Table 1 Summary of variability modeling methods

表 1 可变性建模方法汇总

编号	来源	年份	分类	方法名称	编号	来源	年份	分类	方法名称
1	Other	1990	Feature	FODA	28	WOS	2005	GOAL	GoalMaps
2	WOS	2008	Feature	FD	29	WOS	2009	GOAL	IStarLPS
3	IEEE	2006	Feature	IVMArchi	30	ACM	2010	Multiple	NUI
4	CSA	2004	Feature	IVMAspect	31	ACM	2011	Multiple	MultiDvm
5	WOS	2005	Feature	FVP	32	IEEE	2008	Multiple	MultiMeta
6	IEEE	2003	Feature	Fuzzy	33	Springer	2004	Multiple	Covamof
7	ACM	2004	Feature	Pure::Variant	34	ACM	2010	Multiple	VPI
8	ACM	2006	Feature	Cycle	35	IEEE	2009	Multiple	VPO
9	Springer	2004	Feature	ReqiLine	36	ACM	2004	Multiple	VPModel
10	ACM	2005	Feature	Templ	37	CSA	2006	Semantic	SemanticM
11	CSA	2005	Feature	Cardi	38	IEEE	2009	Semantic	PDO
12	CSA	2002	UML	ExtUML	39	SD	2007	Semantic	Kumbang
13	CSA	2001	UML	UMLM	40	IEEE	2008	Language	CVL
14	ACM	2011	UML	UMLMDA	41	Springer	2009	Language	EmbedDSL
15	CSA	2005	UML	UMLC	42	CSA	2001	Configuration	PCL
16	Thesis	2010	UML	MSVCM	43	Springer	2009	Configuration	Questionnaire
17	IEEE	2008	UML	UMLArchi	44	WOS	2004	Configuration	Koalish
18	Springer	2006	UML	Consolid	45	ACM	2004	other	Orthogonal
19	ACM	2008	Aspect	AspUsecase	46	IEEE	2009	other	FOrdLogic
20	WOS	2008	Aspect	AspNatsuko	47	IEEE	1999	other	Patterns
21	CSA	2008	Aspect	AspIris	48	ACM	2010	other	SPLGraph
22	ACM	2009	Aspect	Crosscut	49	Springer	2003	other	VSL
23	IEEE	2007	Aspect	AspVisual	50	ACM	2009	other	SaaS
24	电子学报	2009	Aspect	EAspect	51	ACM	2010	other	SimCOVAMOF
25	Springer	2011	Decision	DoplerVML	52	Springer	2008	other	Brazilian
26	ACM	2000	Decision	Kobra	53	Springer	2007	other	Ebusiness
27	Springer	2003	Decision	VManage	54	IEEE	2002	other	Automotive

注:“来源”是该文献的来源数据库,其中,SD 是 ScienceDirect 的缩写,WOS 是 Web of Science 的缩写,CSA 是 Citeseer library 的缩写;“Other”表示技术报告等来源;“年份”记录该文献的发表年份或者在网络上公布的年份;“方法名称”是该建模方法的缩略词.

2 系统综述结果分析

对第 1 节选取的文献进行分类,见表 1 所示,按照文献来源、发表年份、分类、方法名称进行组织.

本文对于可变性建模技术进行了分类.根据各个建模方法的主要技术特征,并参考文献[7]的部分分类类型,同时,本文考虑不同可变性建模方法的可变性表达方法,将可变性建模方法分为以下类型:面向特征(feature)的可变性建模、基于 UML 的可变性建模、面向方面(aspect)的可变性建模、基于决策模型(decision)的可变性建模、基于目标(GOAL)的可变性建模、多维度可变性建模(multiple)、基于领域建模语言(language)的可变性建模、基于语义(semantic)的可变性建模、基于配置(configuration)的可变性建模和其他方法.在分类过程中,存在一些方法属于多个分类的情况,本文只标识其主要技术特征的分类.

2.1 可变性建模方法分类

可变性建模技术主要实现领域内核心资产的可变性表示,可变性建模需求主要包含:

1) 可变点的表示.标识可变性发生的位置;
2) 可变体的表示.可变体(variant)定义了该可变点相关的可变性实现的不同选择;
3) 可变体的选择关系表示.定义了跟可变点关联的可变体的选择关系,基本的关系包含:可选性(optional)、选择性(alternative)、多选性(or).可选择性表示可变点上只有一个可变体,并且需要最终确定是否被选中;选择性表示可变点上有多个相关的可变体,且只有一个可以被选中;多选性表示可以从多个相关的可变体中选中多个;
4) 可变体之间的依赖规则表示.定义了可变体之间的依存关系,可以利用代数表达式等表示;
5) 可变性模型和核心资产的追踪.维护可变性模型和核心资产的追踪关系.

建模和研究人员在选择可变性建模技术的过程中,首先确定应用领域,根据领域特点选择相应的系统建模

© 中国科学院软件研究所 http://www.jos.org.cn

方法,进行系统建模;然后,基于该领域内的系统模型确定可变性建模的对象,开展可变性分析;最后,基于建模工具实现可变性建模.因此,从建模和研究人员的角度出发,主要关注应用领域、建模阶段、建模对象类型、建模技术和建模工具(第 2.2 节).其中,建模阶段包括领域/应用需求分析阶段、领域/应用设计阶段和领域/应用实现阶段,建模对象类型包含用例、类图、活动图、软件体系结构、构件、代码和测试用例等.

本文接下来主要从支持的建模阶段、建模对象、建模技术角度对每种分类的不同方法进行对比,分析了不同方法的建模阶段和建模对象,重点对于可变性建模方法的可变性表示机制进行了分析,即可变点、可变体、可变体选择关系和依赖规则的表示,以分析不同分类和不同方法的特点.

2.1.1 基于特征的可变性建模方法

基于特征的可变性建模是最早用于可变性建模的方法,特征定义为用户可见的该软件系统独有的质量或者特征.1990 年,特征模型随着 FODA(feature oriented domain analysis)[8]的提出而被引入软件产品线的研究中.特征模型用特征、特征之间的关系和依赖规则表示软件产品线中所有可能产品的信息.其中:特征之间的关系利用强制性(mandatory)、可选性、选择性、多选性;特征之间的依赖规则利用依赖(require)表示在产品中包括特征 A 意味着在这个产品必须包括特征 B,利用互斥(mutex)表示特征 A 排斥特征 B,两者不能在同一个产品中同时存在.

在基于特征的可变性建模方法中,主要关注特征模型的组织结构、特征之间的关系、特征与核心资产的关联等.各种方法主要是扩展了基于特征的建模方法,在特征之间的关系表示和特征与核心资产的关联方面开展研究.其中,Fuzzy'03,Pure::Variant'04,Cardi'05,ReqiLine'05 等方法主要从特征之间的关系角度进行了研究.IVMAspect'04 和 FD'08 对于复杂系统的建模支持进行了研究.Templ'05,IVMArchi'06,Cycle'06,FVP'05 等方法主要从特征与核心资产的关联角度进行了研究.

特征模型中的每个特征可以进行赋值或与类型/值进行关联,Fuzzy'03[9]的特征权重利用模糊数值表示,并根据特征的使用频率确定.Pure::Variant'04[10]利用特征模型和家族模型实现可变性建模.每个特征利用类型/值表示非布尔型的特征信息,利用约束(restrictions)表示特征之间除了基本关系之外的任意关系,家族模型描述构件的内部结构和构件与特征的依赖.基于基数的方法扩展了特征之间可选性的表达能力,Cardi'05[11]方法使用类似 UML 多重性机制扩展特征模型,包含特征基数和组基数.在 ReqiLine'05[12]方法中,引入新的特征之间的关系,利用层次关系(composition,implementation,specialization)描述父特征和子特征之间的关系,利用依赖关系(implication,modification)描述特征之间的依赖关系,扩展了描述可变特征之间关系的能力.

特征模型在描述具有横切特征的代码时,会出现可扩展性问题和支持复用的问题.IVMAspect'04[13]提出的编程语言 Caesar 结合了特征模型和面向方面的优点,实现可变性建模.FD'08[14]利用上下文可变性模型描述产品所在的上下文的可变性,并与特征模型进行组合形成多产品线特征模型.这两种方法主要是扩展了特征模型对于复杂系统的建模能力.

基于特征的建模方法可以提供产品家族产品特征的简洁表示,特征模型的结构良好,通过识别特征之间的关系,将各个单独的特征组织成一个有机的整体.该方法的一个缺点是无法在特征模型中实现特征到核心资产的映射.为了实现特征到核心资产的映射,Czarnecki 等人提出一种基于模板的建模方法 Templ'0[15],利用特征模型和模型(例如 UML 模型)模板实现可变性的建模,每个模型模板中的元素与特征关联.同一个特征跟需求规格、设计模型和实现等核心资产都有关联,FVP'05[16]利用特征可变性模式表示跟某个特征相关的核心资产,模式由角色组成,角色跟跨越多个核心资产的模型元素关联,方便同一个特征的不同核心资产之间的追踪.特征模型需要与产品线体系结构进行关联,Cycle'06[17]和 IVMArchi'06[18]都利用决策模型(捕获所有模型的可变性元素之间的约束的表格)将特征和设计层次元素进行关联,IVMArchi 方法集成了决策模型、特征模型、ADL 和约束模型,分别利用高层和底层决策变量实现特征和体系结构的配置.

基于特征的可变性建模方法是一种描述共性和可变性的建模方法,但是该方法主要用于表示软件需求层次的可变性.利用特征对软件可变性建模,使得软件在较高层次上具有良好的可裁剪性,实现需求层次上的复用.但是由于其只关注于需求层次,因此,为实现产品构建(根据可变性模型和核心资产实现具体产品的组装),该

© 中国科学院软件研究所 http://www.jos.org.cn

方法需要建立特征到产品线体系结构等核心资产的映射关系,实现不同阶段可变性模型元素的可追溯性.

2.1.2 基于 UML 的可变性建模方法

UML 是广泛使用的系统建模语言,利用 UML 提供的建模机制可以进行系统的可变性建模,例如:利用类的组合、基于插件、基于 UML 模型模板和基于实例化(specification)和重定义(redefinition)的可变性建模[19].UML 的上述建模机制无法满足多种 UML 模型的可变性建模需要,因此,基于 UML 的可变性建模方法的重点在于如何针对不同的模型,利用 UML 的扩展机制,表示可变点、可变体以及可变体的关系.接下来分别对用例模型、活动图和类图,对比基于 UML 进行可变性建模的不同方法的扩展机制.

用例模型是需求阶段重要的核心资产,UMLMDA'11[20]扩展 UML,利用构造型«variant»表示可变的用例,用构造型«alternative»,«specialization»和«option»区分 3 种可变性类型,并且与«extend»扩展关系结合实现可变性建模.由于该方法基于«extend»扩展关系实现可变性建模,因此并没有实现用例可变点的建模.

软件的动态行为的可变性也是可变性建模的一个重要方面,活动图适用于对系统的动态行为建模.ExtUML'02[21]利用构造型«variable»和标记值 feature 对于活动图和构件图进行可变性建模,«variable»和 feature 标识图中的可变部分.在对于构件图的建模过程中,一组特征被建模为包(package),而特征本身建模为构件,可选构件利用«variable»表示,构件之间的依赖关系利用特征模型的依赖关系{or},{xor}表示.

设计层次的类图的可变性描述是可变性建模的重点,UMLM'01[22]方法利用«variationPoint»和«variant»构造型描述类图的可变点和可变体,可变点通过泛化(generalization)/参数化(parameterization)关系关联到可变体.可变体之间的依赖关系通过«mutex»或«require»构造型描述.元素之间的演化约束利用构造型«evolution»表示.同样是对类图建模,Consolid'06[19]定义«optional»,«interaction»,«variant»等构造型实现可变性建模,由于该方法还定义了一种基于模型元素替换的可变性建模机制,因此适用于对类图之外的 UML 模型的可变性建模.

产品线体系结构是产品线的重要核心资产,因此对于软件体系结构的可变性进行建模,是产品线可变性建模的关注点之一.UMLArchi'08[23]指出,体系结构的可变性元素包含构件、连接器和接口,可变点类型包含可选性«opt vp»和选择性«alt vp»,可变性建模在不同层次的体系结构实现,抽象层次的体系结构只是利用«alt vp»表示出可变点信息,第 2 层体系结构表示了可变点和可变体在内的详细可变性信息.

除了扩展 UML 实现可变性建模,基于 UML 的方法也可以跟其他方法结合来实现可变性建模,例如特征模型 UMLC'05[24]和面向方面技术 MSVCM'10[25].MSVCM'10[25]扩展了用例图,基于文本描述的用例模型包含用例名称、用例和面向方面用例,在面向方面用例中定义了连接点(joinpoints)和插入到基础用例模型中的用例.基于配置知识和配置过程,实现了功能、数据、控制流的可变性.

表 2 总结了基于 UML 的可变性建模方法的构造型和扩展,从表中可以看出,除了少数方法不支持可变点和可变体之间的依赖的表示外,虽然不同方法的构造型的表示形式不同,但是都能提供可变点、可变体和依赖关系的表示.不同方法的区别之处还在于提供了"其他"构造型,实现演化、基数可变性的支持.如果利用 UML 实现特定领域的可变性建模,需要结合领域系统模型的可变性建模需求,确定相应的构造型实现建模.

Table 2 Stereotype of variability modeling methods

表 2 不同建模方法的构造型

方法	可变点	可变体	可变体之间的依赖	可选元素	其他
UMLM	«variation point»	«variant»	«mutex», «require»	«optional»	«evolution»
ExtUML	Feature model	«variable»	Feature model	Feature	
Consolid	«optional»	«variant»	«interaction»	«TypeAlt»	«range», «it[1…6]»
UMLArchi	«alt vp» «opt vp»	«variant»		«optional»	«altv vp», «optv vp»
UMLMDA		«variant»		«alternative»	«option», «specification»
UMLC	Variation point	«optional»	Exclusion	Dimension-Value	Generalization
MSVCM	Pointcuts	Advice			Scenarios

表 3 总结了基于 UML 的可变性建模的不同方法支持扩展的 UML 图,从表中可以看出,不同方法对于建模对象的支持程度不同.对用例图、类图和构件图等静态模型的支持度比较高,但是对于顺序图、状态图和活动图的支持度并不高.一方面是因为在动态模型中,模型的可变性类型比较多,不容易进行表示;另一方面,可能是

© 中国科学院软件研究所 http://www.jos.org.cn

因为该方法的应用领域没有对于动态行为进行可变性建模的需求.因此,如果要实现动态模型的可变性建模,需要结合顺序图、状态图和活动图的元模型,分析其可变性类型并扩展 UML 的构造型.

Table 3 UML diagrams annotated with stereotypes

表 3 建模方法扩展的 UML 图

方法	用例图	类图	顺序图	状态图	活动图	构件图
UMLM	–	+	–	–	–	–
ExtUML	–	–	–	–	+	+
Consolid	+	+	+	+	+	+
UMLArchi	–	–	–	–	–	+
UMLMDA	+	–	–	–	–	–
UMLC	+	–	–	–	–	–
MSVCM	+	+	–	–	–	–

注:“+”表示支持,“–”表示不支持,表 6 中“+”,“–”的含义与此相同

基于 UML 的可变性建模方法通过扩展 UML 实现可变性建模,该方法的优点是模型表达比较直观,能够促进用户和开发者之间的有效沟通,可以实现不同类型系统(嵌入式、信息系统等)的不同开发阶段的模型的可变性建模,并可以利用现有 UML 建模工具实现建模.缺点是构造型(stereotype)和标记值的大量使用,会导致模型变得杂乱.另外,该方法对产品构建的支持度不高,实现可变性管理和配置的工作量较大.

2.1.3 面向方面的可变性建模方法

近年来,基于关注点分离的面向方面的可变性建模方法受到越来越多的关注,并且与其他建模方法有结合点.AspUsecase’08 和 MSVCM’10 是基于 UML 和面向方面结合的可变性建模方法.软件产品线中的共性和可变性概念类似于面向方面中的核心(core)关注点和横切(crosscutting)关注点的概念.其中,共性与核心关注点对应,可变性与横切关注点对应,可变性跟系统的共性是正交关系(即相互独立),能够独立地与系统共性部分进行组合.面向方面的可变性建模方法借鉴面向方面的概念,利用方面容器(aspect container)表示可变体,定义类似于面向方面中切入点的机制,实现可变性的绑定,可以提供需求模型、产品线体系结构、系统实现等可变性建模的全生命周期的支持.

对于用例模型,AspUsecase’08[26]方法定义«aspect»关系 UC_{adv}×*acond*×*apcuts*×*baseUCs*,通知用例 UC_{adv} 在 *acond* 条件满足的情况下,通过切入点表达式 *apcuts* 实现与基础用例 *baseUCs* 的连接(join),«variability»关系是«aspect»的实例化.AspVisual’07[27]扩展需求建模语言 ADORA,利用方面容器表示可变体,基于标记 O 和 X 实现可选择性的表示,利用决策变量表示连接关系,即实现方面容器内可变体与共性体的绑定.在用例的可变性建模中,可变性特征与用例场景之间存在相互交织的现象,每个可变特征跨越多个用例场景.增加一个可变特征会导致多个用例场景的修改,Crosscut’09[28]利用横切机制实现用例场景的可变性建模,将场景可变性分为数据的可变性、功能的可变性和控制流的可变性,并且每种可变性表示为用例模型、特征模型、产品配置的组合,实现了可变性管理与场景规格的关注点分离.

对于产品线体系结构,AspNatsuko’08[29]利用可变体方面和可变体规则集对体系结构建模.可变体方面对系统功能建模并用构造型«aspect»表示,可变体规则集表示可变体方面之间的关系.利用构造型«optional»和«aspect»的组合表示可选方面,利用{alternative}表示选择性方面.规则集包含可选规则和选择性规则,在满足规则的前提下,可变体方面才可以被选中.EAspect’09[30]是复旦大学提出的扩展方面机制的体系结构建模方法,构件的可变性通过 Options 和 Variants Schema 来描述.每个 Optional 或者 Alternative 变体元素都关联一个 Guard 条件.当条件被满足时,Optional 元素或 Alternative 的可变体将被包含在体系结构中.

对于系统实现,AspIris’08[31]基于面向方面的思想,在元模型层和模型层,利用名字匹配和切入点表达式确定切入点,使用 XWeave 模型编织工具进行模型合并,并实现代码合并.在产品构建过程中,利用 Pure::Variant 工具实现特征模型的配置,并基于配置结果,利用 XWeave 进行模型编织实现可变性.

基于面向方面的可变性建模方法的优点是:可以在方面中封装可变性信息,并在切入点实现可变的方面和基础模型绑定,比在单个模型中显式地表示可变性适应性更强.但是,如果不同阶段的系统模型采用不同的可变

© 中国科学院软件研究所 http://www.jos.org.cn

性建模方法,可变性管理的错误率将增加.因此,如果采用面向方面的可变性建模方法,需要对产品线生命周期的各个阶段的可变性模型都使用面向方面建模,并实现不同阶段的可变方面的转换.

2.1.4 基于决策模型的可变性建模方法

基于决策模型(decision model)的可变性建模方法利用决策模型表达可变性.决策是实现特定目的的一系列选择,可以表示可变性模型中的可变点.基于决策的可变性模型包含决策模型和核心资产模型,决策模型表示问题空间的用户高层次的可变性,核心资产模型表示解空间的技术层次的可变性,两个模型之间存在追踪关系.该方法一般包含 3 要素:决策、核心资产和决策依赖,在建模过程中,一般将决策与决策变量关联,通过给决策变量赋值实现决策的过程;核心资产表示跟决策相关的核心资产(构件等);决策依赖表示决策之间的依赖关系.

表 4 是基于决策的可变性建模方法对比,从核心资产模型、决策变量、决策依赖、规范化程度和工具支持等 5 个方面进行了对比.可以看出:不同方法的建模对象不同,但是都偏向于实现层次;不同方法都实现了工具支持,但是不同方法在决策依赖和决策模型定义的规范化程度方面存在差异.

Table 4 Comparison of decision modeling approaches

表 4 基于决策的可变性建模方法对比

对比分类	Korba	DoplerVML	VManage
核心资产模型	UML 模型	软件构件、测试用例、文档等	构件描述文件
决策变量	Boolean, Enumeration	Boolean, String, Integer, Enumeration, Real, Set	Boolean, String, Integer, Enumeration, Real
决策依赖	赋值影响	基于规则	基于逻辑
规范化程度	差	较好	较好
工具支持	Decision modeller	DecisionKing	V-define, V-Resolve

对于 UML 模型,Korba'00[32]的决策模型通过表列(tabular notation)描述,包含基于文本的问题和可能的答案.该方法区分高层决策和简单决策,高层决策的解析是给简单决策赋值,简单决策解析实现核心资产元素的绑定.决策依赖利用不同层次决策间的赋值影响表示.该方法的缺点是决策模型的定义不规范.

Dhungana 提出的 Dopler 是可扩展的面向决策的可变性建模语言.DoplerVML'06[33]关注软件产品的自动化构建,决策包含决策名称和一系列相关问题,决策变量定义了问题的可能的答案类型,通过决策变量赋值回答问题,并实现决策的选择.该方法通过定义领域特定元模型,实现核心资产类型和属性的描述.每个决策都通过包含(inclusion)条件直接跟核心资产关联.

欧洲软件研究所和 IKV++ Technologies AG 公司提出 VManage'03[34]方法,利用 XML 描述的决策模型记录了领域分析阶段的用户需求和相关的规则与约束,每个决策用决策标识与描述、决策变量与默认取值以及决策间的依赖表示.该方法包含两种决策间的依赖关系:一个决策的取值影响另一个决策的取值和一个决策的取值决定另外一个决策的取舍.不同于 Dopler 方法,Vmanage 支持决策间的逻辑关系表达式.

基于决策模型的可变性建模方法主要实现可变性建模和产品构建,由于决策模型直接与核心资产关联,因此可以更好地实现产品构建.决策模型可以给出问题以及选择答案的理由,通过回答问题进行配置,比较符合人的思维习惯,与其他方法相比,可以更好地辅助产品构建.但是,由于决策模型的建立需要具有丰富领域知识的领域专家参与,如何捕获各种决策,并结合系统可变性分析完善决策模型是一个难题.

2.1.5 基于目标的可变性建模方法

在对产品线的需求进行可变性建模的过程中,需要获取组织、功能和非功能属性并表示其可变性.目前的可变性建模方法对非功能属性的考虑较少,基于目标的可变性建模方法能够有效地捕获非功能属性,为实现不同需求间关系的推理奠定了基础.基于目标的可变性模型与特征模型类似,基于目标模型的子目标之间的关系实现目标之间的依赖关系.

为实现需求阶段的可变性建模,GoalMaps'05[35]基于目标驱动的过程模型 Map 实现,利用目标、策略和分区的组合实现可变性建模.其中,策略是实现目标的方法,在模型中表示为目标间的连接.分区是三元组$\langle G_1,G_2,S_{ij}\rangle$,表示在策略 S_{ij} 指导下从目标 G_1 到目标 G_2 的路径,每个路径对应一个特征.Map 的多线程拓扑结构对应特征的 Optional 关系,多路径拓扑结构对应特征的 Alternative 关系.该方法的缺点是还没有实现对产品构建的支持.基

© 中国科学院软件研究所 http://www.jos.org.cn

于目标的建模方法虽然与基于特征的建模方法在某些思路上一致,但是基于目标的建模方法在可变性获取方面具有优势.IStarLPS'09[36]采用基于基数的 *i**框架来实现可变性建模,并从中抽取特征模型.该方法通过在 *i**模型中利用基数实现可变性,将目标利用基数(1…*)标记表示目标内的属性可以出现多次.

基于目标的可变性建模主要关注产品线需求的可变性建模,可以从用户的角度出发实现可变性建模,清晰地表示依赖、可追踪性和多关注点.但是,基于目标的可变性建模方法也存在对于产品构建支持不足的缺陷.

2.1.6 多维度可变性建模方法

软件系统的实质是一个多维度结构化的制品集合,用户、系统分析师和开发人员等不同的利益相关者从不同视角对于系统有不同的理解,没有一种表示方法能够覆盖所有利益相关者的关注点.可变性建模的另一个挑战是对包含有大量可变点和依赖关系的企业级模型的管理,随着可变性模型复杂度的提高,多维建模方法和多视角建模方法被提出,该方法可以基于不同的视图更好地描述产品线的可变性.

多维度可变性建模方法分为两类(表 5):一类是扩展特征模型,从不同视角增强可变性的表达能力,包括方法 NUI,MultiDvm,VPI 和 VPO;另一类是基于不同模型实现多维度可变性建模,包括方法 MultiMeta, VPModel 和 Covamof .从表 5 可以看出,不同建模方法的可变性维度并不相同,这是不同领域的描述需求不同导致的.因此,在应用于新领域时,需要根据系统的利益相关者及其具体需求,确定可变性建模的维度.

Table 5 Comparison of the dimension of variability modeling methods

表 5 多维可变性建模方法视图对比

方法	类型	视图名称
NUI	特征模型	业务视图、层次/行为视图、依赖/交互视图、中间视图
MultiDvm	特征模型	利用继承、叠加、聚合实现不同维度的建模
VPI	特征模型	特征模型的子模型(业务用户、安全、硬件视角)
VPO	特征模型	特征模型的子模型(操作系统、语言等视角)
MultiMeta	不同模型	用例模型、静态模型、协作模型、状态模型和特征模型
VPModel	不同模型	需求视图、构件视图、静态视图、动态视图
Covamof	不同模型	核心资产(特征模型、体系结构、构件实现)、可变性(可变点视图、依赖视图)

特征模型在表示特征的绑定时间、特征交互、特征模型与体系结构模型的映射等方面都不能提供很好的支持.NUI'10[37]方法对特征模型的每个特征以不同视图进行扩充,业务视图增加了项目管理和成本效益分析,行为视图添加了每个特征的行为模型,交互视图展示了特征之间的依赖和交互关系,中间视图增加了特征模型和体系结构模型映射采取的步骤.该方法弥补了特征模型的一些缺点,但是提供了 4 种视图.特征模型可以划分为任意不同的维度,MultiDvm'11[38]利用多维度建模语言 Velvet 实现特征模型的不同维度的建模.

基于视角(viewpoint)的可变性建模方法从不同的角度实现建模,VPI'10[39]将视角定义为与利益相关者相关的特征模型的子模型,子模型包含父特征模型中的根节点,并通过模型组合实现不同视角的模型合并[39].对于大型并且持续演化的产品线,特征模型变得复杂并且难于管理,VPO'09[40]方法利用与 VPI'10 方法同样的思路,利用不同的视角描述特征模型的子模型,定义了冲突消解规则实现不同视角的特征选择.以电话领域为例,定义了操作系统、语言等不同的视角.

基于不同模型实现多维度可变性建模的方法主要考虑多个模型如何有效的管理.针对不同阶段的可变性模型,MultiMeta'08[41]标识了需求模型、分析模型和设计模型的共性和可变性.定义了每种模型的元模型,并在元模型的元类属性中定义 kernel,optinal 和 variant 这 3 种构造型表示可变性.该方法在元模型层定义了各个模型之间的关系,实现不同模型元素的一致性管理.VPModel'04[42]主要在设计层次对于可变点进行建模,在需求视图中,可变点利用字符 vp 与 m(mandatory)或者 p(optional)组合表示.在构件视图中,利用.vpo 或.vpm 表示可变点.在静态视图中,利用 UML 标记值{variation point=vp⟨m∨o⟩⟨VariationPointName⟩}表示可变点.为了更好地实现对复杂依赖关系的建模和依赖之间交互的建模,Covamof '04[43]对产品线不同抽象层次进行建模,包含可变点视图、依赖视图和依赖之间的关系视图.可变点跟核心资产关联,利用包含描述信息的属性表示,描述了可变点的状态(open,close)、实现和绑定;依赖与一个或多个可变点关联,描述可变体的选择关系,包含 5 种类型:可选、

© 中国科学院软件研究所 http://www.jos.org.cn

选择、可选可变体、可变体和值;定义依赖交互(dependency interactions)表示依赖之间的关系.

多维度可变性建模方法可以实现不同阶段的可变性建模,但是偏向于需求层次.该方法从不同的视角进行可变性建模,可以实现关注点分离,并提高可变性建模的效率.但是,为保证多维度模型的一致性,多维度建模还需要解决如何将多视图组合在一起、如何表示多个不同视图的可变性之间的关系以及单个视图的改变如何影响另外一个视图并保证多个视图之间的一致性.

2.1.7 基于语义的可变性建模方法

在可变性建模的过程中,基于语义的描述可以实现对于建模元素的语义表示并实现形式语义(formal semantics),能够清晰地表达可变点和复杂依赖,支持平台无关建模和模型之间映射以及自动化推理.

为了实现可变性模型的形式化表达、处理和分析.SemanticM'06[44]提出基于 Triple 语义模型的语义模型,包含本体层、标记层和动态标记层这 3 个层次.本体层定义了产品线领域内的概念;标记层描述核心资产的属性和质量信息;动态标记层描述行为的可变性信息,并且连接本体层和标记层.该方法没有实现不同领域的本体的映射,也缺少建模实例.基于本体的可变性建模技术利用本体表示领域内的概念,Kumbang'07[45]利用 Kumbang 模型和 Kumbang 配置实现可变点的描述和配置,Kumbang 扩展了 UML 元类的构造型,将扩展 UML 的元模型的 Profile 定义为本体,建立了领域内特征、体系结构元素和他们之间关系的语义基础,实现模型和配置的检验,降低了错误的发生率.PDO'09[46]从领域、过程和服务的角度对于可变性进行识别和建模,定义领域本体、过程本体和服务本体,其中,领域本体定义了实体概念、操作、非功能需求和它们之间的语义规则,实现对可变性建模的语义支持,该方法并没有实现推理功能.

基于语义的可变性建模方法可以实现领域内可变性相关知识的重用,实现对可变性建模的语义支持,提高可变性模型的表达能力,并且可以实现自动化推理等分析功能.但是,目前的主要工作集中在基于语义实现领域内可变性相关知识的重用,并没有很好地实现基于语义的一致性检验.

2.1.8 基于领域建模语言的可变性建模方法

基于领域建模语言的建模是利用领域特定语言(DSL)实现可变性建模的一种方法,利用领域特定语言允许产品线建模在系统建模语言层次实现可变性.基于领域建模语言的可变性表示方法有两种:一种是在 DSL 的元模型定义可变性;一种是在分离的模型中定义 DSL 模型元素的可替换性,实现关注点分离.

在元模型定义可变性的方法中,EmbedDSL'09[47]在嵌入式系统中实现了基于模型的领域特定语言(matlab/simulink)的可变性,在 DSL 的元模型中定义可变性,借鉴特征模型的概念,引入 Variant 和 VP 等符号表示可变性.此方法的缺点是需要结合 DSL 才能定义可变性机制.基于模型分离的思想,Haugen 等人提出通用可变性建模语言 CVL(common variability language).CVL'08[48]利用模型元素替换实现可变性建模,包含基础模型、可变性模型、解析模型这 3 种模型.可变性模型描述了基础模型的元素如何被替换以产生新的产品模型,定义的替换包含:值、引用和片段,值和引用被用于修改模型元素的属性和引用,片段的替换通过定义需要替换的片段和替换片段实现.给定一个解析模型后,利用替换片段替换需要被替换的片段.CVL 的基础模型可以是 UML 模型.基于 DSL+CVL 的可变性建模方法可以使领域建模人员关注于领域建模,实现关注点分离.

领域特定语言一般关注特定系统的某个方面,由于关注范围小,很难实现重用.而且建立领域特定语言是一个迭代的复杂过程,因此需要实现模型层次的重用.基于领域建模语言的可变性建模方法可以针对特定领域,在元模型或分离的模型中定义领域特定语言的可变性.但是该建模方法的可扩展性较低,在应用于新领域时需要考虑领域特点,对元模型进行修改来实现可变性建模.

2.1.9 基于配置的可变性建模方法

关于可变性的研究最早在传统的机械领域和电子领域开展,称为产品配置,主要关注一个通用的产品如何经过配置生成符合用户需求的产品.因此,软件产品线领域的可变性研究可以借鉴产品配置技术.基于配置的可变性建模包含基于配置语言的方法 PCL'01 和 Koalish'04,以及基于配置模型的方法 Questionnaire'09.

基于配置语言 PROTEUS,PCL'01[49]实现硬件、软件和文档的可变性建模,定义了 6 种实体类型(family, version description,tool,relation,class,attribute type)和实体之间的关系.实体的属性分为信息属性和可变属性,其

© 中国科学院软件研究所 http://www.jos.org.cn

中,前者标识实体的固有信息,例如共性,后者标识实体结构和构造实体过程的可变性.可变属性决定构造实体的可变体时采取的步骤,例如,可以根据可变属性的值判断子系统如何映射到程序文件,但是规则只能利用条件表达式 if-then-else 实现.该方法的缺点是需要维护大量的配置文件.

在软件体系结构层次,Koala 是嵌入式软件的组件模型和体系结构描述语言.可变性建模语言 Koalish'04 扩展 Koala,并利用多种可变性机制表示可选构件和选择性构件.基于配置参数,系统可以配置出具有不同功能的系统,构件之间的连接随着配置参数的不同而不同[50].Questionnaire'09[51]的可变性通过基于问卷的配置模型实现.模型中包含多个问题,每个问题的答案是多个事实,一个事实被置为真,则作为问题的答案.可变性表现为多个事实之间的选择.该方法的难点在于缺少配置模型建立过程的详细描述.

基于配置的建模方法与产品构建结合紧密,可以支持偏向实现层次的可变性建模.由于配置模型中的配置规则往往需要映射到系统实现文件,当系统规模变大时,跟配置规则相关的系统模块的变更都可能会导致配置模型的变化,因此,可变性配置信息的维护难度加大,需要对配置模型进行有效的管理.

2.1.10 其他可变性建模方法

除了上述方法,还有许多其他类型的可变性建模方法,包括 VSL[52],SaaS[53],SimCOVAMOF[54],Brazilian[55],Ebusines[56],Automotive[57],SPLGraph[58],Patterns[59],Orthogonal[60],FOrdLogic[61]等方法.在这些方法中,正交可变性建模 Orthogonal'04[60]方法最为典型,它在单独模型中定义可变性信息,并在单独模型中定义了可变点的依赖关系,所有的核心资产都可以基于该方法实现可变性建模.该方法的缺点是无法很好地实现可追踪性的建模,并且没有提供指导步骤实现可变性的配置.

2.2 重要的可变性建模技术对比

在软件产品线可变性建模方法的研究与应用中,在特定的应用场景下,如何选择合适的可变性建模方法成为建模人员和研究人员关心的问题.可变性建模包含多种不同的方法,各种建模方法缺少比较,哪种方法适合于何种类型的产品线还不是特别明确,因此,建模和研究人员如何选择建模方法并没有太多的依据.本节从第 2.1 节的分类中选取了一些典型的方法,对于各种方法的差异进行了详细的分析.

典型方法的选取原则是尽量从每个分类中选择一个方法.首先,基于文献信息抽取过程中收集的文献引用率,对于相关方法进行排序;然后,根据第 2.1 节中建模和研究人员对可变性建模技术的关注角度,基于该建模方法是否存在工具支撑、是否存在比较典型的工程应用进行筛选;最终,Pure::Variant,UMLC,DoplerVML,IStarLPS,MultiDvm,Covamof,Kumbang,CVL,Orthogonal 这 9 种建模方法被用于进行分析,基本涵盖了第 2.1 节的所有分类.虽然面向方面的方法研究较多,但是由于缺少工具支撑和典型应用而没有被用于进行对比.

在领域工程和应用工程中,包含领域/应用需求分析阶段、领域/应用设计阶段和领域/应用实现阶段等.每个阶段都包含可复用的软件核心资产,领域/应用需求阶段主要包含特征和用例模型,领域/应用设计阶段包含静态模型(静态图)、动态模型(顺序图、协作图和状态图等)以及软件体系结构,领域/应用实现阶段包含构件、代码、配置参数和测试用例等.因为文档在各个阶段都是重要的核心资产,本文将文档独立出来.不同的建模方法对不同建模阶段的支持程度不同,见表 6.

Pure::variant 与 Orthogonal 都支持不同阶段的建模对象的可变性,Pure::variant 支持软件生命周期各阶段的可变性管理,由于 Orthogonal 在单独的模型中表示可变性,可以用于各种软件制品的可变性建模.UMLC 与 IStarLPS 支持偏向于需求层次的建模,无法实现其他阶段的可变性建模.DoplerVML 支持偏向于实现层次的建模,可以通过扩展元模型实现对动态模型的支持,目前公开的文献资料中还没有该方法对需求层次的用例模型的支持.MultiDvm 通过构件、框架和预编译命令等实现基于特征的软件产品线,偏向于实现层次.Covamof 与 Kumbang 面向基于构件的产品线,都支持体系结构和构件实现层次的建模,并在特征层次实现可变性描述.CVL 的基础模型是 UML 或 DSL 模型,所以支持用例模型、动态图的可变性建模.

© 中国科学院软件研究所 http://www.jos.org.cn

Table 6 Comparison of modeling stage and modeling asset

表 6 不同可变性建模阶段的建模对象对比

方法	领域/应用需求分析		领域/应用设计			领域/应用实现				文档
	特征	用例模型	静态模型	动态模型	软件体系结构	配置参数	代码	构件	测试用例	文档
Pure::Variant	+	+	+	+	+	+	+	+	+	+
UMLC	+	+	–	–	–	–	–	–	–	–
DoplerVML	+	–	–	+	+	+	+	+	+	+
IStarLPS	+	–	–	–	–	–	–	–	–	–
MultiDvm	+	–	–	–	–	+	+	+	–	–
Covamof	+	–	–	–	+	–	–	+	–	–
Kumbang	+	–	–	–	+	–	–	+	–	–
CVL	+	+	+	+	–	–	–	–	–	–
Orthogonal	+	+	+	+	+	+	+	+	+	+

从表 6 可以看出,各种可变性建模方法对需求阶段和实现阶段模型的支持度比较高.其中,对特征层次的支持度最高.支持设计阶段的静态与动态模型以及文档的可变性建模方法并不多,对动态图而言,一方面是因为设计模型的动态图在不同领域的应用不同,不同的动态图元模型不同,无法建立统一的可变性描述方法;另一方面是因为动态图往往比较复杂,并且其状态转变很难理解,同时,动态模型的验证也比较困难,无法保证其正确性.除了特定的领域外,对动态图的可变性建模需求并不高.由于文档内容的可变性机制与模型不同,而现有可变性建模方法主要针对系统模型,因此无法适用于文档.不同可变性建模方法对于不同阶段模型的支持度不同,不同的可变性建模方法的技术细节也不同,本文接下来对于不同建模方法的技术特征进行对比.

可变性的生命周期包含:可变性识别、可变性分析与设计、可变性定制与配置和可变性实现.可变性建模主要将可变性分析和设计的结果进行描述,并且基于产品构建进行可变性的定制和实现.参考 Sinnema[3]和 Dhungana[4]提出的建模方法对比规则,本文从建模、工具和应用 3 个角度对可变性建模技术进行对比,见表 7.从建模角度进行对比是因为可变性建模主要关注可变性如何表示、建模的抽象层次和质量属性如何建模;从工具角度进行对比是因为工具与可变性建模紧密相关,主要实现可变性模型的表达以及如何支持应用工程中可变性模型的应用;从应用角度进行对比是因为可变性建模方法与领域应用密切相关.各个对比角度的解释如下:

- 建模角度
  - ✧ 建模机制:可变性建模方法采用何种建模机制(第 2.1 节)实现可变性的建模;
  - ✧ 可变点:可变性通常会关联在某个可变点上,可变点表示软件产品线的可变性发生的位置;
  - ✧ 可变体:是实现可变性的不同选择,表明了可变性的选择范围;
  - ✧ 可变体关系:表示方法一般分为可选性、选择性和多选性;
  - ✧ 依赖规则:可变体之间存在依赖、互斥等关系;
  - ✧ 抽象层次:产品家族一般包含大量的可变点,并且可变性发生在不同粒度或层次上,例如特征、体系结构和构件等;
  - ✧ 质量属性:是产品的重要特征,可变性模型需要对产品家族的共性和可变的质量属性进行建模.
- 工具角度
  - ✧ 工具视图:在可变性建模过程中,单个视图不能描述模型的全部内容,因此,建模工具需要实现多视图支持;
  - ✧ 推理引擎:工具能否基于可变性模型的依赖规则和已经采取的配置步骤产生当前的配置选项,实现配置过程中的一致性管理.
- 应用角度
  - ✧ 产品实现:产品的实现语言以及可变性实现采用的方式,例如配置文件、静态链接库、预处理命令等;
  - ✧ 应用领域:该方法的应用领域或该方法已经应用的系统;
  - ✧ 演化支持:该方法是否支持可变性的演化管理.

© 中国科学院软件研究所 http://www.jos.org.cn

Table 7 Comparison of important variability modeling methods

表 7 重要的可变性建模方法对比

	对比项	Pure::Variant	UMLC	DoplerVML	IStarLPS	MultiDvm	Covamof	Kumbang	CVL	Orthogonal
建模	建模机制	模型分离	特征模型	决策模型	目标模型	特征模型	模型分离	模型分离	模型分离	模型分离
	可变点	特征	特征或用例	决策(decision)	特征目标	特征	属性描述	特征	组合可变性	符号 Vp
	可变体	特征	可变体	选择(choice)	特征目标	特征	可变体	特征	组合可变性	符号 Va
	可选性	支持	支持	支持	支持	支持	支持	支持	支持	支持
	选择性	支持	支持	不支持	支持	支持	支持	支持	支持	支持
	多选性	支持	支持	支持	支持	支持	支持	支持	支持	支持
	依赖规则	代数表达式	逻辑命题	逻辑表达式	战略依赖依赖互斥	命题公式	代数表达式	代数表达式	依赖	依赖或互斥
	抽象层次	不同层次家族模型	用例图特征模型	决策模型,核心资产模型	无	内部、外部可变性	特征、体系结构、构件	体系结构特征层次	3 个不同模型	不同模型
	质量属性	无	质量需求分析	无	软目标	无	依赖实体	无	非功能属性	Fama OVM
工具	工具视图	建模视图配置视图	用例图特征模型	元模型编辑可变性模型	无	组合视图	建模视图配置视图	建模视图配置视图	建模视图配置视图	建模视图配置视图
	推理引擎	一致性	模型检查	一致性检验	无	具体化	支持	支持	无	Fama OVM
应用	产品实现	C 和 C++	无	Java, C++ 和 C#	无	Java	C, C++, C#	无	Java 和 C	无
	应用领域	Pure:: systems	手机软件	Siemens VAI	商店系统	SQLite	Web 服务智能交通	Robert Bosch GmbH	船舶控制	Radio frequency warner
	演化支持	无	维护可追踪性	允许领域演化,演化模型	无	无	无	无	无	无

建模机制包含基于单个模型实现可变性建模和基于模型分离实现可变性建模两种:基于单个模型的建模利用单个模型表示系统的可变性,基于模型分离的可变性建模在分离的模型中表示可变性.其中,基于单个模型的建模方法包括 UMLC,IStarLPS,MultiDvm 方法,而基于模型分离的方法包括 Pure::Variant,DoplerVML, Covamof,Kumbang,CVL,Orthogonal 方法.虽然基于模型分离的方法可以清晰地表示可变性,但是增加了与核心资产关联的成本.

可变点表示、可变体表示和可变体的选择关系是实现可变性建模的核心,根据建模方法采用的建模机制和建模方法所属的分类,Pure::Variant,MultiDvm 和 Kumbang 方法基于特征表示可变点和可变体,因此支持可选性、选择性和多选性 3 种可变体的关系.UMLC,IStarLPS,Covamof,Orthogonal 和 CVL 都支持 3 种可变体关系.不同方法的可变点和可变体的表示机制与该方法的建模机制有关,其中,UMLC 采用特征和用例描述;IStarLPS 利用特征和目标描述;Covamof 利用实体属性描述可变点,并利用 Variant 表示可变体;Orthogonal 定制符号 Vp 和 Va 分别表示可变点和可变体;CVL 定义了类似特征模型的机制,利用组合可变体(composite variability)表示可变点和可变体.DoplerVML 的决策数据类型 Boolean 和 Enumeration 可以表示可选性和多选性,但是没有机制实现选择性.尽管不同方法的可变性的表示方法不同,但基本都能支持可变点、可变体和可变体关系的表示.

可变体的依赖规则包含依赖与互斥、代数表达式和逻辑命题等.对于依赖与互斥,IStarLPS 和 Orthogonal 都可以利用依赖和互斥表达依赖规则,IStarLPS 还可以利用目标模型中的战略依赖实现依赖关系.CVL 表示了依赖性依赖规则,但是没有表示互斥关系;对于代数表达式,Pure::Variant 利用逻辑编程语言 Prolog 表示特征之间关系,Covamof 的依赖关系包含逻辑、数值和名词性依赖,依赖规则利用基本的算术或逻辑操作符实现.Kumbang 定义了约束语言表达模型中的复杂依赖关系,并基于代数表达式实现;对于逻辑命题,UMLC 通过逻辑运算符和谓词 T 和 B 的逻辑关系式表示.DoplerVML 用 If 〈condition〉 then 〈action〉实现决策之间的依赖, condition 是用布尔表达式表示的决策变量.MultiDvm 利用关键字 Constraint 和命题公式表示,包含与、或、非、蕴含、等价等操作符.依赖性与互斥性能够表示简单的依赖关系,但是对于复杂的依赖关系则无法表达.基于代

© 中国科学院软件研究所 http://www.jos.org.cn

数表达式能利用算术或逻辑操作符实现表达式,但是由于定义了新的操作符,不便于使用人员的理解.基于逻辑命题表达依赖规则具有灵活性等特点,但是在表达复杂依赖规则的时候,需要保证公式的正确性.

为管理产品线中大量不同类型的核心资产,采用层次化技术,并利用不同的抽象层次实现可变性建模.Pure::Variant,Covamof,Kumbang 和 CVL 都实现了包含特征层次的层次化可变性建模,Pure::Variant 包含特征模型和家族模型,家族模型利用不同层次描述产品家族的软件元素.Covamof 实现了特征、体系结构和构件 3 个层次.CVL 包含基础模型、可变性模型和解析模型,可变性模型采用类似特征模型的机制实现.虽然上述方法实现了不同层次的建模,但是除了 Covamof 和 Kumbang 方法外,层次间模型的可追踪性都没有很好地进行处理.Orthogonal,DoplerVML 和 MultiDvm 也实现了系统模型的层次化建模,Orthogonal 支持不同的核心资产的可变性建模,因此也就支持层次化建模,该方法的缺点是没有实现可追踪性.DoplerVML 在核心资产模型中支持不同粒度的软件制品的可变性建模,该方法对于可追踪性的支持度较好.MultiDvm 对系统的不同维度进行建模,并区分内部和外部可变性.不同的方法在一定程度上都实现不同抽象层次的建模,可以实现关注点分离;而不同的应用的抽象层次不同,需要结合具体的领域确定.

质量属性对于软件产品线至关重要,在可变性建模管理过程中,需要对质量属性进行建模,并以此对模型进行分析和检查,减少缺陷的发生,但是目前只有较少建模方法关注质量属性.Covamof 对质量属性进行了建模,并且表示为依赖实体,仅有该方法考虑了质量属性可能受到功能属性的影响的情况.IStarLPS 利用目标模型中的软目标实现质量属性建模和推理,该方法非常适用于质量属性的建模.上述两种方法都支持质量属性的定量和定性表示.UMLC 指出,可变性建模过程中需要进行质量分析,并在建模过程中考虑了质量属性.CVL 提供了安全关键系统的非功能属性的确认和验证.Orthogonal 在单独模型中对质量属性进行建模,并将可变性模型和质量属性关联,提供 FaMa OVM 工具对可变性模型进行质量分析[62].由于产品线的核心资产的缺陷可能会传播到构建的产品中,因此,质量属性在可变性建模过程中应该得到重视.

可变性建模方法需要实现建模工具以指导建模过程,并提供不同的视图支持.UMLC 在建模工具中提供用例图和特征模型的建模,但是没有提供配置视图.Pure::Variant,Covamof,Kumbang 和 CVL 都提供工具支持可变性建模和配置过程,并提供了建模视图和配置视图.DoplerVML 支持团队协作建模.MultiDvm 提供组合视图支持可变性的建模和模型组合.Orthogonal 提供了建模工具 VARMOD-MODELLER 实现可变性建模,并为应用工程人员提供配置工具 VARMOD-DEVELOPER 实现产品配置.

推理引擎在配置过程中辅助配置并提供一致性检查的功能,大致上可以分为 3 类:

第 1 类是根据当前的配置持续判断模型的一致性[3],UMLC 提供了配置过程的模型检查,实现绑定过程中的关系语义检查.MultiDvm 采用逐步配置,在逻辑命题的约束下逐步将模型具体化.Orthogonal 实现了配置的追踪机制,基于工具 FaMa OVM 对配置是否满足约束进行验证;

第 2 类是根据当前的配置,通过反馈机制给用户提供决策,Pure::Variant 实现配置过程的一致性机制检测,如果检测到不一致将反馈给用户.DoplerVML 实现了一致性演化机制,提供配置过程中的反馈;

第 2 类虽然给用户提供了决策,但是只能在用户选择配置选项后才能给出反馈,并不能防止用户做出错误的选择,即第 3 类,实现此功能需要更好的推理机制,Covamof 基于推理引擎,能够根据高层次可变点的绑定,自动决定实现层次可变点的绑定.Kumbang 将配置知识转换为约束规则语言(weight constraint rule language),基于推理引擎减少可能的选择,能够防止用户做出错误的配置.

在产品配置成功之后,还需要通过可变性实现技术,实现具体的产品,因此对各种方法提供的实现机制和应用领域进行对比.Pure::Variant 支持 C 和 C++语言的可变性机制,例如预处理指令,该系统的应用为商用产品线 Pure::systems.DoplerVML 应用在 Siemens VAI 公司的钢铁自动化、工业自动化和客户关系管理系统等,分别用 Java,C++和 C#开发.UMLC 和 IStarLPS 分别在手机领域和商店系统中有应用.MultiDvm 基于 Java 实现了系统并应用在 SQLite 领域.Covamof 支持 C,C++和 C#,基于配置文件和预处理指令实现可变性,并且应用在 Web 服务领域和智能交通领域.Kumbang 在 Robert Bosch GmbH 的汽车周边系统领域进行了应用.Orthogonal 实现了 Radio frequency warner 产品线.虽然正交变化模型 Orthogonal 的建模思想易于实现可变性建模,但是实际应用

© 中国科学院软件研究所 http://www.jos.org.cn

并不多,因为该方法在特定领域中缺少工具的支持,并且维护可变性模型到基础模型的追踪比较困难.

在演化方面,软件产品家族是不断变化的,需要实现模型的演化管理,并维护代码与模型的追踪关系.当需要实现新产品线的时候,演化信息就变得非常重要,但是大部分建模方法都没有实现对于演化的支持.UMLC 对特征模型和用例模型之间的可追踪性进行维护,同时也对系统可能的演化提供了支持.DoplerVML 可以实现在元模型层次的领域演化,允许增加新的核心资产类型,并对演化模型的语义一致性进行检验.

从以上分析可以看出,不同的可变性建模方法在建模对象、建模技术、建模工具和应用领域的不同角度各有特点,通过表 6 和表 7 可以看出不同方法的特点.其中,Pure::Variant 和 Orthogonal 的优点是可以用于全生命周期的模型的可变性建模,但是 Orthogonal 在应用领域方面存在不足;UMLC 和 IStarLPS 主要用于需求层次,但是并没有提供产品构建工具支持;Covamof 和 Kumbang 主要实现体系结构层次的建模,但是 Kumbang 并没有特别典型的领域应用;DoplerVML 在 4 个对比角度都具有优势,并且可以应用于不同类型的系统中;MultiDvm 支持的建模对象偏向于实现层次,该方法的优势在于具有多视图建模的能力;CVL 在建模技术方面具有优势,但是该技术的实现和应用方面存在不足.

在选择可变性建模技术的过程中,可以根据该方法对不同角度的支持程度进行评判,典型的应用步骤包含:

首先确定应用领域,根据表 7 中的应用领域判断是否存在相关的应用,并从中选择该领域相关的可变性建模方法;

其次,明确建模对象,分析该领域的配置需求,获取可变性建模对象,并根据表 6 对不同的方法进行选择;

然后,判断可变性建模技术能否满足建模需求,分析建模对象的可变性类型,并根据表 7 中的建模技术对比,确定相应的可变性建模方法;

最后,根据建模工具对比,选择可变性建模方法相关的工具.

3 可变性建模技术研究趋势

为明确可变性建模技术发展的促进因素以及可变性建模技术的研究趋势,本节从研究的动机方面对现有的文献进行分类,按照存在的问题进行组织,从建模的需求和建模技术发展的角度出发,探讨可变性建模的发展趋势.主要包含多维度可变性建模、现有建模方法的集成、多产品线的支持、面向服务的支持、面向方面支持、体系结构层次建模、可扩展性和领域适应性等方面.表 8 统计了不同研究问题的相关文献,以分析可变性技术的研究趋势.

Table 8 Trends of variability modeling methods

表 8 可变性建模技术发展趋势

问题	方法
多维度建模	Covamof, MultiDvm, NUI, MultiMeta, IVMArchi, VPI, PCL
现有建模方法集成	UMLC, MultiMeta, IVMArchi, PCL, AspUsecase, MSVCM, Kumbang
多产品线的支持	FD, MultiDvm
面向服务的支持	SaaS, SimCOVAMOF
面向方面支持	AspUsecase, AspNatsuko, AspIris, Crosscut, AspVisual
体系结构层次建模	UMLArchi, Kumbang, EAspect
可扩展性	Pure::Variant, NUI, Covamof, DoplerVML, CVL
领域适应性	Automotive, Brazilian, Ebusiness, Covamof, DoplerVML, Pure::Variant

- 多维度建模

随着系统复杂度的提高,相应的可变性模型变得越来越复杂.为了避免高复杂度的可变性模型,不同利益相关者可以区分不同的可变性维度(例如硬件可变性和软件可变性),相对独立地进行可变性建模和配置.对于不同的维度分别进行可变性建模,一方面降低了不同利益相关者建模时的交互成本,另一方面,一些可变性维度是应用或领域无关的,可以更好地实现这些维度的可变性模型的重用.MultiDvm,NUI,VPI 等方法从不同的视角实现基于特征的可变性建模,并基于模型组合方法集成可变性模型.COVAMOF,MultiMeta,IVMArchi,PCL 等研究主要集中在利用不同的模型对于可变性进行建模.

© 中国科学院软件研究所 http://www.jos.org.cn

虽然不同领域的可变性建模的维度不同,但是都需要组合不同维度的可变性模型、处理不同可变性模型的交织问题.因此,研究问题主要是如何准确地划分可变性模型的维度、如何实现不同的维度的模型组合以及如何对于组合后的可变性模型进行分析,以保证不同维度的模型之间的一致性.虽然多维度建模可以降低单个可变性模型的复杂度,但是如果不同维度的可变性模型非常多,管理成本将迅速增加,可以根据不同的需求,实现不同维度的可变性模型的按需组合.

- 现有建模方法集成

可变性建模的多种不同方法关注的建模阶段不同,集成不同的方法可以实现不同阶段的可变性建模,并弥补各自的缺点.因此,如何集成现有的可变性建模方法是研究的重点.MultiMeta 与 UMLC 集成了 UML 可变性模型和特征模型,Kumbang 结合了基于语义和配置的可变性建模方法,IVMArchi 集成了决策模型和特征模型等建模方法,AspUsecase 和 MSVCM 组合了基于 UML 和面向方面的建模方法.

建模方法集成主要集中在特征模型、决策模型和基于 UML 方法的集成.基于特征的建模方法主要关注需求层次的可变性建模,并且在大部分领域都适用;基于决策的建模方法由于跟产品构建过程紧密结合,所以在实现可变性模型的配置过程指导方面具有优势;基于 UML 的可变性建模方法比较简单.不同的可变性建模方法各有优势,实现不同抽象层次模型的可变性建模,但是需要维护不同模型间的一致性,并解决集成不同方法带来的可变性建模复杂度提高的问题.

- 多产品线支持

随着系统的系统(system of systems)的发展,系统变得越来越复杂,并且包含多个不同的子系统,形成多个产品线.因此,多个产品线如何管理将是一项重要的课题,对于可变性的协同建模的研究将变得十分重要.FD 方法和 MultiDvm 方法都基于特征的建模方法实现多产品线的建模,并提供了描述多产品线依赖的机制.在多产品线的场景下,不同的系统开发人员专注于自己的系统,很少了解其他系统,但是不同的系统存在交叉,具有相似的软件模块,因此形成一些横切的可变性模型,需要对多个产品线的横切可变性模型进行单独的建模与管理.多产品线需要协同建模的支持,并考虑不同可变性模型之间的依赖关系,实现跨越系统边界的依赖关系的管理,保证不同产品线的可变性模型之间的一致性.

- 面向服务的支持

面向服务领域也成为产品线技术应用的一个重要方向,由于业务需求具有动态特性,为了实现按需服务,面向服务的系统必须具有适应性,而可变性管理是实现适应性的重要手段.SaaS 和 SimCOVAMOF 实现面向服务领域的可变性建模.面向服务和传统软件的软件开发方式具有较大的差异,前者更加关注业务流程的建模,基于业务流程驱动灵活的系统的组合,为用户提供个性化服务.面向服务的可变性建模将促进面向服务的按需应用,研究问题主要集中在如何表示面向服务系统中的不同的可变性类型,尤其是对 Web 服务和业务流程的可变性进行建模,以及如何基于可变性模型实现 Web 服务的柔性组合和系统的定制.

- 面向方面的支持

面向方面建模与可变性建模的思路基本一致,实现横切关注点和可变性与系统共性的正交分离.另外,面向方面技术对于需求阶段、设计阶段和实现阶段的支持促进了可变性建模和实现技术的发展.AspUsecase, AppVisual,Crosscut 实现了需求阶段的可变性建模,EAspect 和 AspNatsuko 实现了体系结构层次的可变性建模,AspIris 实现了面向方面在可变性实现阶段的应用.

面向方面中的关注点分离实现了程序的适应性和可配置性,在单独的方面中表示可变性,可以清晰地表达可变性.但是,如果只利用面向方面实现某个阶段的可变性建模,而其他阶段采用别的可变性建模方法,则很容易引起不一致和错误.为了实现不同阶段系统模型的可变性表示的一致性,需要实现面向方面的可变性管理框架,实现不同阶段面向方面的可变性追踪与管理.

- 体系结构层次建模

在以构件为基础的软件产品线中,体系结构是整个开发过程的蓝图,体系结构层次建模跟产品线的设计紧密相关,是重要的核心资产.UMLArchi,EAspect 和 Kumbang 都在不同程度上支持体系结构层次的可变性建模,

© 中国科学院软件研究所 http://www.jos.org.cn

它们分别从扩展 UML、扩展方面机制和扩展 Koala 组件模型实现了体系结构层次的可变性建模.随着以决策为中心的体系结构设计方法的发展,在产品线体系结构的设计与维护过程中更加强调设计决策知识的作用,设计理由(rationale)是设计决策背后的原因,理由模型能够表示不同架构师的关注点,体现体系结构设计的不同选择,因此,基于问题和理由的可变性建模是体系结构可变性建模的一个重要研究方向.另外,在体系结构层次,由于动态行为模型和质量属性的可变性难以获取,成为可变性建模的一个挑战.

- 可扩展性

在产品线的应用领域中,往往都是复杂性较高的系统,包含成百上千的可变点,并且处在不断的增长中,因此,可变性建模需要支持大型系统,实现可变性建模的可扩展性(scalability).可扩展性的含义包含粒度方面和复杂度方面,即:1) 可变性模型需要支持不同规模的产品线;2) 产品线中需要添加新的系统时,不会对当前可变性模型的一致性造成影响,并能够实现可变性建模的增量式应用.基于特征的建模方法已经应用于包含 120 个特征的系统中.DoplerVML 方法支持 160 万代码行的应用[33],在大型的应用实例中,支持 324 个构件、160 个配置属性和 78 个决策,决策和核心资产间包含 699 个依赖关系.Covamof 实现了在 Dacolian B.V.公司的大型智能交通系统的应用,该系统包含 1 100 万代码行.CVL 实现了船舶控制领域的应用,因此也能适应大型系统的建模.NUI 提供了对于大型系统的建模支持,并提供了多个扩展接口.

可追踪性支持对于实现可扩展性非常重要,在可变性模型和基础模型分离的建模方法中,实现良好的可追踪性,可变性信息将会非常容易地获取,并且较容易地实现可变性模型的扩展.此外,可变性模型的演化能力是实现可扩展性的重要条件,良好的演化支持可以较好地实现可变性模型的复杂度方面的扩展.

- 领域适应性

不同领域的可变性建模方法不同,而同一领域内的不同类型系统的可变性建模方法也可能不同:Automotive,Brazilian 和 Ebusiness 应用在汽车、卫星发射和电子商务领域;Pure::Variant 的商业化产品 Pure::Systems 在 DaimlerChrysler AG 和 Spacebel 公司有应用;Covamof 在 Robert Bosch GmbH,Thales Nederland B.V 和 Dacolian B.V.有企业化应用,主要是自动化、作战管理和智能交通领域;Dopler 有包含 Siemens VAI 在内的工业化应用实例.不同领域的系统模型各有特点,同一个系统的可变性建模需求也可能不同,导致相应的可变性建模方法不同,因此,在可变性建模技术的研究过程中应该结合具体的领域特点开展.

4 结 论

软件产品线可变性建模是软件产品家族管理的重要技术.随着软件产品线研究的深入,对于产品线建模技术的研究从不同角度都有不同的研究,但是目前没有可变性建模技术的系统对比,如何选择合适的建模技术变得越来越难.本文利用系统文献综述的方法,按照系统文献综述的步骤开展可变性建模技术的系统综述,系统地分析了软件产品线可变性建模的相关技术,从建模的角度、建模的工具角度和应用的角度 3 个方面对重要可变性建模技术进行了对比,提供了各种建模技术的评价,最后探讨了可变性建模技术的研究趋势.本文的工作对于研究人员和工程应用都有借鉴意义,为建模人员和研究人员对于建模方法的选择和研究提供了一定的指导.为了实现软件产品线可变性建模技术的大范围应用,相关方面还需要继续进行深入的研究,包括可变性建模技术的改进、建立可变性建模的参考过程和实现可变性模型的演化支持等.

References:

[1] Lisboa LB, Garcia VC, Lucrédio D, de Almeida ES, de Lemos Meira SR, de Mattos Fortes RP. A systematic review of domain analysis tools. Information and Software Technology, 2010,52(1):1−13. [doi: j.infsof.2009.05.001]

[2] Haugen Ø, Wąsowski A, Czarnecki K. CVL—Common variability language. In: Proc. of the Int'l Conf. on Software Product Line. 2012. 266−267. [doi: 10.1145/2364412.2364462]

[3] Sinnema M, Deelstra S. Classifying variability modeling techniques. Information and Software Technology, 2007,49(7):717−739. [doi: 10.1016/j.infsof.2006.08.001]

© 中国科学院软件研究所 http://www.jos.org.cn

[4] Dhungana D, Grünbacher P, Rabiser R. Domain-Specific adaptations of product line variability modeling. In: Situational Method Engineering: Fundamentals and Experiences, Vol.244. Boston: Springer-Verlag, 2007. 238−251.

[5] Chen LP, Babar MA, Ali N. Variability management in software product lines: A systematic review. In: Proc. of the Int'l Conf. on Software Product Line. Carnegie Mellon University Pittsburgh, 2009. 81−90.

[6] Kitchenham BA, Dybå T, Jørgensen M. Evidence-Based software engineering. In: Proc. of the Int'l Conf. on Software Engineering. Washington: IEEE Computer Society, 2004. 273−281. [doi: 10.1109/ICSE.2004.1317449]

[7] Chen LP, Babar MA. A systematic review of evaluation of variability management approaches in software product lines. Information and Software Technology, 2011,53(4):344−362. [doi: 10.1016/j.infsof.2010.12.006]

[8] Kang KC, Cohen SG, Hess JA, Novak WE, Peterson AS. Feature-Oriented domain analysis (FODA) feasibility study. Pittsburgh: Carnegie Mellon University Software Engineering Institute, 1990.

[9] Robak S, Pieczynski A. Employing fuzzy logic in feature diagrams to model variability in software product-lines. In: Proc. of the 10th Int'l Conf. and Workshop on the Engineering of Computer-Based Systems. 2003. 305−311.

[10] Beuche D, Papajewski H, Schröder-Preikschat W. Variability management with feature models. Science of Computer Programming, 2004,53(3):333−352. [doi: 10.1016/j.scico.2003.04.005]

[11] Czarnecki1 K, Helsen S, Eisenecker U. Formalizing cardinality-based feature models and their specialization. Software Process: Improvement and Practice, 2005,10(1):7−29.

[12] Von der Maßen T, Lichter H. RequiLine: A requirements engineering tool for software product lines. Software Product-Family Engineering, 2004,3014:168−180.

[13] Mezini M, Ostermann K. Variability management with feature-oriented programming and aspects. In: Proc. of the 12th Int'l Symp. on Foundations of Software Engineering. New York: ACM Press, 2004. 127−136.

[14] Hartmann H, Trew T. Using feature diagrams with context variability to model multiple product lines for software supply chains. In: Proc. of the 12th Int'l Conf. on Software Product Line Conf. 2008. 12−21. [doi: 10.1109/SPLC.2008.15]

[15] Czarnecki K, Antkiewicz M, Kim CHP, Lau S, Pietroszek K. fmp and *fmp2rsm*: Eclipse plug-ins for modeling features using model templates. In: Proc. of the Int'l Conf. on Object-Oriented Programming, Systems, Languages, and Applications. 2005.

[16] Hammouda I, Hautamäki J, Pussinen M, Koskimies K. Managing variability using heterogeneous feature variation patterns. In: Proc. of the 12th Int'l Conf. on Fundamental Approaches to Software Engineering. 2005. 145−159.

[17] Lee J, Muthig D. Feature-Oriented variability management in product line engineering. Communications of the ACM—Software Product Line, 2006,49(12):55−59.

[18] Dhungana D. Integrated variability modeling of features and architecture in software product line engineering. In: Proc. of the 21st Int'l Conf. on Automated Software Engineering. 2006. 327−330.

[19] Bayer J, Gerard S, Haugen O, eds. Software Product Lines: Research Issues in Engineering and Management. Berlin: Springer-Verlag, 2006. 195−241.

[20] Azevedo S, Machado R, Bragança A, Ribeiro H. The UML «extend» relationship as support for software variability. In: Proc. of the Int'l Conf. on Software Product Lines. 2010. 471−475.

[21] Robak S, Franczyk B, Politowicz K. Extending the UML for modeling variabilities for system families. Int'l Journal of Applied Mathematics and Computer Science, 2002,12(2):285−289.

[22] Clauß, M. Generic modeling using UML extensions for variability. In: Proc. of the Int'l Workshop on Domain Specific Visual Languages. 2001. 11−18.

[23] Razavian M, Khosravi R. Modeling variability in the component and connector view of architecture using UML. In: Proc. of the Int'l Conf. on Computer Systems and Applications. IEEE Press, 2008. 801−809. [doi: 10.1109/AICCSA.2008.4493618]

[24] Zou SX, Zhang W, Zhao HY, Mei H. Modeling variability in software product family. Ruan Jian Xue Bao/Journal of Software, 2005,16(1):37−49 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/37.htm

[25] de Almeida RB. Odeling software product line variability in use case scenarios an approach based on crosscutting mechanisms [Ph.D. Thesis]. Recife: Federal University of Pernambuco, 2010.

[26] Anthonysamy P, Somé SS. Aspect-Oriented use case modeling for software product lines. In: Proc. of the Int'l Workshop on Early Aspects. New York: ACM Press, 2008.

© 中国科学院软件研究所 http://www.jos.org.cn

[27] Stoiber R, Meier S, Glinz M. Visualizing product line domain variability by aspect-oriented modeling. In: Proc. of the Int'l Workshop on Requirements Engineering Visualization. 2007. [doi: 10.1109/REV.2007.9]

[28] Bonifácio R, Borba P. Modeling scenario variability as crosscutting mechanisms. In: Proc. of the Int'l Conf. on Aspect-Oriented Software Development. New York: ACM Press, 2009. [doi: 10.1145/1509239.1509258]

[29] Noda N, Kishi T. Aspect-Oriented modeling for variability management. In: Proc. of the Int'l Conf. on Software Product Line Conf. Washington: IEEE Press, 2008. 213−222. [doi: 10.1109/SPLC.2008.44]

[30] Shen LW, Peng X, Zhao WY. Software product line architecture modeling and component composition implementation with extension of aspectual mechanism. Chinese of Journal Electronics, 2009,37(Z1):140−145 (in Chinese with English abstract).

[31] Groher I, Voelter M. Using aspects to model product line variability. In: Proc. of the Int'l Workshop on Software Product Line. 2008.

[32] Atkinson C, Bayer J, Muthig D. Component-Based product line development the KobrA approach. In: Proc. of the Int'l Conf. on Software Product lines. Kluwer Academic Publishers Norwell, 2000. 289−309.

[33] Dhungana D, Grünbacher P, Rabiser R. The DOPLER meta-tool for decision-oriented variability modeling a multiple case study. Automated Software Engineering, 2011,18(1):77−114. [doi: 10.1007/s10515-010-0076-6]

[34] Mansell JX, Sellier D. Decision model and flexible component definition based on XML technology. In: Proc. of the Int'l Workshop on Software Product-Family Engineering. Berlin: Springer-Verlag, 2003. 466−472.

[35] Bennasri S, Souveyet C, Rolland C. Modeling variability in requrements with maps. In: Proc. of the Advances in Information Systems. 2005.

[36] António S, Araújo J, Silva C. Adapting the *i** framework for software product lines. In: Proc. of the Int'l Workshop on Advances in Conceptual Modeling—Challenging Perspectives. 2009. 286−295. [doi: 10.1007/978-3-642-04947-7_34]

[37] Bashroush R. A NUI based multiple perspective variability modeling CASE tool. In: Proc. of the European Conf. on Software Architecture. 2010. 523−526.

[38] Rosenmüller M, Siegmund N, Thüm T, Saake G. Multi-Dimensional variability modeling. In: Proc. of the Int'l Workshop on Variability Modeling of Software-Intensive Systems. New Yord: ACM Press, 2011. [doi: 10.1145/1944892.1944894]

[39] Niu N, Savolainen J, Yu YJ. Variability modeling for product line viewpoints integration. In: Proc. of the Int'l Conf. on Computer Software and Applications Conf. 2010. 19−23. [doi: 10.1109/COMPSAC.2010.41]

[40] Mannion M, Savolainen J, Asikainen T. Viewpoint-Oriented variability modeling. In: Proc. of the Int'l Conf. on Computer Software and Applications Conf. Washington: IEEE Press, 2009. 67−72. [doi: 10.1109/COMPSAC.2009.19]

[41] Gomaa H, Shin ME. Multiple-View modelling and meta-modelling of software product lines. IET Software, 2008,2(2):94−122.

[42] Webber DL, Gomaa H. Modeling variability in software product lines with the variation point model. Science of Computer Programming, 2004,53(3):305−331. [doi: 10.1016/j.scico.2003.04.004]

[43] Sinnema M, Deelstra S, Nijhuis J, Bosch J. Covamof: A framework for modeling variability in software product families. In: Proc. of the Int'l Conf. on Software Product Lines. Berlin: Springer-Verlag, 2004. 197−213.

[44] Roshchin M, Graubmann P, Kamaev V. Semantic modelling for product line engineering. In: Proc. of the Information Theories & Applications. 2008. 387−390.

[45] Asikainen T, Männistö T, Soininen T. Kumbang: A domain ontology for modelling variability in software product families. Advanced Engineering Informatics, 2007,21(1):23−40. [doi: 10.1016/j.aei.2006.11.007]

[46] Cao BQ, Li B, Xia QM. A process-driven and ontology based software product line variability modeling approach. In: Proc. of the Int'l Conf. on Grid and Cooperative Computing. Washington: IEEE Press, 2009. 385−390. [doi 10.1109/GCC.2009.67]

[47] Botterweck G, Polzer A, Kowalewski S. Using higher-order transformations to derive variability mechanism for embedded systems. In: Proc. of the Int'l Workshop on Models in Software Engineering. 2009. 107−121. [doi: 10.1007/978-3-642-12261-3_8]

[48] Haugen Ø, Møller-Pedersen B, Oldevik J, Olsen GK, Svendsen A. Adding standardized variability to domain specific languages. In: Proc. of the Int'l Conf. on Software Product Line Conf. Washington: IEEE Press, 2008. 139−148. [10.1109/SPLC.2008.25]

[49] Tryggeseth E, Gulla B, Conradi R. Modelling systems with variability using the PROTEUS configuration language. In: Proc. of the Software Configuration Management. 1995. 216−240.

© 中国科学院软件研究所 http://www.jos.org.cn

[50] Asikainen T, Soininen T, Männistö T. A Koala-based approach for modelling and deploying configurable software product families. In: Proc. of the Software Product-Family Engineering. 2004. 225−249.

[51] La Rosa M, van der Aalst WMP, Dumas M, ter Hofstede AHM. Questionnaire-Based variability modeling for system configuration. Software and Systems Modeling, 2009,8(2):251−274. [doi: 10.1007/s10270-008-0090-3]

[52] Becker M. Towards a general model of variability in product families. In: Proc. of the Int'l Workshop on Software Variability Management. Groningen, 2003.

[53] Mietzner R, Metzger A, Leymann F, Pohl K. Variability modeling to support customization and deployment of multi-tenant-aware software as a service applications. In: Proc. of the Int'l Workshop on Principles of Engineering Service Oriented Systems. Washington: IEEE Press, 2009. 18−25.

[54] Suna CA, Rossingb R, Sinnemab M, Bulanovb P, Aiellob M. Modeling and managing the variability of Web service-based systems. Journal of Systems and Software, 2010,83(3):502−516. [doi: 10.1016/j.jss.2009.10.011]

[55] Burgareli LA, Melnikoff SSS, Ferreira MGV. A variability management strategy for software product lines of Brazilian satellite launcher vehicles. In: Proc. of the Software Engineering Research, Management and Applications. 2008. 1−14.

[56] Schnieders A, Puhlmann F. 6 Variability modeling and product derivation in E-business process families. In: Proc. of the Technologies for Business Information Systems. 2007. 63−74.

[57] Thiel S, Hein A. Modeling and using product line variability in automotive systems. IEEE Software, 2002,19(4):66−72.

[58] Maman I, Botterweck G. SPLGraph towards a graph-based formalism for software product lines. In: Proc. of the Int'l Workshop on Product Line Approaches in Software Engineering. New York: ACM Press, 2010. 40−47.

[59] Keepence B, Mannion M. Using patterns to model variability in product families. IEEE Software, 1999,16(4):102−108.

[60] Pohl K, Böckle G, van der Linden F. Software Product Line Engineering: Foundations, Principles and Techniques. New York: Springer-Verlag, 2005. 50−66.

[61] Elfaki AO, Phon-Amnuaisuk S, Ho CK. Modeling variability in software product line using first order logic. In: Proc. of the Int'l Conf. on Software Engineering Research, Management and Applications. Washington: IEEE Press, 2009. 227−233.

[62] 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, 2011,20(3):519−565. [doi: 10.1007/s11219-011-9156-5]

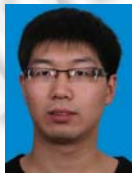
附中文参考文献:

[24] 邹盛享,张伟,赵海燕,梅宏.面向软件产品家族的变化性建模方法.软件学报,2005,16(1):37−49. http://www.jos.org.cn/1000-9825/16/37.htm

[30] 沈立炜,彭鑫,赵文耘.扩展方面机制的软件产品线体系结构建模及构件组装实现.电子学报,2009,37(Z1):140−145.

聂坤明(1985−),男,山东临朐人,博士生,CCF 学生会员,主要研究领域为软件产品线,产品线可变性建模,产品线体系结构.
E-mail: niekunming@cse.buaa.edu.cn

张莉(1968−),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为软件工程,需求工程,软件体系结构,过程建模和优化.
E-mail: lily@buaa.edu.cn



樊志强(1983−),男,博士生,主要研究领域为软件体系结构.
E-mail: fanzhiqiang@cse.buaa.edu.cn

© 中国科学院软件研究所 http://www.jos.org.cn