特征功能分析在产品线构件设计中的应用研究*

费洪晓，李　可
（中南大学 软件学院，长沙 410083）

摘　要：以功能交互的视角解剖特征模型结构，从中分析特征之间存在的功能交互，并根据其中的功能实现特点设计了一套特征—构件模型映射算法来构造面向特征的高内聚构件模型。这一过程实现了特征的内在功能完整性，其之上的功能耦合也随之解除。这不仅提高了可复用软件资产自动化效率，且大大减低了日后的维护成本。

关键词：特征模型；特征功能；功能交互模型；构件化；功能耦合
中图分类号：TP311.5　　**文献标志码**：A　　**文章编号**：1001-3695(2013)03-0799-04
doi:10.3969/j.issn.1001-3695.2013.03.040

Feature function analysis for application and research in product line component design

FEI Hong-xiao, LI Ke
(*School of Software Engineering*, *Central South University*, *Changsha* 410083, *China*)

Abstract: With analyzing the structure of feature model in perspectives of function interaction, this paper discussed function interaction between features, and proposed a feature-oriented componentization method in terms of characteristic of functional implementation. In this way it implemented feature's complete function with decoupling functions. This method not only enhances auto-efficiency of reusing software artifacts but also makes a great saving on the maintaining-cost for the artifacts in future.

Key words: feature model; feature function; function interaction model; componentization; functional coupling

0　引言

目前面向特征的领域建模在领域工程中保持着相当的热度，并推动着软件复用技术快速发展[1,2]。例如领域工程中的实践方法 FODA[3]、扩展了 FODA 的 FORM[4]、软件产品线方法[5]（software product lines，SPL）等，其中，在管理着相似系统集的共性、个性特征的 SPL 中，特征模型显得尤为重要。这主要是由于特征模型不仅在描述领域系统结构有着良好语义，同时在可复用构件开发中也有着天然优势——基于产品族共性及个性点的可剪裁机制。利用特征模型可剪裁性，软件产品便能通过定制获得，从而实现软件产品构造的自动化，大大提高产品开发效率和产品质量。由此可见，领域特征建模技术在产品线构件设计中发挥着重要作用。

领域建模是面向需求层次复用的过程，有效提高了软件复用水平。然而近年来利用特征组织需求结构并设计出领域特征模型的方法仍存在许多不足之处。文献[6]将特征以参数化的形式映射到应用模型上，实现了特征与构件模型的映射，从而达到应用阶段的软件资产可配置性目的，这在某种程度上提高了特征决策层次——通过特征选取得到符合特定需求的构件模型。但该方法中的特征之间存在着功能相互交织情况，例如一个模板模型的实现往往涉及多个存在条件及相关元表达式，反之亦然，由此，当修改某个特征的存在性时，其牵扯的相关建模元素将是众多的。显而易见，构建该模板模型及其往后的维护等相关工作量，尤其对于上规模的系统而言将非常巨大。文献[7]着重多粒度构件的实现，而非特征功能的实现，可以看出该方法弱化了特征模型可剪裁性。文献[8]提出了一种基于角色的特征模型构件化方法同样完成特征到构件模型的映射，该方法将特征操作化为一组角色来实现特征自由组合，然而作为特征抽象功能的替代，这组角色所识别的功能太过零碎，以致在构建构件模型的同时，需要其他特征角色片断似地提供功能支持，随着时间推移，往后构件的功能维护将会涉及多个特征结构的调整，这对庞大而复杂的系统来说，其工作量同样是不可承受的。

综合上述，问题的关键在于解除特征之间的功能耦合，实现特征功能的完整性。据此，本文就特征功能分析在产品线构件设计的应用展开研究，该研究方法首先将特征操作化为一组具有简单处理逻辑的功能单元（简单任务），然后根据特征模型中的特征依赖关系建立特征功能交互模型，并借此设计出面向特征功能单元的构件模型指导规范。该规范划定了功能单元的特征指代，实现了特征间功能的松耦合，为特征模型向需求模型的转换提供有力支持；此外，还成功阻断了特征模型中的功能混杂问题流向更高模型层次，这提升了决策（特征）模型的决策能力，有效缓解了以后系统的维护问题。

1　特征模型

特征是指代一组能标志系统提供某种服务（或功能）特性

收稿日期：2012-07-15；**修回日期**：2012-08-27　　**基金项目**：国家自然科学基金资助项目(61073186)；中南大学研究生教育创新工程资助项目(2011ssxt131)；中南大学米塔尔学生创新创业资助项目(10MY26)

作者简介：费洪晓(1967-)，男，副教授，主要研究方向为信息过滤、网络安全、软件工程(hxfei@csu.edu.cn)；李可，男，硕士研究生，主要研究方向为软件工程.

的需求[9],同时赋予特征之间结构上的关联语义,从而形成可描述领域系统中共性和个性的特征模型[4]。随着领域工程实践方法的发展,特征模型的结构语义得到了丰富扩展[10]。综合这些特点,以及存在的实际应用案例,这里给出一个电子商务关于音乐网站平台的特征模型片段,如图 1 所示。

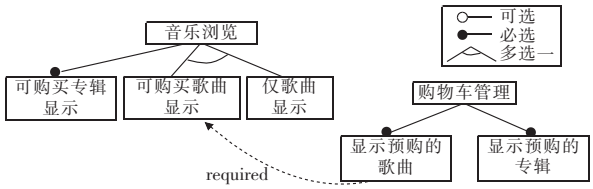

图 1 音乐网站平台的特征模型片段

从图 1 中可看出,特征模型是从需求层次上对系统功能进行描述的,纵向的特征关联表示系统功能抽象层次的分解,而横向关联则说明了特征间的功能依赖。简而言之,特征抽象了系统功能,并就功能实现问题上,特征之间存在着功能依赖。

1.1 特征功能

由于每个特征指代的一组需求具有相关性,而需求功能的实现往往涉及多个特征的实现。因此,现实中存在的特征与构件模型的对应关系往往为多对多,如图 2 所示。

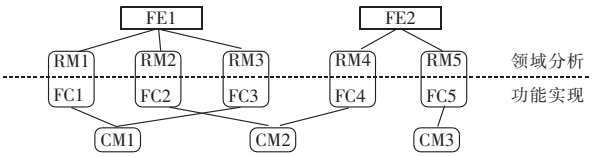

图 2 特征与构件模型的对应关系

在图 2 中,FE 表示特征,RM 表示单个需求,FC 表示对应需求 RM 所识别的简单功能单元,CM 表示构件模型。其中,构件模型 CM2 的实现涉及特征 FE1 和 FE2,而特征 FE1 的实现则涉及构件模型 CM1 和 CM2,显然地,特征 FE1 和 FE2 存在功能耦合,这也是本文力图要解决的问题。

1.2 特征/功能单元交互

作为描述系统结构的特征模型,其特点在于特征之间关联的良好语义[10],而文献[11]较好地诠释了特征的两种依赖类型,即静态依赖和动态依赖,并从侧面指出特征交互关系是以需求功能间的联系为基础,即源自特征的功能单元交互等同于特征间的交互。而功能单元是对数据的逻辑处理,因此可认为功能交互本质上是基于数据传递或数据存取的。这里首先将图 1 特征模型中的特征操作化为一组功能单元和数据资源,进而达到深层次分析特征功能交互的目的,如图 3 所示。

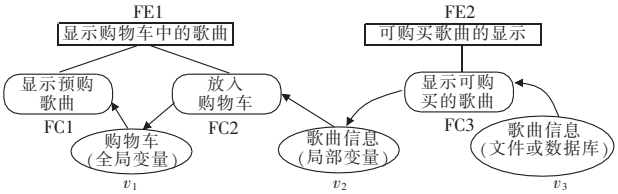

图 3 特征操作化

图 3 说明了特征之间功能上的交互关系。图中矩形表示特征 FE;圆角矩形为操作化自特征的功能单元 FC;圆形表示相关数据,用变量 v 指代;箭头表示数据信息的流向。数据信息的具体状态和存储类型及相关描述如表 1 所示。

表 1 数据的相关描述及影响

存储类型	状态	符号表示	描述
系统输入/输出	无状态	stateless	其状态与 v_2 的状态“瞬时”相似,它们之间的主要区别在于,该变量要么作为系统窗口到功能单元的输入,要么作为功能单元到系统窗口的输出,即其不能说明任何相关功能单元间的紧密性
局部变量	瞬时	transientState	如图 3 的 v_2,说明该数据类型的生存周期依赖于某一功能单元的处理周期,这象征功能函数中的参数传值,可表示功能 FC2 和 FC3 为紧密耦合关系,也就是完整功能块得到实现(本文称逻辑紧密的一系列功能单元完成的特定处理为完整功能块)
全局变量	暂存	saveState	如图 3 的 v_1,该存储类型的生存周期长于局部变量,其指代数据信息是功能单元的汇合点(完整功能块的输出或输入)
文件或数据库	持久	persistentState	如图 3 的 v_3,其性质几乎与 v_1 一样,唯一不同点在于其生存周期更长

1.3 特征功能单元聚集

根据表 1 中的描述进一步分析图 3 中特征功能聚集过程,该聚集方式有两种情况:

a)对于功能单元 FC2 和 FC3 所操纵的数据资源 v_2,由于其存储类型为局部变量,状态为瞬时的,所以 v_2 经 FC3 产生后,该数据需瞬间被 FC2 处理,来达到 FC2 功能实现的目的;同时 FC2 也是对 FC3 功能实现的延续,故有理由认为功能单元 FC2 和 FC3 在功能实现上是连贯的、一致的;又由于 FC2 依赖于 FC3(数据流向自 FC3 至 FC2),因此可认为 FE1 影响范围为 FC2、FC3,故将这两个功能单元聚集在一起,并指代为特征 FE2 的功能完整性实现。

b)当数据资源的存储类型为全局变量时,如 v_1,可认为 FC1 与 FC2 的紧密性较为松散,因为 v_1 存活周期较长,在一定程度上可供多个功能单元(如果存在除了 FC1 的功能单元)共享。这里与 v_1 相关联的两个功能单元前后依赖性为单方面的,也就是说,FC1 只是简单地需要 FC2 提供的服务,并且 FC1 不是 FC2 功能上的延续,又 FC1 操作化自 FE1,故将功能单元 FC1 的特征指代为 FE1。

其他方面,状态位为无状态,持久分别与瞬时、暂存的情况类似。依据上述聚集规则,可划分出特征功能层面上的完整性界线。可以说,数据信息的状态位对划分紧密性功能模块有着重要影响,也促使着特征功能完整性的实现,从而消除特征间的耦合。

2 构件模型设计

依据 1.2 节的特征功能交互过程来建立一套特征功能交互模型,然后基于该模型结构分析出特征抽象的功能结构,再利用 1.3 节的特征功能单元聚集规则,解除特征间的功能耦合,最后结合 2.3 节的构件模型生成算法,完成构件模型的建立。

2.1 特征功能交互模型

首先将特征操作化为一组活动,活动本质上是功能单元(也称为简单任务,为更贴近特征功能交互模型建立,故采用活动一词),其职能是对一些输入数据进行处理或操纵后再对外输出。同时,为从功能视角描述活动的组合,故将活动所需或产出的数据资源指代为变量。

定义 1 活动。定义为三元组(Ips,Ots,Ops),如图 2 的 FC1、FC2 等。其中:Ips 为活动所需的输入数据资源集合,也可称为活动所使用的变量集合;Ots 为活动输出的数据资源集合,即活动生成的或变更的变量集合;Ops 是活动对数据资源进行处理的操作集合。同时,鉴于活动是特征概念的简单功能

规约,也是对一系列操作的一种抽象,并结合面向对象工程中的数据信息封装概念,故设定活动输入流或输出流仅存在一种数据类型。

$$AS = \cup \{a_j \mid \exists i \exists j \cdot f_{i.}\text{fid} = a_{j.}\text{fid}\}$$

其中:a_j 为活动集合 AS 的某个元素,f_i 为特征模型中的某一个特征,$a_{j.}\text{fid} = f_{i.}\text{fid}$ 表示活动 a_j 演化自特征 f_i。

定义 2　变量。$v \in V$ 表示所有活动在执行职能时所需的或对该资源操纵后生成的、变更的数据资源的逻辑指代。

$$\bigcup_{i=1}^{n}(\text{Ips}(a_i) \cup \text{Ots}(a_i)) \subseteq V$$

其中:n 说明了特征模型中的活动数目,$\text{Ips}(a_i)$ 和 $\text{Ots}(a_i)$ 分别表示活动 a_i 的输入变量集合和输出变量集合。

定义 3　变量的产生者和消费者。本文称向变量写入更新的或生成的数据的活动为产生者 w;反之,从变量读取数据的活动为消费者 r,故对变量进行操作的生产者集合为

$$\text{PRDS}(v) = \{a_i \mid \exists a_i \cdot v \in \text{Ots}(a_i)\}$$

消费者集合为

$$\text{CONS}(v) = \{a_i \mid \exists a_i \cdot v \in \text{Ips}(a_i)\}$$

定义 4　变量状态位。$v.\text{stat}$ 表示变量经活动处理后的一种存储状态,其对特征功能完整性实现有着重要影响。各元素具体信息请参见表 1。

$$v.\text{stat} \in \{\text{"stateless"}, \text{"transientState"}, \text{"saveState"}, \text{"persistentState"}\}$$

定义 5　数据源点。$ds \in \text{DSS}$ 表示变量在数据流动过程中作为活动区域的数据流源点,活动区域可视为完整功能块所涉及的相关活动。

$$\text{DSS} = \cup \{v \mid (v.\text{stat} = \text{"stateless"} \wedge \exists v \in \text{Ips}(a_i) \cup (v.\text{stat} = \text{"saveState"} \vee v.\text{stat} = \text{"persistentState"})) \wedge v \in V\}$$

结合上述模型元素定义,图 4 展示了一种特征功能交互模型结构。考虑到特征间的各种依赖关系,该模型结构展现了典型的几种功能单元交互情况,较好地诠释了特征之间的依赖关系,基于该模型结构建立的构件模型算法将能适应其他各种特征功能交互模型结构。

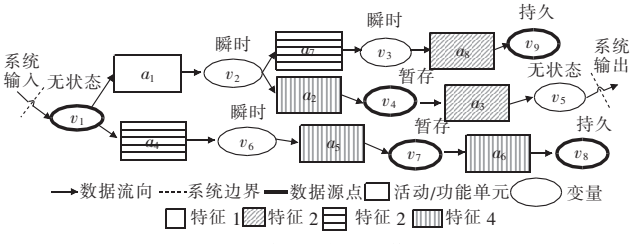

图 4　特征功能交互模型

2.2　变量状态转换表

变量状态转换表描述了活动对变量的操纵影响,也可以说是变量描述的数据从进入活动到活动产出的一个状态变化过程,如图 3 所示。

定义 6　变量状态转换表。R_t 为二维列表,活动 a_i 作为行,变量 v_j 作为列,$R_t(a_i, v_j, v_j)$ 表示列属性变量 v_j 被行属性活动 a_i 操纵后的状态变化过程,同时将活动 a_i 输出的数据资源放入变量 v 中,

各种表格值的说明如下:

$$R_t(a_i, v_j, v) = \text{null} \tag{1}$$

表示变量 v_j 经活动 a_i 操作后输出不是作为系统中某个活动的输入流,而是面向实体(如界面用户)的展示。

$$R_t(a_i, v_j, v) = \text{"transientProc"} \tag{2}$$

表示变量 v_j 经活动 a_i 处理后产出的变量 v 只是简单作为下一个活动的前置条件,可以说该步骤主要作用是辅助相关活动功能进一步得到完整实现,最后令 $v.\text{stat} = \text{"transientState"}$。

$$R_t(a_i, v_j, v) = \text{"saveProc"} \tag{3}$$

表示变量 v_j 经活动 a_i 处理后,其表现形式或性质发生改变,抑或简单地对该变量存储类型进行变更,令 $v.\text{stat} = \text{"saveState"}$。

$$R_t(a_i, v_j, v) = \text{"persistentProc"} \tag{4}$$

表示变量 v_j 经活动 a_i 处理后,其数据内容发生改变,令 $v.\text{stat} = \text{"persistentState"}$。

2.3　面向特征功能单元的构件化算法

本算法主要目的为利用特征模型分解的变量集合和活动集合作为中间介质来将特征映射到构件模型上,面向该介质的特征功能交互模型清晰地列出了特征抽象功能的分布情况,并根据功能单元聚集规则来实现特征功能完整性调整,有效地解除了特征间功能的耦合,以及成功阻断特征模型中的功能耦合流向更高的模型层次——构件模型。

该算法步骤主要内容为:以某个数据源点为起点,根据数据流动路线,到达另一数据源点或无消费者的变量作为终点,期间所覆盖的活动区域即应用特征功能聚集规则后生成的构件模型,并将该构件模型用最先并入其内部的活动所属特征来标志。具体合成算法如下:

定义 7　构件。作为软件体系结构的一个组成单元,构件具有相对独立功能,能向外界提供相关能力或资源。构件 Comp 定义为四元组 $\{\text{CFID}, \text{CProd}, \text{CConm}, \text{CAS}\}$。其中:CFID 表示构件受控的特征标志,CProd 表示构件提供的资源集合,CConm 表示构件请求的资源集合,CAS 表示构件所包含的活动集合。

构件生成算法:

a)初始化源点变量集合 $V\text{Set} = \text{DSS}$ 和活动集合 $A\text{Set} = AS$,而变量暂存集合 $\text{VST} = \varnothing$ 和两个活动暂存集合 $\text{AST1} = \text{AST2} = \varnothing$ 都是堆栈似存取的集合,即这三个集合取出元素时满足后进先出规则。

b)如果 $A\text{Set} = \varnothing$ 或者 $V\text{Set} = \varnothing$,算法结束;否则进入下一步骤。

c)从集合 $V\text{Set}$ 取出一个源点变量,得到对变量进行操作的活动集合 $\text{AST1} = \text{CONS}(v_k)$。

d)如果 $\text{AST1} = \varnothing$,移除 $V\text{Set}$ 中的 v_k,转至步骤 b);否则,将变量 v_k 放入集合 VST 中,新建一个构件 C,$C.\text{CProd} = \varnothing$,$C.\text{CConm} = \{v_k\}$,$C.\text{CAS} = \varnothing$,从集合 AST1 中取出一个活动 a_i,放入集合 AST2 中,令 $C.\text{CFID} = a_{i.}\text{fid}$,移除集合 AST1 中的 a_i,进入下一步骤。

如果集合 $|\text{AST1}| > 1$,考虑 AST1 中每个活动的特征所属问题,当某一变量为数据源点时,以该变量作为输入的多个活动可判定为属于不同的特征。根据定义 1,每个活动只能接收一种类型数据变量,故这些功能单元(活动)有其特殊的作用,满足其所属特征的抽象概念。例如图 4 中活动 a_1、a_4 都需对数据源点 v_1 进行处理,由于其各自具有特定功能目的,可认为 a_1、a_4 操作化自不同特征,现实情况也是如此,这将影响后续功能单元的特征划定。

e)如果集合 $\text{VST} = \varnothing$,转至步骤 d);否则,从集合 VST 取出一个变量 v_m,以及集合 AST2 中取出活动 a_j,根据定义 5 的变量状态转换表得出如下计算结果:

(a)当结果为式(2)时,则 $C.\text{CAS} = C.\text{CAS} \cup a_j$,$\text{VST} = \text{VST} \cup v$,移除集合 $A\text{Set}$ 中的 a_j,移除集合 AST2 中的 a_j,AST2 =

AST2 ∪ CONS(v)。如果集合 CONS(v_m) ∩ AST2 = ∅,则移除集合 VST 中的 v_m,转至步骤 e)。

(b)当结果为式(1)(3)(4)时,则 C. CAS = C. CAS ∪ a_j, C. CProd = C. CProd ∪ v,移除集合 ASet 中的 a_j。移除集合 AST2 中的 a_j,如果 CONS(v_m) ∩ AST2 = ∅,则移除集合 VST 中的 v_m,转至步骤 e)。

3 实例研究

针对图 4 所示的特征功能交互模型,可推出特征与构件模型原本的对应关系,如图 5 中顶部的构件模型结构 Comp1 所示,该构件实现了四个不同特征,这些特征间存在功能层次上的协作关系(图 4)。之后,依据特征功能交互模型,应用特征—构件映射算法后成功实现了构件模型中特征间的解耦合。整个操作过程如图 5 所示。

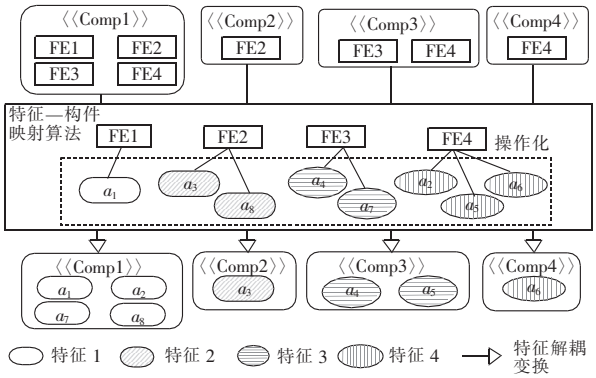

图 5 构件模型中特征解耦过程

最开始构件模型 Comp1 的实现依赖于其内部四个特征各自所识别功能单元的片段似支持,同样情况发生在 Comp3,而 Comp2、Comp4 则存在着特征功能部分冗余。应用构件生成算法,特征所标志的功能单元被重新调整,这有效地完成特征功能解耦,消除特征功能冗余,提升特征—构件映射效率,对产品线资产的自动化和可管理性起到了重要作用。特征—功能单元—构件模型三者的对应关系如表 2 所示。

表 2 特征—功能单元—构件模型的对应关系

特征(调整前)	功能单元	数据源点	数据终点	特征(调整后)	构件模型
FE1	a_1				
FE4	a_2	v_1	v_4,v_9	FE1	Comp1
FE3	a_7				
FE2	a_8				
FE2	a_3	v_4	v_5	FE2	Comp2
FE3	a_4	v_1	v_7	FE3	Comp3
FE4	a_5				
FE4	a_6	v_7	v_8	FE4	Comp4

此外,根据构件模型自身的功能特性进一步组合,可产生符合客户特定需求的系统。

变量 v 是构件请求或提供的资源,其可视为构件向外提供或请求的某项服务,故构件的服务提供集合为

$$PServ(c) = \{v_i \mid \exists v_i \cdot c.\mathrm{CProd}.v_i \notin \mathrm{Ips}(c.\mathrm{CAS}.a_j) \wedge c.\mathrm{CProd} \neq \varnothing\} \quad (5)$$

构件的服务请求集合为

$$RServ(c) = \{v_i \mid \exists v_i \cdot c.\mathrm{CConm}.v_i \notin \mathrm{Ots}(c.\mathrm{CAS}.a_j) \wedge c.\mathrm{CConm} \neq \varnothing\} \quad (6)$$

根据式(5)(6),对于任意两个构件模型 c_1、c_2,满足条件 $PServ(c_1) \cap RServ(c_2) \neq \varnothing$,或者 $PServ(c_2) \cap RServ(c_1) \neq \varnothing$,则认为构件之间存在服务调用关系,可通过构件组合得到特定需求系统。

经过以上步骤,得到面向特征的构件模型,如图 6 所示。

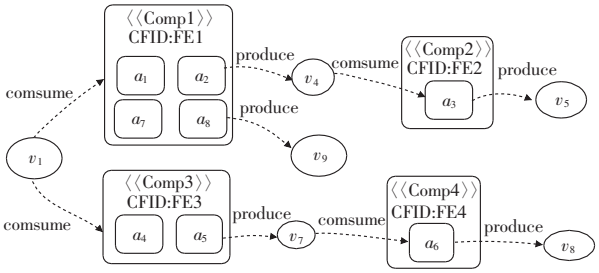

图 6 面向特征的构件模型

本算法旨在建立面向松耦合特征功能的构件模型,阻断特征功能混杂问题扩展至更高模型层次,极大地方便后续系统的维护、演化。

4 结束语

本文就特征在领域工程实践方法中的应用进行了深入分析,尤其细致描述了特征模型对产品线方法的重要影响——软件资产自动化。显然地,该自动化技术的质量结果主要取决于特征到构件的映射效率。考虑到目前若干特征—构件映射算法实现,并结合特征功能混杂交织以致影响软件资产复用效果和维护成本过高等问题,提出了一种面向特征功能单元的构件化方法。该方法从功能依赖角度深入解剖特征功能实现的过程,并由此构造功能紧密的构件模型,解决构件模型内部特征功能交织问题,提高特征模型的决策能力和构件模型的演化能力,从而满足目标用户的特定软件功能需求,以及提升系统相关人员建模效率和降低维护成本。

参考文献:

[1] 杨芙清,梅宏,李克勤. 软件复用与软件构件技术[J]. 电子学报, 1999,27(2):69-75.

[2] 张伟,梅宏. 一种面向特征的领域模型及其建模过程[J]. 软件学报,2003,14(8):1345-1356.

[3] KANG K C, COHEN S G, HESS J A, *et al*. Feature-oriented domain analysis(FODA) feasibility study, CMU/SEI-90-TR-21[R]. Pittsburgh: Software Engineering Institute, Carnegie Mellon University, 1990.

[4] KANG K C, KIM S, LEE J, *et al*. FORM: a feature-oriented reuse method with domain-specific reference architectures[J]. Annals of Software Engineering, 1998, 5: 143-168.

[5] CLEMENTS P, NORTHROP L. Software product line: practices and patterns[M]. Boston: Addison-Wesley, 2002.

[6] CZAMECKI K, ANTKIEWICZ M. Mapping features to models: a template approach based on superimposed variants[C]//Proc of the 4th International Conference on Generative Programming and Component Engineering. Berlin: Springer-Velag, 2005: 422-437.

[7] 王忠杰,徐晓飞,战德臣. 基于特征的构件模型及其规范化设计过程[J]. 软件学报, 2006,17(1):39-47.

[8] 张俊,刘淑芬,姚志林. 一种基于角色的特征模型构件化方法[J]. 电子学报, 2011, 39(2):304-308.

[9] TURNER C R, FUGGETTA A, LAVAZZA L, *et al*. A conceptual basis for feature engineering[J]. Journal of Systems and Software, 1999, 49(1): 3-15.

[10] SCHOBBENS P Y, HEYMANS P, TRIGAUX J C, *et al*. Generic semantics of feature diagrams[J]. Computer Networks, 2007, 51(2): 456-479.

[11] ZHANG Wei, MEI Hong, ZHAO Hai-yan. A feature-oriented approach to modeling requirements dependencies[C]//Proc of the 13th IEEE International Conference on Requirements Engineering. [S. l.]: IEEE Computer Society, 2005: 273-282.