基于领域特征本体的构件语义描述和组装

彭 鑫, 赵文耘, 钱乐秋
(复旦大学计算机科学与工程系软件工程实验室, 上海 200433)

摘 要: 构件的功能语义是复用者了解并判断构件可复用性的重要依据,因此必须在构件开发者和复用者共同的知识基础上进行构件描述. 目前已有一些方法引入本体作为构件语义描述的基础,但仍然存在语义描述不够精确以及无法支持语义的组装推导等不足. 引入了领域分析中提出的基于本体的领域特征模型作为构件语义描述基础, 在此基础上给出了构件端口语义、静态语义、语义协议的定义以及语义组装算法. 基于构件静态语义和组装算法可以在构件组装时进行语义合成,从而辅助开发者进行基于语义的构件适配和组装分析.

关键词: 构件; 业务语义; 领域; 特征; 本体; 语义描述; 组装
中图分类号: TN311 文献标识码: A 文章编号: 0372-2112 (2006) 12A-2473-05

Semantic Representation and Composition of Business Components Based on Domain Feature Ontology

PENG Xin, ZHAO Wen-yun, QIAN Le-qiu
(Department of Computer Science and Engineering, Fudan University, Shanghai 200433, China)

Abstract: Functional semantics of a component is an important factor for the reuser to understand and judge the reusability. So it is important to give the representation of components on the same knowledge base of component developers and reusers. Now there have been some approaches, which introduce ontology as the base of semantic representation for components. However, they have some shortages in precision and support for semantics reasoning when composition. A meta- model of domain ontology supporting semantic representation of components is presented. Based on the meta-model port semantics, static semantics and semantic protocol of components and the semantics composition algorithm are given. Then semantics of components can be composed along with component composition according to the static semantics and the algorithm. It can help developers to perform the semantics-based adaptation and composition analysis of components.

Key words: component; business semantics; domain ontology; feature; semantic representation; composition

1 引言

软件复用面临的一个关键问题是怎样判断构件的有用性,并了解如何去使用被复用构件[1]. 构件的有用性体现在语义和语法两个方面[2],当前的研究工作以及一些流行的构件标准(例如 CORBA、COM、EJB 等)更多地集中在语法有用性上. 构件从本质上说是一种可以独立地理解和复用的自包含实体[2],而语义也是决定构件可用性的关键因素, 尤其是实现某种业务逻辑的特定领域构件. 由此造成的结果是复用更多的发生在一些完成底层功能的通用构件上,例如打印构件、算法构件等,领域构件很难进行复用, 特别是跨组织的复用.

构件库中的构件描述是构件检索技术的基础[3],同时也为构件复用者理解、使用构件提供了手段. 文献[2]提出了一种基于领域特征空间的构件语义表示方法,从领域空间、定义空间和语境空间三个侧面刻画构件语义. 这种描述方法指明了业务领域、构件定义以及语境三者的关系,但还比较宽泛,可操作性不强, 且不支持语义组装. 文献[4]提出在构架描述语言中引入构件行为协议来描述构件的行为, 并支持构件行为协议的组装推导. 这种行为协议从通信行为上对构件进行刻画, 但行为协议的描述基础是不含业务信息的交互事件, 语义性较弱. 文献[3]所提出的构件描述框架使用编目方法(枚举、刻面等)描述问题空间, 在此基础上通过对构件的初始状态以及操作执行所产生的状态变化来刻画接口的语义. 这种方法只适合描述领域相关性不强的构件(例如抽象数据结构),且不支持构件语义描述的组装推导. 由此可见, 基于领域知识(体现为领域本体或领域特征空间等)的构件语义描述已经得到广泛认可, 然而现有的方法仍然存在不足, 主要体现在: 构件服务语义的描述不够精确,无法体现业务操作的各种

收稿日期: 2006-08-15; 修回日期: 2006-12-12
基金项目: 国家高技术研究发展计划(863 计划)(No. 2005AA113120); 国家自然科学基金(No. 60473061, No. 60473062)

细节特征;没有明确构件的请求语义及语义组装方法,因此无法支持构件语义的组装推导.

构件通常可以看作提供(还可能向外界请求)一组服务的黑盒实体[4].从语义的角度看,构件是在领域概念空间内,以一定的环境支持为基础完成特定功能的可复用实体.从复用者的角度看,构件的可用性主要体现在对外提供的服务语义上,复用者可以利用构件服务组装得到更大粒度的构件直至整个软件系统.同时,构件的服务语义也是建立在组装环境的基础上,包括构件请求语义被其它构件满足的情况以及构件服务被组装的情况.因此,构件可以认为是一个完成特定领域语义的实现体,同时包含环境需求以及一部分基于环境不确定性而产生的语义可变性.构件在静态组装或动态组装(运行时,例如体系结构的动态调整)过程中,随着环境不断地具体化,其自身语义的可变部分也逐渐确定下来.

可复用构件的功能描述有两种方式[5],即基于使用方式或基于计算.前者描述复用者可能的使用方式,即在业务环境中表达功能描述,好处在于使用与复用者相同的表达方式,因此更容易被复用者理解[5].后者描述构件的固有属性,即从计算空间(解空间)的角度进行描述,复用者需要首先完成问题空间到解空间的映射[5],无法适用于实现复杂业务逻辑的领域构件.由此可见,基于业务需求和业务上下文的构件描述更容易被复用者理解,而共同的业务语义基础是前提条件.因此,本文引入我们在前期的领域工程研究中[6,7]所提出的基于本体(使用 OWL 描述语言)的领域特征模型(简称特征本体)作为领域内构件描述的业务知识基础,在此基础上定义构件的概念语义模型以及语义组装过程,从而支持构件语义的自描述和组装.

2 领域特征本体

对于特定领域构件来说,一个统一规范的领域模型是准确刻画其业务语义的基础.在基于特征的领域分析过程[7,8]中,特征被定义为领域需求空间的一阶实体.领域特征模型则描述了领域内的共性业务知识,同时也刻画了所允许的可变性以及相关约束(体现为特征依赖关系).领域内的构件共享领域特征模型中的知识,还可以在相关可变点上做出相应的实现选择,但必须满足相应的约束.在基于本体的领域模型(图 1)中,特征被表示为概念(Concept),并进一步细分为业务动作(Action)、动作刻面(Facet)和术语(Term)等建模元素.其中 Action 是业务动作的语义主体,直接表示功能特征;Facet 是 Action 的语义刻画面,用于精确描述业务动作的细节属性;Term 是 Action 上语义刻面的术语取值,即属性的值.为了表达数值型和布尔型特征,数值类型(xsd: number)和布尔类型(xsd: boolean)也可作为动作刻面的值域存在.除了操作特征,特征模型中还包括作为操作目标的业务对象(BusinessObject),它与 Action 之间是操作(ActOn)关系.这几类特征是特征模型的主要组成部分.

定义 **1**(语义刻面集) 对于 $\forall a \in \text{Action}$, a 的语义刻面集可定义为 $\text{FacetSet}(a) = \{f \mid f \in \text{Facet} \wedge (a \text{ rdfs: SubClassOf } f.\text{domain})\}$,其中 $f.\text{domain}$ 表示动作刻面 f 的定义域(rdfs: domain).注意,rdfs: subClassOf 在 OWL 中表示自反、传递的概念泛化关系.

定义 **2**(语义确定化) $\forall a1, a2 \in \text{Action}$ 如果满足($a2$ rdfs: subClassOf $a1$)则对于 $\forall f \in \text{FacetSet}(a1)$,都有 $f \in \text{FacetSet}(a2)$,且对于非数值型刻面 f 都有($a2.f$ rdfs: subClassOf $a1.f$).其中 $a1.f$ 表示业务动作 $a1$ 在语义刻面 f 下的术语取值.$a2$ 与 $a1$ 的这种关系称为语义确定化,即一种对可变语义特征的绑定过程.

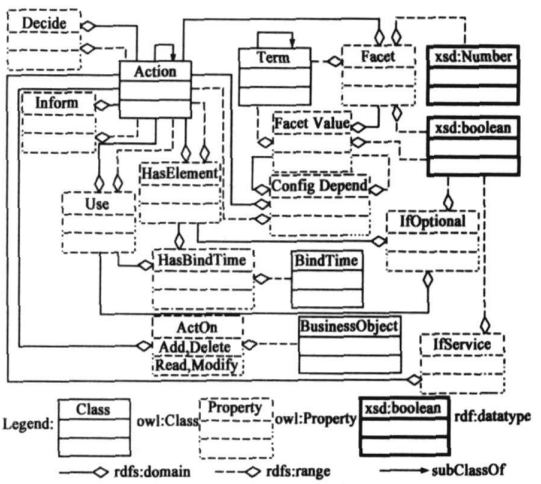

图 1 基于本体的特征模型元模型

由此可见,领域本体中业务动作之间依照动作刻面上的确定化关系形成业务动作的语义层次.高层业务动作的上定义的刻面对其所有子动作均有效,而子业务动作在对父业务动作进行语义确定化的同时还会定义新的语义刻面,因为随着语义的精化可能产生新的语义特征.除了这些基本元素,基于本体的特征元模型还包含特征依赖、绑定时间等其它模型元素,具体请参考文献[6,7].

图 2 是基于图 1 中的元模型的考试阅卷领域本体局部示例.从中可以看出,阅卷(Grade)工作分为试卷准备(PaperPrepare)、分配(PaperAssign)、评分(PaperGrade)和复审(Review)四个业务动作,其中复审是可选的.口语阅卷(OralGrade)和笔试阅卷(WrittenGrade)是两种阅卷业务,它们同样可以分为试卷准备、分配、评分和复审这四个业务动作,其中试卷分配和复审动作是相同的,而在试卷准备和评分上不同.两种阅卷以及试卷准备、评分子动作的区别是通过定义答卷格式(PaperFormat)属性实现的,它们分别支持音频(Audio)和图片(Image)两种格式.除了图 1 元模型中定义的 Facet、HasElement 等关系,领域本体中还可以定义一般的本体关系,例如图 2 中两个术语 AutoAssign 与 FetchMode 之间的 HasAlgorithm 关系,这些关系可能在构件的语义协议描述中使用.

3 基于领域特征本体的构件语义描述

构件的语义信息包括服务端口语义、请求端口语义和语义协议三部分.服务语义体现了构件的可复用性,不仅决定于构件本身的实现同时还取决于组装关系以及运行时的调用参数.请求语义体现了构件对组装环境的约束和期望,同时也包含了一部分可变性.语义协议则体现为服务语义对请求语义

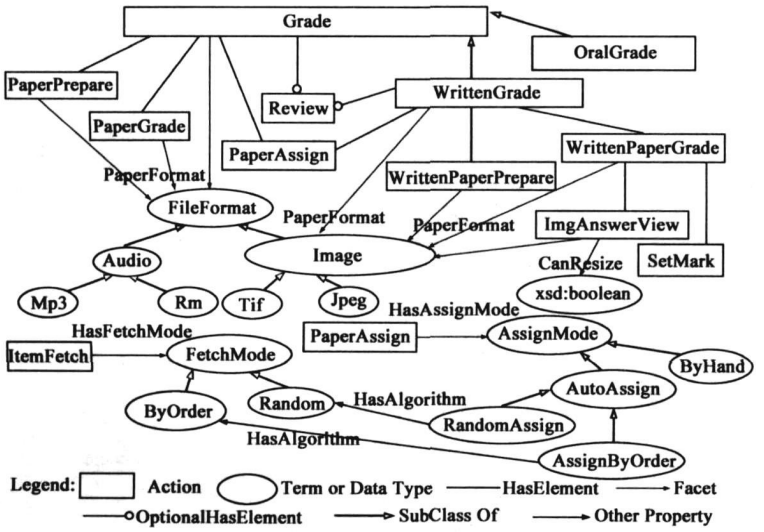

图 2　考试阅卷领域特征本体局部示例

的依赖关系.

定义 **3**(构件端口语义)　构件任何端口 P 的端口语义 PortSemantics(P)是一个二元组< a, FacetValue> ,其中 $a\in$ Action, FacetValue 是 a 的语义刻面集 FacetSet(a)到 Term 的取值函数,且 $\forall f\in$ FacetSet(a)都有(FacetValue(f) rdfs: subClassOf $a.f$). 其中 $a.f$ 表示特征本体中 a 在刻面 f 上的术语取值.

由此可见 PortSemantics(P)也是对 a 的语义确定化,且语义刻面集与 a 相同. 构件的服务端口和请求端口语义都满足端口语义的一般定义. 从构件实现的角度看,构件的服务语义并不总是完全由构件实现体提供,许多情况下构件需要将一部分语义功能委托给其它构件,体现为构件的请求端口. 服务端口和请求端口之间的语义关系决定了组装时以及运行时服务语义与请求语义之间的确定化关系.

定义 **4**(服务端口语义协议)　构件任一服务端口 SP 上的语义协议 SProtocol(SP)是 SP 上各语义刻面取值运算式的集合,即 SProtocol(SP) = { FProtocol(sf) | sf∈ FacetSet(SP) }. 其中 FProtocol(sf) 表示 SP 上的某一语义刻面 sf 的取值运算式,用 BNF 范式可以定义为(其中" < > " 表示非终结符):

FProtocol(sf) ::= < TermItem> ∩ SP. sf

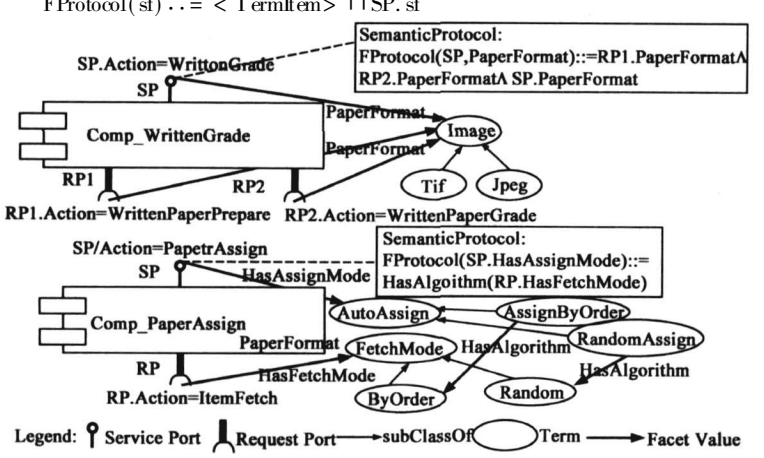

图 3　构件静态语义描述示例

< TermItem> ::= < TermItem> ∩< TermItem> | < RPTermItem>

< RPTermItem> ::= RP. rf| OntoProp(RP. rf)

其中, ∩ 表示即本体概念的交集, SP. sf 表示 SP 的端口语义中 sf 语义刻面的取值, RP. rf 表示该构件的某一请求端口 RP 的语义刻面 rf 的取值术语. 出现在 SP 的语义协议中的请求端口构成了 SP 对请求端口的依赖集. OntoProp 是本体中定义的关系, OntoProp(term) 表示领域本体中与术语 term 有 OntoProp 关系的术语. 由此可见,组装是对构件服务语义的进一步确定化.

定义 **5**(构件静态语义)　构件的静态语义是一个三元组< SPortSemantics, RPortSemantics, SProtocol > ,其中 SPortSemantics 是构件服务端口语义集, RPortSemantics 是构件请求端口语义集, SProtocol 是构件各服务端口的语义协议集.

图 3 描述了两个构件的静态语义,分别是 Comp_WrittenGrade(笔试阅卷构件) 和 Comp_ PaperAssign(答卷分配构件). 其中,笔试阅卷构件提供笔试阅卷服务并请求笔试答卷准备和笔试评分服务. 从语义协议看,该构件所支持的答卷格式是 Image 与所请求的两个服务所支持的格式的交集.

4　构件语义组装推导

构件开发完成后,静态语义也就确定下来了. 此后,构件语义主要在两个阶段进行确定化:组装时以及运行时. 组装时的确定化主要由组装环境决定,即服务端口及请求端口上的组装关系. 而运行时的确定化主要由构件服务的调用参数决定,这些参数将最终决定本次构件服务调用的交互语义. 本文的构件语义描述主要是为了辅助构件复用以及组装过程,因此我们主要考虑组装时的语义推导.

4.1　构件语义组装过程及算法

组装时的语义确定化是一个以满足系统直接需求为目标的链式过程. 但从组装分析的角度看只需要提供单个构件的语义组装机制即可完成整个过程. 从单个构件组装过程看,语义确定化分为端口语义适配和内部语义适配两个步骤. 前者完成构件的请求/服务端口语义与其它构件的服务/请求端口语义的匹配,后者依据相关端口的语义协议完成请求/服务端口对同一构件的服务/请求端口的语义确定化. 端口匹配过程在两个构件之间进行,可以看作求请求构件的请求语义与服务构件的服务语义的交集的过程. 由于端口语义基于领域本体进行描述,因此匹配过程也是基于领域本体进行的. 因此可以定义端口语义适配算法如下,其中 intersection ($C1$, $C1$)表示两个概念 $C1$ 和 $C2$ 在本体中的概念交集.

算法 **1**(端口语义适配算法)

输入:构件服务端口语义 sp 和另一个构件的请求端口语义 rp

输出:适配后的端口语义 p

(1) 在领域本体中求 action= intersection(sp. Action, rp. Action) ,如果 action 为空概念则算法结束; (2) 令 p. Action= action, 对 $\forall f \in$ FacetSet(action), 令 p. FacetValue(f) = action.f;

(3) 对每一个 $f \in$ FacetSet(sp), 求 term= intersection (sp. FacetValue(f) , p. FacetValue(f)) ,如果 term 为空概念则算法结束, 否则令 p. FacetValue(f) = term;

(4) 重复步骤 3 直至遍历 FacetSet(sp) 中的每一个语义刻面;

(5) 对于每一个 $f \in$ FacetSet(rp) , 求 term= intersection(rp. FacetValue(f) , p. FacetValue(f)) , 如果 term 为空概念则算法结束, 否则令 p. FacetValue(f) = term;

(6) 重复步骤(5) 直至遍历 FacetSet(rp) 中的每一个语义刻面.

如果算法在完成适配过程前停止, 则端口语义适配不成功, 表明当前的组装关系不能成立. 端口语义适配完成后, 进一步确定化的请求/ 服务端口语义将按照构件语义协议对同一构件的服务/ 请求端口进行语义确定化, 可以分别称为自下而上和自上而下的内部语义适配. 下面以自下而上的内部语义适配算法为例说明内部语义适配过程.

算法 **2**(自下而上的内部语义适配算法)

输入: 构件服务端口语义 sp 以及请求端口上经过端口确定化之后的语义 rp

输出: 内部适配后的服务端口语义 sp

(1) 对于每一个 sf∈ FacetSet(sp) , 如果 FProtocol(sf) ∈SProtocol(sp) 则根据相关请求端口上已确定的语义计算 FProtocol (sf) 的取值 term;

(2) 如果 term 为空概念则适配失败, 算法结束; 否则令 sp. FacetValue(sf) = term;

(3) 重复步骤(1) ~ (2) 直至遍历 FacetSet(sp) 中的每一个语义刻面.

如果该算法在完成适配过程前停止, 则构件内部语义适配不成功, 表明该构件的静态语义描述(服务端口语义、请求端口语义和语义协议) 存在不一致. 自上而下的内部语义适配算法与算法 2 类似, 不再赘述.

4.2 构件语义组装实例

图4 描述了一个构件语义组装实例. 其中构件 Comp_WrittenPrepare(笔试答卷准备构件) 和 Comp_ WrittenPaperGrade (笔试评分构件) 的服务端口分别被组装到 Comp_ WrittenGrade 构件(笔试阅卷构件) 的两个请求端口上. Comp_ WrittenPaperGrade 构件还需要对外请求答卷查看服务(ImgAnswerView) , 并由一个支持 Tif 答卷格式的答卷查看构件 Comp_ TifAnswerView 满足. 因此 SP2 端口上评分服务支持的答卷格式确定化为 Tif, 与 RP2 端口经过端口语义适配后, RP2 端口的服务语义中答卷格式确定化为 Tif. 由于 Comp_ WrittenPrepare 构件的 SP1 端口支持图形阅卷(同时支持 Tif 和 Jpeg 格式) , 因此 RP1 端口经端口语义适配后语义未发生变化. 此时根据 Comp_ WrittenGrade 构件的语义协议可以进行 SP 端口的语义确定化, 即取两个请求端口上支持答卷格式的交集后将 SP 端口语义中的答卷格式确定化为 Tif 格式.

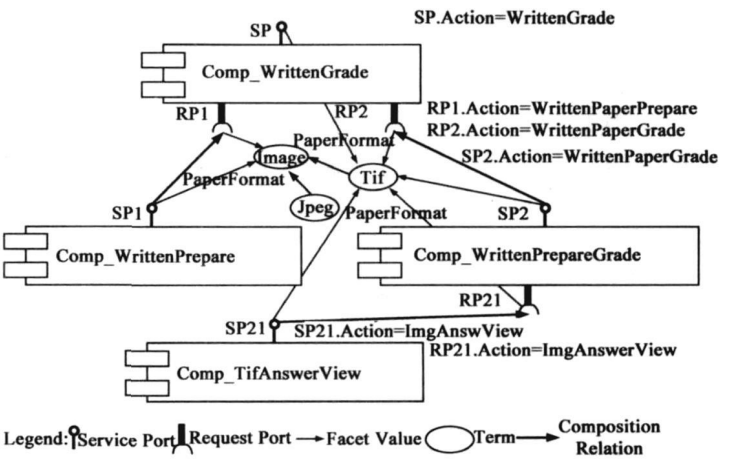

图 4 构件语义组装实例

5 总结和展望

构件语义描述对于构件检索、理解、组装等都具有重要的意义, 是构件开发者和复用者之间交流构件功能的重要信息来源. 领域特征本体作为领域分析阶段的产物, 不仅是面向领域构件和应用开发的需求模型, 还可以为领域内的构件开发者和复用者奠定统一的构件语义基础. 本文引入领域特征本体作为构件语义描述的知识基础, 从而使特定领域业务构件的开发、发布、检索、理解和组装都能在一致的语义基础上进行. 此外, 基于本体的领域特征模型和构件语义描述可以很容易地形式化(例如基于 OWL) 而且支持自动推理, 这也为构件语义匹配和组装过程的自动化创造了条件. 在此基础上, 我们可以进行构件的动态选取和组装, 或者以功能语义分解指导业务构件的设计, 例如我们在特征驱动的体系结构设计中的工作[6]. 下一步我们将进一步研究基于特征语义的构件行为协议描述以及运行时的构件特征交互分析方法, 并在动态软件体系结构以及领域模型驱动的体系结构设计等研究中应用并进一步发展相关成果.

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): 68- 75.
Yang Fuqing, Mei Hong, Li Keqin. Software reuse and software component technology[J]. Acta Electronica Sinica, 1999, 27(2): 68- 75.

[2] 贾育, 顾毓清. 基于领域特征空间的构件语义表示方法[J]. 软件学报, 2002, 13(2): 311- 316.
Jia Yu, Gu Yu-qing. Domain feature space based semantic representation of component[J]. Journal of Software, 2002, 13(2): 311- 316.

[3] 张涌, 王渊峰, 钱乐秋. 一个集成式的软件构件描述框架[J]. 计算机学报, 2002, 25(5): 502- 507.
Zhang Yong, Wang Yuan-feng, Qian Le-qiu. An integrated soft-

ware component description framework[J]. Chinese J Computers, 2002, 25(5): 502- 507.

[4] Frantisek Plasil, Stanislav Visnovsky. Behavior protocols for software components[J]. IEEE Transactions on Software Engineering, 2002, 28(11): 1056- 1076.

[5] Hafedh Mili, Ali Mili, Sherif Yacoub, Edward Addy. Reuse-Based Software Engineering: Techniques, Organization, and Controls[M]. Beijing: Publishing House of Electronics Industry, 2003. 102- 102.

[6] 彭鑫, 赵文耘, 刘奕明. 基于特征模型和构件语义的概念体系结构设计[J]. 软件学报, 2006, 17(6): 1307- 1317.
Peng Xin, Zhao Wen-yun, Liu Yi-ming. Feature model and component semantics based conceptual architecture design[J]. Journal of Software, 2006, 17(6): 1307- 1317.

[7] Xin Peng, Wenyun Zhao, Yunjiao Xue, Yijian Wu. Ontology-based feature modeling and application-oriented tailoring[A]. The 9th International Conference on Software Reuse[C]. Berlin Heidelberg: Springer-Verlag, 2006. 87- 100.

[8] 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.
Zhang Wei, Mei Hong. A feature-oriented domain model and its modeling process[J]. Journal of Software, 2003, 14(8): 1345- 1356.

作者简介:

彭　鑫　男, 1979 年 10 月出生于湖北省黄冈市, 博士. 现为复旦大学计算机科学与工程系讲师. 主要研究方向为领域工程、软件产品线以及软件维护和再工程.
E-mail: pengxin@fudan.edu.cn

赵文耘　男, 1964 年 9 月出生于江苏省常熟市. 现为复旦大学计算机科学与工程系教授, 博士生导师. 主要研究方向为软件复用和构件技术. E-mail: wyzhao@fudan.edu.cn