计算机科学 1997 Vol. 24 №. 1

软部件描述:模型、方法和语言

李留英 毛新军 齐治昌
(国防科学技术大学计算机系 长沙 410073)

摘 要 The paper brings together the current research on software component description from three aspects such as model, method and language. The prospect of further research and development is given.

关键词 Component description model, Component description method, Component description language.

软件重用技术是软件工程领域的一项关键技术。目前,支持软件重用的方法主要有两种:块式重用,即使用可重用的程序块;模式重用,即通过程序转换获得可重用性。其中,最为系统化、最为工程化的软件重用活动是基于重用库的块式重用。块式重用通过构造可重用库实现源代码形式的软件重用,其前提是确认和描述可重用成分即软部件。通过软部件的组织和管理,辅助软件开发人员用这些软部件组装构造应用软件系统。

软部件的有效组织和管理是软件重用的关键,软部件描述是软部件组织和管理的基础和核心。一方面,为了开发和使用可重用软部件,软部件必须是易于理解的,需要对软部件的属性及部件间的关系加以描述;另一方面,软部件描述不仅能帮助用户理解、书写、修改和重用软部件,而且能促进软件重用工具对软部件进行有效的组织和管理。

软部件的描述涉及三个方面的内容:模型、方法和语言。模型是方法和语言的基础,给出软部件描述的抽象对象,即 What to describe。方法是描述的依据,亦是语言的设计基础,指出了如何描述软部件模型,即 How to describe。语言是软部件描述的工具,即 With what to describe,可以对模型加以描述和刻画。

本文从模型、方法和语言三个方面概括了软部件描述的当前研究状况和已有成果,分析了它的进一步研究和发展方向。

1 软部件模型

软部件描述模型指出了软部件的抽象对象,体现了软部件描述语言模型化的不同方法和目标。目前描述单个软部件的模型有:3C 模型和 REBOOT 模型。3C 模型是基于软部件属性的一种指示性模型,REBOOT 模型是基于已有软部件的一种分类的检索模型。

1.1 3C 模型

3C 模型由概念、内容、环境三部分组成。其中,内容给出了软部件的内部属性,环境给出了软部件的外部属性。

· 概念(Concept) 对软部件功能的描述。软部件的概念描述由软部件的接口规范描述及软部件提供的操作语义描述(形式化或非形式化)两部分组成。

· 内容(Content) 对应于软部件的实现,亦称为软部件的体。

· 环境(Context) 对软部件属性的附加描述。通过显式定义和描述软部件环境,用户可以更好地选择和修改软部件以便于重用。它包括三个方面的内容:①概念环境。一个软部件的接口和语义与其它软部件的接口和语义间的关系。②操作环境。操作数据的特性。如:操作数据的类型,可用的操作。③实现环境。一个软部件实现时对其它软部件的依赖关系。如:软部件间的调用关系。

1.2 REBOOT 模型

软部件的 REBOOT 模型是一种面分类和检索的模型。面(facet)是从某个角度对软部件属性的刻画。如软部件的操作、操作的对象等等。Prieto-Diaz 为函数型或过程型的软部件提出了一个三元组的 REBOOT 模型(Function, Object, Medium/Agent),分别表示部件提供的操作,操作对象以及该部件与其它部件间的关系。对于 OO 软部件,给出一个四元

组的面分类方案：〈Abstraction，Operations，OperatesOn，Dependencies〉，分别表示软部件对象，软部件提供的操作，操作对象及软部件间的关系。

1.3 模型比较和扩充

3C 模型和 REBOOT 模型在许多方面是等同的。3C 模型的概念对应于 REBOOT 模型中的面，均指出了软部件的功能。3C 模型的环境描述了软部件的外部属性(即一个软部件与其它软部件间的关系)和数据特性，这与 REBOOT 模型中的 OperatesOn 面和 Dependencies 面是一致的。但两个模型也有不同点。首先，建立模型的目的不同。3C 模型建立了软部件描述语言应该刻画的类属部件属性，而 REBOOT 模型只适合于描述已有的软部件，并且必须是在给定的面框架结构中描述。从描述方法和语言的角度看，3C 模型更适合于形式化的描述方法和语言，而 REBOOT 模型则适合于非形式化的方法。其次，REBOOT 模型没有给出相应的面以对应于 3C 模型中的环境。

3C 模型和 REBOOT 模型都是抽象的软部件描述模型，没有给出特定应用领域中的软部件的特定属性，因而适合于所有应用领域，是两个通用模型。如果要描述和刻画某一应用领域(如实时系统)软部件的抽象特征，需要对上述模型进行进一步的扩充，如增加抽象描述特性(并发属性等)。3C 模型和 REBOOT 模型为软部件描述模型的建立提供了一个抽象模板。通过对模板的扩充和具体化，可以使软部件描述模型适合于不同应用领域的软部件描述需要。

对大型软部件 LSC，如子程序，甚至整个系统及部件创建信息的重用，可采用层次结构模型，每个结点包含三个相互关联的元素：设计实例、设计框架、领域资源。领域资源包括一个领域模型和一组相关的领域实体，其中，领域模型由一组属性构成，它们定义出现在设计框架内的对象的需求或约束；领域实例描述属性值。设计框架描述如何将底层实体组合为设计实例；由一组对象以及对象间的关系构成。设计实例是设计框架中对象的具体化，由规范说明和体系结构组成，规范说明描述设计语义以及设计接口的语法和语义，体系结构描述设计的组织结构。在这种层次模型中，底层结点是高层设计实例内对象的详细设计。它支持分析重用、设计重用、实现重用及维护重用。

2 软部件描述方法

软部件描述是指运用某种表示法描绘软部件的抽象特性。软部件描述方法影响可重用部件的生成、检索和修改。在选择描述方法时，要考虑以下几个问题：对象或系统的结构、需要描述的操作类型、描述的一致性、可理解性、可表达性、部件的集成性，以及描述方法对描述语言的影响。经长期研究，人们已提出了许多可行的部件描述方法，如非形式化方法和形式化方法。与描述语言紧密相关的是形式化方法。

2.1 非形式化方法

非形式化方法包括：传统库和信息科学方法；基于知识的方法；超文本方法等。

常用的传统库和信息科学方法有：分类表示法、关键字表示法。分类表示法常用有限个术语描述部件，部件库按术语分类，形成层次式或网络式结构。层次结构简单灵活，易于理解，但不利于修改和扩充；网络结构易于修改、扩充，但是随着部件库的扩大，部件的检索和维护会越来越困难。关键字表示法使用各种术语描述软部件的特性，如用动词描述软部件提供的操作，用名词描述操作对象，用形容词修饰对象特征。它由语法和语义两部分组成。语法部分描述不同词性术语的组合规则，语义体现在各术语的意义上。关键字表示法直观、简捷，表达能力强，易于用户理解和描述，便于工具的组织和检索，因而能有效地提高用户的重用意愿(intention)。但该方法对软部件的描述不够精确、易有二义性，不能自动解决一词多义、多词同义，为此用词汇表列出术语的同义词和术语间的概念关系，便于术语语义匹配和部件的检索。

人工智能领域已开发出许多知识表示法来描述软部件，常用的有：语义网、规则和框架。衡量知识表示法好坏的两个因素是表示充分性和试探能力。表示充分性指某个表示法描述了多少信息。试探能力描述了表示法的推理能力。基于知识的表示方法能有效地表达部件间的关系，而且能帮助用户理解软部件间的关系。

语义网是一个有向图，结点表示对象，弧表示对象间的关系，它有较好的表示能力，但基本语义网的试探能力差，可增加一阶谓词逻辑来增强其推理能力，如 STARS 程序用结点表示领域概念/软部件，用 IS_A 弧表示类间成员关系，用功能弧表示单个概念的属性和概念间的非类关系。规则是已知的最好知识表示机制，通常用规则对软部件进行分类。Bollinger 和 Barnes 开发的基于 Prolog 的系统中，一条 Prolog 规则对应一个部件或一组相容的部件。框架是一种数据结构，包括槽和填充值。通常将部件

的框架描述存放在知识库中。一个框架对应一种部件，槽表示部件类型，填充值表示具体的部件。

超文本是一种数据结构，节点表示软部件，连线表示节点间的关系。它允许用户在文本内通过连线来回移动查找所需部件。

2.2 形式化方法

软部件重用的形式化方法用形式化的规范说明语言描述软部件。规范语言指明软部件的内容而不考虑实现细节。形式化方法避免了语义的二义性、模糊性；软件系统的形式化规范为系统原型的快速开发奠定基础。下面给出几个典型的软部件形式化描述方法。

(1)代数规范方法。代数规范 SP=(S,OP,A)描述重用软部件的核心思想是用类型 S(sort)和函数集 OP 描述软部件功能。S 的元素作为 OP 中函数的输入/输出集，公理集 A 描述了函数的功能。代数规范的语义允许一些类型可见，一些类型不可见；对代数规范可进行三个方面的扩充：增加规范的操作，参数化代数规范，带接口的规范。所以可以从不同的抽象层次和不同的角度来描述软部件。

Hennicker 和 Nickl 提出的纯代数方法中，行为规范是指基本规范或对基本规范施加规范操作(丰富/enrichment、改名/renaming、输出/exporting、联结/combination)而形成的结构化规范。基本规范 SP 包括 S、可辨别集 OBS⊆S、构造子操作集 C、运算符号集 F、公理集 A。其中(S,C∪F)是它的特征(signature)sig(SP)，A 是一阶谓词逻辑集。Beh(SP)表示规范模型 SP 的类。对基本规范 SP 实施一定的操作形成另一规范 SP'或规范对(IMP',SP')。带接口的行为规范指一个行为规范对(IMP,SP)，sig(IMP)⊆sig(SP)。若(IMP,IP)和(IMP',IP')满足如下条件：

· $sig(SP')=sig(SP)$，$Obs\text{-}sorts(SP)\subseteq Obs\text{-}sorts(SP')$，$Beh(SP)\subseteq Beh(SP')$

· $sig(IMP)=sig(IMP')$，$Beh(IMP)\subseteq Beh(IMP')$

称(IMP',SP')为(IMP,SP)的一个实现，记为(IMP,SP)～～>(IMP',SP')。我们用一棵(无序)树描述软部件，每个结点表示带输入接口的行为规范(IMP,SP)。这样每个子结点(IMPson,SPson)就是父结点(IMPfather,SPfather)的一个实现。重用软部件的根代表最抽象的规范，叶结点代表不同实现的最具体的规范。描述部件之后，可用正文归纳证明器证明相邻结点实现关系的正确性。如果完全正确，说明可重用部件是正确的。

(2)多形式体系方法。其核心思想是从多个角度来描述软部件。在面向对象的结构化软件中，为了能完整地描述对象的数据抽象、并发、异常特性，必须提供一个形式化的机制，从多个角度描述对象的特性。π 语言提供了这种能力。它用 CEM-规范描述对象的特性。一个 CEM-规范包括三个视图(view)：

· 类型视图(Type view) 描述了对象的静态特性，包括输出、公共参数和体三部分。输出部分定义了对象的基本功能；公共参数部分定义了与基本对象有关的其它对象的功能，即定义了对象的概念环境；体部分定义了基本对象的一种可能实现，这种实现要涉及其它有关的对象，这些对象由输入部分描述。输入部分定义了实现环境。

· 命令视图(Imperative view) 把对象看成某个特定类型值的抽象存储，把操作看成修改、检索相关对象类型的过程。

· 并发视图(Concurrency view) 描述与对象类型有关的操作的执行次序。

由于规范的公共参数部分和输入部分不涉及其它软部件的规范，即与实现环境无关，那么对象的具体功能会随环境的不同而异，所以采用配置规范(Configuration specification)描述对象的功能与环境中其它对象功能间的映射关系。在配置规范内，即使对同一部件的描述，不同的映射关系也会产生不同的软部件。

(3)多层次规范方法 (multilevel specification)方法。由 Gabrienlian 于 1991 年首次提出，适用于大型软部件 LSC 的重用。多层次规范方法允许每个高层规范对底层规范施加一些约束。在垂直方向，设计实例的规范必须继承父对象的规范；在水平方向，设计实例的规范可以继承设计框架的规范，设计框架的规范必须继承领域模型的规范。这种方法可以提供不同抽象层次的规范，根据继承原理，最底层的规范就是所需的规范。

3 软部件描述语言

本节介绍几种有影响的部件描述语言。

3.1 库互连语言 LIL(Library Interconnection Language)

LIL 是 Goguen 为开发大型软件系统设计的部件描述语言，支持代码重用和经验重用，允许垂直软件合成和水平软件合成。垂直合成是指不同抽象层次间软部件的组合，水平合成是指同一抽象层次部件间的修改和组合。

3.2 部件语言CDL

CDL语言是Alvey Eclipse项目小组开发的基于对象结构范型(object-structured paradigm)的语言。部件包括两部分:部件接口和部件体。部件接口描述部件的输出以及与其它部件的关系;部件体用类-Ada符号描述,实现了接口部分所体现的思想。CDL语言是设计级语言,可通过本身工具箱将CDL转换为实现语言。

3.3 CIDER语言

CIDER语言是面向对象的部件描述语言,具有面向对象语言的特点,允许继承、输入、实例化。部件由部件接口和部件体组成,可用显式或隐式方式定义部件接口。部件接口包括数据类型接口和对数据类型进行操作的接口。部件体描述部件的功能,即部件的概念。部件也可以是资源、子程序、过程、函数的集合。其中,资源包括枚举类型、常量、数组和子类型的说明。子程序的概念包括形式参数、返回类型和异常处理三部分。函数和子程序级的环境分为几个部分,分别描述部件的继承性、输入参数和输出参数。一个部件可以继承另一个部件。

CIDER具有如下特征:①多重继承。用两个运算符product和uion描述部件间的耦合。uion指明部件资源的操作间的关系,product表明对某个资源操作后对部件内其它资源的影响。②置换。用置换运算符where编辑和置换部件资源。主要采用两种方式:类属实例化或创建新部件。

3.4 Resolve语言

Resolve语言用基于方法的数学模型表达部件的形式化规范说明。它给出部件的类型、部件操作所满足的前置条件和后置条件、部件的实现、部件模型和部件表示法的对应关系。在Resolve中,用数学模型描述部件的接口需求、接口需求与一个可能的实现间的关系,更适合于重用部件的设计。Resolve语言无法用抽象的数学模型描述两个部件概念间的继承关系,故从一个部件产生另一个部件具有很多的弱点。

用户选择部件描述语言时必须考虑如下因素:部件概念描述;易于部件的理解;接口和实现的区分,接口的稳定性;设计的灵活性,允许设计者能清楚地表达设计思想;结构直观,语法简洁。

上述四种语言均能描述部件接口,但支持形式化的程度不同。LIL用view描述部件概念,Resolve用代数规范描述部件概念,CDL相对弱一些。上述四种语言均能实现3C模型。CIDER和Resolve用类属机制、输入方式描述3C模型的操作环境和实现环境,Resolve还可以描述部件与数学模型间的关系,这些均利于部件的理解。

4 进一步发展和研究

3C模型和REBOOT模型主要是针对源代码软部件重用而开发的,不适合于其它可重用对象(如需求分析阶段的信息、设计阶段的信息等)的描述。有效地重用软件开发过程中其它开发阶段的信息和知识,尤其是软件体系结构和大型软部件将是今后提高重用效率,发挥重用潜力的关键。开发支持这些阶段的可重用信息和知识的描述模型将是今后研究的一个重点。

软部件描述方法不仅应刻画软部件的语法和语义特性,还应提供机制和设施描述软部件间的组合特性。通过灵活的软部件介绍和组合设施,用户可以方便地组合基本的软部件形成新的软部件,为软件重用提供更强的功能。

软部件描述语言应与部件规范语言一致,并尽可能与部件实现语言相似,部件描述尽量采用二元偶(Specification,Component),这样,检索软部件时不但能进行特征(signature)匹配还能进行语义或规范匹配,为系统原型的开发和软件自动化奠定基础,而且,有效的软件重用将彻底改变软件开发过程,代之以生产和消费可重用软部件的开发模型、方法、过程和技术。

参考文献

[1]Ben Whittle ben, Models and Languages for Component Description and Reuse, ACM SIGSOFT Software Engineering Notes, 20(2)

[2]W. B. Frakes and P. B. Gandel, Representing reusable software, Information and software technology, 32(10)

[3]Joachim Cramer 等, formal Methods

[4]H. Weber, Uniformity and invariame in support reuse, IEEE Trans. on SE, 18(7)1993