

横切关注点管理中的若干关键问题综述

何成万

(武汉工程大学计算机科学与工程学院 武汉 430205)

摘要 面向方面的软件开发方法区别于面向对象的软件开发方法的关键在于横切关注点的管理。横切关注点的管理应贯穿面向方面的软件开发的全过程。分析了横切关注点在软件开发各阶段的定义及表现形式,阐述了在需求阶段横切关注点的识别、行为约束和演化机制等横切关注点管理中的几个关键问题的研究现状,讨论了今后的研究方向,并提出了一些可能的解决方案。

关键词 面向方面的软件开发,横切关注点的识别,行为约束,演化,目标模型

中图分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.7.001

Survey of Management of Crosscutting Concerns

HE Cheng-wan

(School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430205, China)

Abstract The key difference between AOSD (Aspect-Oriented Software Development) and OOSD (Object-Oriented Software Development) is the management of crosscutting concerns. The management of crosscutting concerns should run through the whole process in AOSD. Firstly, the definition and manifestations of crosscutting concerns at different stages of software development were summarized. Then, the most recent research advances of several key issues such as identification of crosscutting concerns in the requirements phase, behavioral constraints, and evolutionary mechanisms in the management of crosscutting concerns were surveyed. Finally, the future research directions were outlined, and the possible solutions to those key issues were proposed.

Keywords Aspect-oriented software development, Identification of crosscutting concerns, Behavioral constraints, Evolution, Goal model

1 引言

关注点分离机制可提高软件的可扩充性和可理解性^[1,2]。面向方面的程序设计 (Aspect-Oriented Programming, AOP)^[3,4] 方法把软件系统的功能和非功能需求、平台特性等诸多不同的关注点相互独立,很好地实现了横切关注点 (crosscutting concerns) 的模块化。面向方面的软件开发 (Aspect-Oriented Software Development, AOSD) 方法认为横切关注点应该贯穿于软件开发的各个阶段。AOP 被认为是对面向对象技术的一种必要的补充。

面向方面的软件开发方法区别于面向对象的软件开发方法的关键在于横切关注点的管理,包括需求横切关注点的识别、建模以及核心级模块的正确组合等问题。

在面向方面的软件开发方法发展的初期, AOP 语言发展较快,实现层中的相关方法和技术(如 Aspect 的重用、切入点的定义、遗留代码的方面挖掘等)受到了大多数研究人员的关注。但是,实现层中的方面 (Aspect) 从何而来? 能否在软件开发的早期阶段(需求和设计阶段)发现并标识横切关注点? 需求阶段的横切关注点如何映射到设计和实现阶段的 Aspects? 显然,方面需要在更高的层次进行管理。很多学者对

这些问题给予了关注,提出了 Early Aspects^[5] 的概念,用于表示面向方面的需求工程和体系结构设计。国际上多个研究小组正在开展这方面的研究工作,如英国兰卡斯特大学 (Lancaster University)、香港中文大学等。很多国际权威期刊(如 IEEE Software^[6], IEEE Transactions on Software Engineering^[7], IEICE Transactions on Information and Systems^[19]) 和顶级国际学术会议(如软件工程国际会议 ICSE^[8,9], 面向方面的软件开发国际会议^[10], 需求工程国际会议^[11,12], Automated Software Engineering^[20]) 都多次报道了这方面的工作。现有的软件需求工程理论和方法,如基于多视点 (viewpoints)^[13]、用例 (use cases)^[14] 和目标 (goals)^[15] 的需求工程方法,都没有很明确地处理横切关注点的机制,难以有效地适应面向方面的软件这种新的形态。

在实现层,面向方面的程序在实际应用中面临两个关键问题:由于程序演化可能导致的结构冲突和 Aspect 织入过程中可能导致的行冲突。其中,结构冲突问题也称为“脆弱的切入点问题”(fragile pointcut problem)^[16,17,60,61]。“脆弱的切入点问题”首先由 Koppen C 等人在文献^[4]中提出, Kellens A 等人在文献^[17]中对这一问题作了进一步的阐述。脆弱的切入点问题主要表现为现在的联结点模型 (Join Point Model,

表1 横切关注点的定义和表现形式

开发阶段	横切关注点定义	表现形式
需求阶段	横切其它多个需求制品的需求	需求 (目标、用例等)
设计阶段	为其它多个模块提供服务的模块	模块
实现阶段	横切其它多个核心功能模块的实现的代码	代码(Aspect)

JPM)对基础程序的演化不够健壮^[18]。其原因在于切入点(pointcut)的定义紧紧地依赖于基础程序的结构,这种紧密的耦合严重阻碍了软件的演化。如果基础程序的结构发生了变化,就有可能产生连接点错配(join point mismatches)问题,它包括无意识的连接点匹配(unintended join point capture)和意外的连接点丢失(accidental join point miss)两个方面的问题^[17]。其中,无意识的连接点匹配是指,由于基础程序结构的变化而无意识地增加一个新的连接点;意外的连接点丢失是指原来能够和切入点定义匹配的连接点变得不能匹配。行为冲突问题也称为 Aspect 组合的语义冲突问题^[31,32]。这种冲突表现为:原本能够正确运行的程序在织入 Aspect 后不能正确运行;或者是多个 Aspect 织入到同一个连结点,织入的先后顺序可能导致 Aspect 间发生冲突;或者两个 Aspect 之间是一种互斥的关系,不能同时织入到同一个基础程序等。Aspect 组合的语义冲突有可能发生于 Aspect 和基础程序之间,也可能发生于 Aspect 和 Aspect 之间。随着软件规模的扩大以及 Aspect 数量的增加,采用手工的方法控制 Aspect 和基础程序的正确组合很容易出错,甚至是不可能的。

一般来讲,横切关注点的管理应该贯穿于软件开发的各个阶段。图1描述了需求、设计、实现阶段横切关注点和核心关注点之间的关系。我们认为上述的需求阶段的横切关注点识别、实现阶段横切关注点演化和行为约束是面向方面的软件开发中的3个基础和关键问题。在需求阶段,横切关注点的识别是横切关注点管理的起点,行为约束是面向方面程序正确执行的保证,而演化机制为面向方面程序的演化提供了手段。文中阐述这3个问题的研究现状以及对其的一些思考,提出了今后的研究方向及挑战性问题。

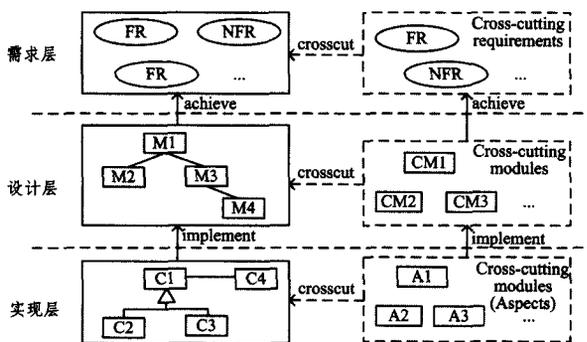


图1 横切关注点的管理层次

2 横切关注点的定义和表现形式

在软件需求分析、设计和实现阶段,横切关注点的定义和表现形式各不相同。

设计阶段的横切关注点指被多个核心模块所调用,为这些核心功能提供服务的模块^[3,55]。实现阶段的横切关注点指分散在其它多个核心模块的实现中,且和核心模块相互缠绕的代码^[3],分散(scattering)和缠绕(tangling)是其主要特征。按照文献^[6]的定义,需求阶段的横切关注点是指横跨其它需求制品的需求,即在需求分解层次中处于较低层,且是作为实现其它多个高层次需求的必要条件的需求^[12]。表1描述了软件开发各阶段的横切关注点的定义和表现形式。

3 研究现状

3.1 需求阶段横切关注点的识别

由于需求阶段的横切关注点有多种表现方式^[21,22],因此横切关注点的识别并非易事。而为了识别横切关注点并确定横切关注点的影响范围,显然要对需求文档中的关注点及关注点间的关系进行建模。

现阶段有很多学者对需求阶段识别横切关注点的方法进行研究,并取得了一些成果。基于候选关注点的检索方法^[23,24]的基本流程如下:依据领域经验将常见的横切关注点作为候选横切关注点,然后以它们或其特征在需求规范中进行匹配检索,如果规范中某个需求依赖或者包含被检索的候选横切关注点,则可确定此候选横切关注点为待开发系统的横切关注点。此类方法的局限性很明显,即若待开发系统的横切关注点并未确定为候选横切关注点,则依据此类方法并不能将此横切关注点识别出来。

Theme/Doc^[26]方法提供了对面向方面软件开发中分析和设计阶段的支持。在分析阶段,Theme/Doc 抽取需求文档中的动作短语和需求,并建立动作短语和需求的关联。如果一个动作短语和一个以上的需求关联,则把该动作短语所描述的行为作为一个候补横切关注点。但是对于一些非功能需求,如安全、性能等,往往没有动作短语和它们关联。在这种情况下,Theme 方法会要求重写需求文档,使之包含动作短语,这使得 Theme/Doc 方法缺少对非功能需求的有效处理。

文献^[25]提出了基于聚类的识别方法,其通过分析各需求间的相似度及二次聚类来识别横切关注点,但操作复杂而不便于应用。在基于 Petri 网的方法^[27]中,首先通过视图组织需求文档中关注点和需求间的关系,然后以各需求间的依赖或限制关系建立 Petri 网模型,最后通过观察 Petri 网模型的变迁结果来识别横切关注点。文献^[42]采用自然语言处理技术,在相关需求文档中通过动词短语匹配获取横切关注点。

Yu Yijun 等学者提出了一种 V 型目标模型^[12]。该模型将系统需求分别用功能目标、非功能目标以及对它们均起作用的任务组成。当完成目标分解后,若某任务(task)与软目标间存在链接并与功能性目标存在多余 1 条的链接,则该 task 横切了功能性目标。显然基于 V 型目标模型的方面识别方法利用 task 的扇入(fan-in)度对横切关注点进行识别。虽然该方法较好地利用了横切关注点扩散的特点,但其只能识别非功能性横切关注点,而没有实现功能性横切关注点的识别。

C. Zhang 等学者验证发现了横切关注点在 KAOS 目标模型中的部分特征^[28]。首先分析某中间件的源码,然后基于 KAOS 对该系统进行需求建模,最后通过比较横切关注点在源码和需求模型中的表现形式来发现或分解目标在映射为代码层时具有的横切行为。Bounour Noraz^[29]总结了各种用于遗产系统源代码的方面挖掘方法,由于软件设计与实现是由需求分析过渡而来的,因此可以提出一种体现横切行为的需

表3 Theme/doc与AspectQuery的比较

方法	输入	输出	横切关注点的组合	处理对象	工具支持
Theme/Doc	结构化需求文档	动作视图 (Action View)	不支持	功能性需求	半自动化
Aspect-Query	目标模型	横切关注点的XML描述	支持	功能性和非功能性需求	半自动化

求模型以支持需求阶段横切关注点的识别。文献[30]提出了GPRN面向方面需求建模框架,其指出系统需求由非功能目标、可操作目标组成。可操作目标是一种功能目标,且与一组精细化子目标一一对应,同时该目标元模型显示可操作目标也可能是一种横切关注点。

文献[8,10]提出了一种基于领域驱动的设计切入点(pointcut)接口的方法。该方法基于领域模型和用例(use case)模型发现领域抽象,并在体系结构设计阶段把这些领域抽象映射为切入点接口。该方法中的领域模型仅作为领域词汇,用于与在用例模型中发现的公共用例进行比较。但该方法仅仅是重用领域模型中的概念,缺乏大粒度重用领域模型(把领域模型作为初始需求模型)的机制。

EA-Miner采用自然语言处理技术从需求文档中挖掘不同类型的关注点^[56]。该工具作为Eclipse插件开发,具有可视化和编辑模型,并生成规格文档等功能。Jonthan Lee和Kuo Hsu提出了一种目标驱动横切关注点识别方法GEA^[63]。该方法先从需求文档中识别出目标,对目标进行规格化;接着使用GDUC模型建立起目标和用例关系表;再通过聚类算法计算出目标之间的相似度,得到相似度相近的目标簇;利用目标和用例之间的相互作用值来确定目标簇的出现频繁度,根据阈值确定出候选横切关注点;最后分析候选方面,得到横切关注点。

在国内,很多学者围绕面向方面的软件开发方法和技术作了大量研究工作。文献[47-49]阐述了基于知识的需求抽取、元模型制导的需求建模,以及结合逻辑和领域本体的需求分析方法,并对现有典型的面向方面的需求工程方法进行了深入分析和比较。文献[50]提出了一种在基于构件的软件开发中系统化支持面向贯穿特性的关注点分离技术,其用面向Aspect的软件开发中的概念Advice对贯穿特性进行建模和模块化封装。文献[51]使用面向方面技术来解决科学计算程序中的误差处理问题,有效地提高了系统的模块化程度和可维护性。

我们在现有研究成果的基础上,提出了一种基于目标模型^[57,58]的横切关注点识别方法AspectQuery^[52]。在分析目标模型中影响横切关注点识别的主要因素的基础上,给出了基于目标模型的横切关注点识别规则(见表2)。若一个目标满足下述两个条件之一,则这个目标是一个横切关注点:1)该目标有助于实现一个非功能需求(软目标);2)该目标至少有两个父目标。

表2 横切关注点识别规则

目标是否与非功能需求相关	父节点是否为(候选)横切关注点	扩散度		判定结果
		总扩散度	AND扩散度	
是	是	\	\	横切关注点
否	是	不小于2	不小于2	(候选)横切关注点
		不小于2	小于2	横切关注点
	否	不小于2	小于2	候选横切关注点
		小于2	小于2	基本关注点

Theme/doc方法^[26]是现在被广泛引用的一种在需求阶段识别横切关注点的方法。开发人员需要依据领域知识从根据词法分析得到的动作视图中提炼出横切关注点。表3从方法的输入和输出、处理对象、横切关注点的组合等方面对Theme/doc方法与AspectQuery进行了比较。

3.2 横切关注点的行为约束

虽然语义冲突在一定程度上可以在软件测试阶段被检查出来,但是测试只能检查出程序有错误,不能保证程序没有错误。即,软件测试不可能检查出所有的语义冲突;而且,由于AOP系统把横切关注点进行了分离和封装,软件测试中的错误定位变得很复杂。我们认为,软件测试虽然是保证程序满足用户需求的有效手段,但软件测试不完全适合于Aspect间的语义冲突检测。

文献[31]提出了一种基于模型的冲突检测方法,它从面向方面软件的UML模型中抽取Aspect和类之间的关联信息,基于这些信息分析Aspect间可能发生的冲突。但该方法还不完善,只能检测出作用于同一个连接点的多个Aspect间的冲突。文献[32]使用COMPOSITION FILTERS模型^[33]解决Aspect间的语义冲突,但该方法 and 文献[31]一样,只能检测作用于同一个连接点的多个Aspect间的冲突。

契约式设计(Design by Contract, DBC)^[34]在面向对象领域应用得非常广泛,它通过指定先验条件(pre-condition)、后验条件(post-condition)、不变式(invariants)来保证某个方法的行为的正确性。同时,它能够精确地对违反契约的程序进行定位。

文献[35]基于行为子类型(behavioral subtyping)^[36]概念,实现了面向对象程序中的行为契约(Behavioral Contracts)。JML(Java Modeling Language)^[37]是一种行为接口规格描述语言,使用它不仅可以定义Java代码的语法接口,如方法名、参数、返回类型等,而且可以定义Java代码的行为,如先验条件、后验条件等。在AOP领域,虽然有些研究^[38]把DBC引入到了AOP,但它的重点在于是对JML的扩展(称为Pipa语言)。

文献[39]提出一种基于行为子类型思想的语义冲突检查方法,给出了面向方面程序中的行为子类型条件。通过提取基础程序和Aspect中定义的前置和后置条件,并把它们自动转换成行为子类型条件检查代码,来实现语义冲突的检查,保证织入Aspect后程序行为的正确性。

我们提出了一种基于行为子类型思想的语义冲突检查方法^[39]。行为子类型(behavioral subtyping)表示一种子类型关系,它既从结构上(类和方法的名称)也从行为上考虑父类型和子类型之间的关系。例如,类C1中包含A方法,C2继承C1,那么在结构上C2是C1的子类型,在C2中可以对A方法进行重定义。如果C2的实例能够替换C1的实例,则称C2是C1的行为子类型。即,假设父类C1中A的前置/后验条件分别表示为R1和E1,子类C2中A的前置/后验条件分别表示为R2和E2,则满足逻辑关系:(R1⇒R2)∧(R1⇒(E2⇒E1)),即子类C2中A的先验条件变弱,后验条件变强。

可以把契约式设计和行为子类型的概念应用到面向方面的软件中,把织入某个Aspect之前的程序视为父类型,织入

后的程序视为子类型,只要它们之间满足行为子类型的条件,就可以保证织入 Aspect 后的程序行为的正确性,以此实现 Aspect 间行为冲突的自动检测。

3.3 横切关注点的演化机制

重构^[40,41,62]是指在影响程序行为的前提下对软件结构进行的重组,它是提高软件质量、促成软件演化的重要方法之一。重构会导致基础程序结构的变化,比如类名的变更、方法名的变更(包括参数及返回值的变更)、方法的追加、移动及删除等。在 AOP 系统中,这些变化都会导致连接点的错配。

与切入点定义有关的研究大多集中在如何提高切入点定义的表现力上。其目的是提供一种基于高层次信息的切入点定义方法,使程序员能够更直接地定义切入点。这样的切入点描述称为语义切入点(semantic pointcuts)。现已有很多这样的提案,如 AspectJ 中的 if 和 cflow^[4]、描述数据依赖关系的 dflow^[43]等。但是这些方法的侧重点在于提高切入点的描述能力,对基础程序的演化而导致的对切入点定义的影响考虑得较少。

从提高 Aspect 的可重用性的角度来看,文献[44]描述了一种分离切入点和 advice 的方法,以此达到提高 Aspect 的重用性的目的。但该文并没有给出具体的实现方法。

文献[16]介绍了一种处理“脆弱的切入点问题”的方法:使用可视化的工具,自动检测并显示 AOP 应用程序演化前后的两个版本之间的切入点语义的区别。该方法使程序员能够快速准确地发现程序的错误,进而对程序进行修改,但它只适用于切入点语义变化的检测,需要分析软件演化前后的两个版本;而且没有给出“脆弱的切入点问题”的解决方法,即检测到切入点语义的变化后,需要手工逐一修改,没有实现软件体系结构的自适应。

文献[18]指出,基础程序和 Aspect 之间应该是一种松散的耦合,并提出一种基于 UML(Unified Modeling Language)描述的设计信息的连接点模型,其提高了 Aspect 的可重用性。该方法虽然对程序员而言显得更直观和易于理解,但存在的问题是如何把高层次的设计信息转换成低层次的代码,以及如何在这两者之间保持同步。

文献[17]提出了一种基于模型的切入点定义方法,它把脆弱的切入点问题转换到更容易解决的概念层。其采用的方法是在切入点和基础程序之间加入一层概念层,切入点的定义不是使用基础程序的构造元素,而是使用概念层的模型元素,从而实现切入点和基础程序的解耦。开发人员使用该方法时,必须预先定义好概念模型。同时,在实现方法上,它把逻辑语言作为实现语言。

基于分析的连接点定义^[45]通过静态程序分析匹配连接点。文献[46]给出了一种把基于名称的连接点定义转换成基于分析的连接点定义的方法。文献[61]基于程序元素间的结构共性抽取固定模式,并将其应用到下一个软件版本,以发现没有被原有切点匹配的连接点。

我们提出了一种基于反射和元数据的面向方面程序演化方法^[53]。其基本思路是:在切入点和基础程序之间加入一层概念层,以实现切入点(pointcut)定义和基础程序结构的解耦。概念层用于描述逻辑信息(概念、逻辑功能等)、物理信息(实现某各功能的类、方法等)以及这两类信息之间的映射关系等。在基础程序中使用元数据(使用 Java Annotation)对逻辑

信息进行标注,同时,切入点的定义也使用逻辑信息。通过程序变换,根据程序中定义的元数据自动生成元对象,生成的元对象及其方法的名称使用逻辑信息。通过元对象实现对基础程序行为的控制。同时,不把 Aspect 插入到基础程序,而将其插入到对应的元对象中。这样,切入点的定义不需要作任何修改就能够支持基础程序结构的变化。相反,如果方法和属性没有被加标注,则不会生成与这些方法和属性对应的元层程序。由于我们把最初的逻辑信息作为中间层(元层)的结构信息,即使程序被多次修改,也不会影响 Aspect 的定义。

4 发展与挑战性问题

4.1 基于领域知识的横切关注点获取

横切关注点的获取是面向方面的需求工程需要解决的一个关键问题。现有的横切关注点的获取方法主要是利用自然语言处理技术,从大量的需求文档中获取横切关注点,如文献[26,54-56],或基于各种需求模型(如目标模型、Petri 网模型等)获取横切关注点^[27,52]。这些方法缺乏对领域知识重用,需要做大量的需求分析前的准备工作(如完备的用自然语言描述的需求文档)等,导致横切关注点的获取变得复杂。

我们认为,领域知识重用和基于用户目标需求模型的关联分析的有效结合,是获取横切关注点的一种有效手段。如何重用领域知识为用户提供一个初始的目标需求,以及如何基于目标需求模型获取横切关注点,是需要解决的关键问题。

领域模型是针对某一具体领域的公共需求。目标模型(goal model)^[13,16]描述了软件系统应实现的目标以及其与运行环境之间的关系,基于目标的需求工程认为它是需求获取和分析的原点。

基于领域知识的横切关注点获取的基本思想是:以领域模型为基础,选择最合适的领域模型作为用户的初始需求模型,实现需求模型(包括横切关注点)的自动获取;通过对目标需求模型中目标间的关联分析,获取横切关注点。

一方面,在领域模型库中查找合适的领域模型作为初始需求模型,其中,查找算法是关键。以领域目标模型为例,目标名称由两部分组成:操作本体与实体本体。用户描述其需求的类自然语言通过需求分析往往可以得到描述其期望目标需求的动宾短语^[59]。因此,如何在领域模型库中找到合适的领域模型的问题,可以转化为根据动宾结构的期望目标需求在领域模型库中选择合适的由操作本体和实体本体标识的领域模型的问题。另一方面,如果在领域模型库中查找不到相应的领域模型,可以通过对目标模型中功能性目标、非功能性目标间的关联分析,查找影响其它目标的个数大于 $n(n \geq 1)$ 的目标,得到候补横切目标。

4.2 基于目标的语义连接点模型

现有的 AOP 语言提供的连接点模型(Join Point Model, JPM)不能很好地适应基础程序的演化^[18]。在 Web 服务及云计算技术正在普及的今天,迫切需要一种具有语义的连接点模型。如图 2(a)所示,假设类 A 的函数 I 和函数 J 以及类 B 的函数 M 都实现了某个功能目标(如输出字符串),仅实现方式(如输出格式)不同。能否定义连接点同时匹配这 3 个函

数? 如果功能目标具有层次关系,能否定义匹配不同层次目标的连接点? 虽然可以考虑使用通配符定义连接点,但这种连接点定义并不能够从语义上准确反映程序员的意图,而且不同组件中实现相同功能目标的函数的名称可能没有任何可以匹配的模式。更进一步的例子,如图 2(b)所示,服务 C 和 D 都实现了同样的功能目标,能否在不修改连接点定义的前提下,用服务 D 替换服务 C? 面向方面软件的演化需要更加健壮、更具有语义的连接点模型。

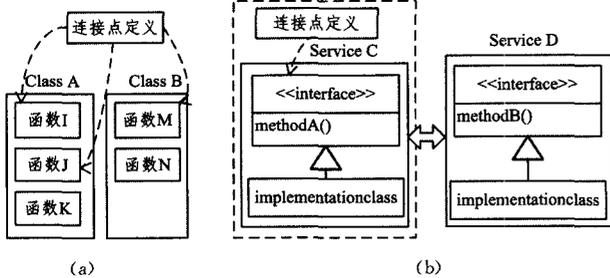


图 2 面向方面软件的演化

目标(goal)相对于需求、设计和实现来说更加稳定^[58]。我们认为基于目标的连接点模型及其实现方法可以让基础程序和连接点定义解耦,从而有助于实现面向方面软件的演化。图 3 描述了基于目标的连接点模型的基本思想。基于目标的连接点定义被转换成 AspectJ 的连接点定义后,使用 AspectJ 的编译和执行工具运行程序。

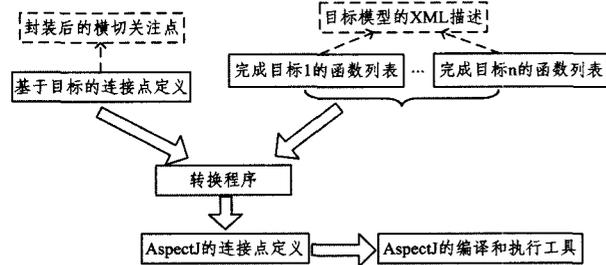


图 3 基于目标的连接点模型

结束语 面向方面的软件开发正逐步走向实用化阶段。如何在不同的软件开发阶段有效管理横切关注点是研究和开发人员面临的新挑战。文中分析了横切关注点的识别、行为约束和演化机制等横切关注点管理中的几个关键问题的研究现状,分析了其发展趋势和挑战性问题,并提出了可能的解决方案。

今后,将在现有工作的基础上,在需求阶段横切关注点的识别、语义连接点定义及实现方法方面开展更深入的研究工作。

参考文献

[1] Tarr P, Ossher H, Harrison W, et al. N Degrees of Separation: Multi-Dimensional Separation of Concerns[C]// Proc. 21st International Conference on Software Engineering. May 1999; 107-119

[2] Parnas D L. On the criteria to be used in decomposing systems into modules[J]. Communications of the ACM, 1972, 15(12): 1053-1058

[3] Kiczales G, Lamping J, Menhdhekar A, et al. Aspect oriented programming[C]// Proceedings of ECOOP'97. Number 1241 in

Lecture Notes in Computer Science, 1997; 220-242

[4] AspectJ team. The AspectJ Programming Guide[OL]. (2003). <http://eclipse.org/Aspectj>

[5] Aspect-Oriented Software Association. Aspect-oriented software development[OL]. (2015). <http://aosd.net>

[6] Baniassad E L A, Clements P C, João Araújo, et al. Discovering early aspects[J]. IEEE Software, 2006, 23(1): 61-70

[7] Zhang C, Jacobsen H A. Mining Crosscutting Concerns through Random Walks[J]. IEEE Transactions on Software Engineering, 2012, 38(5): 1123-1137

[8] Landuyt D V, Truyen E, Joosen W. Automating the Discovery of Stable Domain Abstractions for Reusable Aspects[C]// Workshop on Aspect-Oriented Requirements Engineering and Architecture Design. Early Aspects at ICSE, 2009; 1-7

[9] Mussbacher, Amyot G. On Modeling Interactions of Early Aspects with Goals[C]// Workshop on Aspect-Oriented Requirements Engineering and Architecture Design. Early Aspects at ICSE 2009, 2009; 14-19

[10] Van Landuyt D, Op de beeck S, Truyen E, et al. Domain-driven discovery of stable abstractions for pointcut interfaces[C]// Proceedings of the 8th International Conference on Aspect-oriented Software Development (AOSD'09). ACM, 2009; 1-10

[11] Weston N, Chitryan R, Rashid A. A Formal Approach to Semantic Composition of Aspect-Oriented Requirements[C]// Proceedings of the 16th IEEE International Requirements Engineering Conference. 2008; 173-182

[12] Yu Yi-jun, Sampaio J C, Mylopoulos J. From Goals to Aspects: Discovering Aspects from Requirements Goal Models[C]// Proceedings of the 12th IEEE International Requirements Engineering Conference. 2004; 38-47

[13] Kotonya G, Kotonya G, Sommerville I. Requirements Engineering with Viewpoints[J]. Software Engineering Journal, 1996, 11; 5-18

[14] Schneider G, Winters J. Applying Use Cases; a practical guide [M]. Addison-Wesley, 1998

[15] Mylopoulos J. Goal-Oriented Requirements Engineering [C]// Asia-Pacific Software Engineering Conference (APSEC). 2005; 3

[16] Koppen C, Stoerzer M. Pcdiff: Attacking the fragile pointcut problem[C]// EIWAS. 2004; 1-8

[17] Kellens A, Mens K, Bricchau J, et al. Managing the Evolution of Aspect-Oriented Software with Model-based Pointcuts [C] // Proceedings of the 20th European Conference on Object-Oriented Programming (ECOOP'06). 2006; 501-525

[18] Cazzola W, Pini S, Ancona M. AOP for Software Evolution: A Design Oriented Approach[C]// Proceedings of the 2005 ACM Symposium on Applied Computing. 2005; 1346-1350

[19] Ubayashi N, Yasutaka K. An Extensible Aspect-Oriented Modeling Environment for Constructing Domain-Specific Languages [J]. IEICE Transactions on Information and Systems, 2012, E95-D(4): 942-958

[20] Ubayashi N, Otsubo G, Noda K, et al. AspectM: UML-Based Extensible AOM Language[C]// 23rd IEEE/ACM International Conference on Automated Software Engineering. 2008; 501-502

[21] Baniassad E, Clements P, Araújo J, et al. Discovering Early Aspects[J]. Software, IEEE, 2006, 23(1): 61-70

- [22] Li Gang. Identification crosscutting concerns in requirement specifications-a case study[D]. Queen's University, September 2009
- [23] Rosenhainer L. Identifying crosscutting concerns in requirements specifications[C]// Proceedings of the Aspect-Oriented Requirements Engineering and Architecture Design Workshop. 2004;24-28
- [24] Cleland-Huang J,Settimi R,Zou Xu-chang, et al. Automated classification of non-functional requirements[C]//Proceeding of the 4th WSEAS International Conference on Computer Engineering and Applications(CEA'10). 2010;137-142
- [25] Duan Chuan,Cleland-Huang J. A clustering technique for early detection of dominant and recessive cross-cutting concerns[C]// Aspect-oriented Requirements Engineering and Architecture Design. 2007
- [26] Baniassad E, Clarke S. Theme; an approach for aspect-oriented analysis and design[C]// Int'l Conf. on Software Eng. . 2004; 158-167
- [27] Abdelzad V, Aliee F S. A method based on petri nets for identification of aspects[J]. Information Science and Technologies Bulletin of the ACM Slovakia, 2010, 2(1), 43-49
- [28] Zhang C, Jacobsen H A, Yu Y. Linking goals to aspects[C]// Presented at Workshop on Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design, Held in Conjunction with AOSD Conference. 2005;1-6
- [29] Nora B, Said G, Fadia A. A comparative classification of aspect mining approaches[J]. Journal of Computer Science, 2006, 2(4); 322-325
- [30] He Cheng-wan, Tu Cheng-mao. GPRN: A hierarchical framework for aspect-oriented requirement modeling[J]. International Journal of Digital Content Technology and its Applications, 2011, 5(2); 165-172
- [31] Tessier F, Badri M, Badri L. A Model-Based Detection of Conflicts Between Crosscutting Concerns; Towards a Formal Approach[C]// International Workshop on Aspect-oriented Software Development. Peking University, China, September 2004
- [32] Durr P E A, Bergmans L M J, Ak şit M. Reasoning about Semantic Conflicts between Aspects[C]//Proceedings of the First Aspect, Dependencies, and Interactions Workshop. 2006;10-18
- [33] Aksit M, Bergmans L, Vural S. An Object-Oriented Language-Database Integration Model; The Composition-Filters Approach [M] // ECOOP'92 European Conference on Object-Oriented Programming. Springer-Verlag, 1992; 372-395
- [34] Mitchell R, McKim J. Design by Contract by Example[M]. Boston: Addison-Wesley, 2002
- [35] Findler R B, Latendresse M, Felleisen M. Behavioral contracts and behavioral subtyping[C]//Proceedings of the 8th European Software Engineering Conference Held Jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering. 2001;229-236
- [36] Liskov B H, Wing J M. A behavioral notion of subtyping[J]. ACM Transactions on Programming Languages and Systems (TOPLAS), 1994, 16(6); 1811-1841
- [37] Leavens G T, Cheon Y. Design by Contract with JML[OL]. (2006). <http://www.eecs.ucf.edu/~leavens/JML/jmldbc.pdf>
- [38] Zhao Jian-jun, Rinard M. Pipa: A Behavioral Interface Specification Language for AspectJ[C]// 6th International Conference (FASE). 2003;150-165
- [39] He Cheng-wan, Li Zheng. Implementation of Behavioral Subtyping in Aspect-oriented Program[J]. International Journal of Digital Content Technology and its Applications, 2011, 5(2); 206-215
- [40] Bois B D, Van Gorp P, Amsel A, et al. A Discussion of Refactoring in Research and Practice[OL]. (2004). <http://win.ua.ac.be/~lore/refactoringProject/publications/ADiscussionOfRefactoringInResearchAndPractice.pdf>
- [41] Mens T, Tourwe T. A Survey of Software Refactoring[J]. IEEE Transactions on Software Engineering, 2004, 30(2); 126-139
- [42] Ali B S, Kasirun Z M D. An approach for crosscutting concern identification at requirements level using NLP[J]. International Journal of the Physical Sciences, 2011, 6(11); 2718-2730
- [43] Masuhara H, Kawauchi K. Data flow pointcut in Aspect-oriented programming[C]// APLAS'03. 2003;105-121
- [44] Beugnard A. How to make Aspects re-usable, a Proposition [C]// Proceedings of the Aspect-oriented Programming Workshop at ECOOP. 1999
- [45] Aotani T, Masuhara H. SCoPE: An AspectJ compiler for supporting user-defined analysis-based pointcuts[C]// International Conference on Aspect-Oriented Software Development. 2007; 161-172
- [46] Wang Lin, Aotani T. Automatic translation from name-based pointcuts to analysis-based pointcuts for robust aspects[C]// Proceedings of the 8th Workshop on Reflection, AOP and Meta-Data for Software Evolution. 2011
- [47] Jin Zhi, Bell D A, Wilkie F G, et al. Automated Requirements Elicitation; Combining a Model-driven Approach with Concept Reuse[J]. International Journal of Software Engineering and Knowledge Engineering, 2003, 13(1); 53-82
- [48] Jin Zhi. Revisiting the Meaning of Requirements[J]. Journal of Computer Science and Technology, 2006, 21(1); 32-40
- [49] 金芝, 刘璘, 金英. 软件需求工程: 原理和方法[M]. 北京: 科学出版社, 2008
- [50] Mei Hong, Cao Dong-gang. BC-S2C: Enabling Separation of Crosscutting Concerns in Component-Based Software Development[J]. Chinese Journal of Computers, 2005, 28(12); 2036-2044 (in Chinese)
梅宏, 曹东刚. ABC-S2C: 一种面向贯穿特性的构件化软件关注点分离技术[J]. 计算机学报, 2005, 28(12); 2036-2044
- [51] Cui Zhan-qi, Wang Lin-zhang, Liu Hui-gen, et al. Computational Error Handling as Aspects: A Case Study and Evaluation[J]. Journal of Software, 2011, 22(11); 2639-2651 (in Chinese)
崔展齐, 王林章, 刘慧根, 等. 面向方面的计算误差处理技术: 实例研究与评估[J]. 软件学报, 2011, 22(11); 2639-2651
- [52] He Cheng-wan, Tu Cheng-mao. AspectQuery: A Method for Identification of Crosscutting Concerns in the Requirement Phase [J]. IEICE Transactions on Information and Systems, 2013, E96-D(4); 897-905
- [53] He Cheng-wan, Zhang Li-jun, Zhang Hui. An Approach to Aspect-Oriented Software Evolution Based on Metadata and Reflection[J]. Acta Electronica Sinica, 2011, 39(8); 1771-1777 (in Chinese)

参考文献

- [1] Duygulu P, Barnard K, de Freitas J F G, et al. Object recognition as machine translation; Learning a lexicon for a fixed image vocabulary[C]// Computer Vision (ECCV 2002). Springer Berlin Heidelberg, 2002; 97-112
- [2] Barnard K, Duygulu P, Forsyth D, et al. Matching words and pictures[J]. The Journal of Machine Learning Research, 2003, 3 (2): 1107-1135
- [3] Carneiro G, Chan A B, Moreno P J, et al. Supervised learning of semantic classes for image annotation and retrieval[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(3): 394-410
- [4] Jeon J, Lavrenko V, Manmatha R. Automatic image annotation and retrieval using cross-media relevance models[C]// Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval. ACM, 2003; 119-126
- [5] Feng S L, Manmatha R, Lavrenko V. Multiple bernoulli relevance models for image and video annotation[C]// Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004 (CVPR 2004). IEEE, 2004, 2
- [6] Grangier D, Bengio S. A discriminative kernel-based approach to rank images from text queries[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 30(8): 1371-1384
- [7] Makadia A, Pavlovic V, Kumar S. A new baseline for image annotation[M]// Computer Vision (ECCV 2008). Springer Berlin Heidelberg, 2008; 316-329
- [8] Guillaumin M, Mensink T, Verbeek J, et al. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation[C]// IEEE 12th International Conference on Computer Vision, 2009. IEEE, 2009; 309-316
- [9] Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database[C]// IEEE Conference on Computer Vision and Pattern Recognition, 2009 (CVPR 2009). IEEE, 2009; 248-255
- [10] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]// Advances in Neural Information Processing Systems. 2012; 1097-1105
- [11] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]// CVPR. 2015; 1-9
- [12] He K, Zhang X, Ren S, et al. Delving deep into rectifiers; Surpassing human-level performance on imagenet classification[C]// ICCV. 2015; 1026-1034
- [13] Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional architecture for fast feature embedding[C]// Proceedings of the ACM International Conference on Multimedia. ACM, 2014; 675-678
- [14] Duygulu P, Barnard K, de Freitas J F G, et al. Object recognition as machine translation; Learning a lexicon for a fixed image vocabulary[M]// Computer Vision (ECCV 2002). Springer Berlin Heidelberg, 2002; 97-112
- [15] Metzler D, Manmatha R. An inference network approach to image retrieval[M]// Image and video retrieval. Springer Berlin Heidelberg, 2004; 42-50
- [16] Yavlinsky A, Schofield E, Ruger S. Automated image annotation using global features and robust nonparametric density estimation[M]// Image and video retrieval. Springer Berlin Heidelberg, 2005; 507-517
- [17] Carneiro G, Chan A B, Moreno P J, et al. Supervised learning of semantic classes for image annotation and retrieval[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(3): 394-410
- [18] Liu J, Li M, Liu Q, et al. Image annotation via graph learning[J]. Pattern recognition, 2009, 42(2): 218-228
- [19] Zhang S, Huang J, Huang Y, et al. Automatic image annotation using group sparsity[C]// 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2010; 3312-3319
- [54] Baniassad E, Clarke S. Finding Aspects in Requirements with Theme/Doc[C]// Proceedings of Early Aspects 2004; Aspect-oriented Requirements Engineering and Architecture Design. 2004
- [55] Elrad T, Filman E R, Bader A. Aspect-oriented programming: Introduction[J]. Communications of the ACM, 2001, 44(10): 29-32
- [56] Sampaio A, Rashid A, Chitchyan R. EA-Miner: Towards Automation in Aspect-Oriented Requirements Engineering [J]. Transactions on Aspect-Oriented Software Development III, 2007, 4620; 4-39
- [57] Mylopoulos J. Goal-Oriented Requirements Engineering [C]// 12th Asia-Pacific Software Engineering Conference (APSEC). 2005; 3
- [58] Anton A I, McCracken W M, Potts C. Goal Decomposition and Scenario Analysis in Business Process Reengineering[M]// Advanced Information Systems Engineering. Springer-Verlag, 1994; 94-104
- [59] Liu Wei, He Ke-qing, Wang Jian, et al. Heavyweight Semantic Inducement for Requirement Elicitation and Analysis [C] // Third International Conference on Semantics, Knowledge and Grid. 2007; 206-211
- [60] Rebelo H, Lima R, Leavens G T. Modular contracts with procedures, annotations, pointcuts and advice[C]// Proceedings of the 2011 Brazilian Symposium on Programming Languages. 2011; 1-16
- [61] Khatchadourian R, Greenwood P, Rashid A, et al. Pointcut Rejuvenation; Recovering Pointcut Expressions in Evolving Aspect-Oriented Software[J]. Sactions on Software Engineering, 2012, 38(3); 642-656
- [62] Wang L, Aotani T, Suzuki M. Interaction awareness for aspect refactoring[C]// Proceedings of the 8th International Workshop on Advanced Modularization Techniques-AOAsia/Pacific. New York; ACM, 2013; 15-18
- [63] Lee J, Hsu K. GEA: A Goal-Driven Approach to Discovering Early Aspect[J]. IEEE Transactions on Software Engineering, 2014, 40(6); 584-602

(上接第 12 页)

何成万, 张立军, 张慧. 基于元数据和反射的面向方面软件演化方法[J]. 电子学报, 2011, 39(8): 1771-1777