软件体系结构层次的结构度量研究

高 晖,张 莉
GAO Hui,ZHANG Li

北京航空航天大学 软件工程研究所,北京 100083
Software Engineering Institute,Beihang University,Beijing 100083,China
E-mail:gaohui@cse.buaa.edu.cn

GAO Hui,ZHANG Li.Research on software architecture level structure metrics.Computer Engineering and Applications, 2007,43(24):19-23.

Abstract: In this paper,a set of exercisable measurement set is developed,which can be used to measure the structure features (such as complexity,coupling,morphology) of software architecture models.The relationships possibly existed in different features are educed,after we analyze measured data using traditional correlation analytical method,which is commonly used in software measurement.The result in this paper provides an essential foundation for the research on the relationships between the structure features of software architecture and quality attributes.

Key words: software architecture;software metrics;software quality

摘 要:提出了一套可操作的度量组,利用该度量组可以对软件体系结构模型的结构特征(复杂性、耦合性、形态)进行测量。利用软件度量中常用的相关性分析方法对度量的经验值进行分析,初步得出了各种度量间可能存在的相关关系。此研究为研究软件体系结构的结构特征与软件其它质量属性的关系提供了必要的基础。

关键词:软件体系结构;软件度量;软件质量
文章编号:1002-8331(2007)24-0019-05 **文献标识码**:A **中图分类号**:TP301

1 介绍

一个成功的软件产品除了能够实现功能上的需求,还必须满足用户在可移植、可扩展、易用、易维护、安全、可靠等诸多非功能质量属性的要求。实践人员和研究人员一致认为:在产品的结构和质量之间可能存在着一定的联系,所以,不少人都在软件结构性质的测量方面做着各种努力[1]。在传统的软件测量研究中,按照测量的对象分类,度量技术可以分为两类:第一类是程序代码度量[1],比如:代码行(LOC)、Halstead 的程序长度/词汇量/容量度量等;第二类设计模型的度量,比如:模块内部度量[1]、基于构件技术中的构件模型度量、面向对象度量技术[2-4]等。随着软件体系结构技术、模型驱动体系结构(MDA)开发方法的发展,软件体系结构成为设计过程中对软件质量具有重要影响的一个设计制品。软件体系结构通过构件、连接件等基本元素对软件系统的结构和行为等方面进行抽象的描述,其结构特征直接影响了软件可扩展性、易维护性、性能等质量属性。所以,研究软件体系结构模型的结构特征,并研究其测量方法是非常必要的。

由 Basili 等人提出的 GQM 方法(Goal-Question-Metric)[5]为研究软件体系结构度量提供了的基础。各种度量技术根据 GQM 框架提出了软件体系结构层次上的不同度量,比如:在 PASA[6,7]中根据性能目标为评估性能特征定义了量化的标准,这些量化的标准包括计算机资源需求、作业驻留时间、利用率、吞吐量、队列长度等性能相关的度量;在 SACMM[8]中为了对软件可更改性及软件体系结构的演化性进行评估,利用 graph kernel 函数定义距离度量对软件体系结构的相似性进行量化;ALRRA[9]中为了进行可靠性风险分析,研究了两类度量:一类是与软件体系结构动态复杂性相关的一组度量,包括:构件的操作复杂度、连接件的输出耦合度;另外一类是与故障相关的一组度量,包括:构件和连接件的故障模式、故障严重性级别。通过复杂性度量和故障严重性级别;可以计算可靠性风险因子。Hany Ammar 等人利用熵的概念提出了软件体系结构的静态和动态度量[11];另外,Losavio 和 Chirinos 根据 ISO9126 定义的质量子特征定义了一套软件体系结构层次上的度量组[10]。这些软件体系结构层次上的度量技术都解决了研究中的特定问题,然而对于软件体系结构所体现的一些通用结构特征,比如:结构复杂性、耦合性、软件形态等,需要定义度量组对这些特性进行测量。

本文开发了一套软件体系结构层次的结构度量组,量化地表示软件体系结构的结构特征;为了验证度量组,利用该度量组对软件体系结构模式(某些模式需要进行应用假设)进行测量,并对这些测量结果与软件体系结构模式的质量特征进行分析。

基金项目:新世纪优秀人才支持计划资助(Supported by Program for New Century Excellent Talents in University)。
作者简介:高晖(1977-),男,博士生,主要研究领域为软件体系结构,UML 相关技术;张莉(1968-),女,教授,博士生导师,主要研究领域为企事业过程工程,软件体系结构。

2 软件体系结构概念及描述方法

结合在软件开发过程中的实践与其他研究人员对软件体系结构的定义[12],笔者认为软件体系结构是软件开发早期的一项软件制品,它通过一组元素(构件、连接件等)来表示软件系统的结构信息和运行时交互信息,并描述了这组元素如何达到系统需求,特别是系统的软件质量需求。模型是对现实世界的抽象,软件体系结构模型也就是对软件体系结构的一种抽象表示。软件体系结构模型就是软件体系结构层次上进行度量的对象。在软件体系结构模型中,最核心的概念是构件和连接件。另外,角色、端口和角色类型都是软件体系结构中比较重要的概念。表 1 中给出了这些主要概念。这些基本概念在软件体系结构研究中已经稳定,并得到研究人员的认可。为了使度量与具体的描述技术无关,本文在第 3 章中将利用这些概念对度量进行定义。这样定义的度量组可以不受限于具体的软件体系结构描述语言,只需要选用的软件体系结构描述语言能够支持这些概念即可进行度量。

表 1 软件体系结构的主要概念

名称	定义	职责
构件 SA_Component	构件指系统中负责进行计算或数据存储的单元。一个构件可以有自己的状态和数据,同时需要与其它构件进行通信和交互	用于描述软件系统中主要的计算单元和数据存储
连接件 SA_Connector	连接件是对构件之间通信、协调和合作机制的一种抽象	描述软件系统中各个构件的连接和交互关系
端口 SA_Port	端口是构件在参与某个交互过程中需要的上下文环境	利用端口对构件具备的某种职责进行描述,用于表示构件的外部可见特征
角色 SA_Role	角色是参与连接件实现的需求时的某种职责的抽象表示	利用角色对特定需求的参与者进行描述,用于表示连接件中职责的划分
角色类型 SA_RoleType	角色类型是对角色职责的抽象表示,一般利用接口的方式表达	利用角色类型说明端口模板和角色模板的职责

虽然度量组不依赖于语言,但本文中为了能够具体地说明如何测量软件体系结构模型,采用了一种基于 UML2.0 的软件体系结构描述语言 TADL[13]来描述软件体系结构模型。TADL 采用外廓方式扩展 UML 得到的一种软件体系结构描述语言,并且可以支持表 1 中的所有概念。TADL 不仅可以描述软件体系结构模型,也能够描述软件体系结构模式。在 TADL 中,软件体系结构模型有 5 个视图:构件依赖视图、构件结构视图、连接件角色视图、交互行为视图和角色类型视图。这几个视图从不同角度对软件体系结构进行描述,在软件体系结构层次的结构度量中,除了交互行为视图,其它四个视图都为度量提供了所需信息。图 1 中给出了利用 TADL 描述 MVC 模式的多视图模型。

3 软件体系结构层次的结构度量组

软件体系结构层次的结构度量指的是对软件体系结构模型的静态结构进行的度量。静态结构提供了以下 3 部分信息:构件的外部可见特性及其关系; 构件实例之间的连接关系;连接件中角色之间的连接关系。利用这些信息,本文从复杂性、耦合性和结构形态 3 种类型的结构特性对软件体系结构模型进行度量。本文通过四个部分对度量进行描述:定义、说明、观点、经验数据。定义部分描述了度量的计算方法或公式;说明部分描述了该度量定义的一些基本原理以及在 TADL 中计算的方法;观点部分描述了该度量可能产生的影响,包括与软件质量属性之间的关系以及与设计策略之间的关系;经验数据部分则是对 7 种常用的软件体系结构模式,包括:分层模式 Layer、管道过滤器模式 Pipeline、黑板模式 Blackboard、代理者模式 Broker、模型–视图–控制器模式 MVC、表示–抽象–控制模式 PAC、微核模式 Microkernel[14]进行测量的结果,以柱状图形式给出。

3.1 结构复杂性度量组

结构复杂性度量组主要考虑软件系统各部分之间的相互作用,并将这些相互作用量化。本文定义了两个度量对系统之间的相互作用进行测量。

3.1.1 度量 1:系统角色复杂性(CP_SYS_R)

定义 软件体系结构模型 M 中,特定连接件 x 中的角色集合为 $Role(x)$,则系统角色复杂性定义为

$$CP\_SYS\_R=\frac{\sum_{x\in CN}(|Role(x)|)^2}{|CP|} \quad (1)$$

其中 CP 为构件集合,CN 为连接件集合。

说明 连接件中的角色表示了软件系统在实现某个特定业

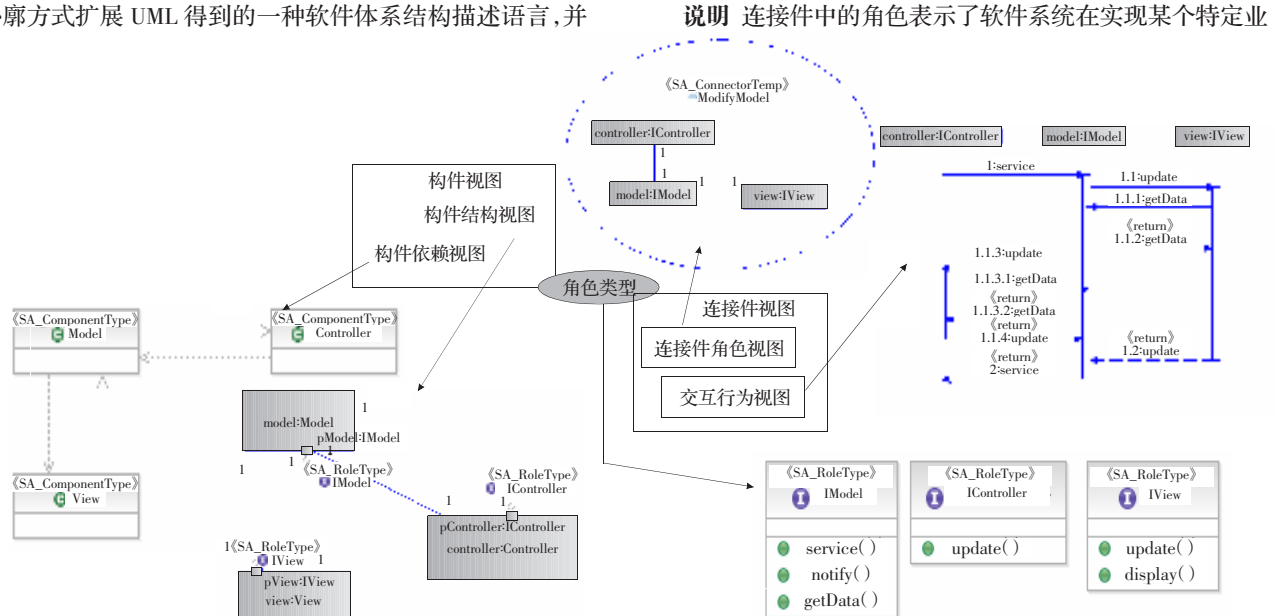


图 1 MVC 模式的多视图模型

务过程时的职责划分,连接件中角色越多说明系统模块之间的交互越复杂,系统总的角色复杂性则是连接件角色平方和的均值。该定义的基本原理是基于程序间连接的平方数增加时复杂性增加的发现[1]。在 TADL 描述的软件体系结构模型中,从连接件角色视图可以得到各个连接件的角色个数,从构件视图可以得到构件个数,则该值通过公式(1)计算。

观点:

(1)当连接件中的角色数量越多,则软件系统中的交互会增加,构件之间的依赖关系也会越复杂。如果系统角色复杂性越高,则软件系统各部分的交互关系会非常复杂,可能对系统的可扩展性、可维护性等质量属性会产生影响。

(2)根据对 7 种常用的软件体系结构模式[14]进行测量的结果,系统角色复杂性的取值在 3~8 之间。如果该值太大,则可以通过对特定构件的进一步分解,降低系统角色复杂性。

经验数据:利用该度量,对 7 种常用的软件体系结构模式进行度量,结果见图 2。

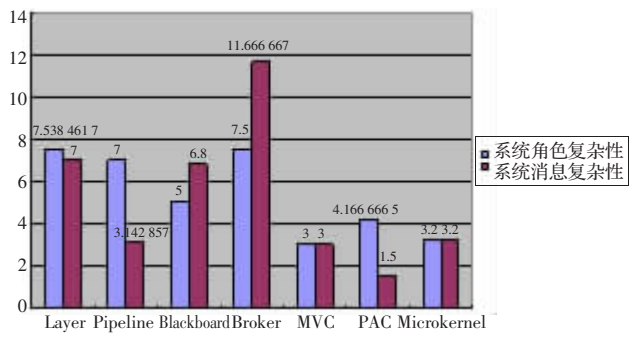


图 2　系统复杂性度量经验数据

3.1.2　度量 2:系统消息复杂性(CP_SYS_OM)

定义　软件体系结构模型 M 中,特定连接件 x 中的输出消息集合为 $Messages(x)$,则系统消息复杂性定义为

$$CP\_SYS\_OM=\frac{\sum_{x\in CN}(|Messages(x)|)}{|CP|} \tag{2}$$

其中 CP 为构件集合,CN 为连接件集合。

说明　连接件中各个角色之间交互时,当角色 ra 调用角色 rb 的一个方法时,认为 ra 向 rb 发送一个输出消息。由于软件体系结构层次是一个非常抽象的设计层次,能够获取的信息有限,为了简化系统消息复杂性的计算,本文利用角色类型的函数数量来计算角色之间的输出消息。该定义的基本原理是对系统角色复杂性的细化,利用更丰富的信息来定义系统交互的复杂性,从测量结果可以初步判断系统消息复杂性比系统角色复杂性更加准确。在 TADL 描述的软件体系结构模型中,从连接件角色视图可以得到各个连接件的角色并根据角色类型可以得到接口函数的数量,从构件视图可以得到构件个数,则该值通过公式(2)计算。

观点:

(1)当连接件中的输出消息越多,则软件系统中的交互会增加,构件之间的依赖关系也会越复杂。系统消息复杂性可能对系统的可扩展性、可维护性等质量属性会产生影响。

(2)根据对 7 种常用的软件体系结构模式[13]进行测量的结果,系统消息复杂性的取值在 1.5~11.67 之间。如果该值太大,则可以通过对特定构件的进一步分解,降低系统消息复杂性。

经验数据:利用该度量,对 7 种常用的软件体系结构模式进行度量,结果见图 2。

3.2　结构耦合度量组

耦合是指两个模块之间的相互依赖程度,是一个模块对的属性。全局耦合则利用软件系统中所有可能的模块对耦合度进行计算。本文定义系统耦合度是对软件体系结构一个整体属性的测量。

3.2.1　度量 3:系统耦合度(CL_SYS_MEDIAN)

定义　软件体系结构模型 M 中,系统耦合度定义为:

$$CL\_SYS\_MEDIAN=median(\{CL\_CP\_IND(x,y)\}) \tag{3}$$

其中 $median$ 计算集合的平均值:

$$CL\_CP\_IND(x,y)=i+\frac{n}{n+1} \tag{4}$$

说明　系统耦合度的定义是基于构件间耦合度的,式(8)给出了构件间的耦合度定义。其中最坏耦合关系的系数借鉴了文[1]中的分类:R0:无耦合关系;R1:数据耦合关系;R2:标记耦合关系;R3:控制耦合关系;R4:公共耦合关系;R5:内容耦合关系。在给出了构件间耦合的基础上,对所有构件对的耦合关系求平均值就得到系统的耦合度。在 TADL 描述的软件体系结构模型中,从构件依赖视图得到构件间连接关系,并确定连接关系的类型,则该值可以通过公式(3)即可计算系统耦合度。

观点:

(1)系统耦合度越大,则软件系统中构件之间的关系会越复杂。系统耦合度可能对系统的可扩展性、可维护性和可更改性产生影响。

(2)根据对 7 种常用的软件体系结构模式[14]进行测量的结果,系统连耦合度的取值在 0.9~2.5 之间。如果该值太大,可以利用设计模式等技术改善系统设计。

经验数据:利用该度量,对 7 种常用的软件体系结构模式进行度量,结果见图 3。在测试时,本文中假定构件间的最坏偶合关系系数为控制耦合关系,即构件间存在参数传递,所以 i 取值为 3。

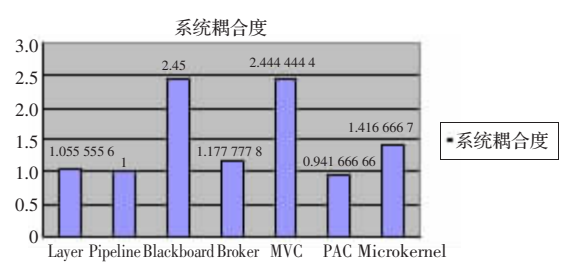


图 3　系统耦合度量经验数据

3.3　结构形态度量组

Yourdon 和 Constantine 采用"形态(morphology)"这一概念来指用图示的方式表示的总体系统结构的"形状",形态特性可以用来区分软件设计的好坏[1]。在软件体系结构的设计实践中,设计师也常常利用"形态"的概念来指导自己的设计。本文定义了以下几个度量对软件体系结构模型的形态进行测量。

3.3.1　度量 4:系统深度(MP_SYS_DEPTH)

定义　软件体系结构模型 M 中,特定连接件 x 中的角色集合为 $Role(x)$,则系统深度定义为:

$$MP\_SYS\_DEPTH=\max(|Role(x)|),x\in CN \tag{5}$$

其中 CN 为连接件集合。

说明　本文将软件体系结构的深度定义为系统各部分完成

交互所包含的最大角色数,所以计算各个连接件中包含的角色数,并取最大值就得到系统深度。系统交互包含的角色数越多,说明软件的实现越复杂。系统深度利用角色数来表示系统中最复杂的交互关系。在TADL描述的软件体系结构模型中,从连接件角色视图得到各个连接件的角色数量,则该值可以通过公式(5)取得最大值即可得到系统的深度。

观点:

(1)当连接件中的角色数量越多,则软件系统中的交互会增加,在完成该特定业务时复杂性也增加。系统深度可能对系统的功能分解结构和软件体系结构的可构造性产生影响。

(2)根据对7种常用的软件体系结构模式[14]进行测量的结果,系统深度的取值在3~7之间。如果该值太大,则说明业务太复杂或者设计时角色分配不够合理。

经验数据:利用该度量,对7种常用的软件体系结构模式进行度量,结果见图4(a)。

3.3.2 度量5:系统宽度(MP_SYS_WIDTH)

定义 软件体系结构模型 M 中,系统宽度定义为

$$MP\_SYS\_WIDTH=\max(|CP(t)|),t\in CP\_TYPE \quad (6)$$

其中 CP_TYPE 是构件类型结合,$CP(t)$是属于构件类型 t 的构件集合。

说明 本文将软件体系结构的宽度定义为系统中具有相同特征的构件集合的基数,所以将构件划分为不同的类型,并计算集合的基数就得到系统宽度。在软件体系结构设计中一个重要的特点就是构件可以按照其特征进行更高层次的分类,同一类的构件具有相似的外部特性、实现技术和开发成本。在TADL描述的软件体系结构模型中,从构件视图中可以每个构件所属的构件类型,通过构件类型将构件划分为不同的集合,则该值通过公式(6)取得这些集合的基数的最大值即可得到系统的宽度。

观点:

(1)系统宽度可能对软件开发的风险产生影响。因为系统宽度很高时,如果对应的该类构件的设计出现缺陷,会造成很多开发资源的浪费。采用不同的实现技术可以降低实现的风险,比如:设计和实现时对该属于相同类型的构件可以采用继承、模板等程序设计语言提供的抽象机制。

(2)根据对7种常用的软件体系结构模式[14]进行测量的结果,系统宽度的取值有很大差别,例如:在分层模式中该值很大,这是由于分层模式常用于信息系统中,而信息系统的需求从其实现技术上看具有很大的相似性,一般每个需求的实现都由"UI+逻辑处理+数据实体"这三类构件实现,系统的宽度就随着系统需求的规模而增大。本文在对分层模式进行度量时,假设了较大的需求规模,所以该值相比其他模式的度量都大。

经验数据:利用该度量,对7种常用的软件体系结构模式进行度量,结果见图4(b)。

3.3.3 度量6:系统连接性密度(MP_SYS_CD)

定义 软件体系结构模型 M 中,系统连接密度定义为

$$MP\_SYS\_CD=\frac{\sum_{x\in CN}(dependence\_count(x))}{|CP|} \quad (7)$$

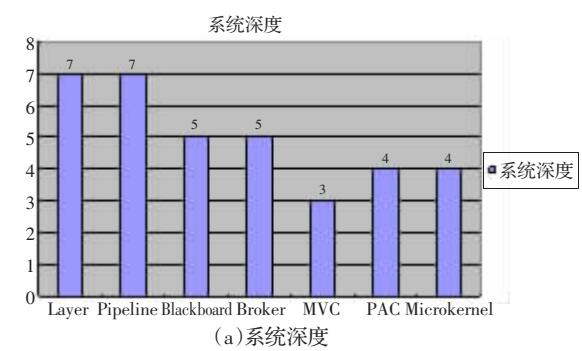
其中 CP 为构件集合,$dependence_count(x)$是构件 x 依赖的构件数量。

说明 本文利用构件依赖图中的边和结点之比作为系统连接性密度度量。连接性密度表示了构件之间关系的复杂程度。理论上,根据图的性质,当图退化为树时,连接性密度的取最小值$(n-1)/n$;当图为完全有向图时,连接性密度取最大值 $n-1$ 之间,n 是构件的个数。在TADL描述的软件体系结构模型中,从构件视图中的构件依赖图可以得到每个构件与其它构件的依赖关系,则该值通过公式(7)以计算。

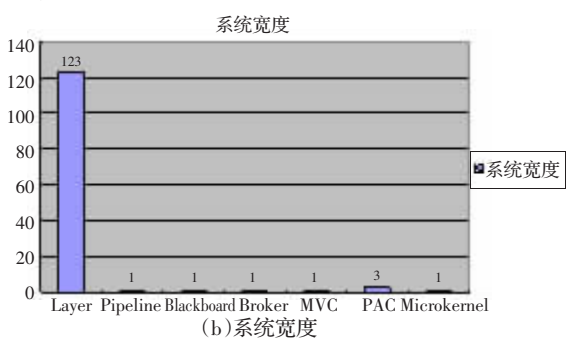
观点:

(1)系统连接性密度越大,则软件系统中构件之间的关系会越复杂。系统连接性密度可能对系统的可扩展性、可维护性和可更改性产生影响。

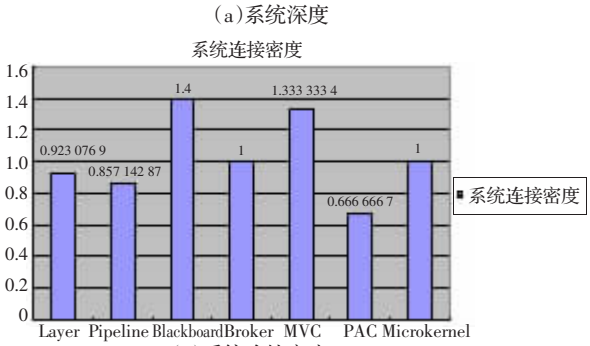
(2)根据对7种常用的软件体系结构模式[14]进行测量的结果,系统连接性密度的取值在0.5~1.5之间。如果该值太大,则



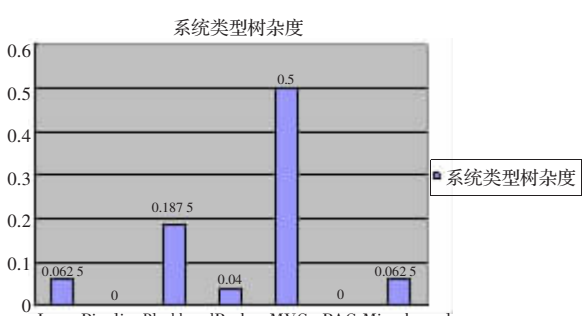
(a)系统深度



(b)系统宽度



(c)系统连接密度



(d)系统类型树杂度

图4 系统形态度量经验数据

表 2　Spearman 和 Kendall 等级相关系数

	消息复杂性	耦合度	深度	宽度	连接性密度	类型树杂度
角色复杂性	0.71/0.52	−0.36/−0.24	0.90/0.82	0.40/0.33	−0.29/−0.20	−0.38/−0.25
消息复杂性	−/−	0.29/0.24	0.55/0.41	−0.04/−0.07	0.34/0.29	0.15/0.05
耦合度	−/−	−/−	−0.33/−0.31	−0.58/−0.46	0.99/0.98	0.89/0.75
深度	−/−	−/−	−/−	0.25/0.20	−0.30/−0.26	−0.39/−0.32
宽度	−/−	−/−	−/−	−/−	−0.58/−0.44	−0.25/−0.21
连接性密度	−/−	−/−	−/−	−/−	−/−	0.87/0.72

注:表格中有数值的表项中前一个数字为 Spearman 相关系数,后一个数字为 Kendall 相关系数。

说明构件的设计不合理。

经验数据:利用该度量,对 7 种常用的软件体系结构模式进行度量,结果见图 4(c)。

3.3.4　度量 7:系统类型树杂度(MP_SYS_TIP)

定义　软件体系结构模型 M 中,系统类型树杂度定义为:

$$MP\_SYS\_TIP=\frac{\text{多于构件依赖图生成树的边数}}{\text{多于生成树的最大边数}} \quad (8)$$

说明　Ince 和 Hekmatpour 认为系统偏离单纯的树结构的距离越远、离图结构的距离越近,设计的质量就越糟糕。所以利用树杂度来衡量给定的图偏离树结构的程度[1]。本文借鉴[1]中定义的树杂度来定义软件体系结构层次构件依赖图的树杂度。在 TADL 描述的软件体系结构模型中,从构件视图中的构件依赖图可以得到每个构件与其它构件的依赖关系,则该值通过公式(8)以计算。

观点:

(1)树杂度越大,则软件系统中构件之间的关系会越复杂。系统类型树杂度可能对系统的可扩展性、可维护性和可更改性产生影响。

(2)根据对 7 种常用的软件体系结构模式[14]进行测量的结果,系统深度的取值在 0~0.6 之间。如果该值太大,则说明构件的设计不合理。

经验数据:利用该度量,对 7 种常用的软件体系结构模式进行度量,结果见图 4(d)。

4　数据相关性分析

首先,通过对各种度量的定义的分析,可以直观地假设具有相关关系的是:(1)角色复杂性和消息复杂性、角色复杂性和深度、消息复杂性和深度,这组度量存在相关关系的原因是都与连接件中的角色相关,上述 3 个度量值都可能随着角色数量的增加而增加;(2)耦合度和连接性密度、耦合度和类型树杂度、连接性密度和类型树杂度,这组度量存在相关关系的原因是都与构件依赖图的结点与边的关系相关。

在进行相关性分析时,由于数据较少不能判定数据是否满足正态分布,所以本文没有采用 Pearson 相关系数,而是采用了在软件度量中对未知分布的数据进行相关分析时常用的两种关联稳健度量:Spearman 等级相关系数和 Kendall 等级相关系数,结果见表 2。从表 2 中数据可以看出,角色复杂性和消息复杂性、角色复杂性和深度、消息复杂性和深度、耦合度与连接性密度、耦合度与类型树杂度、连接性密度和类型树杂度这 6 项能够与前面的假设一致;而另外 2 项,耦合度与宽度、宽度和连接性密度也可能存在负相关的关系。由于数据的原因,还需要进一步的工作进行判定。

5　总结

本文的主要贡献是在软件体结构层次上提出了一套可操作的度量组,利用该度量组可以对软件体系结构模型的结构特征进行测量。利用软件度量中常用的相关性分析方法对度量的经验值进行分析,初步得出了各种度量间可能存在的相关关系。本文的研究为研究软件体系结构的结构特征与软件其它质量属性的关系(如:软件适应性、软件可扩展性等)提供了必要的基础。本文中的这些度量及其相关性还有待收集更多的经验数据进行进一步的验证。(收稿日期:2007 年 5 月)

参考文献:

[1] Fenton N E,Pfleeger S L.Software metrics a rigorous and practical approach 2nd[M].[S.l.]:China Machine Press,2004.

[2] Chidamber S R,Kemerer C F.A metrics suite for object oriented design[J].IEEE Transactions on Software Engineering,1994,20(6).

[3] Churcher N I,Shepperd M J.Comments on a metrics suite for object oriented design[J].IEEE Transactions on Software Engineering,1995,21(3).

[4] Yacoub S M,Ammar H H,Robinson T.Dynamic metrics for object oriented designs[C]//Sixth International Software metrics Symposium,1999.

[5] Basili V R,Rombach H D.Goal/question/metric paradigm:the TAME project:towards improvement-oriented software environments[J].IEEE Trans Software Eng,1998,14(6).

[6] Williams L G,Smith C U.PASA:a method for the performance assessment of software architectures[J].Software Engineering Research and Performance Engineering Services,2002.

[7] Williams L G,Smith C U.Performance evaluation of software architectures[C]//Proceedings of the 1st International Workshop on Software and Performance,1998.

[8] Nakamura T,Basili V R.Metrics of software architecture changes based on structural distance [C]//11th IEEE International Software Metrics Symposium(METRICS 2005),2005.

[9] Yacoub S M,Ammar H H.A methodology for architecture-level reliability risk analysis [J].IEEE Transactions on Software Engineering,2002,28(6).

[10] Losavio F,Chirinos L.Quality characteristics for software architecture[J].Journal of Object Technology,2003,2(2).

[11] Ammar H,Shereshevsky M,Mili A W,et al.Software Architecture Level Metrics,2001.

[12] 孙昌爱,金茂忠,刘超.软件体系结构研究综述[J].软件学报,2002,13(7).

[13] Buschmann F,Meunier R,Rohnert H,et al.Pattern-oriented software architecture volume 1:a system of Patterns [M].[S.l.]:China Machine Press,2003.