一种基于CBD的软件测试方法*)

曹严元　张为群
(西南师范大学计算机与信息科学学院　重庆400715)

摘　要　基于构件的开发方法(CBD)的提出和大量应用,对传统的测试技术提出了新的挑战。本文通过对CBD方法及特点的研究,提出了CBD软件的测试样式,通过构件规格说明构架和构件交互图对构件间交互关系建模,并在此基础上给出了构件软件系统的测试技术和方法。
关键词　CBD,基于构件的软件,软件测试,测试样式

Research for the CBD-Based Testing Method

CAO Yan-Yuan　ZHANG Wei-Qun
(College of Computer and Information Science, Southwest Normal University, Chongqing 400715)

Abstract　With the advent of component-based development, arose more challenge to traditional testing technology. This paper researches the method and feature of CBD, presents the testing type of software based on component. The interaction of components has been modeled through component architecture and the component interaction graph, then proposes the testing technology and method of component-based software.
Keywords　CBD, Component-based software, Software testing, Testing type

1　引言

在最近的一份报告中Gartner小组预计,到2003年,所有的软件解决方案中将有70%是使用像预建的构件和模板这样的"积木"来建造的[1]。基于构件的开发(CBD)和基于构件的软件工程(CBSE)正在被大量地采用,并逐渐成为软件开发的新趋势。

CBD方法相对于传统的开发方法体现出了许多新优势,由于对软件功能性、开发效率、质量、可靠性、可移植性等方面的良好支持而受越来越多软件开发组织的重视,目前已有许多工作致力于分析、设计和构建CBD软件,但对CBD软件的测试和维护问题的研究还很少。CBD软件的新特点给测试带来了新的挑战。首先,用于组装的构件的源代码往往不能获得,因此不能使用传统的集成测试方法。其次,即使得到了构件的源代码,构件也可能用不同的编程语言编写,运行在不同的平台上,这要求测试方法和工具与平台和语言无关。最后,还要考虑构件测试环境的配置、构件测试的可重用性和充分性、冗余测试问题,以及分布式系统中构件的竞争和死锁等等问题。

基于构件的软件系统的异质性(heterogeneity)和实现透明性(implementation transparency)使得用传统的测试技术进行测试有一定的困难,需要研究一些基于构件开发方法的测试技术。本文中,我们从构件生产者和使用者的角度分析构件本身的测试和构件集成软件的测试两个方面,通过对构件交互关系的图形建模,提出了一种基于CBD的软件的测试设计样式,给出了CBD软件的测试技术和方法。

本文第2节介绍相关背景;第3节从构件本身测试和构件集成软件测试两方面研究测试样式,并把重点放在构件软件的集成测试方面,提出一些基于构件开发方法的测试技术和方法,并结合一个实例说明这些方法的使用;最后给出结论并展望未来。

2　背景

对于构件的定义,已有很多的描述。Anders Hejlsberg认为构件是一个自包含的软件单元,它不仅是代码和数据,还包含提供关于构件如何适应特定宿主环境,如何持久化的动态附加信息。Szyperski和Pfister[2]指出,构件是一个契约上有具体接口,有明确的上下文依赖的组成单元,能够被独立配置且被第三方支配构造。对于CBD而言,构件远非模块化编程方法中的子程序、面向对象方法中的对象和类、或系统模型中的包。在CBD中,构件的概念既包含了这些思想又扩展了它们。这里采用一个相当广泛、全面的构件定义:

定义1　构件是一个独立发布的功能部分,可以通过它的接口访问它的服务。构件可以定义为一个五元组:{*spc*, *imp*, *dep*, *sta*, *pac*},其中*spc*为规格说明,*imp*为构件实现,*dep*表示可被部署,*sta*为构件标准,*pac*表示可被包装(如图1所示)。

构件规格说明建立在接口概念上,是关于构件提供的服务的抽象描述,以作为服务的客户方和提供方之间的契约;构件实现符合规格说明;软件构件存在于一个定义良好的环境(或称构件模型)中,必须遵守的一套规则,即构件标准,已有的构件标准包括Microsoft的COM+,Sun的Java Beans和Enterprise Java Beans(EJB),以及OMG的CORBA构件标准;构件可以按不同的方式分组来提供一套可以替换的服务,它可以包装;成品构件安装在一个运行环境中,它们就将被部署。

CBD方法是一种面向复用、基于构件开发的方法,通过有计划地集成现有的软件部分来进行软件开发,以"选择并集成"而不是"设计并构建"为指导思想,采用的是构件组装的方法。作为一种软件复用的体现方式,它在参照标准、质量控制、开发过程、CASE工具及技术观念等方面使软件工程的研究与实践产生了极大的更新和改进,对于软件系统的开发者和用户来说,能够降低开发费用、提高生产率以及在快速的技术

*)本课题得到重庆市自然科学基金资助(项目编号:CSTC-2004BB0146)。

演化面前提供受控的系统升级。CBD方法是基于在模块化系统、结构化设计以及最近的面向对象系统等方面的长期工作的基础上逐渐形成的，并扩展了这些概念，强调要根据构件的功能来设计解决方案，其他构件只能通过定义良好的接口来访问它，关注使用基于接口的设计技术来组装构件。

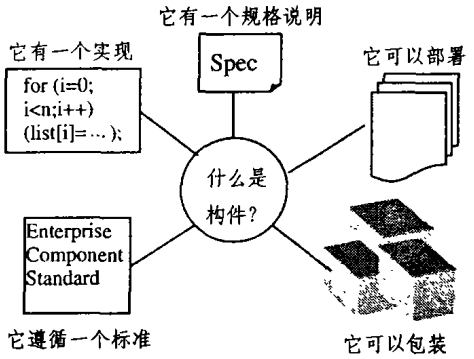


图1

3 基于构件开发方法的测试

基于构件的开发方法关注于解决方案的结构、确认应使用什么样的构件、对已有的构件进行选择、对构件加以包装和集成以及对最终解决方案的测试和部署等。它主张构件的规格说明与其设计和实现分离，重点关注于接口的设计方法。

基于CBD的以上特点，对测试CBD软件需要一些新的测试技术和方法，这些方法以现在已经成熟并大量使用的传统测试技术和面向对象的测试技术为基础，并对其进行相应的扩展和修改。

Harrold认为应该从构件生产者和构件使用者两个不同的角度来看待构件软件的测试问题[3]。构件生产者把构件看作是独立于使用构件的环境的，因此生产者用上下文独立的方式测试构件的所有功能。相反，构件使用者的应用程序提供了构件的运行环境，所以使用者不能把构件看作独立的单元，要考虑与应用程序相关的构件功能。另外，构件生产者拥有构件的源代码而构件的使用者通常都没有构件的源代码。

对于商业构件来说，构件的生产者和使用者很可能不同，这会导致一些问题：生产者对构件本身的测试不充分，使用者对此难以确定；生产者和使用者使用构件的环境不同，包括编程语言、操作系统、硬件平台等的不同；对于构件使用者，源代码很可能未知。这些问题也许是构件软件测试问题与传统的软件测试问题的最大区别。

3.1 构件测试

3.1.1 目标 充分测试构件，为构件使用者提供必要的测试信息，如构件规格说明，测试文档等。

3.1.2 上下文 构件生产者拥有构件的源代码，因此测试构件类似于传统的单元测试，但由于构件是独立发布的功能单元，通过接口和其他的构件进行交互，实现构件系统的组装，构件生产者一方面须把构件作为独立于使用它的上下文的单元来进行有效测试，使构件的使用者对构件质量充满信心，减少使用者对构件的测试负担。另一方面构件生产者还可能对构件以后将被使用的环境作一些正确和全面的假设，并在这种假设环境中对构件进行测试，对构件使用者提供有效的测试信息。

3.1.3 策略 构件生产者对构件的测试可以用传统的测试技术和方法。

对构件充分测试，为构件使用者提供测试规格说明和文档。在文[3]中给出了构件提供者测试构件并为使用者提供summary information的几种测试方法：程序分片法、控制依赖分析和数据流测试法等。

3.2 构件软件的测试

3.2.1 目标 集成测试由构件组装的软件系统，揭示构件的互操作性错误，确保构件的正确交互。

3.2.2 上下文 基于构件的软件是通过组装各种构件形成的，构件通过接口进行交互，接口概括地描述了构件应如何与其它构件进行交互，同时却隐藏了底层的实现细节。构件的使用者通常没有构件的源代码，但可以根据构件提供者提供的测试规格说明和文档信息得出接口和交互的相关信息，用构件规格说明构架对构件构架建模，通过对系统构件的相依性分析，建立系统的构件交互图，通过遍历该图测试构件软件，显示最低限度的互操作性。

3.2.3 策略

(1) 与交互特性相关的主要元素

接口：封装了执行服务的一个或多个方法函数，描述特定环境中构件的行为和责任，对使用者隐藏了实现细节，用符号"I"表示。

事件：对一个接口的调用产生一个接口事件，另外事件还包括异常处理事件和用户行为事件，用符号"E"表示。

控制依赖：事件 E_1 和 E_2 之间存在控制依赖关系，如果存在一条执行路径当触发 E_1 的时候将直接或间接触发 E_2；另外控制依赖还存在于接口和接口、接口和事件的关系中。测试控制依赖可以确定不同构件中的互操作错误。

数据依赖：构件接口的调用其结果最终是构件执行功能的调用，如果接口 I_1 中包含有功能 f_1，I_2 中包含有功能 f_2，f_1 依赖于 f_2，当且仅当 f_2 中定义的变量值在 f_1 中使用，我们就说接口 I_1 和 I_2 之间存在数据依赖关系。

(2)构件交互图

定义2 构件交互图 $G=(V,E)$，V 表示系统中的构件结点，构件接口和事件间的交互用一条边连接，以 E 表示。$V=VI\cup VE$，VI 代表构件接口，VE 代表由构件产生的事件。构件交互图中的一个路径或子路径是一个确定的有限序列$(v_0,v_1,\cdots,v_k)$，其中 $e_{i,i+1}=(v_i,v_{i+1})\in E, i=0,1,\cdots,k-1$。一个简单路径是路径$(v_0,v_1,\cdots,v_k)$，对于任意 i 和 j，$i\neq j$ 时 $v_i\neq v_j$。

在构件交互图中，接口和事件的交互关系可以直接得出。对于控制依赖和数据依赖关系在构件交互图中给出如下定义：

定义3 如果在图中存在一条简单路径 $v_0,v_1,\cdots,v_k$，其中 $v_i\in V, i=1,2,\cdots,k-1$。边 $edge=(v_i,v_{i+1})\in E$，就说两接口间或接口与事件间有控制依赖关系。

定义4 ①如果 I_1 调用 I_1，I_2 调用 I_2，且 I_1 依赖 I_2，则接口 I_1 和 I_2 有数据依赖关系。②如果 E 控制依赖于 I_1，并且 I 依赖于 I_1 或者 I_1 依赖于 I，那么接口 I 和事件 E 有数据依赖关系。③如果 E_1 控制依赖 I_1，E_2 控制依赖 I_2，并且 I_1 依赖于 I_2，那么事件 E_1 和事件 E_2 有数据依赖关系。

(3)测试过程。基于以上测试模型，我们得到对基于构件的软件进行测试的方法：

① 通过构件提供者提供的构件规格说明和测试信息，确定系统中的构件集合以及它们之间的物理依赖关系，得出构件规格说明构架。

② 建立构件交互图。 用构件规格说明构架得到构件的物理依赖关系后，还要进一步确定构件间的交互关系。我们可以根据在开发构件系统的过程中，分析和设计阶段的一些系统构架的分析模型和文档，对构件间交互关系进行相依性分析，分为直接相依性和间接相依性，直接相依性包括构件接口和事件的交互，间接相依性则包括它们之间的控制依赖和数

据依赖关系。建立构件交互图对构件间的交互建模，对测试基于构件的软件提供模型支持。

③测试系统构件的直接相依性。在构件交互图中，每个接口都要保证至少被测试一次；每个接口调用事件也必须被至少测试一次，其它的和接口调用无关但对构件有影响的事件也必须被测试。

④测试系统构件的间接相依性。在构件交互图中，确定构件间的控制依赖和数据依赖关系，选择测试用例。

对于控制依赖，当两个接口或者一个接口和一个事件间有控制依赖关系时，这种控制依赖可能由不同的场景提供，也就是说在构件交互图中可能有多于一条的简单路径存在于接口和事件中，我们有两种选择测试覆盖的方法，一种是仅仅只需要覆盖至少一条简单路径；另一种是要求覆盖所有的简单路径。理论上要求达到对所有简单路径的覆盖，但在现实中，由于达到此覆盖的测试用例数量相当大，因此一般采用至少覆盖一条简单路径。

对于数据依赖，由于构件的源代码不可知，我们只有通过近似假设的方法，一个完备的近似假设每个接口对都存在一个数据依赖关系，另外简单近似假设每个接口都不存在与其他接口的数据依赖，对数据依赖的测试要把构件交互图和其它的分析和设计文档还有具体的功能性描述等信息相结合。

(4)应用实例。我们通过一个银行系统的例子说明以上方法。

根据银行系统中包含的构件的规格说明和构件提供者提供的测试信息，我们得到图2所示的银行系统构件规格说明构架的简化实例。Security 构件包含并确定帐户和 PIN 号码的信息，Keypad 构件提供 PIN 号码和交易数量的输入，Transaction 构件处理所有银行事物类型的交易，SavingsSystem 构件提供数据管理服务，Account 构件包含帐户卡和现金服务。SavingsSystem 构件和系统其它构件都有交互，另外，Account 构件和 Security 构件、Transaction 构件之间有交互关系。

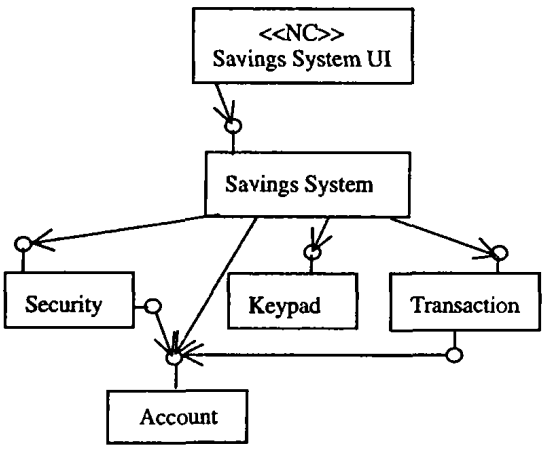
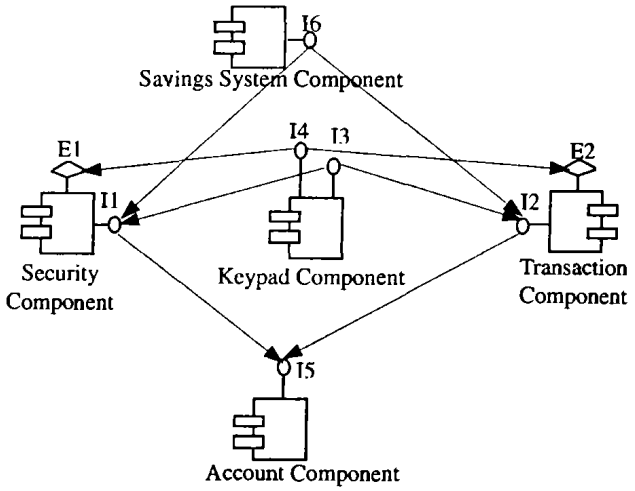


图2

由构件规格说明构架我们得到银行系统中的构件以及它们之间的物理依赖关系。对应构件规格说明构架我们建立构件交互图，通过构件交互图我们可以更进一步找出系统中的直接和间接依赖关系，为测试用例的选择提供有利的模型支持。

图3是与图2对应的构件交互图，Security 构件有接口 I-Validation，事件 Cancel，该事件将终止当前操作开始一个新操作。Transaction 构件有接口 I-Transaction，一个 Cancel 事件。Keypad 构件有两个接口 I-Init 和 I-GetValue，事件 Cancel。Savings System 构件有一个接口 I-SaveService。Account 构件有接口 I-Account。



(I_1:I-Validation；I_2:I-Transaction；I_3:I-Getvalue；I_4:I-Init；I_5:I-Accout；I_6:I-SaveService；E_1,E_2:E-Cancel)

图3

在构件交互图中，根据方法③我们对系统构件间的直接相依性进行测试。在图3中，以下用例覆盖了所有接口和事件。

Case1=$\langle I_4, E_1\rangle$

Case2=$\langle I_4, E_2\rangle$

Case3=$\langle I_5, I_1, I_3\rangle$

Case4=$\langle I_6, I_2\rangle$

在图3中，上面所列的用例 Case1、Case2、Case3、Case4覆盖了图中5个构件的所有接口和事件，但不能覆盖 V_6和 V_5间的控制依赖关系，我们可以加上两个用例覆盖控制依赖：

Case5=$\langle I_6, I_1, I_5\rangle$

Case6=$\langle I_6, I_2, I_5\rangle$

结论 在 CBD 方法的基础上，提出了测试 CBD 软件的测试样式，主要考虑构件集成软件的测试方面，通过构件规格说明构架和构件交互图对构件间的交互建模，对测试构件系统的互操作性错误提供支持。根据构件交互图得出测试构件系统的直接和间接相依性的方法。文中所举的例子只是一个简化了的实例，目的是对方法进行说明，在实际的开发测试过程中还需要更多更详细的分析。

目前构件软件的测试技术本质上还是借鉴传统的测试技术，根据构件的特点进行一些相关改进。在未来的工作中，我们应以方法研究为起点，结合实例研究和工具的开发，在测试技术和方法上发展和完善构件测试的理论基础，研究测试的充分性判据，解决跨平台、跨语言的测试问题。

参 考 文 献

1 Phipps D. Workgroup AD in the Next Century: Death and Rebirth, Gartner Group Research Note, Nov. 1999

2 Clement S. Component Software: Beyond Object-oriented Programing. Addison-Wesley, 1998

3 Harrold M J, Liang D, Sinha S. An Approach To Analyzing and Testing Component-Based System [C]. In: Proc. First Intl. ICSE Workshop, 1999. 134～140

4 Wu Y, Chen M H, Offutt J. UML-based Integration Testing for Component-based Software. http:// www. iccbss. org/2003/presentations/Wu_121305VS. pdf

5 Bhor A. Software Component Testing Strategies. http://www1. ics. uci. edu/～abhor/ics221/comp-test. htm

6 Liu C, Richardson D. Software Components with Retrospectors. http:// www. ics. uci. edu/～cliu1/research/ talks/ROSATEA-slides. ppt

7 Rosenblum D. Adequate testing of component-based software; [Technical Report TR97-34]. University of California at Irvine, 1997

8 景涛，白成刚，等. 构件软件的测试问题综述. 计算机工程与应用，2002，24

9 Binder R V 著. 华庆一，等译. 面向对象系统的测试. 人民邮电出版社，2001

10 Brown A W 著. 赵文耘，张志，等译. 大规模基于构件的软件开发. 机械工业出版社，2003