维普资讯 http://www.cqvip.com

计算机科学2004Vol. 31№. 12

测试面向对象系统的配置管理

朱庆生　罗　佳
（重庆大学计算机学院　重庆400044）

摘　要　论文提出一种在面向对象软件系统测试中的配置管理模式。该模式在充分考虑了面向对象系统特征的前提下，对测试的不同阶段的配置管理进行了详细的阐述。首先保证对测试配置项进行有效的管理，通过对配置项的管理，即通过对类、对象、界面的管理来分析和搜索测试路径，达到完全覆盖所有测试路径。其次，探讨了如何在配置管理中处理测试时的变动和并发现象。

关键词　配置管理，软件测试，面向对象

Configuration Management for Testing Object-Oriented System

ZHU Qing-Sheng　LUO Jia
(College of Computer Science, Chongqing University, Chongqing 400044)

Abstract　This paper proposes a software configuration management model used in the object-oriented software test. Considering the object-oriented feture, we expound the configuration management method at each test stage in the model. The model firstly ensures an effective management in the test configuration. Making use of management of class, object, and interface, we use the configuration management to analyze the test path and try to cover all possible test paths. Then the authors discuss how to deal with changes and concurrent phenomenon in the configuration management.

Keywords　Configuration management, Software test, Object-oriented

1　引言

众所周知，软件开发在经历了几十年的发展以后，现在的软件开发已经不再是昔日的作坊式的产出。随着软件规模不断扩大，软件设计的复杂程度不断提高，软件开发已由最初的“程序设计阶段”经历了“程序系统阶段”进而演变为当前的“软件工程阶段”。目前广泛采用的面向对象技术是一种全新的软件开发技术，正逐渐代替被广泛使用的面向过程开发方法，被看成是解决软件危机的新兴技术。面向对象的类、继承、封装和多态思想，使程序具有很好的可重用性。但它同时也是一把双刃剑，程序的错误也可能随着软件代码的重用率，造成错误的繁衍。为了保证软件开发的质量，我们需要对软件开发进行严格的测试。就管理而言，相应的配置管理不仅可以对测试的过程进行严格的规范性管理来提高测试的效率和配置项的管理，而且可以通过配置管理对测试工作进行一定的支持来保证测试的完备性。下面我们就针对面向对象系统的特点对软件测试中的配置管理进行探讨。

2　面向对象系统的软件测试对配置管理技术的需求

面向对象系统的测试配置管理所涉及到的配置项有测试规格说明、测试计划和步骤、测试用例、测试结果。与传统的面向过程的测试配置计划不同，面向对象的测试配置计划更着重对类和图形界面的管理。测试配置计划中应明确对类和界面相关概念和存储模式的配置管理办法、测试的工具、测试的工作环境要求、测试的步骤和对测试结果的保存办法等[1]，具体来看在对测试的配置管理工作中，主要需要进行下列工作：

① 确定配置项。测试中的配置项有测试对象、测试路径、测试用例、相应的预期测试结果，然后将其保存到配置库中。首先要确定测试对象，在系统中要明确记录各个测试项的配置管理状态和管理权限，这样在测试的过程中才能清楚地识别未测试的项，以及明确谁可以对其进行检出。由于类的继承机制，错误可能沿着类的继承而向下传递，因此对类这个测试配置项的保存，要设计合理的存储方式，要能反映对象之间的继承关系和触发关系，使之可以追根索源，在变动时可以保证所有相关的对象都能随之变动。然后在此基础上根据类和图形界面的配置项来制定测试路径，再根据测试路径设计测试用例，最后对每个测试用例分析预期的测试结果。

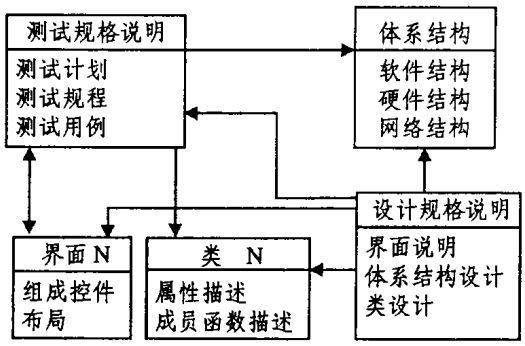
②测试的时候可能出现多个人使用同一模块的情况。配置管理要注重对测试过程中可能涉及到的这种并发情况进行控制。对测试过程进行控制，主要是对测试中的并发现象进行管理。

③ 根据测试中出现的错误或发现的缺陷进行变更。测试的结果可能引起测试对象的变动。配置管理应能够有效地维护其演化历史。软件测试过程与整个软件开发过程基本上是平行进行的，测试过程其实质是设计过程和实现过程不断交互的过程，由于测试中发现的问题往往是在系统分析或设计时就已经产生了，当然更多的是在编码阶段产生的。因此如果软件在测试时发现了问题，很有可能对原来的设计进行修正[2]，如图1所示，如果是设计之初的问题，我们需要变动设计规格说明，设计规格一旦变化，则相应的必须修改类结构，

朱庆生　教授，博导，研究方向为多媒体技术、电子商务、软件工程。罗　佳　研究生，研究方向为电子商务。

维普资讯 http://www.cqvip.com

界面，甚至可能修改整个体系结构，同时测试计划和用例等都会随着做出调整。这都要求配置管理对其进行详细的记载和管理，对版本的变动和变动的审核进行管理。防止死锁和修改被回写的情况。



单箭头表示属于关系，双箭头表示交互关系

图1 面向对象系统的配置对象图

本文针对测试的各个阶段提出了面向对象的测试配置管理系统模型，提出了在测试过程中如何解决测试配置项的分析和管理、配置项并发控制管理、变更管理等问题。下面我们就面向对象系统的特点对各个不同阶段分别进行阐述。

3 单元测试的配置管理

面向对象系统的单元测试主要是对类的测试，而对类的测试又以对类的实现，即对象的测试。如前所述由于类的继承特性，类的错误可能随着类的继承而被延伸。因此首先要保证对所有类的子类以及子类的子类进行测试。我们在配置管理中可以对类之间的继承关系建立它们的数据表来进行管理。(如下图2所示)在表中我们可以记录每个类的所有子类，还可以记录这些类的一些相关信息，如类的一般性描述，功能说明，在哪些程序中使用了这些类，以及类的成员函数是完全继承父类的成员函数，还是对成员函数有所改变，对于这方面我们可以采用标志位的方法来解决。

对类，也即测试配置项的命名中我们采用对类分级的命名方法，这样我们可以很清晰地明确类之间的继承关系。当我们建立了这样的一系列表以后。我们发现类之间的继承关系实际上就形成了一个向下生长的树，我们对这些类之间的测试实际上就变成了对这棵树的遍历。通过这种配置管理方式，我们可以保证对类的所有子类的测试，避免有些类的漏测。而对子类的测试配置首先是要识别哪些子类需要重新测试。这里我们分情况对待，对父类中已经测试过的成员函数，有两种情况需要在子类中重新测试：①继承的成员函数在子类中做了改动；②成员函数调用了改动过的成员函数的部分。由于已经在上述表中记录了相关信息，所以其实现是可行的。

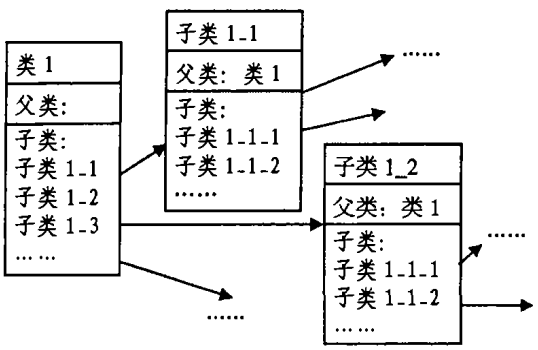


图2 类之间的继承关系表

我们都知道对类的测试实际上是体现在对类的实现，也即是对象的测试。而对象的实现都体现在各个界面或是全局函数中。由于类的复用性，类的实现可能出现在多个不同的地方，为了测试这样的类的实现可能出两个问题，一是当发现有错误的类时，如何对所有相关程序进行调整，如果没有有力的配置管理办法，则可能不能将这种缺陷有效的通知相关程序。二是在测试时可能出现其他开发成员正在使用该类的实现，当我们完成了对类的修改后，其他成员又进行了保存，使前面的修改失效，我们如何解决这种操作中的并发性问题。

对于第一个问题，我们仍然采用图2中的相关记录，当我们发现类的缺陷后，马上根据它在表中记录的所有使用过它的程序进行修改，然后又对它的所有子类的所有实现进行修改，同时进行版本更新。对于第二个问题，我们采用图3中的配置管理办法对类的测试进行管理。当我们对类进行测试时，我们对它进行共享性的锁用，此时不允许其他开发人员对该类进行修改功能，这样既保证了对它的测试不会影响其他开发人员的开发工作，又保证了当发现该类有缺陷时，不会出现其他开发人员对类的修改进行回写。在测试完成后，如果类有缺陷，我们立即对该类由共享锁升级为独占锁，对该类修改结束后再解开锁。

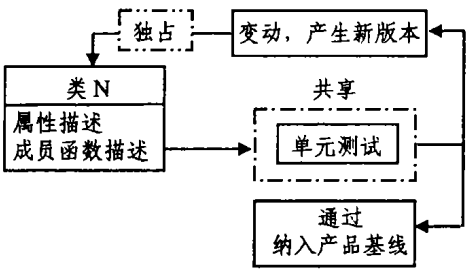


图3 类的测试管理流程

4 页面测试的配置管理

页面测试的目的是寻找构成页面的可视对象及其消息传递的错误，以保证每个页面作为一个单元能正确运行。面向对象的页面测试是根据对象的事件驱动设计的测试方案，一般先进行考查页面的布局和风格是否和设计要求一致，如果有出入就需要对其进行变更。然后是对各个页面的功能实现进行测试，由于已经进行了可视对象的单元测试，因此页面测试着重考察对象间消息的传递和各个类并行时的情况。则对页面的测试配置管理就体现在两个方面，一是利用配置系统中的配置项，根据页面上对象之间的消息触发队列制定测试方案。在制定测试方案时，应将所有的消息和消息传递途径考虑进去，避免出现漏测的情况，这就要求对配置系统中合理的设计配置项的保存形式。二是当出现错误或缺陷时如何进行变更管理。识别对哪些配置项，如何对这些配置项进行变更，同时如何处理并发情况。

我们在测试之前先根据页面中的对象之间的触发关系在配置系统中建立一张能反映这种关系的表(如表1所示)来，在表中记录页面中各个对象可能引起的触发，然后我们对这张表根据触发与被触发的关系进行遍历就可以得到测试所需的该页面的所有消息队列。比如在表1中，我们以 A 出发，由 A 触发 C，再由 C 触发 E，最后由 E 触发 F。现假设该表就此结束，那么这条消息队列即为 A→C→E→F。然后按照这种方法将所有可能的消息队列找出来。在上述工作完成以后，我们再

维普资讯 http://www.cqvip.com

对这些消息队列来制定相应的测试方案和测试用例，将其纳入配置管理的范畴。

表1 对象触发关系图

引起触发的对象	被触发的对象1	被触发的对象2	被触发的对象3	……
A	C	F	K	
C	B	E		
E	F	K	D	
……				

在页面测试中当有错误或缺陷出现时，首先应保存测试结果和错误情况说明，然后分析错误发生的原因。如果是由于页面内的对象间消息传递发生的错误，则我们可以采用类似于单元测试中的变更管理办法进行配置管理；如果错误是由单元测试时没有发现的错误，则应按照在单元测试阶段的配置管理办法进行管理；如果发现该类的错误是由于继承父类中的成员而产生，则应反向找到产生该成员的父类，进行修改，然后同样还需对其子类进行修改。

5 集成测试的配置管理

集成测试是在完成页面测试的基础上，把页面按照系统设计要求组装起来作为一个整体来进行的测试，目的是发现页面之间与消息传递有关的错误。这样的错误可能不仅仅有编码的错误，还可能有设计错误和需求分析错误。因此在对集成测试进行配置管理时，有两个问题是必须要解决的，一是确定配置项，保证系统所有的页面进行测试，对这些页面如何进行组织，二是当出现变更时，由于此时的变更可能涉及到对早期的分析和设计进行变动，因此如何对此时的变更进行管理是我们需要考虑的问题。

目前软件系统越来越庞大，结构也越来越复杂。系统大多页面较多，关联消息较复杂，所以我们采用分组测试的方法。这里所用的分组的方法和上述页面测试中对测试路径的分析方法类似，但是由于面向对象方法设计的系统采用事件驱动的机制，在集成测试里是页面之间的相互调用，可能出现如图4中所示的情况，图4中是以1为出发点的一个有向图，在这张图中可能出现环路现象。对于这样的情况，我们仍按照页面测试中的办法来制定测试路径(即分组)。先在配置系统中就配置项(图形界面)之间的调用关系建立表，然后根据表中的调用关系遍历。当出现某个页面已在测试路径中出现过时，我们的路径分析就此结束。比如：1→2→3→4→8→3，因为3已在前面的测试路径已经出现过，路径分析停止。

在明确了测试路径后，接下来就是针对测试路径设计测试用例并分析预期的测试结果。当在测试的过程中出现的问题。如前所述，现阶段的变更大多将涉及到早期的设计和分析。众所周知，此时的变更将引起软件的很大修改，而且工作量是早期阶段变更所需工作量的倍数级。如图1中所示，就配置管理而言，首先根据变更要求修订设计规格说明，然后将设计规格说明的改动具体到体系结构，类或相应的界面中，相当于重新进行了一次分析和设计。完成此步工作后，才能重新考虑设计测试方案等，也即是对测试规格说明进行修改。

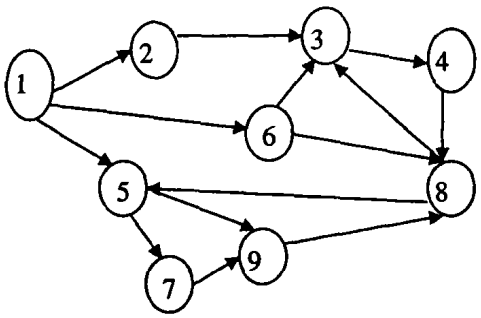


图4 页面之间的调用关系图

6 系统测试的配置管理

系统测试时，配置管理应该尽量搭建与用户实际使用环境相同的测试平台，配置测试环境是系统测试实施的一个重要阶段，测试环境适合与否会严重影响测试结果的真实性和正确性。测试环境包括硬件环境和软件环境，硬件环境指测试必需的服务器、客户端、网络连接设备，以及打印机/扫描仪等辅助硬件设备所构成的环境；软件环境指被测软件运行时的操作系统、数据库及其他应用软件构成的环境。应该保证被测系统的完整性，对临时没有的系统设备部件，也应有相应的模拟手段。对于完成了系统测试的产品，即可对其纳入产品基线，分配版本号。

结束语 本文就测试中的相关配置问题进行了分别阐述，使配置管理不仅做到对测试中配置项进行应有的保存，而且还对测试的有关工作提供了支持。当出现需要对配置项进行变动时，能很好地保证所有相关的配置项同时得到变动，此外还注意处理变动中的并发情况。由于配置工作是一项管理工作，所以需要得到开发单位提供行政组织上的支持，才能使配置工作有效展开。

参 考 文 献

1 计算机软件配置管理规范. GB/T12505-90

2 罗晓沛，侯炳辉主编. 系统分析员教程. 清华大学出版社，2003，6：304 ～ 332