软件危机及其应对措施探讨

杨健,张晓玲,周少云
(大理学院 数学与计算机学院,云南 大理 671003)

摘要:软件危机几乎一直伴随着软件开发的整个历史,对于大型软件项目,它造成的危害更大。正是在此背景下,软件工程和面向对象技术得以产生和发展。作为面向对象的基础,设计模式为可复用面向对象软件开发指明了一条道路。敏捷软件开发和极限编程则作为面向对象的进一步发展而出现,成为近年来软件开发方法学研究的热点。

关键词:软件危机;面向对象;设计模式;敏捷软件开发;极限编程

中图分类号:TP311　　文献标识码:A　　文章编号:1009-3044(2006)35-0135-02

Software Crisis and It's Solution Measures

YANG Jian, ZHANG Xiao-ling,ZHOU Shao-yun
(Faculty of Mathematics and Computer,Dali University,Dali 671003,China)

Abstract:Software crisis nearly continuously is following the entire history of computer software developments. Regarding the large-scale software project, the harm which it creates is bigger. Under this background, the software engineering and the object-oriented technology came into existence and developed. As the foundation of the object-oriented technology, the design pattern has indicated a path for the object-oriented reusable software development. The agile software development and the extreme programming appeared as the further development of the object-oriented technology, becomes a hot spot of software development methodology research in the recent years.

Key words:Software crisis. Object-oriented, Design pattern, Agile software development, Extreme programming

1 引言

“软件危机”几乎从计算机诞生的那一天起就出现在软件开发人员的面前。它主要表现为:软件的性能跟不上硬件发展的速度;软件开发过程的研究与实际应用需求不相适应;在大型软件项目开发面前,人们力不从心,导致预算超支,软件交货时间一再拖延。从软件本身质量来说,大型软件难以维护,可移植性和可复用性差。

从软件本身的特性和软件危机的各种表现中,不难发现其产生的原因[1]:一是最终用户不能够在需求分析阶段就完整精确地描述软件需求,往往要反复改动;二是软件开发人员与最终用户不能很好地进行沟通,因而产生需求与实现的差异;三是软件开发企业缺乏对大型软件开发进行有效管理的经验;四是软件开发人员以及各个开发小组之间存在沟通和协作的障碍;五是缺乏有力的方法学指导和开发工具的支持。在此背景下,人们开始探索用工程的方法管理软件开发,通过在软件开发方法、工具、管理等方面的应用,大大缓解了软件危机造成的被动局面。

从另一个角度考虑,用软件来解决问题,需要把人类的思维模式和处理问题的方法转换为计算机语言表示。如果计算机能够直接表现求解问题的方法,并以人类思维方式建立问题域模型,则软件不仅易于理解,而且易于维护和修改,提高了软件模块化和重用化的可能性【2】。面向对象(Object-Oriented, OO)技术正是在这样一种背景下产生和发展起来的。

2 面向对象技术

OO包括分析(OOA)、设计(OOD)和编程(OOP)三个部分。它不仅包含程序设计的方法论指导,还贯穿于软件开发的整个过程。OO技术认为:客观世界由各种对象组成,对象内部有表示其状态的属性,对象之间有相互作用和联系,形成一个互动整体。而OOA和OOD方法需要将现实世界中的对象映射为问题域中的对象,在解空间中直接模拟问题空间中的对象及其行为,通过对象的互动而得到问题的解。

OO方法学中包含了以下核心的概念:

对象:对象是现实世界中个体或事物的抽象表示,是表示内部状态的属性和对外部提供服务的相关操作的封装。通过对象提供的服务,与其他对象相互作用。

类:类是某些对象共同特征(属性和操作)的表示,是一类事物共有的本质特征的抽象集合。类的概念使我们能够对属于该类的全部事物进行统一描述,而不需对每个具体的事物进行重复的说明。

在面向对象编程语言中,类是一个独立的程序单位,其作用是定义对象,而对象则是一个类的实例。对象和类的关系类似于传统程序设计语言中的变量和数据类型的关系。

继承:类之间的继承关系是现实世界中遗传关系的直接模拟。子类通过继承,拥有父类的属性和操作。子类也可以拥有自己的属性和操作,可以通过覆盖,使得继承的操作呈现出自有的特性,这种特性也被称为多态性。可以说,正是因为多态,才使得继承不再是简单的传承,使得系统有了多样化发展的可能。

消息:消息是向对象发出的服务请求。消息机制为对象之间提供了一个相互动态联系的途径,使它们的行为相互配合,构成一个有机的运动系统。

从上面的概念中可以总结出面向对象的三大特征:封装、继承和多态。面向对象编程语言和设计工具通过不同方式实现这些核心概念,体现这三大特征,完成问题的求解。

3 面向对象程序设计中的接口

与传统软件开发技术相比,面向对象方法能够在软件的模块化、可维护性和可复用性等方面拥有较大优势,并由此提高软件开发人员的效率和软件质量。

面向对象的模块化,不仅仅是功能的划分,而且是数据与功能聚合体的划分。通过这种划分,对象与对象之间聚合的粒度降到合适水准,使得一个对象的修改对另外对象的影响降到最小。面向对象的维护,不是简单的修改源代码,而是通过继承的修改。这种修改,通过采用一定的设计方法,可以将其对已有代码的影响降至最小。面向对象的复用不是对代码的拷贝,也不是对已编译的动态链接库的调用,而是通过继承来复用父类或祖先类的服务。

然而,要实现上述的模块化,代码维护和复用特性,就必须用到接口的概念。

对象操作所定义的所有操作型构(每个操作的名称、作为参数的对象和返回值)的集合被称为该对象的接口。接口描述了对象所能接受的全部请求的集合。接口的功能同类一样,但并不要求必须存在一种父子关系。接口也是引用类型,可以用作形式参数,而具有同样功能的类(实现了该接口)将用作实际参数。类与接口的关联关系是,如果一个类实现了某个接口,则这个类必须

收稿日期:2006-10-08

提供接口中所规定的所有方法【3】。

下面就代码的可维护性来描述接口的作用。假设有一个Transport类，描述交通工具的抽象概念，它有3个子类：car、bike、airplane，分别代表汽车、自行车和飞机。现在假设需要在这个类的层次结构中保存加油的信息，因而需要增加一个refuel()方法，为每一种机动车辆加油。如果在Transport中定义该方法，显然不合适，因为bike不需要加油。在具体的机动车的类中增加该方法也不合适，因为如果有新的需要加油的交通工具出现，则对于提供加油服务的类（例如定义一个油库类提供该服务）来说，必须增加对该种交通工具的判断，也就是说，要修改其他类（油库类）。

如何实现这样的功能而不引起其他类的修改？我们可以定义一个可加油接口。油库类对实现了该接口的类提供服务，如果增加新的交通工具，只要新类实现可加油接口，就可以享用油库类的加油服务了。从这里我们可以看到，可加油接口对于提高这个类的可维护性有着不可替代的作用。

4 在面向对象技术应用设计模式

设计可复用的OO软件比较复杂。人们必须找到相关的对象，以适当的粒度将它们归类，再定义类的接口和继承层次，建立对象之间的基本关系。设计模式使人们可以更加简单方便地复用成功的设计和体系结构，建立起可复用的，更加便于维护的面向对象的应用系统代码。可以说，设计模式是面向对象软件的设计经验的总结。每一个设计模式都集中于一个特定的OO设计问题和设计要点，描述了它的使用条件和约束条件，使用的效果以及如何取舍。

典型的设计模式共有23个[4]，分别从对象的创建，类和对象的组合结构的形成，以及算法实现和对象间职责的分配等三个角度进行了划分。这些设计模式为可复用的面向对象软件开发技术作了有效的补充，它展示了如何使用诸如对象、继承和多态等基本技术，也展示了如何使用各种技术解决设计问题的方法。

设计模式在将一个分析模型转换为一个实现模型时特别有用。然而，一个成熟的设计方法不仅仅要有设计模式，还可有其他类型的模式，如分析模式、用户界面设计模式或者性能调节模式等等。但设计模式是其中最主要的部分。

5 敏捷软件开发和极限编程

为了应对软件危机所造成的被动，软件开发人员在各个方面进行改进，包括采用工程管理的方法来进行软件开发过程控制。面向对象方法增强了软件的可维护性、可复用性，提高了软件开发的效率，从而成为当前主流的软件开发技术。

然而，使用一些固定的规范和人为约定，并不能保证软件项目的开发照着预想的方向发展。很多时候，采用庞大、重型的过程控制方法，反而会产生它本来期望解决的问题[5]。这种过程控制降低了团队的开发效率，使得进度延期，预算超支，严重的会导致项目失败。针对这种现象，敏捷软件开发应运而生。在敏捷软件开发中，不再过多强调过程控制和工具，而是强调开发人员个体及交互的作用；不再强调面面俱到的文档，而是追求可以工作的软件；不再进行合同谈判来避免客户需求改变所带来的损失，而是与客户进行紧密的合作；不再严格遵循开发之前订立的计划，而是随时响应来自客户和其他方面的变化。

在敏捷软件开发技术中，对面向对象设计进行了一些原则性的规范说明，并采用极限编程实践代替了完全以工程化的方法对软件开发的控制。在极限编程实践中，强调短周期计划，简单设计和结队编程，以测试用例推动开发，持续进行整个系统的集成和测试。与此同时，团队中的每个程序员都具有集体代码所有权，能够在任何时候改进任何代码。这种方式，看起来是对传统软件工程方法的一个颠覆，然而，它通过遵循有效的面向对象设计原则，采用多种设计模式，并充分调动每一个参与开发的人员（包括最终的客户）的智慧，最大限度地增强了客户与开发人员的沟通，从而，有效保证了软件开发的效率和质量。可以说，敏捷软件开发为有效缓解软件危机开创了一个新局面。

参考文献：

[1]齐治昌，谭庆平，宁洪.软件工程[M].北京：高等教育出版社，2001. 8-9，142-143.

[2]杨庚，王汝传.面向对象程序设计与C++语言[M].北京：人民邮电出版社，2002. 2-18.

[3]Peter van der Linden,著，邢国庆，等著译.Java 2 教程[M].北京：电子工业出版社，2005.143-146.

[4]Erich Gamma,等著，李英军，等译.设计模式-可复用面向对象软件的基础[M].北京：机械工业出版社，2000. 8-13.

[5]Robert C. Martin,著，邓辉，译.敏捷软件开发-原则、模式与实践[M].北京：清华大学出版社，2003.2-38，79-87.

（上接第111页）

配置时需要指定IP地址类型为IPV6，配置源IP地址或目的IP地址要符合IPv6规范，如1:1::1:1。

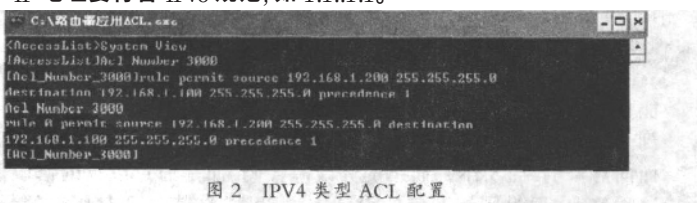

图2　IPV4类型ACL配置

配置IPV6类型ACL步骤等同于配置IPV4类型ACL，实体规则格式大致如下：

rule [rule id] {permit | deny} [source IPv6-source-address IPv6-source-mask] [IPv6-destination-address IPv6-destination-mask] [source-port port-number] [destination-port port number] [precedence value] [time-range time-limited-name] [logging] [fragment]

具体配置如图3所示。

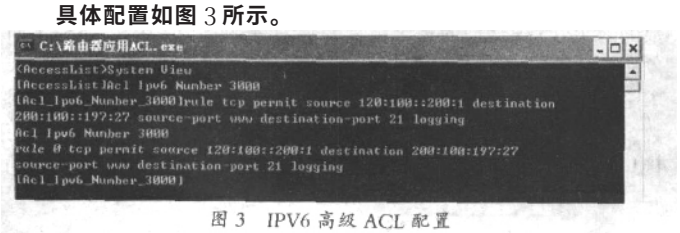

图3　IPV6高级ACL配置

4 配置操作系统ACL

4.1 Windows

windows 2000以上提供ACL组件，服务于交换服务器。Windows支持QOS的要求：首先，不是所有的网卡都能配置成功，网卡必须支持802.1p；其次，必须是域控制器，而且需要安装QOS Packet Scheduler；与此同时，支持操作系统.NET Server或以上版本。

4.2 Linux

Grsecurity是众多Linux ACL系统中比较典型和成熟的一种。grsecurity ACL规则由主题进程和对象组成。主题进程是被执行的进程；对象是文件、资源、能力(capability)和对IP的访问控制。

具体操作请查阅相关文档，在此不详细论述。

5 结束语

ACL技术的应用给QOS的实施带来了便利，能更快捷地配置网络环境和操作系统，达到理想的网络应用和对网络管理的要求。本文是对ACL技术的应用和探讨进行了大体的介绍。主要描写了ACL的分类以及在路由器和操作系统上的配置实现，并且，在现实网络环境下进行过具体的实践操作，并取得预定的结果。

参考文献：

[1] 陈绪，魏琼. 网络层访问权限控制技术[J].http://www.linuxmine.com, 2005(8).