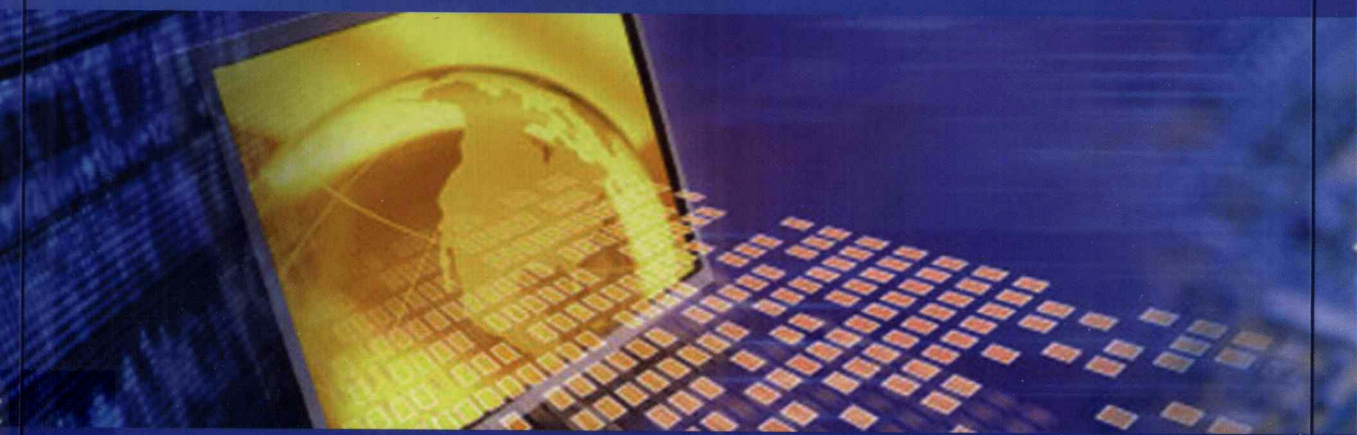
Software Testing

软件测试技术与应用

何月顺 著



中国水利水电出版社
www.waterpub.com.cn

软件测试技术与应用

何月顺　著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书主要介绍软件测试技术，分四大部分：理论篇、设计篇、技术篇和扩展篇。理论篇主要介绍软件测试基础知识，目的是让读者对软件测试有一个初步的了解，了解软件测试的重要性；设计篇主要介绍如何编写测试计划、测试方案、设计黑盒和白盒测试用例，重点阐述在编写测试计划和测试方案时需要注意的问题，以及如何设计测试用例；技术篇主要介绍系统测试过程中其他相关测试技术，包括 Web 测试技术、本地化与国际化测试、兼容性测试和易用性测试；扩展篇主要介绍当前流行的性能测试和自动化测试，通过实例介绍测试的过程。

本书由浅入深、由理论到实践，详细描述测试过程中每个阶段需要注意的地方，如缺陷分析方法，这是改进测试和质量控制的重要手段。希望帮助初学者了解软件测试的过程和相应技术，对软件测试有一个整体的了解；同时也可以帮助中高级测试工程师进一步提高软件测试的技能。

本书适合于软件测试的初学者，同时也适合于一些中高级测试工程师。

图书在版编目（CIP）数据

软件测试技术与应用 / 何月顺著. -- 北京 : 中国水利水电出版社, 2012.7
ISBN 978-7-5084-9831-7

Ⅰ. ①软… Ⅱ. ①何… Ⅲ. ①软件－测试 Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2012)第116655号

策划编辑：陈宏华　责任编辑：李　炎　加工编辑：郭　赏　封面设计：李　佳

书　　名	软件测试技术与应用
作　　者	何月顺　著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（发行部）、82562819（万水）
经　　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	三河市铭浩彩色印装有限公司
规　　格	184mm×260mm　16 开本　19.5 印张　479 千字
版　　次	2012 年 7 月第 1 版　2012 年 7 月第 1 次印刷
印　　数	0001—3000 册
定　　价	38.00 元

前　言

软件测试——目前国内最炙手可热的IT职位，从相关的招聘网站中不难发现，公司对软件测试工程师的需求不断增长。近年来国内软件测试也在迅速发展，在一些软件发展比较快的城市，很多公司都有专门的测试团队。不过现在大多数企业主要是进行黑盒测试。

在与一些朋友和公司交流时，发现两种现象：一是现在很多公司在实施黑盒测试时遇到一定的瓶颈，公司有一套完整的流程，测试过程看似没有问题，但测试质量却没有得到进一步的提高；二是一些朋友做了几年黑盒测试后，能顺利完成测试计划、测试方案和测试用例，感觉黑盒测试没什么需要学习的，但同时又明显感到自己一些东西没有学透，技能也遇到瓶颈。

不管是从公司角度还是个人角度来看，以上两种情况是普遍存在的。其根本原因是未认真分析软件测试的准出条件，在软件测试过程中并不是按要求完成测试计划、测试方案和测试用例，最后执行没有严重问题就认为软件是可以发布的。如关于缺陷，大多数朋友都知道缺陷的处理过程，如何描述缺陷以及在评审过程中关注未解决的严重问题，但是如果这样测试完成后，并且评审过程严重问题都已解决，就说明软件测试可以发布了吗？其实这个时候大多数软件测试工程师可能自己都不确定，自己还在问自己软件真的可以发布了吗？会不会有严重问题遗漏了没有测试到啊？

而这些正是本书需要帮助大家解决的问题，本书由浅入深、由理论到实践，详细介绍软件测试过程中的每个步骤，并且重点介绍测试过程中容易忽视的细节，扩展缺陷分析和其他测试（如文档测试、可安装性测试等）的重要性。

本书分为理论篇、设计篇、技术篇和扩展篇。

理论篇的主要内容有：软件测试的发展、缺陷的引入、修改缺陷的成本、测试成本以及测试工程师的职责和心态，系统生命周期中的测试策略，以及测试的几种模型，软件测试组织的发展，通过介绍软件测试组织的发展来找到自己在软件测试行业中的职业发展。

设计篇的主要内容有：如何编写测试计划和测试策略，结合实际，详细介绍测试用例的设计方法，从功能测试和单元测试两方面介绍测试用例的设计方法，测试用例设计是测试过程中的重要步骤，直接影响着软件测试的质量，这是很核心的一部分内容，而测试用例不仅仅需要设计还需要管理和维护。同时，还介绍测试过程中如何对发现的缺陷进行管理和分析，通过分析缺陷来改善测试流程。

技术篇的主要内容有：Web 测试、本地化与国际化测试、兼容性测试和易用性测试，在Web测试中详细介绍关于 Web 安全性的测试。

扩展篇的主要内容有：介绍常见的功能测试外的其他测试技术，主要介绍性能测试和自动化测试，并且通过案例详细介绍测试过程；接着介绍验收测试和文档测试；最后介绍如何制定自己的职业规划。

由于笔者水平有限，出现错误在所难免，欢迎广大读者批评指正，同时感谢曾经帮助、支持和鼓励过我的朋友。如有任何问题，可以发邮件到 arivnhuang@163.com，作者将尽力答疑解惑。

作者

2012 年 3 月

目　录

第三部分 技术篇

第四部分 扩展篇

第一部分　理论篇

理论篇主要包括三个章节内容，主要对软件测试的基础知识进行详细介绍，目的是让读者从整体上了解软件测试。第一章介绍软件测试的发展、缺陷的引入、修改缺陷的成本、测试成本以及测试工程师的职责和心态；第二章介绍系统生命周期中的测试策略，软件测试的几种模型；第三章介绍软件测试组织的发展，通过介绍软件测试组织的发展找到个人在软件测试行业中的职业发展。

第 1 章　软件测试概述

在讲述后面的章节之前，有必要对软件测试的基础知识进行介绍。本章主要介绍软件测试的基础知识，希望通过本章节的学习，读者可以对软件测试的定义、缺陷的引入等基础知识有一个大概的了解。

本章主要包括以下内容：

- 软件测试发展历史
- 缺陷引入
- 历史教训
- 修复缺陷的成本
- 测试付出的成本

1.1　软件测试发展历史

早期并没有软件测试这个概念，直到 20 世纪 60 年代（软件工程建立前），为证明程序设计的正确性而进行了相关的测试。

1972 年，在北卡罗来纳大学举行了首届软件测试正式会议。

1975 年，John Good Enough 和 Susan Gerhart 在 IEEE 上发表了文章《测试数据选择的原理》，软件测试被确定为一种研究方向。

1979 年，Glenford Myers 在《软件测试艺术》中，对测试做了定义：测试是为发现错误而执行的一个程序或者系统的过程。

20 世纪 80 年代早期，“质量”的号角开始吹响。软件测试定义发生了改变，测试不单纯是一个发现错误的过程，而且包含软件质量评价的内容，制定了各类标准。

1983 年，Bill Hetzel 在《软件测试完全指南》中指出：测试是以评价一个程序或者系统属性为目标的任何一种活动，测试是对软件质量的度量。

20 世纪 90 年代，测试工具盛行起来。现阶段的测试工具主要有两种来源：开源测试工具和商业测试工具。

（1）开源测试管理工具主要有：Bugzilla、Bugfree、TestLink、Mantis 等。

（2）开源自动化测试工具主要有：Watir、Selenium、MaxQ、WebInject 等。

（3）开源性能测试工具主要有：JMeter、OpenSTA、DBMonster、TPTEST、Web Application Load Simulator 等。

（4）商业测试工具主要包括以下几种：

①TestDirector：全球最大的软件测试工具提供商 Mercury Interactive 公司生产的企业级测试管理工具，也是业界第一个基于 Web 的测试管理系统，它可以在公司内部或外部进行全球范围内测试的管理。通过在一个整体的应用系统中集成了测试管理的各个部分，包括需求管理、测试计划、测试执行以及错误跟踪等功能，TestDirector 极大地加速了测试过程。

②Quality Center：基于 Web 的测试管理工具，可以组织和管理应用程序测试流程的所有阶段，包括指定测试需求、计划测试、执行测试和跟踪缺陷。此外，通过 Quality Center 还可以创建报告和图来监控测试流程。合理使用 Quality Center 可以提高测试的工作效率，节省时间，达到事半功倍的效果。

③QuickTest Professional：HP QuickTest Professional 针对功能测试和回归测试自动化提供业界最佳的解决方案，适用于软件主要应用环境的功能测试和回归测试的自动化。采用关键字驱动的理念来简化对测试用例的创建和维护。它让用户可以直接录制屏幕上的操作流程，自动生成功能测试或回归测试脚本。专业的测试者也可以通过其提供的内置脚本和调试环境来取得对测试对象属性的完全控制。

④LoadRunner：一种预测系统行为和性能的负载测试工具。以模拟上千万用户并发负载并实时监测系统性能的方式来确认和查找问题。LoadRunner 能够对整个企业架构进行测试。通过使用 LoadRunner，企业能最大限度地缩短测试时间，优化性能和加速应用系统的发布周期。

其他工具与自动化测试框架还有：Rational Functional Tester、Borland Silk 系列工具、WinRunner、Robot 等。

1996 年提出的测试能力成熟度（Testing Capability Maturity Model，TCMM）、测试支持度（Testability Support Model，TSM）、测试成熟度模型（Testing Maturity Model，TMM）。

①TCMM 于 1996 年，由 Rodger 和 Susan Burgess 在 Testing Computer Software 会议上提出。

②TSM 于 1996 年，由 David Gelperin 和 Aldin Hayashi 提出。

③TMM 于 1996 年，由 Ilene Burnsein 博士在伊利诺伊研究所提出。

TCMM、TSM 和 TMM 是对软件能力成熟度模型（CMM）的有益补充。

到了 2002 年，Rick 和 Stefan 在《系统的软件测试》一书中对软件测试进行了进一步定义：测试是为了度量和提高被测软件的质量，对测试软件进行工程设计、实施和维护整个生命周期的过程。

我国的软件测试技术研究起步于“六五”期间，主要是随着软件工程的研究而逐步发展起来的。由于起步较晚，与国际先进水平相比差距较大。现在国内软件测试还处于初级发展阶段，在 2004 之前大学毕业生出来找工作，可能几乎没有听说过软件测试这个职位，企业也不重视软件测试，近些年随着互联网信息技术和我国外包业务的发展，很多 IT 企业开始重视软件测试，并开始组建软件测试团队。

虽然近些年来，软件测试在国内得到很大的发展，但相对于国外软件测试的发展来说，国内还处于初级阶段，目前国外开发工程师与测试工程师的人员比例为 1:1，而在国内开发人员与测试人员的比例为大概 7:1 左右，相对于国外还有很大差距。据国家权威部门统计，中国软件人才缺口超过 100 万人，其中很大一部分为软件测试人才，缺口达到 30～40 万。

然而符合企业急需的软件测试工程师在国内现有的人才数量中却寥寥无几，由于软件测试工程师属于软件产业化过程中凸显的一个新型软件技术职业，国内传统学历教育在这方面尚处于真空状态，无法满足行业对这一特殊岗位的需求。根据信息产业部门发布的最新报告显示，我国软件测试工程师的行业需求超过 30 万，业内专家预计，在未来 5～10 年中，我国企业对测试人才的需求数字还将继续增大。

在国外，软件测试已经不仅仅是为了发现程序或系统中的错误，还包括对软件质量的度量和评价，而在国内，软件测试更多的是为了发现程序或系统中存在的错误。

目前，国内主要以手工测试为主，主要验证客户需求是否被实现，而随着软件复杂程度不断加大，手工测试的成本越来越高，导致测试的整体成本越来越高，随着国内手工测试的不断发展与完善，这两年国内很多 IT 企业也已经开始研究自动化测试，希望通过自动化测试来提高测试效率。但目前大部分企业通过自动化测试工具录制和回放来实施自动化测试，离开发一个很完善的自动化测试流程框架还有很长的一段路。

目前，国内的软件测试流程还在不断的完善之中，并且软件测试工程师的整体素质还不是很高，未来企业对软件测试工程师的要求会越来越高，不仅仅需要掌握软件测试的相关知识，还需要了解编程技术、业务流程、数据库技术、网络技术等。

正是因为国内软件测试还处于初级发展阶段，所以未来软件测试行业还有着很大的发展潜力。其主要有三个方面：一是进入软件测试行业的机会；二是未来发展和待遇；三是相关培训机构和第三方测评机构不断发展。

第一：进入软件测试行业的机会

现在软件测试的门槛相对于软件开发低很多，这样为想进入软件测试的朋友提供了更多更好的机会。正是由于进入软件测试较简单，所以进入软件测试行业之后受到公司的重视程度与软件开发工程师也有所不同，原因很简单，开发工程师可以生产出产品，产品是可见的，而测试并不能生产出产品，只是发现产品里面的缺陷。因此一些企业不是很重视软件测试，但随着软件测试的发展，软件质量在整个公司的研发过程和整个项目团队中得到充分的重视，那么同级别的测试工程师和开发工程师的待遇将是一样的。在国外一些软件测试工程师的待遇甚至超过开发工程师，因为高级软件测试工程师的要求可能超过同级别的开发工程师，他们不但需要熟悉软件测试，还需要熟悉编程的相关知识。

第二：未来发展和待遇

软件测试工程师的未来是高级测试工程师，而高级软件测试工程师就需要熟悉性能测试或自动化测试。随着软件测试的发展，未来手工测试工程师的竞争力逐渐下降，性能测试和自动化测试是国内软件测试发展的趋势，性能测试工程师和自动化测试工程师的职业逐渐形成，就不仅仅要求他们能设计测试用例，还需要相关的编码能力，并且对数据库、操作系统等都应该有相应的了解。当然性能测试工程师和自动化测试工程师的待遇相对于手工测试工程师也有很大的提高，并且比一般的开发工程师也会高出很多，这同时也说明仅仅依靠手工测试，想要在软件测试行业有一个好的发展变得越来越难了。

毋庸置疑，国内软件测试的未来是美好的，但是并不代表每个软件测试工程师一定会有一个好的发展，准确地说中国的软件测试发展充满机会和挑战，只有不断地提高自身的能力，才能更好地适应软件测试行业的发展。

第三：相关培训机构和第三方测评机构不断发展

国内关于软件测试培训的公司或机构很少，但随着软件测试的发展，企业对软件测试人员的需求不断增多，促使软件测试相关的培训机构也不断发展。而对于企业来说，企业在发展过程中也需要不断完善测试流程，这同样需要相关培训机构的帮助，所以会促使软件测试相关培训机构的发展。

1.2 历史教训

以前大家不把软件当回事，但现在我们不能对软件视而不见，软件几乎渗透到我们生活的每个角落。然而软件始终不能做到完美无缺，总是存在让人厌烦的缺陷。软件开发工程师一个小失误就可能带来灾难性的事故。

1.2.1 1962年，“水手1号”火箭爆炸

经济损失：1850万美元。

1962年7月22日，美国发射了一枚命名为“水手1号”的火箭。火箭在飞往金星途中，突然偏离预定的轨道，任务控制在起飞293秒后摧毁了火箭，凌空爆炸。有关部门立即进行了紧张的调查，调查的结果也出乎人们的意料，导致这次事故的原因是：在控制火箭飞行的计算机程序中错误地省略了一个连字号“-”。仅仅是因为缺少了一个小小的连字号“-”，竟使美国损失了1850万美元。

1.2.2 1978年，哈特福德体育场倒塌

经济损失：7000万美元以及给当地经济造成的2000万美元的损失。

1978年，在上万球迷离开哈特福德体育场仅仅几小时后，钢结构的体育场屋顶就被大雪压塌了。起因是CAD软件程序员在设计体育场时通常错误地假设钢结构屋顶的支撑仅承受纯压力，但当其中一个支撑意外地因大雪而被压塌，引起连锁反应，导致体育场屋顶的其余部分像多米诺骨牌一样相继倒塌，进而导致整个体育场全部倒塌。

1.2.3 “5·19”南方六省断网事件

2009年5月19日，海南、甘肃、浙江、江苏、安徽、广西六个省出现了严重的断网现象，称之为“5·19”断网事件。因此导致打不开网页，QQ、MSN等即时通信工具掉线和无法在网络上收听、收看音、视频等故障。广东、上海、北京等10多个省也受到波及。

事故原因是DNS域名解析故障。DNS域名解析是网络用户访问互联网时服务商所进行的必要工作，普通用户在访问互联网时一般是输入网站的域名，但在后台技术上则需要翻译成数字化的服务器地址，经过这个过程，用户才能看到想要访问的网站。

由于暴风影音客户端软件存在缺陷，加上其高达1.2亿的用户，当暴风影音域名授权服务器工作异常时，导致安装该软件的上网终端频繁发起域名解析请求，引发DNS拥塞，造成大量用户访问网站慢或网页打不开。

此次事故从某种意义上完全是一场“蝴蝶效应”。最开始可能仅仅是一家网游私服为了争夺玩家，不择手段地攻击另外一家私服。黑客在设法黑掉竞争对手网站的情况下，干脆从域名下手，对DNSPod的服务器进行狂轰滥炸。这导致中国电信方面检测到异常的网间流量，从而启动应急机制。

不幸的是这台被攻击的DNS服务器正在为大约10万家网站提供域名解析服务，其中有VeryCD、中国站长、4399.com等知名网站，而最出名、流量最大的恰恰是暴风影音。网民同时向以暴风影音为首的10万个网站的访问请求随即演变为一场灾难。由于DNS服务器已经瘫

瘫，而用户的请求集体转向中国电信的 DNS 解析服务器，从而导致电信服务器很快就瘫痪了。这样的效应逐步扩大，最终导致“5·19”全国南方六省网络瘫痪的重大事故。

1.2.4 2003 年，美加停电事故

著名安全机构 SecurityFocus 的数据表明，2003 年 8 月 14 日发生的美国及加拿大部分地区史上最大停电事故是由软件错误所导致。

SecurityFocus 的数据表明，位于美国俄亥俄州的第一能源（FirstEnergy）公司下属的电力监测与控制管理系统“XA/21”出现软件错误，是北美大停电的罪魁祸首。根据第一能源公司发言人提供的数据，由于系统中重要的预警部分出现严重故障，负责预警服务的主服务器与备份服务器接连失控，使得错误没有得到及时通报和处理，最终多个重要设备出现故障导致大规模停电。

预警系统崩溃后没有接收到更多的警报，更没法向外传播，操作员并不知道预警系统已经失效，他们发现了部分异常情况，但因为没有看到预警系统的警报，而不知道情况有多么严重，以致一个小时后才得到控制站的指示。但此时没完没了的故障干扰已经让操作员反应不过来，无法控制整个局面。正常情况下，出现错误的网络会立即与其他网络分隔开来，这样一来错误就会被固定在一个地方，但是同样由于预警系统失灵，操作员没有做出应有的反应，最终使得错误蔓延，一发不可收拾。

根据北美电力可靠性协会（NERC）公布的有关事故资料，可看出事故起因和发展过程：

在发生大停电事故前 1 小时，即美国东部时间 15:06，美国俄亥俄州的一条 345kV 输电线路（Camberlain–Harding）跳开，其输送的功率转移到相邻的 345kV 线路（Hanna–Juniper）上，引起该线路长时间过热并下垂，从而接触线下树木。当时由于警报系统失灵没能及时报警并通知运行人员，15:32 该线路因短路故障而跳闸，使得克利夫兰失去第二回电源线，系统电压降低。

此后，发生了一系列连锁反应，包括：多回输电线路跳开、潮流大范围转移、系统发生摇摆和振荡、局部系统电压进一步降低，引起发电机组跳闸，使系统功率缺额增大，进一步发生电压崩溃，同时有更多的发电机和输电线路跳开，造成大面积停电的发生。

在首先跳开的 5 回 345kV 线路中，除第 4 回属于 AEP 公司外，其他 4 回均属于 FE 公司。他们认为，虽然有一些线路跳闸，系统也是安全的，因而未与其他相连系统解列，导致事故扩大。

1.3 缺陷的引入

一个项目从启动到发布必须经过这几个阶段：项目启动、需求分析、方案设计、程序编码、验证和发布，如图 1-1 所示。

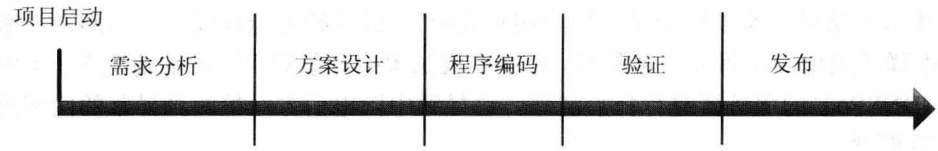


图 1-1 项目开发几大阶段

那么在整个项目开发过程中，缺陷是从什么时候开始引入的呢？一种错误观念是缺陷是只在写代码的阶段开始引入的，有相关调查显示，缺陷是从需求分析阶段开发引入的，并且在

需求阶段引入的缺陷所占比例最大，约 56%，具体见缺陷原因分布图，如图 1-2 所示。

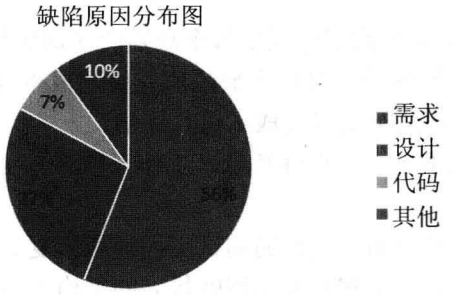


图 1-2　缺陷原因分布图

从图 1-2 中可以看出需求分析阶段引入的缺陷占整个系统缺陷的 56%左右，方案设计阶段引入的缺陷占整个系统缺陷的 27%左右，写代码阶段引入的缺陷占整个系统缺陷的 7%左右，其他方面引入的缺陷占整个系统缺陷的 10%左右。

有这样一个实例，在数据管理界面，如图 1-3 所示，有一个功能是按时间进行检索，可以选择显示最近一天、最近一周、最近一个月和所有数据。但当时需求工程师在分析需求时，忽略了一个问题，即进入数据管理界面时，默认状态下是显示所有数据还是显示最近一天、最近一周或最近一个月的数据，而测试工程师由于不是很了解业务流程，也未发现这个问题，当系统销售一年后，有客户开始投诉进入数据管理界面的时间太长。对反馈的问题进行分析，发现进行数据管理界面时，默认的是显示数据库中所有的数据，而这个过程需要两部分的时间，第一部分时间是检索数据库；第二部分时间是将检索到的数据显示到界面列表中。所以当系统运行的时间越长，系统累积的数据就越多，这样时间就越来越长，如果只是几千个数据这个时间当然不明显，但当系统中数据到达几十万条后，这个时间就很惊人了。最后将其修改为默认显示最近一周的数据，这样问题就得以解决了。通过这个实例可以看出需求定位不明确可能导致缺陷的引入。

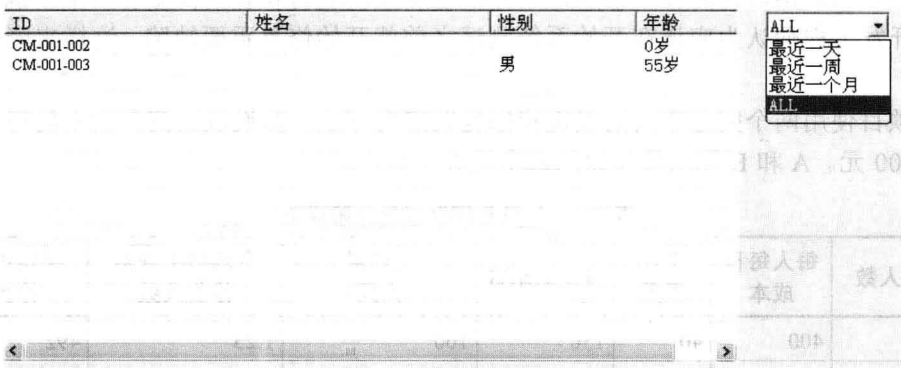


图 1-3　数据管理界面

1.4　修复缺陷的成本

在上节中我们介绍了系统缺陷是如何引入的，那么修复这些缺陷需要花费多少成本呢？

这里包括两层意思：第一，整个系统每个缺陷的平均修复成本；第二，在不同阶段引入的缺陷修复成本之间的关系。

（1）首先，探讨在整个系统测试过程中每个缺陷的平均修复成本。

在系统测试过程中，要精确地计算每个缺陷所花费的成本其实并不是一件简单的事，但通过下面的公式可以对每个缺陷平均修复成本进行评估。

修复一个缺陷的平均成本＝（工程师数量×工作天数）×工程师日平均工资/修复的缺陷数量。

①工程师数量：指参加整个系统测试的测试工程师和修复缺陷的开发工程师的数量；

②工程师日平均工资：指测试和开发工程师每天的平均工资；

③修复的缺陷数量：指被解决与修复的缺陷数量。

之所以说这个公式只能评估每个缺陷大概的平均修复成本，是因为在修复缺陷的过程中，还可能花费其他成本，例如开发或购买测试工具的成本；还有一种情况在实际工作中可能经常出现，开发工程师努力查找一个缺陷的原因，但发现花了几天时间还是无法找到，这其实也是修改缺陷的成本。所以说要精确地计算每个缺陷修复的成本其实不是一件容易的事。

注意：在公式中并没有使用“发现”缺陷的总数量，是因为发现的缺陷数量并不能真实地反应修复每个缺陷的成本，并且发现的缺陷并不一定被全部解决，所以使用被修复的缺陷数量来统计。在测试过程中发现缺陷只是第一个步骤，定位错误、决定如何修复、开发人员测试（又名单元测试）修复的内容、系统测试修复项、寻找由这个缺陷引起的其他缺陷，所有这些都是修复缺陷所需要花费的成本。

让我们看一个实例，假设有 A 和 B 两个开发团队，其中 A 团队一共有 4 名开发工程师，他们在项目开始的前几个月，不修复任何缺陷，除非缺陷阻塞他们继续开发。A 团队认为同时修复测试提出的所有缺陷是更节省成本的方法。因此在产品即将发布的前一个月，他们才开始修改所有的缺陷。

B 团队一共有 8 人，由于更关注市场的变化而受到约束，所以当客户提出需要一个 β 版本时，他们马上开始设计软件。考虑到一个有着许多缺陷的 β 版本会令客户很不满意，所以 B 团队的开发工程师认为应该在开始系统测试之前就开始修复重要缺陷，这样来降低成本和风险。

两个项目使用两个完全不同的发现和修复缺陷的方法。假设该实例中的平均每人每日的成本为￥400 元。A 和 B 团队修复缺陷成本见表 1-1。

表 1-1　A 和 B 团队修复缺陷成本

团队	人数	每人每日成本	工作天数	修复缺陷数	工作量（人天）	平均每个缺陷修复天数	平均每个缺陷修复成本
A	4	400	40	130	160	1.23	492
B	8	400	20	32	160	5	2000

A 团队在系统测试时暴露了大量的缺陷，虽然大部分缺陷是容易修复的，但是还有一些缺陷需要花很长时间才能修复。B 团队在系统测试时暴露出非常少的缺陷，但是每发现一个缺陷都需要较长的时间，这样平均每个缺陷的修复时间很长，导致 B 团队的修复缺陷的成本很高。但仔细分析一下 A、B 两个团队，A 团队虽然修复了 130 个缺陷，但还有一些缺陷未修改，B