目 录

测试团队组建的成功案例

作者：匿名

前言

也许不是每个人都有机会组建一个全新的测试团队，对于小公司而言组建测试团队是再正常不过的事，新创建的公司也需要有组建测试团队经验的人才，本文通过回顾组建团队的整体过程，讲述很多不为人知的细节。

内容提要

对于组建的公司，老板在招募测试管理人才时，一定会问如何组建测试团队的问题，因为这将是老板下一步工作最关心的问题。你的回答可能直接决定了是否能胜任此职位，本文将案例分为三个部分向大家介绍：

（一）如何规划团队愿景：这是一个帮老板下决心的过程，中间的博弈不仅仅是管理技能上的头头是道，更多是心理上的认同感，请拿出你的专业化水准，用大量数据和你的人格魅力证明你的规划切实可行。

（二）招兵买马真枪实弹操练：再好的愿景都是浮云，怎样落地才是管理能力真正的体现。管理者要面对人才招募，人才培训，技能培养，工作的安排等一系列问题。并在短期内让这一切都内井然有序。

（三）思考新问题和新挑战：好的开始不一定就意味着好的结果，因各种原因，缺乏积累和沉淀，势必会面临大量员工离职，面对加薪调整。高层变动等各种困境，同时还得证明团队存在价值并争取更多资源。

关键词

团队愿景规划，团队组织结构规划，总体方案规划。

背景介绍

公司类型：商务贸易公司（以下简称 A 公司）

项目类型：研发内部项目（技术部不是企业核心部门）

公司规模：120 人左右 其中技术部 12 人（开发 7 人，数据库 2 人，网络运维 3 人）

事件背景：A 公司因业务发展需要，承接了数个外部项目，利润可观，公司非常重视此事，特调派有项目管理经验的副总兼任外包部门总经理，从内部项目部抽调了 3 位开发人员，人力需急速扩张至 50 人左右，为保证外部项目质量，

计划招募专业的测试人员。经 HR 审阅和多方比较，锁定了一位叫 Mary 的女士，下面将和 Mary 探讨组建团队的诸多趣事：

正文

一、如何规划团队愿景

1、4H 内做出一个测试规划

上午 10：00 Mary 接到 A 公司技术部主管电话，要求在下午 2 点为总经理做一个 20 分钟的测试规划报告。Mary 接到电话后开始思索报告内容：

1）报告时间为 20 分钟，测试规划命题太大，时间上看不应涉及过多细节；

2）报告对象为总经理，要明确对方最关心什么，最急于解决什么问题；

3）测试规划需贴近公司实际避免空谈，但又能在短时间内展示专业水准。

带着问题，Mary 给技术部主管回电话咨询，希望能了解上述细节，经沟通，Mary 获得了必要的信息，开始构思 PPT 框架：

A、PPT 应控制在 8 页以内，除开始和结束页还剩 6 页，平均每页演说时间约 3 分钟。

B、计划前两页把获取到的必要信息整合展示，后面几页针对前面的信息来做推导，最后做一个简单规划总结。

C、整个 PPT 样式要简洁，使用图表或图形，最好能有 A 公司的 Logo，印象会更好。

梳理简单框架后开始拼凑素材，下方是产出的 PPT 文档结构：

PPT Page1：标题：A 公司测试规划浅谈　公司 LOGO 作者：Mary　时间：X 年 X 月 X 日

PPT Page2：项目概述，图形描述当前承接了几个项目，分属哪个领域，有哪些支持（例如技术支持，设备支持，政府支持，人力支持，财政支持等等）。【目的：开篇总揽全局，描绘出一幅蓬勃发展之景。这样入手不会显得整个测试规划思维狭窄。】

PPT Page3：质量标准：外包行业的重要质量验收指标介绍【目的：强调测试的重要性，用户验收容易让主管和经理们印象深刻，比介绍测试标准更有针对性】

PPT Page4：现状分析，当前 A 公司的测试现状，包括易用性，功能测试，性能测试和安全性测试等【目的：从宏观拉回现实，让主管和经理们意识到现在的差距甚大，加大测试重要性的渲染】

PPT Page5：当前解决方案，针对现状提出实质性的解决方案，该方案要能快速执行，短期见效，抓重点。【目的：给主管和经理们信心，该任务虽然艰巨，但按时完成也是有很大希望的，体现你的执行力】

PPT Page6-7：后期规划，为测试描绘一幅蓝图，从技术层面和管理层面给予一定高度的规划【目的：证明你的能力，这两页是你汇报的神来之笔，好坏直接决定了你在对方公司心目中的定位。】

PPT Page8 结束语：能为 XX 公司打造一支专业化团队是 XX 的目标 谢词：感谢 XXX，XXX 之类的。【目的：表决心，毕竟还没正式进入对方公司，洽谈期间谦虚谨慎为上策】

以上是PPT的内容介绍,PPT演说结束后得到了经理的认可和赞赏,希望Mary可以为A公司做一个详细的组织结构规划方案。这样的结论已经是非常满意的结果，意味着 Mary 的面试算是过了一半，具体组织结构的规划将有机会和公司有更多互动。

心得体会：思考如何给高管留下深刻印象，换位思考和面试前的充分准备是关键。

2、做一个组织结构详细规划方案

这次任务的时间 Mary 给自己定义为 3 天，算是趁热打铁的做法。查阅资料了解外包公司组织结构，结合对 A 公司急切的实际需求，准备选择一种前期人力投入少，见效快，执行力好的组织结构：测试团队独立运作，支持所有项目的工作开展。

文档包含以下几个部分：

第一部分：内容概述，介绍事件背景，阅读对象，组建测试团队的必要性，以及下文的介绍方式。【原则：清晰简明，给阅读者概括性认知】

第二部分：组织结构规划

1）介绍三类测试组织结构：小公司模式，大公司模式，外包公司模式，并举例使用这三类模式的公司（这类公司尽量选择知名的，或者 A 公司的同类别公司）【原则：客观介绍三类组织结构的基本情况，不带评论和倾向色彩】

2）组织结构利弊分析，对比选择不同组织结构对测试人员的各项要求，包括对测试人员业务，技能，质量水平要求，以及资源利用率和个人发展空间等诸多因素。【原则：通过数据和表格对比，客观分析各项选择的优缺点，其实已经有侧重了，你希望的方式会更优描绘】

3）推荐组织结构，结合上述的利弊，选择一种模式作为雏形，在结合公司世界情况进行局部改良，尽量扬长避短。这里推荐的是以大公司组织结构为主体，并吸纳部分小公司组织结构灵活性的优势，做部分弹性调整。【原则：表名组织结构立场和观点，并有所创新，不按部就班】

第三部分 测试质量体系搭建

1）测试质量体系的构成，包括技术手段的数据收集，以及管理上的各类辅助决策。呈上启下，证明推荐组织结构合理可行，要达到高质量要求非推荐模式所属【原则：介绍尽量体现专业化，使用规范图表和相关术语】

2）测试质量平台搭建，勾画在推荐组织结构基础上搭建整体质量平台的可行性和长远指导意义。增强管理者选择该组织结构模式的信心。【原则：有宏观介绍，也有微观执行方案，体现执行力和远见性】

第四部分 测试团队综合管理规划

1）团队人才结构，借鉴人力资源理论，在推荐组织结构的基础上构建团队雏形。给出梯队人才结构比例，以及等级评定制度和 360 度人才选拔方案【原则：尽量图形化，重点放在团队构建规模和各阶梯的人力配比上】

2）人才储备与培养，给出操作步骤，强调如何育人留人，最大限度减少人员流失对企业造成的影响【原则：尽量图形化，简洁化，不是本文重点，点到为止】

3）职业发展规划，进一步强调这样的组织结构对人员的发展益处，进一步增强人员稳定性，包括组织内部的职位提升，也包括外围发展空间的扩展。【原则：不拖泥带水，简洁明了，图形化描述】

第五部分：总结 提炼本文核心思想，强调组织结构，质量体系，团队管理三者在测试组织框架中的联系紧密。把单纯的组织结构描述上升到三维一体的高度。是文章的点睛之笔，把全文很好的串在一起。【原则：体现本文主旨，概况重点，升华高度】

这样的组织结构让经理和主管们很满意，决定推广和实施，Mary 的面试基本算是通过了，但还缺少一份愿景规划，Mary 明白下一份文档是经理和主管向高层汇报使用的，马虎不得。

心得体会：组织结构的规划对今后工作的开展至关重要，既要有全局观又要用发展的远光看问题。

3、给公司一个愿景规划，给自己一个空间

时间很紧张 2 天后要给出来，Mary 开始构思这份愿景规划，她考虑了如下问题：

1）看这份文档的可能是高层，目的是主管和经理向上级汇报；

2）目前应该属于基本方案提报阶段，还无需出炉细致的解决方案。

3）时间有限，采用 PPT 形式，页数也不宜过长，保留 2 个版本，（2-3 页版，10 页版）。

PPT 的编写方式这里就不再累述，在 PPT 里通常高层更在意战略决策，成本控制，收益回报等因素，在编写 PPT 时这些内容都应尽量考虑进去，技术方面最好有模型的宏观概念，专业度上才能有更好的认可。至于制作 2 个版本是方便经理和主管依据时间来做演说，故准备两个版本 2-3 页版本是精华篇，10 页是详细篇。

心得体会：愿景总是要给的，但要给得合情合理，不可太过夸大但也不要过于保守。

二、招兵买马真枪实弹操练

1、测试总体规划方案的出炉

所有工作正式启动前都会有一份指导类文件，总体测试规划方案必不可少，方案大致包含如下几个部分：

第一部分：文档简介。包括背景介绍，编写目的，文档主旨，术语说明等内容；

【编写原则：概述基本情况，表明文档编写目的和文档的重要性】

第二部分：团队组建方案。包括组织结构，团队成员定位，工作职责，团队的工作模式和工作流程等内容。

【编写原则：明确组织结构，突出团队特征，让框架变得有血有肉，有成员定位，有工作职责划分，有工作模式和流程等等】

第三部分：团队管理方案。包括团队建设规模，团队组织关系，团队人才培养等内容。

【编写原则：总体规划组建只是第一步，该部分呈上启下，强调建设规模和组织关系，进一步明确组织结构的特征，同事关注人文，强调人才培养的可持续性】

第四部分：测试质量管理体系建设。包括 A 公司质量管理体系未来的构造，整体平台搭建的宏观框架，以及可实施的顺序和步骤。

【编写原则：升华文章主题，由进及远，概述团队未来发展软性指标】

第五部分：近期计划目标。包含近期工作计划，季度目标，年度目标等内容，

【编写原则：回到当前，拟定含时间点的可行性计划方案，为实施做好充分准备】

心得体会：任何实践都要有指导依据，在组建初期显得尤为重要，该方案的主要阅读人员为公司中高层，以及平级主管，得到他们的认同，下一步的工作才能顺利实施。

2、启动人才招募

一切准备工作就绪，招募人才成为当务之急，按照 Mary 规划的金字塔模型和梯队管理策略，她将第一批需要招募的对象锁定在梯队中高层，有了他们的帮助，第二批次的招募和组建培训等一系列后续工作才能如愿实施。Mary 拟定了如下实施步骤

- 步骤一：拟定中高级测试人员的工作范围和招聘要求；
- 步骤二：通过 HR 渠道发布招募信息；
- 步骤三：拟定中高级测试人员的培养方案和培养计划；
- 步骤四：启动初级测试人员招募；
- 步骤五：全员培训方案和培养计划；

很快 A 公司测试人员招募信息发布，但每天收到的简历数量很少，简历审阅后合适的又减少了一大半。这样的招聘进度有违最初规划的近期目标。如何拓展招聘渠道成为 Mary 要思考的重要问题。

通过观察和分析，包括对前期应聘者的初步了解，发现如下问题：

1）该职位的描述过于严格，不太适合成长中的中小型企业；

2）A 公司并非 IT 公司形象，而是外贸公司，多数面试者缺乏对公司的了解；

3）组建团队的时间是 9 月下旬，不是离职跳槽的高峰期。

Mary 开始改进职位描述，并希望 HR 加大对公司背景的宣传。初级测试人员的招募计划也提前启动。同时将此招募计划发放给公司同事，希望内部推荐能解决燃眉之急。

一周后，Mary 招募到了 2 位高级测试人员，1 位中级测试人员和 1 位初级测试人员，这是一个好的开始，如何让面试者快速了解公司内部情况，以及协助完成前期计划目标中的任务，成为 Mary 要解决的新问题（这部分在第三节“培训人才，善用人才”中会详细提及）。

错过了招聘高峰，短期内要招聘大量初级测试人员成为新困难，Mary 开始积极和本土各类测试培训机构及普通本科院校接洽，希望从中找到合作机会，扩大招募人员渠道。事实证明她的选择确实解决了初级人员招募的问题，简历有了更多筛选空间，一天面试 4 个人也是常态，很快在规定的日期内 Mary 完成了 80%的计划招聘任务。下一节，将关注如何让测试人员各司其职，快速投入新工作。

心得体会：人才招募不仅仅是 HR 的工作，也是主管需要通过各种人脉渠道等积极解决的问题，坐等 HR 招聘可能会空欢喜一场。

3、培训人才，善用人才

人员入职后的第一件事，一般交由 HR 统一安排，也就是所谓的入职流程，每个公司都有自己的入职流程，例如填表，配置电脑账号，安排工位，发放工牌，介绍办公环境和相关同事等等。一切就绪后，真正的工作安排就交由负责招募的使用方。

NO1：第一个重要会议

Mary 计划先召集第一批次的入职人员完成“团队宣言”（共计 9 人，包括 2 人高级， 2 人中级，5 人初级）大致内容如下：

1：欢迎各位加入 A 公司测试团队；

2：团队当前情况和为来规划简介；

3：激励团队，鼓励创新和共同建设。

4：落实近期工作任务安排。

NO2：近期工作安排具体内容

工作周期	重点工作	培训安排	负责人
第一周	熟悉环境，熟悉基本业务 搭建测试环境。	整体业务培训	业务方负责人轮流培训
		工作流程培训	Mary 负责
第二周	参与需求讨论和评审，进一步熟悉相关业务，搭建用例和缺陷管理平台	工具使用培训	两位高级测试工程师各负责
		用例写作要求和缺陷记录标准	Mary 负责
第三周	依据项目进度安排具体测试任务，包括用例编写和测试执行等工作。	新人测试技能培训	Mary 负责
第四周	无缝对接相关业务组，积极参与项目。	新人测试技能培训	Mary 负责

NO3：大会后的小会

为了让工作尽快落到实处，并加强沟通，Mary 计划在第一周和高级测试工程师交流。交流重点如下：坦诚目前团队新建会遇到的种种困难，希望大家能齐心协力共同面对问题，解决问题。

【心得：任务需要团队共同完成，靠一个人的力量是不够的，找到得力的助手会让工作水到渠成】

NO4：如何分配工作任务

如 Mary 所愿，计划基本按进顺利完成，在第三周安排任务工作时有一段插曲，目前有三个项目启动，项目 X-001　项目 X-002　项目 X-003，项目进展如下：

项目名称	项目进展	项目复杂度	人力安排
项目 X-001	项目进入开发阶段中后，很快将有模块提交测试	3 星	高级测试 1 中级测试 1 初级测试 2
项目 X-002	项目进入开发阶段前期，离测试模块提交还有一段时间	3 星	高级测试 1 初级测试 3
项目 X-003	需求阶段，还在讨论设计方案，需求待评审	5 星	Mary 中级测试 1

怎么样安排测试人力更为合理，是 Mary 要思考的问题，通过面试和前两周的观察，Mary 决定将项目 X-001 交由一位高级测试带队，并配备 1 位中级测试，初级测试人员由高级人员自行挑选。项目 X-002 交由另以为高级测试带队，配备 3 名初级测试，等待补员。X-003 暂由 Mary 负责跟进，配备 1 名中级测试，等待补员。

【安排原则：优先考虑项目进度，其次考虑技能优势以及和 PM 的匹配度，X-003 的安排是两手准备，若能在限定的时间内招募到一位合适的高级测试最好，如若不能，则培养一位中级测试出来，避免因管理工作分身乏术。】

心得体会：招聘只是工作的开始，合理安排人员工作，快速投入项目实践是组建阶段的重要内容。

三、思考新问题和新挑战

当一切感觉走上正轨，其实只是一个刚刚开始。团队管理不仅仅是内部协调运作，同外部的沟通协作，对公司方针政策的理解都有很大学问。

1、如何同开发团队协作

有时想法是好的，执行起来也会千差万别，2 个月后部分问题开始暴露出来：

项目名称	问题描述
X-001 项目	测试和开发人员关系融洽，合作愉快，但经观察，发现测试并未按照当初拟定的原则记录 BUG，经常是和开发人员私下沟通解决，记录的规范性上也存在问题。
X-002 项目	测试和开发气氛紧张，测试过于“实事求是”，开发人员对测试抵触情绪很大，双方也发生过 2 次争吵事件。
X-003 项目	基本是公司的元老开发，缺乏测试概念，对测试工作不理解，甚至感觉测试可能影响他们的工作进展，开会也经常把测试遗忘。

面对这些问题，Mary 开始思索解决方案，针对不同项目组的情况各个突破：

→ X-001 项目：

加强监督，每天要求测试负责人汇报当日工作进展和缺陷记录情况，并 Check 缺陷记录的准确性。

同开发团队沟通，告知缺陷记录并不会影响开发绩效，普及缺陷记录对项目的积极意义。

→ X-002 项目

深入实际，了解主要问题出在哪里，抓重点冲突问题先解决。在这过程中既要鼓励测试提出合理问题，又要测试能换位思考问题，大家可以拍桌子讨论，但会后一条心按讨论结果做事。

→ X-003 项目

放低姿态，用虚心求教的态度同开发人员积极沟通，多听意见，但坚持必要的原则，主动参与会议，拿出专业水准。提出业务上的见解，让开发团队接纳测试，

心得体会：问题暴露出来并不可怕，可怕的是不去及时解决，逃避只能让问题变得更严重和不可控。因立场不同，冲突不可能没有，避免冲突不一定是最好的解决方案，拿捏好分寸至关重要。

2、如何面对高层变动

相比同级部门因需求和技术上的问题发生口角或争执，高层变动对整个团队的影响也不容忽视的：

因种种原因 A 公司原本外包部门的总经理不久后离职，且带走了部门技术主管和一批老员工。公司上下议论纷纷，几位主管也是轮流被 CEO 面谈，严重影响了下属工作情绪，X-003 项目也被迫搁置。

对 Mary 来说，原本的规划和步调被完全打乱了，人员招募也被迫暂停。公司高层对新领导的指派也是悬而未决，该如何稳定军心？Mary 做了如下几件事：

1：专注项目本身，召集会议，调整工作分派，将 X-003 的人力投入到 001 和 002 中；

2：加强团队建设，组织活动，技能竞赛，经验分析，项目小结会等；

3：为个人拟定目标，制定能力规划表，针对每个人的特点拟定各自的学习计划和学习方案，辅助指导个人成长。

【目的：转移注意力，让工作本身正常运转，向团队传达积极信号，让团队保持良好的工作态度】

不久新领导上任，技术主管也从下面提拔了一位，X-003 项目由原技术部内部人员负责接手，追赶交付日期。大家都开始忙碌起来，一切似乎又恢复了平静。但 Mary 没有意识到，新旧领导对测试工作的看法和态度是有差别的，依然需要磨合，这注定对后续工作的开展产生影响。

心得体会：高层变动在所难免，不在自己控制范围内的事尽量忽视，专注眼前正在做的事，待高层问题解决后再调整策略，配合新工作的开展。

3、如何育人，留人

3 月是人员流动的旺季，原本 Mary 信心满满想要表功，测试人员也的的确确是辛苦了大半年，X-001 项目也顺利交付客户。但薪资调整名单的公布让 Mary 感到意外，测试人员的调整比例明显低于开发团队，调整金额也少于开发团队，Mary 找了技术主管和总经理理论，得到的回答让她相当气愤：诸如：测试人员的工作没有开发人员辛苦，薪资自然比开发人员低；项目收款还不顺利，优先考虑开发也是应该的。测试的作用到底有多大还有待考证等等。

这样的答复让 Mary 倍感失落，而一张张离职申请更让她感到无助。走人？从气愤到想离职的冲动再到冷静。内心的交战最终让 Mary 决定留下来面对一切。

第一步：分析人员流失的情况，对比同行业薪资水平，安抚现有人员，并向上级主管汇报；

第二步：拟定新的工作方案，按现有人力调配资源，拿一个项目不参与测试（纯开发）；

第三步：加强同新主管的沟通和互动，细化测试工作汇报，强调工作量和工作强度，拿数据说话。

第四步：优化团队结构，建立备份机制，并重点培养核心骨干人员；

第五步：为下属争取更多机会和发展空间，例如组建质量保障团队。

经过 Mary 的努力，三个月后，事情开始好转起来，测试得到了相对公正的对待，人员也做了补充，开始恢复常态化运转，质量保障团队也被提上日程。

心得体会：任何主管都会面临下属离职的困扰，如何正确看待离职，安排好后续工作，控制个人情绪调整好心态是关键。

总结

本文通过一个团队组建的历程想大家展示了组建团队过程中的诸多细节，每段介绍都附带一个心得体会，帮助总结前文内容。本文不是一个万能模板，任何公司的组建都有自己的背景和发展的具体需要，不能按部就班，但里面的诸多经验是可以借鉴及灵活运用的。这也是写这篇文章的初衷，希望能对想要组建团队或有机会组建团队的新主管们一些参考。

180 多个测试 WEB 和 Desktop 应用程序的测试用例—综合测试清单

作者：于芳 译

这是一份做 Web 和桌面程序测试的清单。

让测试清单成为用例写制过程中的一个完整部分。用这个清单，你可以快速的创建成百上千个测试 Web 或者 Desktop 应用程序。这些都是常规的通用的测试用例，应该适用于几乎所有种类的应用程序。 当为项目写测试用例的时候，参照这些测试清单，我相信你能够覆盖大多数测试类型，除了某些需求文档上要求的特定的商业规则的应用程序。

尽管这是一份普通的清单，我推荐除了用应用程序特定的测试要准备一个标准的测试清单来迎合你特别的需求。

用清单做测试的重要性：

为你的应用程序的可复用的测试用例维护标准的库会保证大多数普通的缺陷会更快的被发现。

清单帮助快速完成应用程序新版本的测试用例的写作。

复用测试用例帮助节省写重复性测试用例的钱。

重要的测试用例被覆盖掉经常比较不可能忘记。测试清单可以被开发者饮用来保证大多常见的问题在开发阶段被修复解决掉了。

一些要点：

1）用不同的用户角色去执行这些测试场景，如管理员，客人。

2）对 WEB 应用程序来说，这些场景应该在多种浏览器上测试，如 IE，FF，Chrome 和 Safari 在客户批准的版本上测试。

3）在不用的屏幕分辨率下测试，如 1024*768，1280*1024 等

4）应用程序应该在多种屏幕上测试，如 LCD，CRT，Notebooks，Tablets 和 Mobile Phone。

5）测试应用程序在不同平台上，如 Windows，Mac，Linux 操作系统上。

测试 WEB 和 Desktop 的综合测试清单

假定你的应用程序支持以下功能

有多样字段的表格

- 拥有子窗口
- 与数据库交互
- 多种多样搜索过滤标准和呈现结果
- 图片上传
- 发送邮件功能
- 数据导出功能

通用测试场景

1、所有的强制字段应该得到证实并且用星号标示

2、验证错误消息应当在正确位置恰当地显示

3、所有的错误消息应当用同样的 CSS 格式显示（如用红色）

4、常规的确认信息应当用有别于显示错误信息的 CSS 样式显示（如用绿色）

5、工具使用须知应当有意义

6、下拉字段应当首先以空值或者像‘Select’样的文本赋值开始

7、删除页面上的任意记录功能应当要求确认

8、Select/deselect 所有记录选项应当有如果页面支持记录添加/删除/更新功能

9、数量值应当用正确的货币符号显示

10、默认的页面排序应当有

11、重置按钮功能应当给所有字段设置默认值

12、所有的数字值应当恰当格式化

13、输入字段应当作最大值检查。输入值大于特定的最大值限制应当不能被接受或者存储到数据库中

14、检查所有的输入字段，包括特殊字符

15、字段标签应当是标准的，如字段接受用户名字应当被恰当标记为‘First Name’（名字）

16、在对任意记录进行添加/编辑/删除操作之后检查页面的排序功能

17、检查超时功能。超时的值应当是可配置调节的。在操作超时后检查应用程序的行为。

18、检查在应用程序中使用的 Cookies

19、检查可下载的文件是否指向了正确的文件路径

20、所有的资源键应当可以在 config 文件中配置或者在数据库中配置而不是固定的编码

21、标准的规范应当贯穿在为资源键命名的整个过程始终

22、验证所有 WEB 页面的装饰（验证 HTML 和 CSS 检查语法错误）来确认他是符合标准的

23、应用程序崩溃或者意外的页面应当被转向错误页面

24、检查所有页面上的文本上有没有拼写和语法错误

25、检查数字的输入字段输入以字符输入值。合理的验证信息应当出现

26、如果允许输入数字字段检查负值的数字

27、用小数值检查数量字段

28、检查所有页面上可使用的按钮功能

29、用户应当不能通过快速连续点击提交按钮即提交页面两次

30、用 0 除的错误应当在任意计算中处理

31、输入数据首位置和末位置为空的情况应当正确处理

GUI 和可用性测试场景

1、所有的页面字段（如文本框，选择按钮，下拉列表）应当合理的排列

2、数字值应当正确证实除非某些特定的值除外

3、充足的空间应当留在字段标签之间，列与列之间，行与行之间以及错误信息之间。

4、滚动代码应当且仅当必要时可用

5、字体大小，样式和颜色对标题，描述的文本，标签，字段内数据和网格信息应当符合 SRS 规定的标准

6、描述文本框应当是多行

7、不可用字段应当是灰色，用户应当不能够在这些字段上设置焦点

8、当点击任意输入文本框时，鼠标箭头指向应当改变为光标

9、用户应当能够在下拉列表中输入值

10、用户填写的信息应当保持紧密的当页面提交有错误信息时。通过改正错误用户能够重新提交表格

11、检查合适的字段标签是否正确用在错误消息中

12、下拉字段值应当以定义的排序命令呈现出来

13、Tab 和 Shift+Tab 命令应当正常工作

14、默认的选择选项应当是在页面负载的时候预先被选定

15、字段特定的和页面级别的帮助信息应当可用

16、检查正确字段是否被高亮当遇到错误时

17、检查下拉列表选项是否可读而且不会因为字段大小限制崩溃

18、检查页面上的所有按钮应当可以通过键盘快捷方式访问而且用户应当能够使用键盘执行所有的操作

19、检查所有页面上的崩溃图片

20、检查所有页面上的崩溃链接

21、所有页面应当有标题

22、确认消息应当在执行任意更新或删除操作行为之前显示

23、当应用程序忙时小时应当显示

24、页面文本应当向左靠齐

25、用户应当能够选择任意选择选项和多选框中的任意组合

过滤标准的测试场景

1、用户应当能够使用页面上的所有参数来过滤结果

2、提炼搜索功能应当装载搜索页面和所有用户选中的搜索参数

3、当至少要求一个过滤标准来执行搜索行为时，确保当用户没有使用任何过滤标准就提交页面时合适的错误信息会显示出来

4、当至少一个过滤标准选择不是强制的，用户应当能够提交页面并且默认的搜索标准应当符合查询结果

5、对过滤标准过滤出的不合法的值恰当的证实信息应当有所显示

结果网格的测试场景

1、当页面显示时间超过默认时间来显示结果页面时页面装载时符号应当显示

2、检查是否所有的搜索参数是被用来抓住显示在结果网格上的数据

3、所有结果的数量应当显示在结果网格上

4、用来搜索的搜索标准应当显示在结果网格上

5、结果网格值应当被默认列挑选出来

6、挑选出来的列应当和挑选图标一起显示出来

7、结果网格应当包括所有有正确值的特定的列

8、升序和降序功能对由数据排序支持的列起效果

9、结果网格应当以合适的列和行间距显示出来

10、当每页有多于默认的结果数值时，页数编码应当可以使用

11、检查页面编码功能的下页，上页，首页和末页功能

12、重复的记录不应当显示在结果网格中

13、检查是否所有列可见，还有垂直滚动条在必要时可用

14、检查动态列的数据（这些列是值基于其它列值动态计算的列）

15、对于结果网格显示报告，检查‘所有’行并验证所有列的所有显示报告

16、对于结果网格显示报告，检查页面编码功能可用时和用户导航到下页时‘所有’行数据

17、检查是否使用了合适的符号来显示数值，如%符号应当用作显示百分比数值

18、检查数据范围可用时结果网格的数据

窗口的测试场景

1、检查默认窗口大小是否正确

2、检查子窗口大小是否正确

3、检查是否页面上有任意字段有默认焦点（一般来说，焦点应当设置在屏幕上显示的第一个输入字段上）

4、检查子窗口在父/打开者窗口关闭的时候关闭掉了

5、如果字窗口打开，用户不能够使用或者更新任意背景上或者父窗口上的字段

6、检查窗口最小化，最大化和关闭功能

7、检查窗口是否可以放大缩小

8、检查父窗口和字窗口的滚动条功能

9、检查子窗口的取消按钮功能

数据库测试测试场景

1、检查当页面成功提交时正确的数据保存在数据库中

2、检查那些不接受空值的列的值

3、检查数据完整性。数据应当基于设计存储在单张表中或者多张表中

4、索引名字应当按照标准给出，如 IND_<表名>_<列名>

5、表应当有主键列

6、表的列应当有可用的描述性信息（除了评审列像创建日期，创建者等）

7、对于每个数据库添加/更新操作，应当添加日志

8、应当创建需要的表索引

9、检查数据只有在操作成功执行时数据才符合数据库

10、数据在遇到失败的事务时应当回滚

11、数据库名字应当按照应用程序类型如测试，UAT，沙盒，现场的（尽管这不是标准，它对数据库的维护很有帮助）来命名

12、数据库逻辑性的名字应当根据数据库名字来给出（再次指明，这不是标准，但是对数据库维护很有帮助）

13、存储过程不应当以前缀‘SP_’命名

14、检查表中评审列的值恰当地弹出来了（如创建日期，创建者，更新日期，更新着，是否删除，删除日期，删除者等）

15、检查是否保存时输入数据没有缩短。呈现给用户的页面上的字段长度和数据模式里的长度应当一样

16、用最小值，最大值和浮动值检查数字字段值

17、用负值检查数字字段值（对于验收测试和非验收测试都要做）

18、检查选择按钮和下拉列表选项是否正确的保存在数据中

19、检查书库字段被用正确的数据类型和数据长度设计

20、检查所有的表限制如主键，外键等是否被正确执行

21、用样例的输入数据来测试存储过程和触发器

22、输入字段导向和追踪空间应当在将数据传输给数据库之前缩短截断

23、主键列不允许赋予空值

图像上传功能测试场景

（也适用于其它文件上传功能）

1、检查上传的图像路径

2、检查图像上传和改变功能

3、检查图像与不同格式的图像文件（如 JPEG，PNG，BMP 等）上传的功能

4、检查图像与有空白的图像或者其它任何允许在文件名中含有特殊字符的图像上传功能

5、检查重名图像的上传

6、检查与图像大小超过最大允许大小的图像上传。恰当的错误信息应当显示。

7、检查图像与非图像文件类型（如 txt，doc，pdf，exe 等）的图像上传功能。恰当的错误信息应当显示

8、检查具有特定高度和宽度的图像（如果定义的）是被接受，否则拒绝

9、对较大大小的图像，图像上传进度条应当出现

10、检查取消按钮功能在上传过程中工作正常

11、检查文件选择对话框仅当支持文件列出来时出现

12、检查多个图像上传功能

13、检查上传后的图像质量。图像质量在上传后不应当改变

14、检查用户能够使用或查看上传的图像

发送邮件的测试场景

（写作或验证邮件的测试用例不包含在此）

（确保在执行邮件相关测试时之前用模糊的邮箱地址）

1、邮件模板应当对所有邮件使用标准 CSS

2、在发送邮件前应当验证邮件地址的真实性

3、邮件正文模板的特殊字符应当恰当处理

4、语言性的特殊字符（如 俄语，汉语或者德语字符）应当在邮件正文模板中恰当处理

5、邮件标题不应当为空

6、邮件模板中用到的收件人字段对所有收件人应当被实际值替换，如{名字}{姓}应当为名字和姓合理替换

7、如果有动态值的报告在邮件正文中，报告数据应当正确计算

8、邮件发送者名字不应当为空

9、应当从不同邮件客户如 Outlook，Gmail，Hotmail，Yahoo！等邮件来检查邮件

10、用 TO，CC 和 BCC 字段检查邮件的发送功能

11、检查内容为空的邮件

12、检查 HTML 格式的邮件

13、检查邮件标题和脚注，检查公司 LOGO，隐私政策及其他链接

14、检查有附件的邮件

15、检查发送给单个人，多个人或群发多人的邮件发送功能

16、检查回复邮件的地址是正确的

17、检查发送邮箱的最大容量

Excel 报告功能的测试场景

1、文件应当以恰当合理的文件扩展名导出

2、导出的 Excel 文件文件名应当按照标准来，如如果文件名使用 timestamp，她应当在导出文件的时候用真实的 timestamp 替换

3、检查导出的 Excel 文件中有日期列时的日期格式

4、检查数字或者货币值的数字格式。格式的格式应当 跟页面上显示的一样。

5、导出的文件的列应当有合适的列名

6、默认页面的挑选排序应当在导出文件中也携带

7、Excel 文件数据应当以标题和脚注文本，日期，页面编码在所有页面上合理的格式化

8、检查显示在页面上的数据和导出的 Excel 文件的数据是一样的

9、检查页面编码功能可用时的导出功能

10、检查导出按钮是否按照导出文件类型显示合适其当的图标，如 Excel 文件图标是对应 xls 文件

11、检查大型容量的文件的导出功能

12、检查包含有特殊字符的页面的导出功能。检查这些特殊字符是否恰当地导出在 Excel 文件中了

性能测试测试场景

1、检查页面装载时间在接受的范围之内

2、检查页面在慢速连接上的装载

3、检查在轻负载，正常负载，较多负载和重负载条件上任意动作的响应时间

4、检查数据库存储过程和触发器的性能

5、检查数据库查询的执行时间

6、检查应用程序的负载测试

7、检查应用程序的压力测试

8、检查在峰值负载条件下的 CPU 和内存使用情况

安全测试的测试场景

1、检查 SQL 注入攻击

2、安全页面应当使用 HTTPS 协议

3、页面崩溃不应当泄露应用程序或者服务器信息。为此错误页面应当显示出来。

4、避开输入中的特殊字符

5、错误消息不应当泄露任何敏感信息

6、所有的帐户信息应当通过加密的渠道传输

7、测试密码安全和密码政策加强功能

8、检查应用程序退出功能

9、检查野蛮暴力攻击。

10、Cookie 信息应当仅仅以加密格式存储

11、检查在超时或者退出后会话 Cookie 的时间段和会话的终止。

12、会话令牌应当通过安全渠道传输。

13、密码不应当保存在 Cookies 中

14、测试服务攻击的拒绝

15、测试内存泄露。

16、通过在浏览器地址栏中操作各种各样值来测试未授权的应用程序访问

17、测试文件扩展处理以确保 EXE 文件不可以上传到服务器上也不能在服务器上执行。

18、敏感字段如密码和信用卡信息不应当还有自动完成可用功能。

19、文件上传功能应当使用文件类型限制而且对扫描的上传文件抗病毒

20、检查目录陈列是否被禁止

21、密码和其他敏感字段在输入的时候应当有遮盖

22、检查忘记密码功能是否有安全特性，如在一定时间会有暂时的密码过期和安全在改变或者请求新密码前要问安全问题

23、验证 CAPTCHA 功能

24、检查重要的事件是否记录在日志文件中。

25、检查访问优先权是否正确执行。渗透测试测试用例——在此页我列出了大概 41 个渗透测试的测试用例

我已经尽可能覆盖所有的 Web 和 Desktop 应用程序功能的标准测试场景。但是我知道这仍然不是一份完全的清单。不同项目的测试员根据他们的经验有他们自己的测试清单：）

请自如添加更多的测试场景或者下面建议中的反面测试用例让这个清单完整！

基于数据库的异常数据监控系统

作者：李江

摘要：随着 IT 界研发流程的不断改善，迭代开发已成为许多互联网、软件公司通用的研发流程。迭代是以逐步、快速的方式不断地改善设计、完成发开的过程。在迭代过程中，由于半成品的使用、测试开发的不完善导致的数据库脏数据的出现，会给最终系统的正常运行带来不少损失，本文提出了一种弥补这种不足的监控方案，异步地、通过一些业务规则定时检查数据库中重要的业务数据，以便及时发现设计漏洞、开发测试漏测遗留的线上问题。

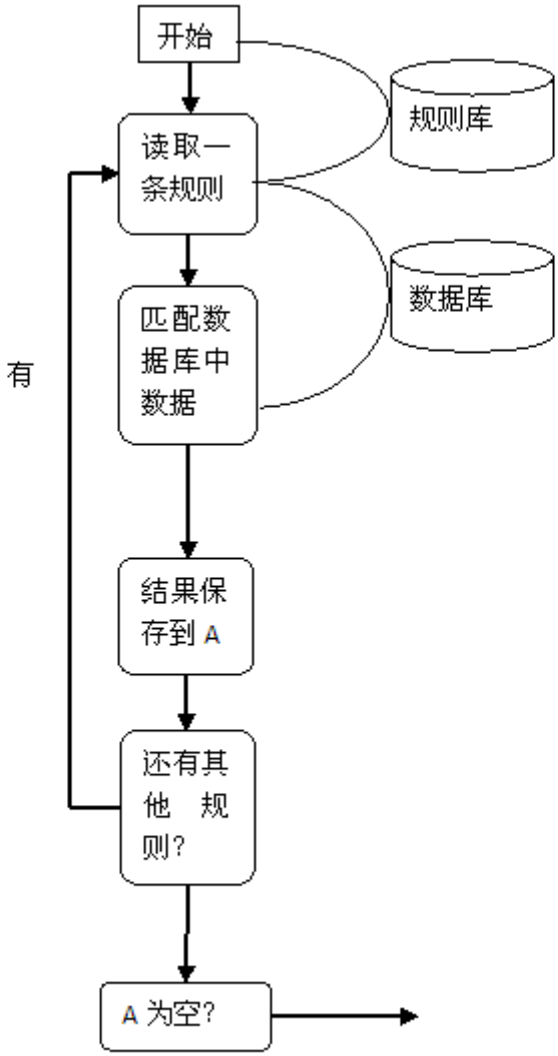
关键词：数据库监控；异常数据；框架；线上监控；测试；

目标：上线之后，通过数据库里数据的业务逻辑规则，筛选出不合逻辑的数据，并发邮件给研发人员报告问题。

方案：

系统功能模块包含：数据扫描读取模块、规则读取模块、分析模块、报警模块。

主要流程：



系统中设定定时任务，每 5 分钟跑一次。

规则库格式包含：1、规则描述 2、规则对应的一系列 SQL 语句。例如：1、买家金额为负数；2、select user_id,user_name,balance from user where balance<0;

结果分析：

该系统框架完成后，通过丰满规则库，可以减少测试人员 UI 测试时对数据正确性的校验；以及帮助线上系统及时发现问题，通知修复，当然具体对每个项目的帮助程度，取决于规则库的完善性。

推广应用：

该系统不仅仅可以应用于测试环境、线上环境的事实监控；还可以用于正常数据、异常行为的挖掘分析等，具有较广的应用场景。

软件测试结果归纳与分析

作者：云霄

摘要：如同代码是程序员的成果之，软件测试报告是测试人员的主要成果之一。一个好的软件测试报告建立在测试结果的基础之上，不仅要提供必要测试结果的实际数据，同时要对结果进行分析，发现产品中问题的本质，对产品质量进行准确的评估。本文详细说明一个软件测试报告究竟需要什么样的测试结果，需要对哪些结果进行归纳分析。

1、软件测试结果的内容

软件测试评估的目的是统计和分析测试结果，确定是否达到软件要求的指标。一般来说，首先需要分析实际测试执行的有效性和充分性，分析测试执行是否完全，软件问题的产生是否因为不符合测试的前提和约束；其次，统计测试过程中的所有软件缺陷，并将缺陷的各种属性进行归纳分析；最后，根据用例的执行情况对软件进行宏观的横向分析，确定软件缺陷的错误来源。

2、测试的有效性和充分性

评价软件测试有效性的主要目的是评价测试人员的工作和使用评价后的结果改进测试过程。在软件测试中，往往会存在一些无效的方面，评价的目标就是识别这些无效和问题以便可以采取修复措施。

在测试的有效性评价工作中，存在两个关键的因素：一是评估的目标，目标是对度量过程的恰当指导。无效的目标会使整个评价过程无效；二是实现度量目标所需的信息类别，信息的收集需要建立专门的小组，整个评价过程也应指派专门的人员负责，因为如果没有专人负责评价过程。那么就无法确保进行正确的数据收集和评估过程。

当所有的软件测试过程结束后，软件测试有效性评价工作就可以开始了，测试阶段的最终执行结果是它的入口条件，表 1 列出了输入所需的一部分信息类型，根据具体项目的不同，也会产生其它的输入。

表 1 测试有效性评价的输入信息

序号	输入的信息类型
1	执行的测试的数量
2	测试中消耗的资源

3	所使用的测试工具
4	发现的缺陷
5	被测试软件的规模
6	修复缺陷的天数
7	没有修复的缺陷
8	在操作中所发现的那些本该在测试中发现的缺陷
9	发现缺陷的阶段
10	所发现的缺陷的名称

我们可以通过一个实际的例子来看看有效性是如何实现的，表 2 列出了软件需求规格说明中常见的几个章节。针对每个需求首先验证是否包含了相应的测试类型，比如在容量和时间要求章节中，是否包含性能测试、强度测试和余量测试等。接下来，列出每个测试类型有多少个测试项进行支撑，通过这一点可以看出各个测试类型在本次测试中的优先级状态。最后，列出每个测试类型包含的测试用例数量，可以反映出需求的覆盖情况。

表 2 软件需求有效性和充分性说明

需求文档章节号	需求名称	测试类型	实际的测试项	设计的用例数	软件缺陷数
3.1	CSCI 外部接口需求	接口测试	X	X	X
3.2	CSCI 功能需求	功能测试	X	X	X
		人机交互界面测试	X	X	X
3.3	CSCI 内部接口	接口测试	X	X	X
3.4	CSCI 数据元素要求	边界测试	X	X	X
3.5	适应性要求	/	/	/	/
3.6	容量和时间要求	性能测试	X	X	X
3.7	安全要求	安全性测试	X	X	X
3.8	保密要求	安全性测试	X	X	X

3.9	设计约束	代码走查	X	X	X
		静态分析	X	X	X
3.10	软件质量因素	/	/	/	/
3.11	人的特性/人的工程需求	易用性测试	X	X	X
3.12	需求可追踪性	/	/	/	/

3、软件缺陷统计与分析

软件缺陷数是最直观反应软件质量好坏的标准，也是最难以分析的标准。由于测试人员的个体差异，测试环境的不可预知和诸多因素的存在，导致测试结果的不准确。正因为如此，所以不能单凭数量上的多少来衡量软件质量的好坏，而需要进行系统的归纳和分析。那么影响缺陷的指标有哪些呢？一般来说，包括缺陷发现的轮次、缺陷的严重级别、缺陷所属的测试类型和每千行代码所含的缺陷数等，如果是系统测试，还应包括缺陷所属的配置项等。简单的说，每个因素都决定了缺陷数一方面，但不完全决定缺陷的有效性。

根据多轮测试发现的缺陷数，可以了解测试的效率。

根据缺陷的严重级别，可以了解测试的深度。

根据缺陷所属的测试类型，可以了解测试的广度。

根据缺陷的缺陷类型，可以了解缺陷的来源。

通过对每千行代码所含的缺陷数分析，可以了解程序代码质量。

由于每个因素片面性的存在，下面就为读者介绍三种综合利用多种因素，全面考核软件质量的方法。

1）软件测试缺陷按轮次和级别统计

根据轮次和级别进行分析，可以清晰地看出每轮测试时缺陷的严重程度，与其他配置项对比分析时，可以轻而易举的判断出在同一时刻下，各个配置项的质量好坏程度。

表 3 缺陷按轮次和级别统计

测试阶段 缺陷级别	首轮测试	第一轮回归测试	第二轮回归测试	总计
致命缺陷	2	1	0	3
严重缺陷	8	3	0	11

一般缺陷	25	12	0	37
建议缺陷	6	2	0	8
小计	41	18	0	59

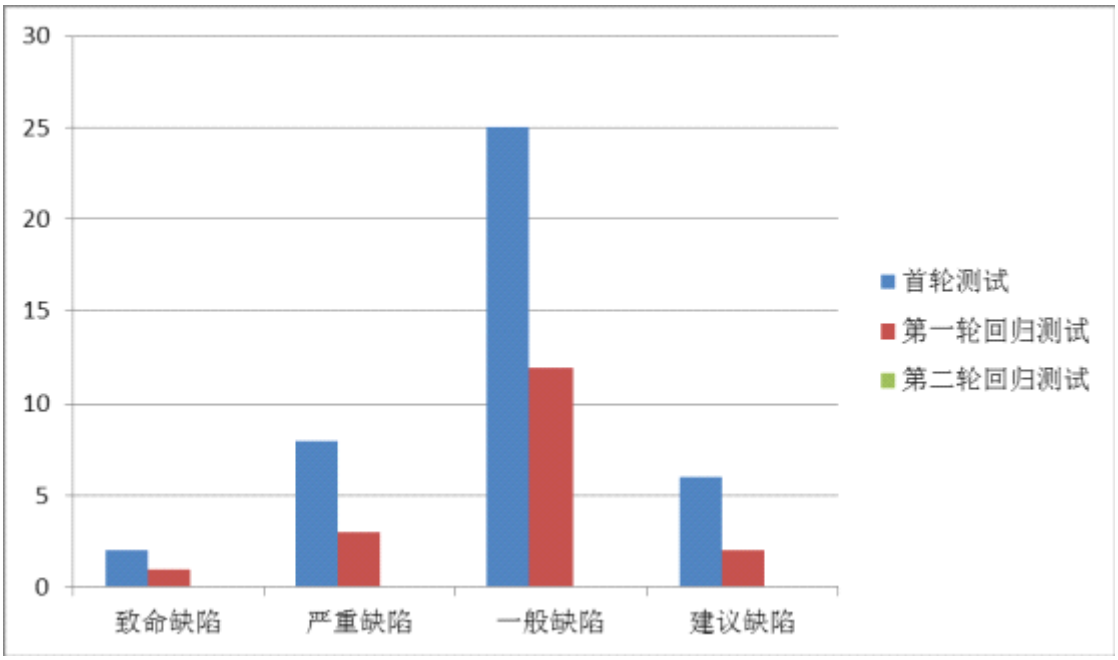


图 1 缺陷按轮次和级别统计

2）软件测试缺陷按所属测试类型和级别统计

根据缺陷所属的测试类型和级别进行分析，可以精确的将缺陷定位到每个测试类型中，从而反应出软件在哪些方面存在的较大质量问题。

表 4 软件测试缺陷按所属测试类型和级别统计

测试结果 测试类型		首轮测试				第一轮回归测试				第二轮回归测试			
		致命	严重	一般	建议	致命	严重	一般	建议	致命	严重	一般	建议
1	文档审查	0	0	2	0	0	0	0	0	0	0	0	0
2	功能测试	1	5	15	1	1	3	9	0	0	0	0	0
3	性能测试	0	1	1	0	0	0	0	0	0	0	0	0
4	边界测试	0	1	2	1	0	0	1	0	0	0	0	0
5	接口测试	1	0	2	0	0	0	1	0	0	0	0	0
6	人机交互界面测试	0	1	3	4	0	0	1	2	0	0	0	0

7	安装性测试	0	0	0	0	0	0	0	0	0	0	0
8	总计	2	8	25	6	1	3	12	2	0	0	0

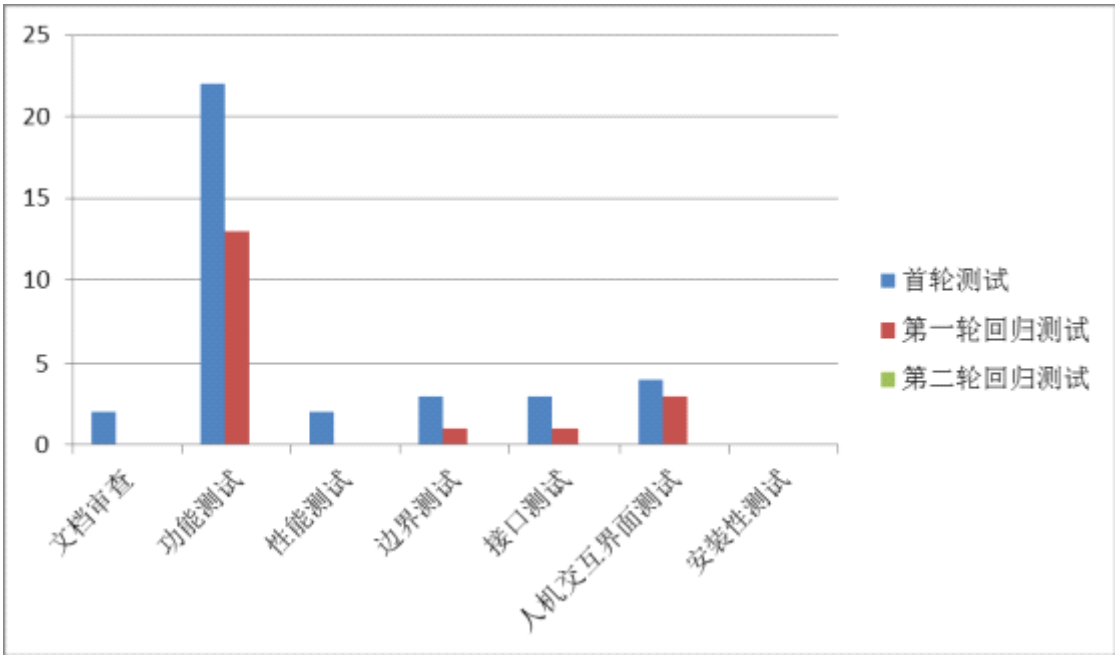


图 2 软件测试缺陷按所属测试类型和级别统计

3）软件测试缺陷按缺陷类型和轮次统计

根据缺陷类型和轮次进行分析，可以将软件缺陷定位到代码层面，通过多轮的对比，从而可以看出软件修改过程中的修改趋势，可以有效避免错误的发生。

表 5 软件测试缺陷按缺陷类型和轮次统计

测试阶段 缺陷类型	首轮测试	第一轮回归测试	第二轮回归测试	总计
程序缺陷	20	12	0	32
文档缺陷	1	0	0	1
设计缺陷	19	6	0	25
其它缺陷	1	0	0	1
小计	41	18	0	59

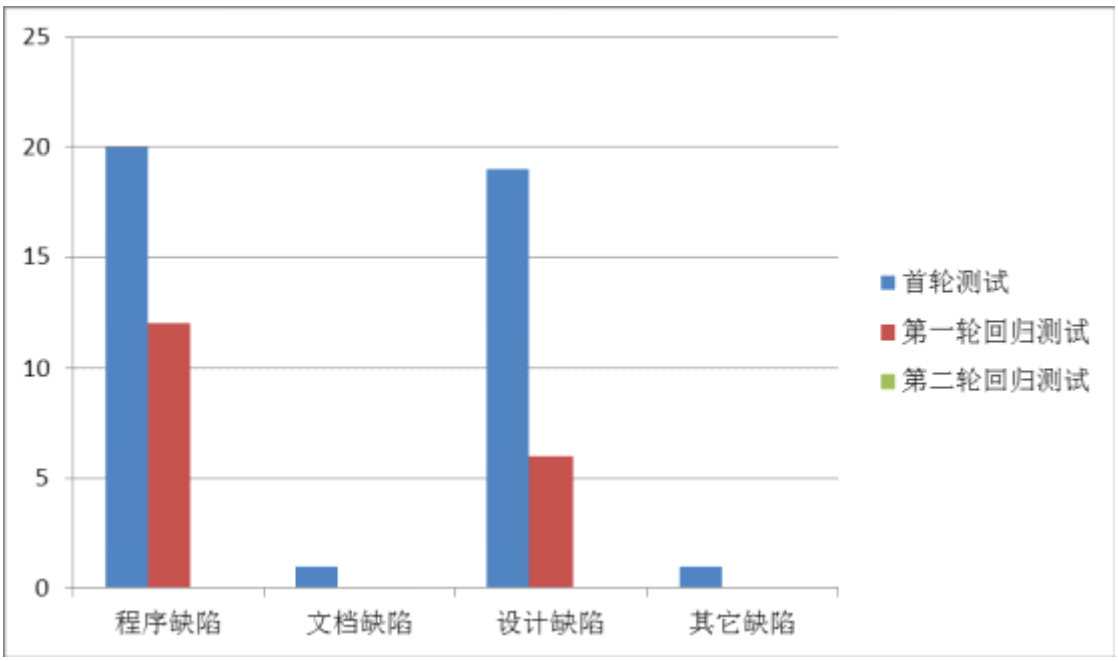


图 3 软件测试缺陷按缺陷类型和轮次统计

4、测试用例执行情况

测试用例的执行情况能够反应测试人员在执行测试的过程中，软件质量对软件在实际应用中产生的效果。与软件缺陷不同的是，缺陷反应的是一种现象和问题，而用例的执行情况则反应的是软件实际操作的使用难度。一个缺陷影响一个用例和一个缺陷影响多个用例，是两个完全不同的概念，所以用例的通过率是用户真正关心的数据。

在用例执行情况中，根据每轮测试的结果，可以分别对用例总数、执行用例总数、未执行用例数、通过数、未通过数和通过率等指标进行考核。用例总数代表了本次测试设计的用例总数，执行用例总数代表了本轮测试需要执行的用例总数，未执行用例数则是前两者的差数。而通过数、未通过数和通过率则反应了本轮测试用例的通过情况。

表 6 用例执行结果统计

序号	测试类型	首轮测试			第一轮回归测试			第二轮回归测试		
		执行用例数	通过数	通过率	执行用例数	通过数	通过率	执行用例数	通过数	通过率
1	文档审查	3	2	66.67%	3	3	100%	3	3	100%
2	功能测试	176	145	82.39%	176	168	95.45%	176	176	100%

3	性能测试	8	6	75%	8	8	100%	8	8	100%
4	边界测试	21	15	71.43%	21	19	90.48%	21	21	100%
5	接口测试	25	14	56%	25	23	92%	25	25	100%
6	人机交互界面测试	13	10	76.92%	13	12	92.31%	13	13	100%
7	安装性测试	5	4	80%	5	5	100%	5	5	100%
8	总计	251	196	78.09%	251	238	94.82%	251	251	100%

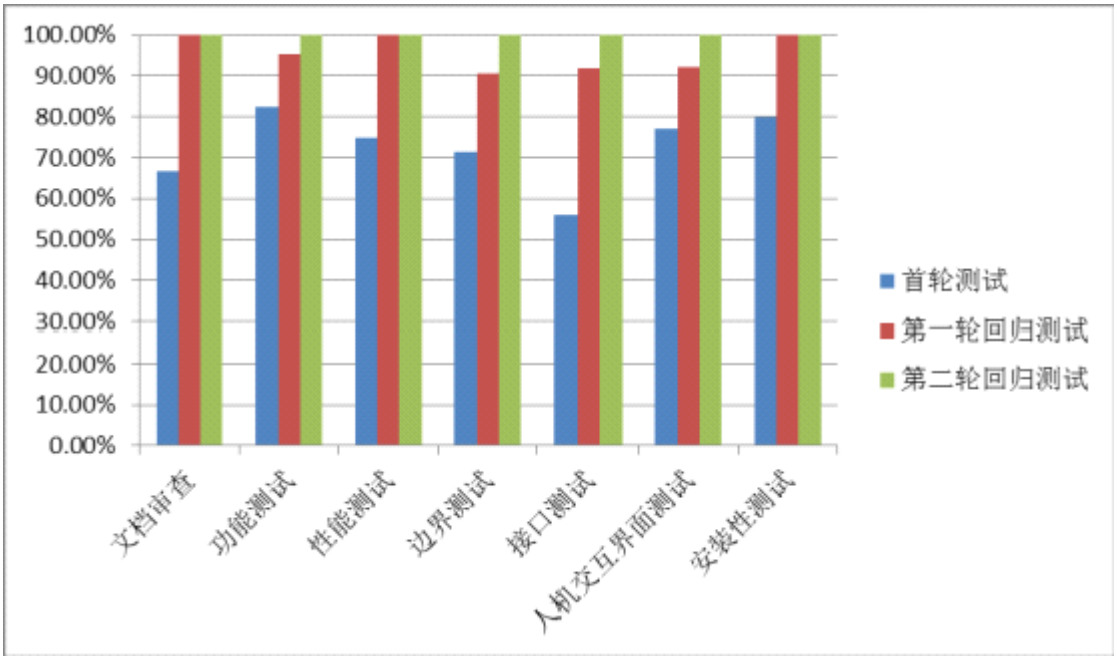


图 4 用例执行结果统计

5、总结

传统的软件测试，只针对软件产品开展，找到缺陷之后再加以改正和修补，这是一种“亡羊补牢”的质量管理方式。而针对开发全过程所开展的软件测试和过程度量，则注重事先分析，通过对已发生的数据对比、统计、时间序列等分析，来判断软件产品质量的未来趋势，并提前予以控制和预防，属于一种“防患于未然”的质量管理方式。对测试的结果进行整理、归纳和分析，可以借助于 Excel 文件、数据库和一些直方图、圆饼图、趋势图来进行分析和表示，并通过对比分析、根本原因查找、问题分类、趋势分析和其他统计分析等来实现。

从 bug 中能得到什么——简述缺陷分析

作者：吴治宾

每一轮测试结束进行缺陷分析是必不可少的，关于缺陷分析和预防的一些方法可以参考：http://www.cnblogs.com/Jackc/archive/2009/02/18/1392657.html；链接中的一些方法讲的有些笼统和理论化，下面主要结合自己的项目测试经验做下简单的总结。

《软件质量管理实践--软件缺陷预防、清除、管理实用方法》这本书中介绍的缺陷度量元感觉很有用处，可以结合一下度量目标以及实际的项目确定适当的度量元。例如，可以按照如下表所示的思路确定组织整体或者项目组个体使用哪些缺陷度量元。

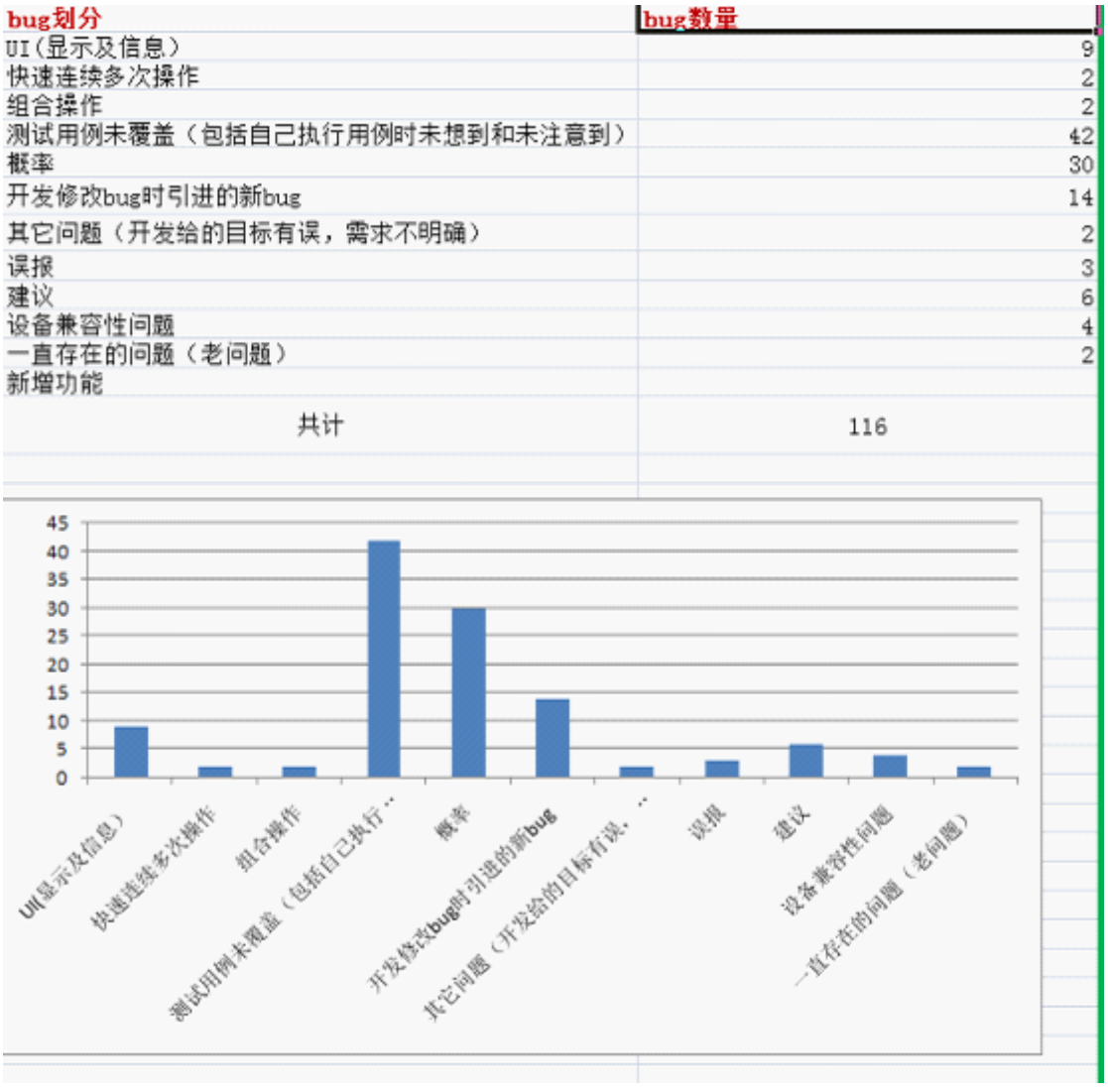
信息需要	可度量概念	度量目的	度量元	派生度量元
通过模块的各类型缺陷数来评价软件质量	模块缺陷分布	反映缺陷按类型、严重程度、所属模块分布情况。通过度量可以客观上看出哪个模块的缺陷比较高，这样可加大对这个模块的开发投入	每个模块的各类缺陷数目	各模块的缺陷个数百分比
通过总体的各类型缺陷数来评价软件质量	总体缺陷分布	反映总体缺陷的分布情况，可看出软件的缺陷主要是哪些方面的缺陷，可帮助项目组找出问题，提高质量	每类缺陷的数目	每类缺陷占总缺陷的比例
通过缺陷密度评价模块稳定性	缺陷密度	通过按模块的缺陷密度倒序排列，通过二八定理确定缺陷密集模块，确定修复重点	每个模块的各类缺陷数目	每个模块的各类缺陷密度及比例
判断缺陷数量的趋势	总体趋势	反映新缺陷数、被解决的缺陷数和遗留的缺陷数的趋势，了解缺陷解决是否及时和全面	各种状态缺陷的数量	各种状态缺陷的数量的比例
判断缺陷驻留时间	缺陷排除情况	判断缺陷产生的原因	缺陷数量排行、缺陷发现时间、缺陷清除时间	整体缺陷清除率、阶段性缺陷清除率、缺陷的驻留时间

确定哪种缺陷发现方式有效	缺陷数量和种类	选择合适的降低缺陷的方法	缺陷种类	缺陷密度、同行评审发现错误率、测试发现的缺陷数、PPQA发现的缺陷数

在我们给出的测试报告中可以结合以上各度量元给出相应的缺陷分析，以此来判断被测产品的质量情况以及缺陷趋势。

作为测试人员也应该在分析指标、统计数据的基础上，对软件缺陷状况进行定性分析，发现问题，并向项目负责人和测试负责人汇报相关情况，以此优化测试流程。

如下图是我根据一个实际项目进行的一次简单的缺陷分析过程：



上面的统计结果只是个参考，因为有些 bug 不好分类，有些 bug 是交叉测试后发现同事以前测试的模块发现的等等。

通过上面的统计大概可以看出 bug 主要集中在测试用例未覆盖到，这时最主要进行的就应该是测试用例的完善更新，确定是不是测试用例的有效性出现了问题（关于测试用例有效性可以参考我写的上一篇文章），当然这里面不只是测试用例设计问题，也应该有自己执行测试用例没有多去思考、增加新的观察点的原因。

对于概率性出现的 bug，要分析测试用例是否覆盖到，如果没有同上一条；如果有，之前执行测试用例没有发现，就不太好解决，对于测试人员来说只能尽可能重复测试场景，并总结经验，分析原因，是不是其它模块也存在潜在的概率性 bug。

回归测试中发现开发人员修改 bug 引进的新 bug，针对这个问题要引起测试人员的重视，如果是开发人员技术和态度问题，那么一个项目多轮次测试时只进行主要功能测试和验证性测试是存在一定的风险。针对测试过程中发现开发人员所犯的一些低级错误，如打包错误，缺陷 reopen 等等，对于这些问题每轮测试结束时要收集上报给项目经理和测试负责人，力求开发人员作出相应的改进。

通过缺陷分析也可以发现一些其它的问题：测试用例执行情况、测试人员态度、缺陷遗漏、测试方法改进等等。

这里举个简单的例子，一个项目多 Build 测试，我们分析缺陷：

是否有些缺陷执行测试用例本应该在前期发现，但是实际在后几轮才发现？（当然，这个可能与测试策略也有一定的关系，这个也有可能是有些测试人员本没有执行测试用例，但是测试报告上填写执行）

是否存在 Not A Bug 的缺陷？（分析是开发人员问题，还是测试人员问题）

是否有些缺陷一个人执行测试用例时没有发现但是另一个人执行却发现了（可能与测试用例编写不明确有关系）

是否有些缺陷是在交叉测试时发现的，而这些缺陷测试用例又覆盖不到？

是否有些缺陷是有些同事不断的引入新的测试方法，测试技术以及测试工具发现的？

分析其它组的缺陷，是否有好的测试方法和测试思路引进？

作为测试人员我们要不断的从缺陷中分析，为自己测试改进提供参考，找出自己测试思路的短板和盲点，不断优化测试过程，提高测试质量和效率。

作者简介：吴治宾，网名“没翅膀的飞鱼”/“Binbby”，两年测试小生，专注于软件测试，对系统测试、Web 测试、性能测试、稳定性测试、测试流程、测试管理、测试改进有所涉及且有独到见解，欢迎测试交流，共同进步。

基于 selenium 的两种 web 自动化测试框架的思考

作者：李江

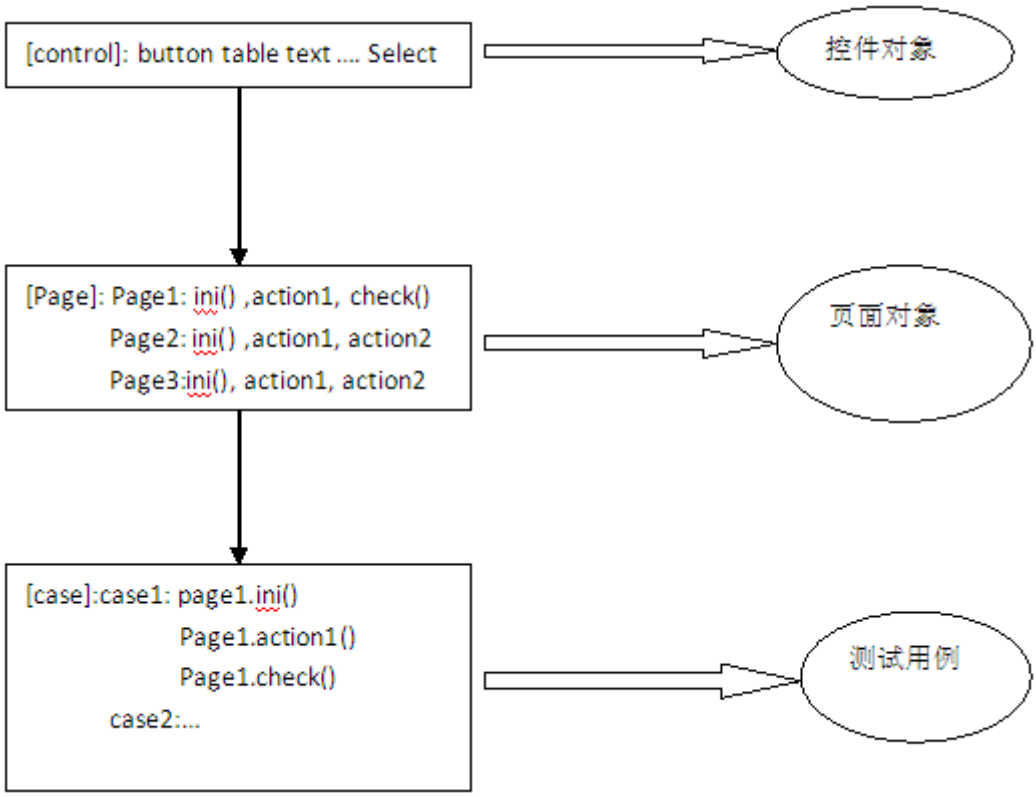
摘要：近年来，随着 selenium2.0 的出现，大量的研发测试人员投身于 web 自动化框架研究中，前端自动化变得容易起来。目前，测试框架大同小异，主体思路大致都是“控件-页面-测试用例”三个层面。然而，从本人经历的项目来看，该框架模式，并非所有场景都适合。因此，本文提出一种新的框架”场景片段-测试场景”，并用两种不同的测试对象，分析说明两种框架的特点。

关键词：selenium；web 自动化；框架；应用场景；

目标：分析“控件-页面-测试用例”，“场景片段-场景”的特点和适用范围。

正文：

我们先来观察下当前主流的“控件-页面-测试用例”框架。



在最底层，需要编写一些常用的控件对象，大多数情况下，这些对象是可以复用的，除个别特殊功能外。

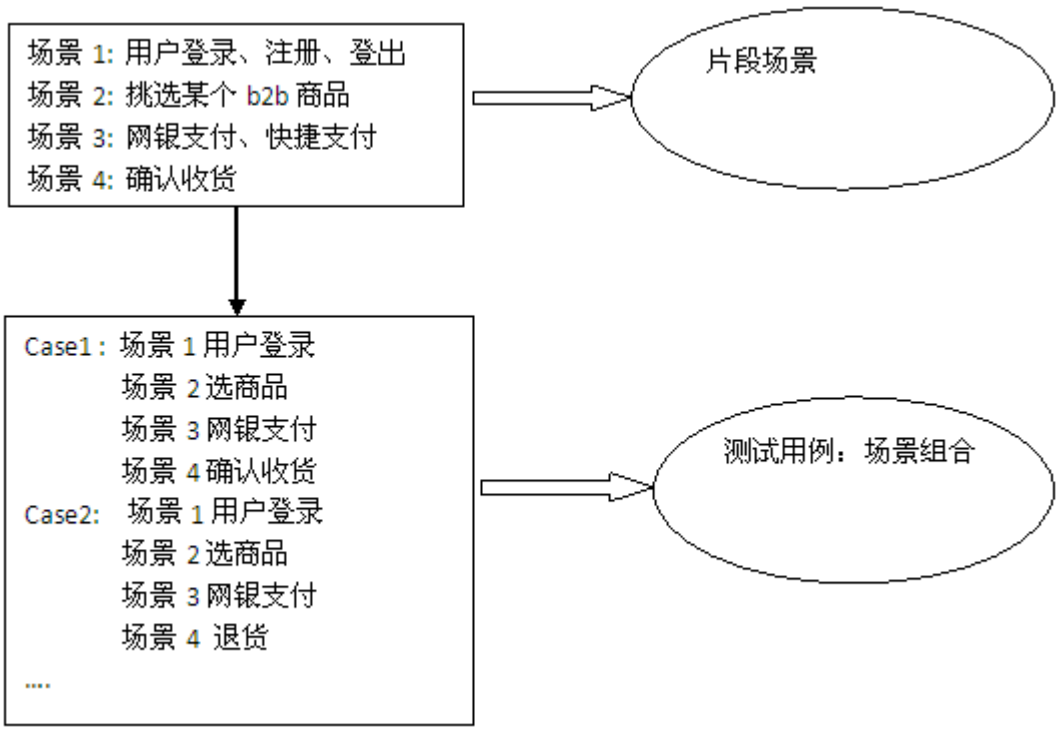
在页面层面，也是测试人员工作量最大的地方，需要对待测试的每个页面进行转化。针对每个页面，编写对象，一般包括页面初始化，初始化/定义一些页面控件；页面操作；操作结果确认。

测试用例层面，通过调用一些页面操作、结果确认组合成 Case。

优点：对每个页面固定的框架，控件及页面层可以在开发之前就确定并不用经常修改，case 编写较容易，层次清晰，针对页面控件的测试写起来比较容易。

缺点：使用该类框架，开发人员修改、新增页面，都需要对页面对象、测试用例进行修改维护。另外，对组合式页面（页面布局分为多个区域，每个区域的控件元素不固定，依赖于上一个页面的输出），页面对象会比较复杂，页面初始化依赖于输入参数，分支多，页面动作分支多，页面检查分支多。于是，对于组合式的页面，使用该类框架，维护复杂。

下面介绍第二种框架，可以弥补上述场景的应用。



片段场景，由一些原子类操作组成，这些操作对应的页面以及控件固定，可以用 seleniumIDE 进行录制，改写成对象接口。对于组合类页面，可以将一个页面的操作分解为多个原子操作的组合。

测试用例，由片段场景组合而成。

优点：页面修改、控件修改，仅需重新录制影响到的原子操作，不影响测试用例层面。对于业务流程的修改，仅需修改测试用例的组合，无需修改控件、页面。

经验认为，录制一个页面的各类操作（第二框架）化成原子操作所需要的时间，远远小于编写一个页面（第一框架）以及操作的时间。对于后期的维护，第二个框架也会比前面框架容易得多。

我的软件测试成长之路

作者：赵婉萍

摘要：记录我从毕业到现在的软件测试成长之路，从初入测试之门，到深入了解测试，到现在的资深测试工程师，每个阶段的收获都有所不同，服役的每个公司学到的东西也不同。总之，都是一笔非常宝贵的财富吧。

关键词：软件测试；成长；

正文：

引子

我 03 年毕业，刚毕业就进了一家国企，在综合信息室的网络中心工作，干了 1 年，觉得没多大前途，主要是感觉在那里学不到东西，大学学的东西根本用不上多少，上班也就是喝茶（我还没这个爱好）、看报，然后就是那个部门有了新邮件了就打电话过来，我给他们收收邮件，（因为我们单位是军工单位，有些牵扯到国家 保密的资料，所以没有上外网）。有时候哪个部门的电脑坏了，就打个报告让我们计算机小组去维修，如果维修不了就报废，我就签个字，其他的一概不管，去这个单位，我收获最大的就是给单位建了个企业网站，以前的网站太破了，刚进去，领导就给我分了这么个任务，以前没接触过这方面的东西，我也是边学边做，其间也是拖拖拉拉，反正厂里也不急，以前在学校学的都是些书本上的理论，真正用的上的也没多少，在此期间，自学了 dreamweaver、photoshop、 flash，还会点 CAD，可怜的我大学期间没有电脑，这些东西在大学就应该会的，我工作了才学。不过学了总比没有学好，和我一同进单位的还有 20 多个学生，他们整整半年也没干个啥，整天上班就是下车间实习，美其名曰：了解流程，但是他们在实习期间因为没有领导管，车间的工人也不怎么认识他们，所以上班期间经常窝在宿舍睡大觉或者集体玩游戏，那时候真羡慕那些下车间的同事，羡慕他们不用上班，现在想想其实那样也未必是好事。也因为我刚进去就有了要干的事情，所以年终给我发的奖金是我们那批学生中最多的，但是这样我还是感觉在那里真的很无聊，完成了单位的企业网站，我就没啥事情做了。春节过完后办公室搬了新楼，我们又为新楼的网络忙活了一阵子，测整个大楼的网络通不通，我们使用的是局域网，好几天都是在整个楼上跑上跑下，测试每个房间的网线通不通，这时候对硬件有一些了解了，忙完了这个后，我就已经觉得真的该跳槽了，否则我可能就要废了。6 月底我就请假去西安找工作，听同学建议说

我刚从国企出来，如果直接转做软件开发，可能不好入门，工作也不好找，做测试可能比较好点，运气还不错，很快我就找了份测试的工作，工作找好了先上了1周的班，感觉自己对这个工作还能应付得了，就回去辞职了，因为是没到合同规定的时间辞职，所以我还交了3000的违约金，那时候我一门心思的想跳出来。

初入测试门，对测试了解很浅显

我的测试生涯就是从04年7月份开始的，刚开始因为自己所在在得单位用到的开发语言是c++，可是我在大学就学了c语言，所以我打算利用闲暇时间自学c++， 每天都有要做的事情，我给自己订了个计划，每天做什么，看多少c++，学多少测试知识等等，虽然每次都没有按时完成（可能我这个人比较懒散吧），但是我每天都在学习，这点我比较欣慰。刚开始接触测试，感觉对测试理解的太浅了，觉得做测试太简单了，就是拿个软件随便在上面用鼠标点一点，没有逻辑性，刚开始也不会设计用例，后来随着测试经验的积累，感觉自己在测试行业还是个门外汉，很多知识需要加强，像配置管理，还有版本测试方面，我根本就不懂，有时候测试环境的搭建都要开发同事来帮忙完成，那时候感觉自己好笨，真想把什么都弄懂了，但是什么事情都不像自己想象的那么简单容易，做了2个大项目的测试，都感觉不怎么理想。有的项目在马上验收的时候才发现bug，所以开发人员只能加班加点的在修改，我知道这是我的失职，但是他们从来没埋怨过我一句，这让我很感动。我还有个缺点就是脸皮薄，刚开始遇到bug，觉得不好意思给别人提，这也是那个项目最后延期的一个原因，也是我在测试中对bug的定位不够导致，应该公私分明，是bug就是bug，不能因为觉得发现开发人员的bug了，就是得罪他了。另一个原因就是个测试人员介入项目的时间太晚了，我是项目开发后期才介入的，对项目的需求搞得不清楚，好多文档都没有，什么都要靠自己去琢磨，没有概要设计和详细设计说明书，以至于到项目后期，设计改了再改，而我这个测试人员有时候却不知情，测出来bug，提交给开发人员的时候，他们却说这个已经不是这样设计的了。搞得自己干的很吃力，也很没有成就感。到了第二年公司开发部经理打算实施 CMMII，全体员工都开始学习软件工程和RUP，RUP2000上面说的那些可能不适合我们这些小公司，开发部经理就让我们在一起讨论，看那些该删除，那些该修改，这样的活动我感觉真的很不错，每周都有讨论交流会，把上周自己学习到的东西或遇到的问题提出来交流，最后公司制定了一套软件开发管理制度，以后项目的开发管理都比较正规了，这也是我感到最高兴的事情，因为管理正规了，测试工作也好开展多了。在此期间，我

还自学了 Rational 那套自动化测试工具，已经能使用 Robot 进行自动化测试了，但是仅限于 GUI 测试，VU 测试还在摸索阶段。第二年 3 月底公司接了个项目，经理决定采用正规的开发流程，需求阶段测试人员就介入，需求说明、概要设计、详细设计和部署都要有详细的文档说明。详细设计规定的一些软件规约都要记录在案以备以后测试或者修改之需，这次测试我感觉做的比较理想，但是肯定还有很多不足之处，那就是版本控制不严格，还有软件的需求变更自己和开发人员沟通的不到位。现在我还在研究 ClearQuest，我想以后公司能够用 ClearQuest 来进行缺陷管理。

慢慢深入了解测试

前面入门篇写的有些多了，上面是从我之前的笔记中摘出来的，感觉那时候对这些的印象还是很深的，所以就罗嗦的写了下来。真正让我打算好好深入了解测试还是在进入软件测试行业的第二年，在那个小公司越做越没有成就感，因为任务不多，有许多时间来自学。也有时间去了解业界测试行业的流程和标准，当时就觉得 HW 的软件测试做的比较正规，就非常想去 HW 学习锻炼下，就开始寻思着跳槽去 HW 了。因为以前学 c++，经常在 csdn 混，在 csdn 认识很多朋友，一个偶然的机会，朋友推荐去 HW，我想也没想就发了简历，然后积极的准备，由于自己态度诚恳，比较好学，面试很顺利，一面、二面、三面都顺利通过了。进入 HW 后，发现自己之前的工作环境全是在 windows 下的，而 HW 的这份工作全是 linux 下，又开始入手学习 linux，边学习边了解被测系统，被测系统很庞大，而且测试周期很长，一个版本测试周期一般至少 1 个月。

测试流程

在这里真的学到了很多测试方面的东西，对测试有了很深刻的了解。整个软件测试流程：从需求评审、项目计划、测试方案设计和评审、测试用例的设计和评审、测试执行、执行阶段的版本控制、测试执行完毕评估，每个阶段都是严格把关。每个阶段都会有专门的部门进行考核。刚开始感觉比较繁琐，很是不适应，慢慢的悟出了其中的道理，真的就像我们俗语说得那样：“快就是慢，慢就是快”。虽然看到每一步都要有专门的部门来评审，感觉过于繁琐，浪费了时间和人力，但是这个评审却减少了很多项目流程和管理中许多的失误，试想，如果没有专门的部门来评审把关的话，问题遗留到后面被发现了，再去寻找解决办法，还要考虑这个解决办法是否可行，好多设计好的东西都要推翻重来，那么，花费的时间和人力将会翻了好几番。这样看来，评审还是必不可少的。

对于测试来说，开始一个新产品的测试，经过如下几个步骤：

1、首先是需求评审

2、开发的设计评审

3、开工会

这个名词可能在别的公司很少见，在这里重点说下开工会都是干嘛的。开工会由项目经理来组织，开工会上主要是明确需求、设计、项目经理、项目开发情况（开发负责人完成）、测试部分

重点说下测试部分：1）测试计划：测试需求分析、估计、知道 TPL、项目开工会；2）测试设计：测试方案设计、测试用例设计，测试用例实例化、测试执行阶段；3）测试执行：测试执行开工会、每轮测试执行、测试设计优化；4）测试评估阶段：bug 收敛程度、测试完成标准。

4、测试需求分析

5、测试方案设计

6、测试用例设计

7、测试用例实例化

8、版本验证测试

9、开始测试执行

10、测试结束评估

测试管理工具

HW 有自己的一套测试管理工具，这套管理工具很强大，有 case 管理、bug 管理、case 版本管理、自动化脚本管理。

Case 版本管理：每个版本测试完成后会把该版本的 case 保存起来，部分 case 合并入基线版本，开始一个新版本的测试，会从基线版本中导出 case，在基线版本的基础上增加新 case。这样 case 版本管理有效的记录了每个不同版本的不同功能点。

Bug 管理：通过 case 发现 bug，会生成一个 bug 单，和该 case 关联起来，在记录 bug 的时候要求步骤简明扼要，表达清晰。在回归该 bug 的时候，必须记录每个操作步骤，如果记录不清晰会被打回的。每个 bug 都会有测试经理把关，bug 先到测试经理那，然后测试经理会把 bug 分到各个开发负责人。

自动化脚本管理：该测试管理工具集成了测试框架 TTCN（好像就叫这个名字），使用的脚本语言是 tcl。

测试培训

这里的测试培训很多，经常会有机会让你参加各种培训，如各阶段的 PTM 培训、测试方案设计培训等，让每个员工都能有学习的机会。不是只是在工作中成长，还有在学习培训中成长。下班后经常有各种相关业务培训。

自动化测试

说到自动化测试，这里的自动化测试基本上是已经做得很好了，自己发挥的余地不多了。而且测试的东西基本都固定了，要测的是一个平台，基本将近 2 年的时间，都在测这个平台，而且平台的各个模块分好后就不经常交换了，这就是有点对自己个人成长不太有利的地方，测试框架已经有测试经理搭建好了，自己就填一些代码。时间久了也就厌烦了，也对自己所负责的模块产生了怠慢心理，有点思维定势了。很难发现一些隐藏的问题。交换下各测试模块，可能就会有助于发现一些新的问题。

性能测试

由于各个模块以及固定，而且被测系统变数不大，这里的性能测试是专门有人来做的，在这里的性能测试了解的很少。

测试更加深入

由于在上面一家公司的自动化测试基本已经非常成熟了，自己发挥的余地不多了，2 年后，我又跳槽进入了互联网行业，这次跳槽，基本是从通讯行业到互联网行业的跨越，通讯行业的特点是被测系统很庞大，测试周期长，测试质量要求很高。而互联网行业的特点是被测系统不是很庞大，测试周期很短（为了适应市场快速发展的需求），测试质量要求没有通讯行业那么高。互联网行业，往往为了抢占市场先机，快速开发一个产品，测试只要没有太大的问题，就能快速上线，这里没有所谓的 bug 收敛度，也没有版本的概念，就是需求测试，新增需求或者修改需求。这里由于存在的变数比较多，所以很多东西都没有像上一家公司那么固化，有自己可以发挥的余地。比如测试自动化、测试策略等。每次接到新需求都和前一个需求存在很大的变数，比如你现在测的这个是使用 c++实现的，下一个你要测的可能就是用 java 实现的，比如现在你接手的这个项目是搜索引擎测试，下一个你接手的项目可能就是离线任务计算的测试。所以需要不断的学习才能适应不断的变化。在这里也是非常锻炼人的。让你不断的学习，不断的接收新的东西，不断的有成就感。总的感觉是工作的很 happy。

测试流程

这里说说互联网行业的测试流程。来了一个新需求后，下面的测试流程一般是必须有的

1、需求评审

产品经理、开发、测试一起来 review 下需求，确认该需求是否可实现，是接受还是拒绝

2、设计评审

需求评审通过后，开发开始根据需求进行设计，并进行设计评审，参与人：开发相关人员、测试人员

3、测试计划

根据需求文档和设计文档，测试工程师准备测试计划。测试计划包括：相关文档链接（需求、设计）、被测系统功能逻辑概要、测试内容、测试环境、测试任务分配及测试进度安排、是否需要性能测试等

4、测试用例设计及测试用例实例化

5、测试用例评审

主要参与人为开发工程师、测试工程师。

6、测试执行

开发提测后，开始测试执行

7、测试完成准备上线

待所有 bug 都关闭测试完成后，测试提交测试总结报告，等待上线。

8、测试总结

总结这次测试做的好的地方，以及做的不好的地方，好的地方继续推广给大家，不好的地方寻找改进措施。避免下次出现类似问题。

性能测试

在这里我能接手性能测试，这让我对性能测试有了比较深入的了解。性能测试不论使用何种测试工具，基本测试思想是一样的。即通过对被测系统加压，寻找系统的最大压力下的服务状态。性能测试，首先是编写性能测试方案，根据性能测试方案来开展性能测试。

性能测试方案一般包括如下几点：

项目术语、功能逻辑以及流程图、性能需求、测试环境网络拓扑图、测试机器以及相关配置、软件服务包、性能指标、测试工具、测试数据、测试方法、测试任务分配及进度、问题以及风险。

下面说说性能方案中几个重要的点

1、功能逻辑以及流程图

这里简明扼要的讲下被测系统的功能逻辑以及流程图，方便大家对被测系统进行快速理解，在 review 性能测试方案的时候帮忙寻找设计不合理的地方。

2、性能需求

如果是产品需求，一般这个是产品经理随着新需求一并提出的。如果是技术需求，如性能和优化，这个是项目经理根据目前服役的系统的性能，期望得到一个优化后的效果。

3、测试网络拓扑图

这个需要参考以后上线后的部署，模拟线上服务的部署图，获取最小模式下的系统的配置。如线上服务部署了 10 台机器，分别 5 个 client 和 5 个 server，这时候就可以用 2 台机器来模拟真实部署。

4、测试机器

性能测试机器最好和线上的机器配置相同，如 cpu、内存等的配置和线上机器相同，这样测试出来的效果才有说服力。如果性能测试机器配置低于线上配置，测试出来的结果很差，分不清是硬件问题还是软件问题。如果性能测试机器配置高于线上配置，测试出来的效果好，也不能说明真的被测系统的性能就能满足线上要求。

5、软件服务包

这里列出的软件服务包是被测系统的包版本，以及所依赖的一些基础包的版本，因为不同发的版本性能结果是不同的，如上个版本的性能结果是满足要求的，新版本上新增了一些功能逻辑，可能性能结果就比上个版本性能结果差，这时候版本的包的版本就很重要了。开发工程师可以从包的版本找到差异，那里的功能逻辑导致性能变差了，如何进行优化等。有些基础包的版本不同也会导致性能结果不同，如一些通讯需要的包的版本不同可能会导致性能结果变差，一般一些基础依赖包需要和线上版本的包保持一致。否则测试出来的结果没有参考价值。

6、测试工具

列举出性能测试需要的工具，如 http_load、loadrunner 等。以及测试命令

7、测试数据

这部分很重要。性能测试根据不同的性能需求构造不同的测试数据，如查询流程为 A-B，如果 A-B 无结果，走 A-C，如果 A-C 还无结果则走 A-D，这时候就要

准备这些数据，而且分几种情况进行，根据线上数据分析，走各个分支的百分比，算出个百分数，然后准备各个分支的测试数据，最后再准备一份给系统带来最大查询压力下的 A-D 的测试数据。

8、测试方法

压力测试还是稳定性测试。

压力测试：加压条件，加压命令，每次加压多长时间，如何进行加压的（这时候网络拓扑图就有他的价值了，从图上可以看出是对那个模块加压的，是系统加压还是分模块加压）

稳定性测试：正常压力下，系统长时间运行，验证系统正常提供服务，内存正常，无 coredump 等。

9、测试任务划分以及进度

划分测试任务，谁来造数据、多久能造好、谁来部署环境、多久能部署好环境、谁来执行压力测试等

10、问题及风险

性能测试中可能遇到的问题，如性能差、测试时间短、测试工程师并行多个测试任务等，这些风险如何预防。

作者简介：毕业后第二年开始从事软件测试工作，一直到现在，服役于各个不同行业的软件测试。

UI Automation 在功能自动化测试中的应用

作者：郭盈

1、引言

目前，在软件测试领域掌握了功能测试和性能测试的精髓，就能在测试市场中占据技术制高点。功能测试已跨越了单靠手工敲敲键盘、点点鼠标就可以完成的阶段，正朝着自动化和智能化的方向发展。市场上涌现了一大批功能测试自动化工具，提高了测试工程师的效率。但是当遇到工具本身的功能与测试工程师的需求发生偏离时，给使用工具带来了较大的挑战。而 UI Automation 作为一个应用程序接口，主要提供对 UI 控件的信息收集与控制访问，它提供的自动化库可以准确识别 windows 平台下的 UI 控件，提供了自定义方式进行自动化测试的方式，弥补了上述的不足。本文主要针对微软提供的 UI Automation 自动化测试技术进行研究分析，首先介绍了并分析 UI Automation 的体系结构，在此基础上给出了 UI Automation 的自动化库进行自动化测试的基本流程，最后基于此基本流程结合 windows 自带的计算器程序给出测试的具体实施步骤。

2、UI Automation 体系结构

UI Automation，微软提供的 UI 自动化库。它包括在.NET Framework 3.0 中，是 Windows Presentation Foundation（WPF）的一部分，可进行 UI 测试自动化。此自动化库一开始就是为可访问性和 UI 测试自动化任务而专门设计的，使用 UI 自动化库来测试运行支持.NET Framework3.0 的操作系统，例如 Windows XP、Windows Vista、Windows Server 2003 和 Windows Server 2008 的主机上 Win32 应用程序、.NET Windows 的窗体应用程序和 WPF 应用程序。

在 UI Automation 中，所有的窗体、控件都表现为一个 AutomationElement、AutomationElement 中包含此控件或窗体的属性，在实现自动化的过程中，通过其相关属性进行对控件自动化操作。对于 UI 用户界面而言，所用显示的桌面上的 UI，其实际上是一个 UI Tree，根节点是 Desktop。在 UI Automation 中，根节点表示为 AutomationElement RootElement。通过根节点，可以通过窗体或控件的 Process Id、Process Names 或者 Windows Name 找到相应的子 AutomationElement，例如 Dialog、Button、TextBox、CheckBox 等标准控件，通过控件所对应的 Pattern 进行相关的操作。

UI Automation 的体系结构如图 1：

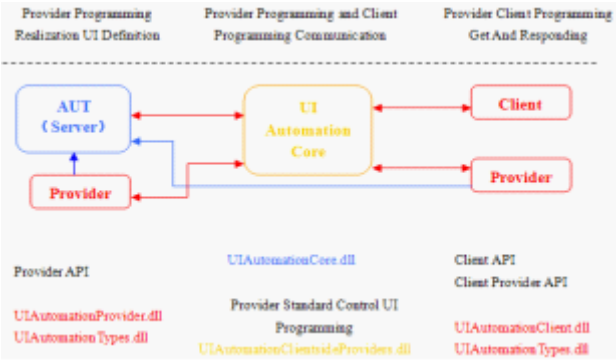


图 1 UI Automation 的体系结构

1）在服务器端由 UIAutomationProvider.dll 和 UIAutomationTypes.dll 提供；

2）在客户端由 UIAutomationClient.dll 和 UIAutomationTypes.dll 提供；

3）UIAutomationCore.dll 为 UI 自动化的核心部分，负责服务器端和客户端的交互；

4）UIAutomationClientSideProviders.dll 为客户端程序提供自动化支持。

在 UI 自动化库体系结构中使用客户端-服务器视点和命名约定。从 UI 测试自动化的角度来看，意味着所测试的应用程序被称为服务器，测试工具被视为客户端，测试工具客户端向所测试的应用程序（服务器）请求 UI 信息。UIAutomationClient.dll 库实际上就是 UI 自动化客户端使用的自动化库。另外，UIAutomationTypes.dll 库包含 UIAutomationClient.dll 和其他 UI 自动化服务器库使用的各种类型的定义。

除 UIAutomationClient.dll 和 UIAutomationTypes.dll 库外，还将看到 UIAutomationClientProvider.dll 和 UIAutomationProvider.dll 库。UIAutomationClientSideProvider.dll 库包含一组与构建时不支持自动化的控件配合使用的代码，这些控件可能包括旧式控件和自定义的.NET 控件。通常一般的应用程序使用标准控件（均设计为支持 UI 自动化），因此不需要此库。UIAutomationProvider.dll 库是一组接口定义，可供创建自定义 UI 控件和希望控件能被 UI 自动化库访问的开发人员使用。

3、UI Automation 的自动化测试流程

下面，主要描述基于 UI Automation 自动化库进行 UI 自动化测试的基本流程：

1）测试代码首先应该启动所测试的应用程序；

```
Process p = Process.Start(“..\\app.exe”);
```

2）获取对主机的桌面窗口的引用作为 AutomationElement，例如：

```
AutomationElement aeDesktop
                    =   AutomationElement.RootElement;
```

3）获取应用程序，例如：

```
aeDesktop.findFirst(TreeScope.Children, new
PropertyCondition(AutomationElement.NameProperty,"标题名(属性信息)");
```

4）获取控件的引用，例如：

```
AutomationElement aeButton = aeForm.FindFirst(
    TreeScope.Children, new PropertyCondition(
AutomationElement.NameProperty","按键的名称"));
```

5）触发控件事件，例如：

```
InvokePattern ipClickButton = (InvokePattern)
  aeButton.GetCurrentPattern(InvokePattern.Pattern);
  ipClickButton.Invoke();
```

6）设置测试结果的验证结构，例如

```
If (result == "期望结果")
    { Console.WriteLine("\nTest:Pass");}
Else
{ Console.WriteLine("\nTest:Fail");}
```

7）关闭所测试的应用程序，例如

```
WindowPattern wpCloseForm = (WindowPattern)
    aeForm.GetCurrentPattern(WindowPattern.Pattern);
wpCloseForm.Close();
```

注意：InvokePattern 是一种 AutomationPattern 类型，用于控件的触发，了解 AutomationPattern 类是了解如何使用 UI 执行测试的关键。常用的模式有 ExpandCollapsePattern、GridPattern、GridItemPattern、InvokePattern、MultipleViewPattern、RangeValuePattern、ScrollPattern、SelectionPattern、SelectionItemPattern、TablePattern、TableItemPattern、TextP-attern、TogglePattern、TransformPattern、ValuePattern、WindowPattern。

4、实例分析

我们结合 windows 自带的计算器小程序讲述如何基于 UI Automation 技术来实施功能自动化测试。

图 2 windows 自带的计算器小程序

测试需求描述：测试图中红色标记的功能，加法功能测试，即 7+8=15。

```
Process p = Process.Start("calc.exe");
AutomationElement aeDesktop = AutomationElement.RootElement;
//触发“7”键，“8”、“+”和“=”键类似代码
AutomationElement Button_7 = aeForm.FindFirst(TreeScope.Children, new
PropertyCondition(AutomationElement.NameProperty, "7"));
InvokePattern ipClickButton7
=(InvokePattern)Button_7.GetCurrentPattern(InvokePattern.Pattern);
ipClickButton7.Invoke();
aeForm.FindAll(TreeScope.Children,  new
PropertyCondition(AutomationElement.ControlTypeProperty, ControlType.Edit));
string Result= TextBox.GetCurrentPropertyValue(ValuePattern.ValueProperty);
//判别上句的字符串是否为 15，是输出 pass，否输出 fail。
```

5、结束语

本文研究了 UI Automation 的自动化测试技术，给出了详细的实施流程，给相关的 UI 功能测试领域起到了抛砖引玉的作用，还有待更深层次的理论研究和实践上的突破，在这种技术的支撑下，使企业可以更好基于此搭建自定义的 UI 功能自动化测试框架。

web 手工测试的经验总结

作者：叶子

前言

本文主要是阐述个人的 web 手工黑盒测试的工作经验

测试目的

测试并不仅仅是为了找出错误，通过分析错误产生的原因和错误的分布特征，可以帮助项目管理者（开发人员）发现当前所采用的软件过程的缺陷，以便改进；从而提高软件的质量，更体现了测试的重要性。

工作经历

工作环境介绍

09 年 3 月刚入职，也是项目初建阶段，项目组 6 个人在一个小房间，5 台台式机 1 个人用笔记本（领导）；2 张桌子比较挤，工作的地方是比较简陋；刚开始熟悉需求，然后是和同事一起部署项目，不过就是看看表结构，学习下怎么把数据入库 Oralce 数据库，后台 Oracle 存储过程开发，在前台配置业务指标配置展现，给用户做个什么小需求等等，都是琐碎的事。不过项目初建比较累，加班较多，事情也多，大概到 8、9 月份才开始正常上下班。 10 年过完年大概 3 月份左右吧，二期的项目要测试（公司给我们项目组划了一片办公地方，挺好的），项目组人手不够，老大让我转测试，问我同不同意，我想了想自己的工作内容比较杂，专注一件事情也是好事，就同意了！开始真正的测试工作。

然后老大从别个项目组调了一个有测试经理级别的人，过来协助测试，我跟着他学习了一点东西。比如测试用例的撰写，用户验收 UAT 用例和测试报告的输出。记得他说过做测试要细心，提出的 bug 要跟踪，注意页面的美观性，按钮、字体大小、字体颜色，风格要保存一致等，虽然他教的少，不过还是挺感谢滴！

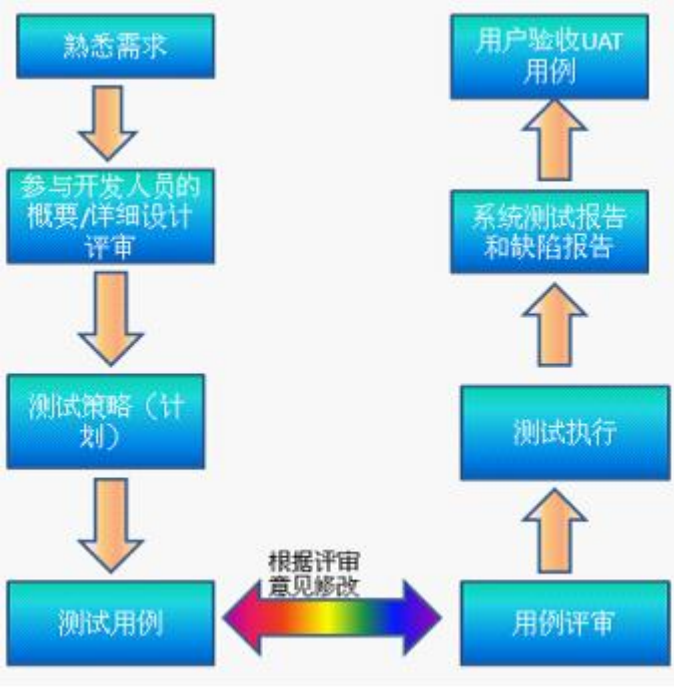
二期项目上线之后，测试工作告一段落，我又恢复了以前的工作，没有了测试工作就做业务需求开发写写 Oracle 存储过程，前台配置展现，维护下测试环境和线上的环境等等。

做了测试之后感觉自己挺喜欢这行滴，因为工作的事情比较杂想学不到什么东西，个人想专注测试，2011 年动摇了要离职的念头，不过老大找我谈了好几次话，也主动给我加了工资，就留下了，遇到这种老大，挺不容易的，对我们组员很好。

11 年项目三期测试；输出测试策略，按照计划的时候输出相应的文档，比如测试用例的输出，然后全员参加用例评审（开发、测试和 PM），会上提出用例的不足或者、与需求不符或者不完善的地方；修改了之后发送 PM，通过后执行测试。测试的时候每天晚上邮件反馈当天的工作进度，采用迭代测试的形式测试，测试环境我自己维护，一轮测试完成后，开发把 bug 修复完成，在提供一个发布包，然后验证，没有新 bug 产生后，就输出一份系统测试报告和缺陷报告（针对开发人员）。如果客户要求做压力测试，需要输出一个压力测试方案（包括场景、测试模块、测试环境等），当然方案也要评审，评审通过后开始 LoadRunner 压力测试，测试完成后输出压力测试报告。然后是一个用户验收 UAT 测试用例的输出。最后上线完成。并输出一个上线总结文件！

在工作期间带了 3 个新同事，Ta 们 3 个都不同，也许是刚开始接触测试，慢慢的成长！有一个女同事很…..我给她定了个学习系统和业务的计划，人家自己不做反而在那里看开发的代码，问她的时候她也总是没问题，偷懒很严重，如果她不是女生就和老大说不用她了……

介绍下以前公司的测试流程：



测试执行：采用迭代测试，在 BugFree 里记录 bug 一轮测试完毕，重新发布测试环境 验证前一次的 bug；如果第二轮测试有新的 bug 产生 则 再次迭代 。

2012 年我坚决离开待了 3 年的公司，去找有专业测试团队的公司，为了我的理想而奋斗。来了新公司之后才发现，测试只要我一个人 ，没得人商量，没有严格的流程，老大是做开发滴，让他把测试环境和线上环境同步，也没同步（说明一下这个公司分工很细，设计、数据库、开发、测试等比较细），上线的时候通宵不说简直是痛苦 ，眼看快过年了，犹豫着要不要重新换一份工作！！！

测试方法

先谈谈个人对测试的理解，“测试”顾名思义就是‘测’和‘试’，需要不断的测量、调试和验证，多测多试。以下介绍工作中用到测试方法。

边界测试

根据汉字的理解，边界就是“边缘界限”；在软件中的理解应用：

数字类型：是最大值加 1，最小值减一，当然也包括最大值和最小值的验证

比如 一个文本输入域允许输入 1－255 个字符。？ ？ ？正常（合理）输入：输入 1 个，小于 255 的字符，254 ，255 个字符

边界（不合理）输入：输入 0 个字符和 256 个字符 。

日期类型：如页面配置是 周 为单位的，需要验证当年的最后一周，次年的第一周，

当年最后一周与次年第一周的组合（如 2012 年 12 月 31 日至 2013 年 1 月 6 日）

等价类划分

等价类分有效和无效两种：有效等价就是对程序合理的、有意义的、正确的输入；

无效等价就是对程序不合理的、无意义、不成立的输入

如一个电话号码输入框，长度要求 15

有效输入：0-9 10 个数字正常输入

无效输入：空值、负数、小数、15 个 0 输入

边界输入：0、16 位长度的值输入

破坏测试

破坏测试个人理解和无效等价类似，输入非法、错误、不正确和垃圾数据

比如在年龄输入框输入特殊字符，字母标点符号等非正整数值。个人比较喜欢这种测试方法，虽然不成立的输入，却可以发现其中的问题，同时也是对需求的更深入理解。

数据验证

1、前后台数据一致 ： 前台正确录入信息保存后，后台数据库相对应的表正常记录（与前台输入一致）

比如：注册一个用户信息提交成功后，用户表 users 中是否正常保存了当前的录入信息。

2、存储过程验证：oracle F8 编译通过，F8 执行后 对应的数据表正常录入数据，无锁表现象（当目标表 B 表从另外一长表 A 表取值，当 A 表数据过大时要借助临时表，避免死锁、耗费资源的现象）

根据开发习惯找错误

同一个开发人员开发的模块，在不同的模块犯了错误，其他的模块也有类似的错误

比如 A 开发人员 主要负责用户、权限模块，在测试用户模块时发现用户名可以重复，现象用户名重复： 注册了两个相同的帐号，但是用户状态不同，一个是不可用状态，一个是可用状态，但是登录的时候两个都不能登录，提示“帐号不可用”。然后再去验证权限模块，角色名称也可以重复，看似小问题，但对于用户来说可能就是大问题了，因为正常状态的用户不能登录。所以开发人员的习惯也是不能忽视的！

LR 压力测试

选择好录制协议，录制脚本，根据需要添加 事物和集合点 ，使用参数化，设置 runtime-setting ，在场景执行的时候 注意观察主机 CPU 和内存使用率。

个人观点

1、立项前的需求分析很重要，与开发人员的沟通也很重要；对需求理解程度越深，对开发的思想理解越透彻，撰写的测试用例就越全面，漏测的几率也会减少。

2、关注用户的需求，注重细节，尽可能找出系统中隐藏的缺陷。

3、总结测试过程中发现的问题，做好漏测记录，避免相同的错误发生。

自动化测试之 QTP 入门宝典

作者：王海兰

摘要：本篇文章主要面向采用 QTP 进行自动化测试的读者朋友们，其中详细描述了 QTP 使用中应该注意到的细节方面。本篇为入门篇，在后续的章节中，我们还会陆续推出提高篇和升华篇。希望各位读者朋友们多多支持。

关键词：软件测试；自动化测试；QTP；

软件测试已发展为当今软件行业中不可缺少的一部分，那么在测试工作背后，我们经常面临得是什么问题？没错！就是反反复复进行相同的测试工作。

每次回归测试后，为了验证先前的功能不受新环境的影响，我们都会去执行以前执行过的案例，这样枯燥机械性的工作相信没有多少人喜欢做。那么在这种情况下，想必大家都能想到解决这一问题的办法了：自动化测试。

OK，那么下面我们就带领大家走入自动化测试的神秘世界吧。

或许刚刚入门的朋友觉得自动化测试很简单，比如我们即将讨论的 QTP，基本上拿过来就可以使用了。但是，工具的使用是有其技巧的，并且维护也是个相当大的工作量。这两点也正是我要写这篇文章的目的。

我们把该篇文章分为三章：理论、实践、维护

理论篇

如果你想在接到领导的任务后，直接着手开始使用 QTP 的话，那么前期的知识储备尤为重要，这块其实很简单，看点书籍、网上视频教学、培训班都是很好的选择，本人就是主要以看书为主，在此推荐《精通 QTP－自动化测试技术领航》这本书，每天坐地铁时都会拿出来仔细研读，甚是喜欢。

另外视频教程的话网上也有，不过资源有限而且讲得不够深入。至于培训班，那就得看你在哪个城市了，目前自动化的培训机构还不是很多，主要集中在大城市并且是权威的培训机构。

实践篇

所谓实践那么就是开始录制工作了，如果公司财务状况乐观的话，建议使用正版的 QTP 软件，这样可以减少很多由于盗版软件引起的麻烦。

在录制 QTP 时有很多问题是初学者无法解决的，下面就为大家分享一些常见的问题以及解决方法

问题 1：在录制时，QTP 没有任何反应并且也没有生成脚本。

解决方法：此问题有可能是 QTP 的版本不支持当前的测试环境，所以要更换与当前匹配的 QTP。

问题 2：QTP 在回放脚本时，找不着对象。

解决方法：QTP 找不着对象分两种，一种是对象库里不存在此对象，所有这种情况很好解决，只要把对象加到对象库就可以了。还有一种情况是对象库里已经有了对象，但是由于在执行脚本时，由于脚本执行太快，对象还未 load 完就去操作对象，这种情况我们要加入 wait 语句，比如 wait（10），即等待 10 秒后再去执行接下来的脚本。

问题 3：有些操作录制不了，比如右键、刷新页面等。

解决方法：此操作涉及到鼠标的动态变化，建议大家采用低级录制，即在点击 record 之后，再去点击工具栏上的低级箭头按钮。

问题 4：Active Screen 不能正常显示。

解决方法：在 run 脚本之前，依次点击 Automation->Update Run Mode，则可以在 Active Screen 里看到对应的 Screen。

问题 5：对于在远程端口里测试的情况，凡是遇到弹出框就不能执行。

解决方法：远程窗口不能最小化。

问题 6：如何更改服务器日期。

解决方法：此问题的解决方法直接上脚本，简单易懂

case1：将服务器的日期更改为指定日期【不用恢复】

```
Dim oShell
Set oShell = CreateObject ("WSCript.shell")
oShell.run "cmd /K date 13-03-10"
Set oShell = Nothing
```

case2：将服务器日期更改为指定日期并要求恢复

```
Dim inc_Date
inc_Date=DateAdd("m", 12,Date)
Set oShell = CreateObject ("WSCript.shell")
oShell.run "cmd /K date "& inc_Date
…//其他操作
inc_Date=DateAdd("m", -12,Date)
Set oShell = CreateObject ("WSCript.shell")
```

```
oShell.run "cmd /K date "& inc_Date
```

问题 7：如何 check web 页面的显示问题。

解决方法：添加 Bitmap Checkpoint。

问题 8：如何做到在每个脚本里都添加固定的一些脚本。

解决方法：当希望在每一个新建 action 时都增加一些头部说明，比如作者、创建日期、说明等，用 action template 来实现。

Step 1：用记事本等文本编辑器，输入如下类似的内容：

```
'Author:
'Product:
'Date: Date
```

Step 2：然后将文件保存为 ActionTemplate.mst，并存放到 QTP 安装目录下的 dat 目录。

维护篇

录制好的 case 在后期的回归测试时往往需要花费大量的时间和精力去维护，这时候我们就需要一套完善的维护计划。

由于录制好的脚本在回归测试时需要不断的被拿过来执行，有时甚至是每天都会执行多次，那么在多次运行脚本的同时也会给我们的测试环境多多少少带来点影响，从而影响测试结果。那么下面我们就针对在回归时出现的各种问题来做个小结吧。

症状一： 前几次运行时在当前页面找得到对象，但是反复执行脚本多后就会报对象不存在的错误。

对症下药：每次运行脚本都会产生新的交易数据，导致系统数据越来越多，以至于当前页面存放的永远是最新数据，而我们的目标数据就被依次放到后续的页面上，所以才会出现找不到对象的情况。面对此类问题请切记，尽量采用搜索特定的交易号去直接唯一的定位到某条数据而不是在当前页面随意的去选一条记录。

症状二：交易数据被锁。

对症下药：如果脚本里涉及到多次交易被锁这种情况的话，我们可以把 unlock 交易数据的 sql 整理一下，在每次批量执行脚本前先执行这些脚本即可解决该类问题。

症状三：对象存在但是仍然找不着，并且在 debug 脚本时可以成功执行。

对症下药：此类问题是由于对象 load 时间不够长，我们要做的就是增加 wait 时间，不要小瞧这个问题，在批量执行脚本时也是经常会发生的。

症状四：个别 case 的脚本失败后导致同一个 folder 下的 case 无法执行。

对症下药：理想的情况是在每天下班后跑脚本，然后第二天来查看执行情况，但是如果出现了如上所述的情况，这个时候我们就无法查看结果了，必然会影响我们的工作效率，所以为了避免悲剧的发生，我们可以把影响面大的 case 放到单独的 folder 下从而避免影响其他 case。

症状五：测试环境更新某个页面对象导致多个 case 受影响，万般无奈下只能一个个 Debug 失败掉的 case。极大得影响了工作效率。

对症下药：可以把这些被多次用到的页面对象放到共享库里，在录制时凡是遇到此对象，我们统一调用对象库的共享对象，那么在后期该对象发生变化时我们只需要更改共享库即可，而不需要一个个去更改页面对象。

以上五条是可以提高自动化测试效率的小策略，希望可以帮助到大家。

参考文献

[1] 余杰，赵旭斌.精通 QTP：自动化测试技术领航[M].第 1 版.人民邮电出版社，2012 年 1 月 1 日