目 录

质量好与更好

作者：罗斯汀

摘要：

本文从测试人员经常需要回答的“版本质量好不好”这个问题引申开去，结合实际案例，分析了“什么是版本质量好？”，“如何基于测试结果评估质量？”以及“如何让质量更好？”三个层次的问题。

关键词：产品质量，质量度量，质量改进

作为测试人员，尤其是测试负责人，你经常需要回答的一个问题是“这个版本的质量好么？可以按计划如期上线么？”要回答这个问题，我们首先要弄清楚“什么叫版本质量好”？其次，我们需要知道“如何有效地基于测试结果评估质量”；最后让我们一起再来探讨一下“如何让质量更好”。

一、什么才是版本质量好

“版本质量好”最直接的一个理解是：bug 少就是质量好。可是按照 bug 无法穷尽的原理，不可能测到没有 bug 的那个时候啊！bug 要少到一个什么程度才是足够好呢？有人说，我们上线前只能是评估个信心指数，关于真正的质量，还是要看版本上线后报出的 bug，如果少，那就是质量好了。听起来好像经过一个用户实际检验的过程，下的结论应该比较准了。但是单纯依赖这个简单的指标，可能会产生错误的判断。我们看看以下两个案例。

案例 1：比较某项目最近的两个版本。版本 A 上线后有 10 个生产环境的 bug，版本 B 有 18 个。我们可以说版本 A 的质量比版本 B 好么？不一定！因为若版本 B 的大部分 bug 可以通过其它方式绕过去；或者即使某个功能有问题，对用户的后续操作也没有太大的影响；而版本 A 里面有几个不紧急修复用户就无法继续操作的问题，那么哪个版本的质量更好一些呢？答案是不言而喻了。所以，除了 bug 数，bug 的严重性或者说对用户操作的影响程度也是衡量产品质量的一个重要因素。

案例 2：在生产服务器上最近一个月，系统 A 和系统 B 都各有 5 个不太严重的 bug。用户 C 同时使用这两个系统，他说系统 A 的质量比系统 B 好多了。为什么呢？原来系统 A 的界面使用很方便，对用户操作的响应也很快。相比之下，系统 B 在这方面就明显有不少不足。哦，原来除了 bug，质量其实还关注系

统的可用性和性能。因为这通常并不是功能测试的重点甚至范畴，所以也容易在项目组内部评估产品质量的时候被忽视。

好的质量，体现在产品的少缺陷、易用、高性能、高稳定性等多个方面。反映在版本中的好的质量，并不是测试出来的，而是在软件开发各个阶段逐步植入的。

二、上线前评估质量

通过各种质量保证手段在软件开发全生命周期内对质量进行植入，当然也还是需要对最终的结果，其中最重要的就是产品，进行质量评估。上线前测试人员最常用的质量评估手段就是基于 bug 评估。

在实际工作中基于 bug 进行质量评估时，往往容易犯两个极端的错误。一是完全依赖测试人员的主观印象，而不去做细致的数据分析。这固然有它的原因，因为一般测试人员还是会试图根据某些指标去度量和分析产品质量的，只是在没有深刻理解指标的内涵和经常得到非常雷同的结论时，慢慢对定量分析就越来越麻痹了，极端的情况下就开始完全相信测试人员的主观感受。二是完全依赖数据分析，而不与主观印象或者经验互相验证。这经常会发生在项目组内除了测试人员以外其它角色对于测试结果的理解时。例如，在某个测试报告中看到罗列的 bug 数就简单地对产品质量做出一些断言。这其实是很危险的。基于这两种极端的弊病，我们推荐综合主观的感觉和客观的数据分析来进行较全面的基于 bug 的质量评估。

1. 主观的感觉

一个好的测试人员对自己测试的模块的质量是有感觉的。他知道哪里比较不稳定，哪里应该在下面的测试中放更多的精力。这些信息有时不能通过测试报告完全表达出来。所以，测试主管或者项目经理最好在阅读测试报告的同时，还和相关的测试人员多一些日常的沟通，了解他们根据以往的测试经验和本次测试的情况，心理的感受有哪些是不曾在报告中提到的或者是有一些差距的。当一个项目有多个测试人员时尤其应该如此。而且越有经验的测试人员，他的主观感觉越应该受到重视，并与客观数据分析得到的结果进行互相验证。

2. 客观的数据分析

数据分析的基础是数据。如果基础错了，结果当然可能谬以千里。例如，如果对 bug 进行逐个审查，有时会吃惊地发现即使有 bug 严重性的指导规范，项目组内的几个测试人员对于 bug 严重性实际上有着不一致的标准。又如，有时我们会发现开发人员在填写 bug 的原因类别等信息时会随意选择，或者因为理解偏差导致信息输入有误。所以，每个测试人员都需要关注自己负责的 bug 的供参考、分析的信息是否真实、正确，以保证统计分析时有正确的基础。

正如一句话说的：“角度决定深度”，合适的数据分析维度也是客观的数据分析的重要保证。例如，bug 的严重性与数量的结合能比单纯的 bug 数量更客观地反应质量状况；bug 的聚集程度和修复 bug 的返工率能从一个侧面反映出版本的稳定性；bug 的原因归类能反映出软件开发流程中哪个环节（需求分析、设计、编码、单元测试等）是目前项目组质量较薄弱的地方；发现 bug 的手段分类及相应 bug 数能反映出测试人员在哪种测试方法下还应该多进行一些测试，而代码中又在哪些方面考虑得较不充分（如操作顺序、数据多样性、UI）等等。在进行 bug 属性分析的时候，IBM 推出的 ODC 分析法值得借鉴。[1]

数据分析的结果往往会呈现在测试报告中。因为测试报告的格式容易复制，这使得一些测试人员在没有深刻理解数据分析指标内涵的时候也能顺利地写出象样的报告。但这样的报告也仅止于“象样”。如果继续追问为什么要这么分析，数据说明了什么问题，为何两个维度得出的结论看似矛盾，项目组最大的质量风险在哪里等等，我们就能知道得出“质量好或不好”这个结论不是罗列一些数据那么简单。在数据和结论之间必须要架起分析这道桥梁。

三、上线后评估质量

上线前的质量评估只是一个预测，完整的质量评估还需要在上线后的一段时间继续进行。常用的上线后评估质量的方法如下。

1. 软件缺陷清除率

软件缺陷清除率=软件开发过程中发现的所有缺陷数/（软件开发过程中发现的所有缺陷数+软件发布后发现的缺陷数）

有资料统计显示，过去美国软件业的平均整体缺陷清除率只约达 85%，而对一些具有良好的管理和流程等的著名软件公司，其主流软件产品的缺陷清除率可以超过 98%。

2. 上线后 bug 数量及对客户使用影响程度严重性分析

亡羊补牢，犹未晚矣。针对上线后发现的 bug 应该每个都给予高度重视，因为它们都反应了内部开发测试的薄弱之处。通常我们邀请开发人员和测试人员一起逐个分析每个 bug，从 bug 发生的根本原因角度反思“此类问题是否有可能在内部发现或者避免；如果可能，最有可能在哪个环节由哪个角色通过何种方法发现或者避免。”有的改进是流程方面的，有的是基于特定的技术方法，也有一些是难以改进的或者从投资回报率的角度不值得改进的。对于前两种，我们都作为知识库的积累，在后续版本开发中不断提醒自己进行持续的关注和改进。

四、如何让质量更好

评价完了质量的好坏之后，当然还要探究一下“如何让质量更好？”有则改

之，无则加勉。对于这一永恒的话题，下面仅介绍一些我们在工作中经历的一些误区及修正的方法，以抛转引玉。

1. 评估当前项目组的质量现状

“如果你不知道你在哪里，一张地图是帮不了你的”。为了知道质量改进的方向，我们也必须知道我们特定的项目在当前最大的质量问题在哪里。

这个问题的答案不能由项目经理或者测试组长或者质量经理一个人包办，而要求集思广益。我们曾经尝试过包办，但后来发现这不能赢得广大项目组员的普遍认同和积极参与，因此后面的改进效果就不理想。实践证明，让每个项目组成员都能发表自己关于质量的体会、意见和建议，并且有机会得到重视和采用，对于让质量的意识深入人心有积极的影响。当然，若每个人都有自己对质量的看法，并且粗细粒度不同，角度各异，还是需要比进行一定的分类、汇总、提取。

2. 确认改进目标

改进目标的设定力求做到如下几点，并得到相关人员的认同：（1）是最需要改进的，且在下一个版本马上可以改进的；（2）少量；（3）遵循 SMART 原则；

这里强调是“我们应当设立下个版本马上要改进的目标”而非长期改进的目标。因为通过实践我们发现有一些老大难的问题，不是我们不知道问题存在，而是始终在某种现状的制约下无法用一种新的方法去解决它，所以当外部条件不具备改进的基础时，我们不如先集中力量做我们下个版本马上能做的事情。否则，追求一个水中捞月的目标只会让士气低迷，也影响其它可行的质量改进。

另外需要强调的是“改进的目标不宜太多，通常 3~5 个已经足够”。如果你的项目组还不是那么完美，每个角色都有至少一个改进的点就更好了，因为这会让所有的人都参与进来。有时即使只有 3~5 点，也不见得在长达几个月的新版本的开发过程中大家都能记忆犹新。适时适当的提醒也是很重要的。

在大致确认了少量重要的可行的改进目标后，对每个目标的设定更为关键。因为此时我们往往容易陷入一个误区。那就是在下一个版本发布后，当我们回顾时，我们无法判断这次改进计划到底执行得怎么样。

例如，开发人员反应上个版本经常出现这样的情况：一些经过讨论后的需求没有及时更新到需求文档，导致在开发阶段遗漏了某些逻辑，报了不少 bug。因此我们设立的目标是“需求分析人员在需求有变化或补充时及时更新需求文档”。新版本开发结束了，我们询问开发人员“你觉得需求文档此次更新及时么？”A 说还可以吧。B 说有些时候挺及时的，但也还是有开发快结束时再补的。C 说很好，一次也没有碰到。那么作为项目经理或者质量经理，对于这个改进目标的执行情况，你能得出什么样的结论呢？很难，是么？当然难，因为我们没有在开始的时候就把这个目标完全遵循 SMART 原则，即 Specific（明确）， Measurable

（可度量）， Attainable（可达到）， Relevant（相关）， and Timely（及时）。至少在 Specific（明确）和 Measurable（可度量） 这两个方面我们可以在设立目标时就做得更好。如果我们把目标改成“要求需求分析人员在需求讨论结束后的 2 个工作日内更新所有需求文档”，是否 B 的评价就不会那么摸棱两可了？如果我们在 bug 跟踪工具中对 bug 原因属性中添加一条 “需求文档未及时更新”，那么通过新版本中 bug 的分析，我们是否就收集到了比口头问答和回忆更为准确的数据来回答“需求文档及时更新状况”？（当然，如果你需要知道并没有反映到 bug 上的需求更新的问题，你还需要想其它相应的方法。）

又如，因为测试人员反映上一版本修复 bug 的返工率很高，我们希望能够降低修复 bug 的返工率，设定了目标“降低修复 bug 的返工率”。拿 SMART 原则一套，我们马上知道这个目标本身存在的问题了。如果修改成“降低修复 bug 的返工率到低于 10%（上一版本为 13%，各项目历次版本平均 bug 修复返工率为 8%）”，是否更清晰一点？

不可度量的改进难以衡量效果；有效果的改进受益于清晰的目标设定。如果你正受困于和上述案例类似的问题，不妨试试开始用 SMART 原则来设定质量改进目标。

3. 持续改进

持续改进意味着回顾与改进是一个闭环。前面我们提到确认改进的目标时要选择下个版本马上能改进的目标，而从持续改进的角度我们需要放眼长远的目标。分解和传承可以将短期目标与长期目标很好地结合起来。

例如，有一个问题“加强单元测试覆盖率来提高开发代码质量”在过去的一年中频繁地在我们进行质量回顾时出现，但迫于项目进度和相关技术熟悉程度的压力，从来都没有纳入过改进的日程。在新的一年里，当项目进度可以稍微放缓的时候，我们就细分了一些具体的小目标在各个版本中逐步地开展改进。如，在版本 3.0 中研究相关单元测试工具和技术，并确定我们适用的工具和技术；在版本 3.1 中在 2 个模块的新代码中进行试用；在版本 3.2 中推广到其他模块的新代码；在版本 4.0 中推广到所有模块的老代码。

又如，我们发现项目 A 最近几个版本都有一些关于改进代码质量方面的一些新举措并取得了一定的效果。于是，我们把它们补充到代码开发的规范中，要求作为基本工作要求，长期坚持。同时，我们介绍和推荐这些方法到其它有类似问题的项目而再次验证并修正我们的方法。当时机成熟，我们甚至可以积累而成整个部门或者公司的代码开发管理规范呢！由于这些改进都源于实际工作中发生过的问题，而且相应的改进方法也被实践验证过，因此传承、推广相关流程和方法能够将质量更好地持续植入每个项目的各个适用的版本，以达到我们追求的

“让质量更好”的目的。

4. 结束语

一个好的测试人员应该能够对产品质量做出正确的明确的判断，因此他应该可以回答“版本质量好还是不好”的问题。而追求更好的质量则是项目组每个成员的共同使命和长期目标。无论你所在的项目组质量状况如何，让我们一起积极探索让它的质量更好的一切方法并共同分享！

参考文献：

[1] M.Butcher, H.Munro, T.Kartschmer, Improving software testing via ODC: Three case studies, IBM systems jornarl, VOL 41, No 1, 2002

【淘测试专栏】测试团队之如何找到合适的人员

作者：含笑

都说在阿里工作日子过得特别快，从记忆中第一次接手电话和 face to face 面试 ，到后来经历了一个全新的测试团队从无到有的组建，从最初的不知所措，到后来的小有成就感，感触颇深。在闲暇之余总结了几点面试的技巧，供即将或正在进行人员招聘的同学做一些参考，希望能够对大家快速筛选合适的人员有所帮助。

一 、面试的形式：

1. 面试前需要事先做好作业

面试是双向的，是双方了解彼此的过程。因此首先要摆正自己的位置，不要给人盛气凌人的感觉。同时，尊重对方意味着面试前要做好功课。大致了解一下对方的简历。千万不要等面试开始了，还不知道对方的姓名，申请的岗位等等。在面试开始的时候，一定要简单扼要的介绍自己。这和作为主人，向登门拜访的客人介绍自己是一样的道理。

2. 破冰，让面试者尽快进入状态

遇到比较内向或者少言寡语的面试者,要尽快让他们放松。比如幽默一下，说说今天的天气，新闻等。这样他们容易进入状态，正常发挥。这里面有一个误区。为了让面试者放松，让他们上来就自我介绍。这个方法有时候会适得其反。那些没有准备过的人会紧张得不知道从何入手。

3. 多听少说，但不失控制权

有不少面试官会在面试中不停的发问。这种方式看似十分主动，但其实不一定能从面试者身上得到有效的信息。如果在整个面试过程中，面试官说的比面试者还多，到底是谁在面试谁呢？比较最有效的方法是让面试者自己多讲，面试官一边倾听，一边根据情况提问，引导并控制面试者的话题。

4. 留点时间 Q&A

无论这个时候你是否已经做出了录用或不录用的决定，都要给面试者一个提问的机会，而且要认真应答。前面说过，面试是双向的。如果你希望面试者能接受这个机会，那么这就是你说服他们的时候了。或者说，这时候是他们在面试你

了。不要因为这个过程中的失误而失去你所要的人。即使你决定不录取面试者，你仍然要完成这最后一关。因为虽然你不录取他们，但你希望他们能对你和公司留下好印象，也许可以帮你推荐更多的人，也许他们改进了以后还会回来。千万不要低估他们的口碑对公司造成的损害。

二 、面试中的忌讳：

1.不要被简历忽悠了

简历是死的，不一定能反映出面试者当下的情况。比如简历上写的是名校毕业的，又有知名企业的工作背景。但这些都是过去，不能说明面试者现在的水平。简历往往有水分，或者描述不精确的地方。比如简历上写的是精通 Java 语言。到底精通到什么程度，只有通过面试才能大致了解。简历上越是把自己写得优秀的地方，越要去挑战一下。

2.不要对面试者有任何假设

不要对面试者有任何假设，包括简历上的信息。唯一的假设就是对方不合格。因此在面试个过程中要想法设法找出面试者的问题，给最终的决定提供有效的判断依据。有些面试官看到对方有多年经验，就假设他们在某个方面是合格的，在心理上已经开始放水。还有面试官看到对方在某些问题上口若悬河，吐沫横飞，就假设对方有经验，有水平，而主动放弃了追问细节的机会。录用以后才发现此人空有理论，没有动手能力。会说不会做。录取一个不合格的人，不仅是对公司不负责，也是对面试者不负责。

3.不要把决定留给下一个人

通常，一个面试者要经过 N 道面试。最终的结果往往要大家讨论，或者领导拍板。于是，有些面试官认为自己是开始的关卡，并不重要。反正决定权在后面。有了这种心理，会很大程度上影响面试的效果。本来自己可以搞清楚的问题，却把责任推给了后面的人。或者有意问一些简单的问题，把难题留给后人。其实，无论最终是大家讨论，还是领导拍板，每个面试官的论点和论据都很重要。否则要为什么你去面试？

4.不要诱导

我们会通常问一些开放式的问题，希望给面试者一个发挥的空间。但提出问题的方法如果弄得不好，就变成了具有诱导性的问题，引导甚至迫使面试者朝面试官想听的方向回答。例如，“你是怎样看待团队合作的？”。绝大多数面试者回答时，都会试图讲述团队合作的好处，因为这是面试官想听的。像这样的问题，答案虽然多种多样，但很难从中得到有效信息。应对如流的人很可能事先准备过，但实际工作中不一定能做到。而那些回答得不太好的人，说不定做得挺好，只是在这么短的时间内总结不出来，表达不清楚而已。因此面试官在准备问题时，一

定要从面试者的角度去考虑一下，看看他们有什么样的选择。如果面试者没有选择，这样的问题问了也是白问。

5. 不要答案，要过程

所谓面试，自然要出一些题目考考面试者。特别是技术类型的面试，出些试题是很必要的。但是，我们要关注的不是面试者的答案，而是他们怎样获得答案。大家熟悉的微软，谷歌等公司面试时的开放式问题，其实就是这个目的，观察面试者如何解题。面试官一定要清楚地知道，哪些答案是死的知识点，哪些答案是活的解决方法。知识点暂时不知道没有关系，是可以通过学习得到的。而方法则不是那么容易学得到的。

6. 不要放弃细节，细节决定一切

最后一点也是最重要的一点，刨根问底的精神。无论简历，还是面试时的介绍，都是面试者事先准备过，总结过的。这些是他们的穿戴打扮。要想看清本质，唯一的办法就是让他们脱掉外面的衣服，充分展示里面的肌肉。面试官一定要注意，追问细节的目的不是拷问对方，寻找满意的答案，而是试图了解面试者对某方面的理解是否源于自己的真实经历，某些说法是否可信等等。举个例子，有不少面试者说自己的优点是学习能力强。然而当具体问到他们最近读了什么书，看了什么博客，或者在项目中学到了什么东西时，却支支吾吾，说不清楚。

三、选择什么类型的人：

我们招聘的方法，技巧再好，如果不清楚要什么样的人，也是白搭。这里列举一些我们认为应该选择的人和应该放弃的人。虽然不同公司招聘不同类型的人，但以下几点恐怕具有普遍性。

1. 有亮点好过万金油

选择：虽然有很多地方不如他人，但在某些点上有过人之处，哪怕只有一点。如果此人在某点上可以比他人做得更好，更透，说明此人有自己独特的方法或见解，在其它事情上同样可以做的更好，更透。面试官的目的就是要去发现这个闪光的亮点。

放弃：什么都懂一点，但什么都不精通。

2. 缺点与信心并存

选择：承认并了解自己的缺点，但充满信心。

放弃：自信心过度膨胀，认为自己没有缺点；或者过度缺乏自信，在面试过程中找不出自己的优势。

3. 知己知彼

当面试有工作经验的人时，他们选择换跳槽的目的很重要。

选择：对面试的公司有一定的了解，对自己的职业规划也很清晰，而且二者

的需求吻合。

放弃：对原公司极度不满，把原公司说得一无是处；对面试的公司完全不了解；工作时间久了，为了换一个环境。这三种人一定不要选择。

4.潜力股

选择：能够明显地看出他在过去的工作中学到了很多东西，能力得到了很大提升。善于从工作中学习的人有很大的潜力。

放弃：虽然有一定能力和经验，但已经很久没有进步了。这种人在环境比较好的外企和国企比较多。由于环境舒适，便安于现状，逐渐失去了进取精神和学习动力。他们虽然有能力，但潜力不大了。最好还是留在原地不动。一动反而会出问题。

自动化测试工具 Sikuli 的介绍

Founding 测试工作室

一、Sikuli 的介绍

Sikuli 是一个利用图片进行可视化检索和自动化图形界面的技术。发布的初版 Sikuli 程序中包含了 Sikuli 脚本语言，一个可视化的适合 Jython 的 API 和一个方便利用截图写出可视化脚本的集成开发环境 Sikuli IDE。Sikuli 脚本可以不经过 API 的编译器直接自动化搜索到任何你能在屏幕上见到的东西。你可以利用程序控制一个 web 页面，或者在各种操作系统上运行的桌面程序，或者是模拟器下的 iphone 程序。（from:http://groups.csail.mit.edu/uid/sikuli/）

Sikuli 可以通过构造脚本，简短的程序来扩展其他程序的功能。使用 Sikuli 需要对通用的脚本语言 Python 有一定了解。但是他不需要对它所扩展的那些语言的知识有任何了解。当程序员需要触发某个应用程序的功能时，她只需要在相关的 GUI 上面画个框框，点击鼠标截取图片，把它直接插入 Python 代码的中去。

Sikuli，在墨西哥惠慈尔土著人的语言中这是“上帝之眼”的意思。使用 Sikuli 的程序员不需要了解 GUI 内部的代码，同样，Sikuli 对它毫无了解。取而代之，Sikuli 使用计算机视觉算法来分析屏幕上正在发生的事情。“它作为一个代理，像人类一样看着屏幕”，Miller 说到。这意味着，不需要任何额外的改动，Sikuli 就可以在任何有图形界面的应用上工作。它不需要在不同的文件格式或者计算机语言之间转换，因为，就像人一样，它只是看着屏幕上的像素。

二、指南和举例

1.安装：

环境：需要升级到最新的 java runtime；

下载 Sikuli IDE：

http://groups.csail.mit.edu/uid/sikuli/download.shtml

2.工作环境介绍

Sikuli IDE 是一个方便编写与执行 Sikuli 脚本的环境，IDE 的窗口分为三个部分：任务栏，编辑窗口，调试窗口。如图 2-1。

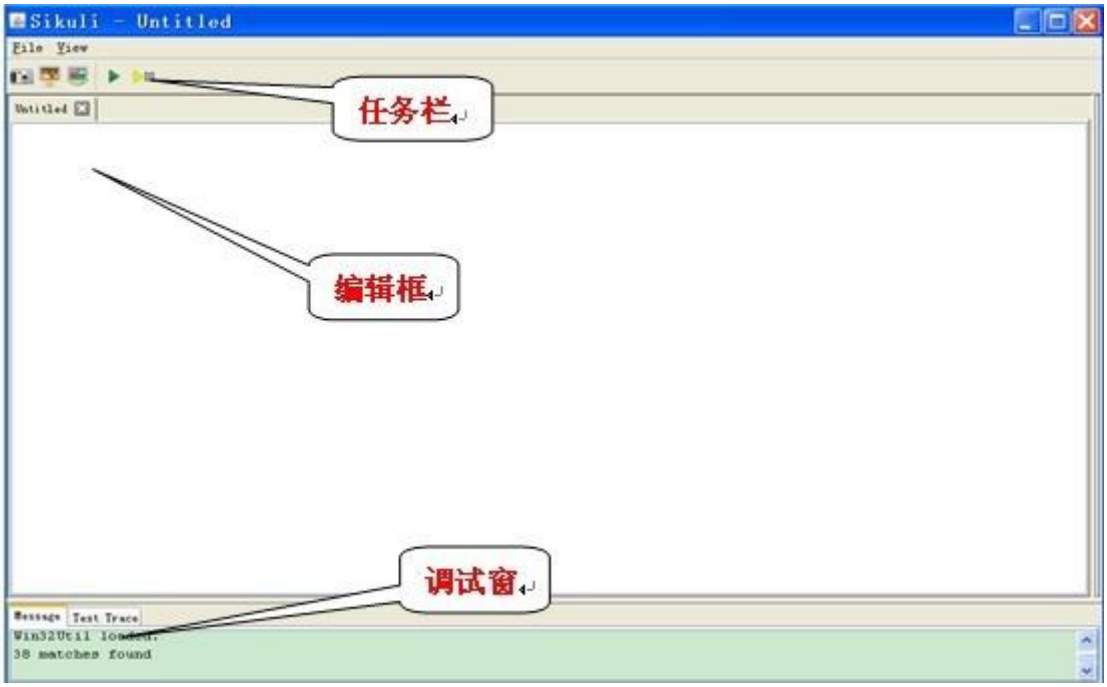


图 2-1

任务栏提供了 5 个不同的功能按钮，分别是

① Capture（捕捉）：捕捉脚本中用到的截图；

② Load（加载）：加载脚本中用到的图像；

③ Region（区域）：限制在定义的区域内搜索；

④ Run（执行）：执行脚本；

⑤ Run(animation)（动态执行）：动态查看执行的每个操作。

3.举例

写最简单的 Sikuli 脚本，我们利用一个简单 hello world 的例子，执行脚本：popup(“hello world”)；

再看一个打开网页的例子，执行脚本：

doubleClick( )

wait(about:blank)

type(“www.baidu.com”)

click( )

三、Sikuli 语法

如果命令由两个词语组成那么第二次的首字母要大写。

1）capture(*args)

截取指定区域中的图像，args 是制定的参数，是 4 个坐标，x，y，w 和 h；

2）click(img, modifiers=0)；clickAll(img, modifiers=0)

单击，多个 modifiers 之间用 or(|)分隔；clickAll()可用于批量的操作某事件；

3）doubleClick(img, modifiers=0)；doubleClickAll(img, modifiers=0)

双击；

4）rightClick(img, modifiers=0)

右击；

5）closeApp(app)；openApp(app)

关闭或者打开应用程序

6）dragDrop(src, dest)

src 为实际需要拖动的按钮，dest 为需要拖动到的目的地，比如应用在调整声音音量；

7）find(img)，findAll(img)

搜索目标范围内最符合的结果；结合使用比如可以找到目标结果然后双击或批量操作；

8）popup(msg)

弹出一个提示信息对话框；

9）type(*args)

输入内容

10）Key class

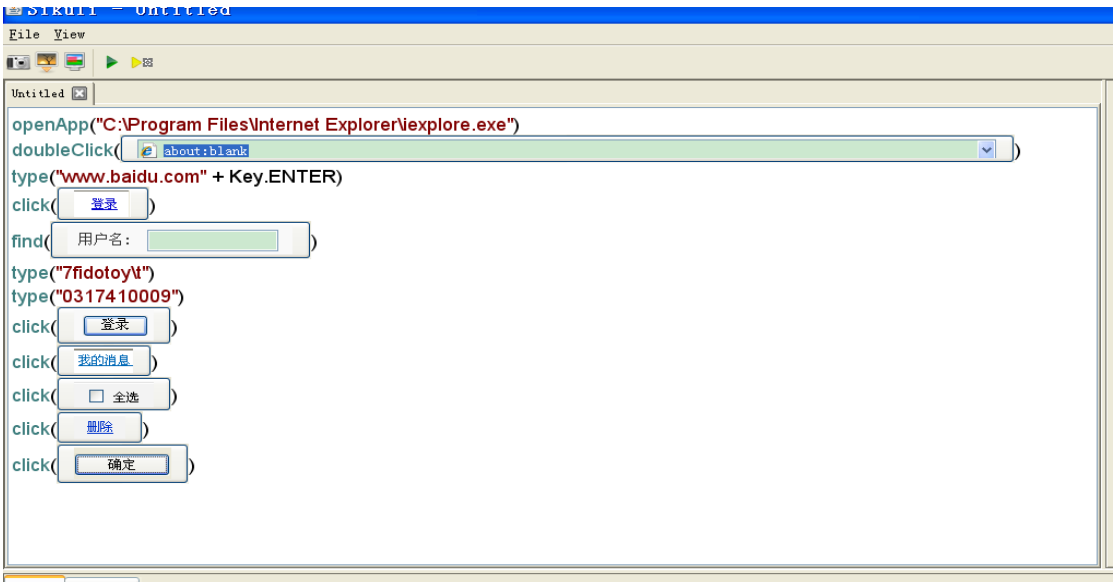
用于键盘输入，这是一个类

应用时可以使用 Key.ENTER，Key.DELETE 来表示输入，删除；一些特殊的键盘按键则用 KEY_CTRL，KEY_ALT 来表示

更多的 Sikuli 语法及脚本命令可以参考：

http://sikuli.org/documentation.shtml#doc/pythondoc-python.edu.mit.csail.uid.Sikuli.html

四、应用



百度登陆，到个人中心删除所有站内信。

五、调试过程中遇到的问题总结：

1、相同的图像如果重复，会导致运行时不能识别正确的图像而无法正常运行；

2、语法 for x in findAll(img)在实际运行时报错；

3、如果没有设置等待时间，网速较慢的情况下超过等待时间会报错；

4、在真实环境下执行脚本而非模拟环境，在记录登陆 cookies 等信息的情况下不利于多次执行脚本；

5、只认当前活动状态下的图标，且需要完全匹配，运行时如果有其余应用程序覆盖了图标，或图像颜色略显不同就会造成运行失败或是得不到预期的果；

6、IDE 环境没有撤销，返回等功能键，一旦撤销缩写的脚本无法恢复，只能重写；

7、若测试登陆界面，无法加密显示密码，安全性不高

8、保存后无法多次执行，尚不知是改变了脚本中图像属性还是因为脚本命名导致，不利于脚本的维护；

但是较适合作为 GUI 的 Unit test 存在，本身仍存在一些使用上的不便利：

参考文献

Sikuli: Using GUI Screenshots for Search and Automation, UIST 2009.

GUI Testing Using Computer Vision, CHI 2010.

当好矛遇上好盾

——测试与开发不得不说的融合关系

作者：朱锦帆

摘要：

好的项目管理，良好的测试与开发的融合与合作犹如制造一双好的矛与盾，矛可以攻击盾，盾可以阻击矛，为何不可以同时造出优良的而且互相了解的矛与盾呢。测试如矛，开发如盾，造之初期，矛若知盾之强或弱点便可适时适地探测，盾若及早告知矛自身弱点何处须重点攻击，若弱点被攻破可及早修补，弱点也成为强点了。

关键词：测试，开发，融合

有很多软件测试界的朋友，经常在面试的时遇到一个相同的问题，在测试项目或者产品的过程如何协调与开发人员的关系，即如何处理工作性质所带来的开发与测试的矛盾。

刚开始刚涉足测试行业时，我很不能体会或是理解这种矛盾，只能浅层的想象测试是挑开发的毛病，开发当然也不乐意老让人挑毛病，所以就有了矛盾。之后翻阅一些书籍，里面谈到测试和开发的协调需要上升到一定的高度，即“双赢”，就是大家本着维护好整个项目让客户达到最佳满意度这个最高理想，就可以很大程度上避免一些“冲突”，实现工作中的双赢。理论上这是个很好很和谐的概念，以此回答面试官也是一个不错的选择，不说加分起码也扣不了分。

但是在实际操作中，很多现实因素会制约这种思想，比如一些公司的测试方面的绩效考核制度就是凭借测试人员的 bug 量，大家为了年终能过的潇洒一些，还不卯足了劲查 bug，免不了有钻牛角尖的情况发生，再要是同时开发人员是看谁出的错少的话，恐怕这测试和开发的“硝烟”那是终日挥散不去啊。不过目前以此种制度的作考量的公司应该是越来越少的，大多数管理层不会如此单纯，如此量化，显得有辱其智慧啊。

还有一种情况，就是测试和开发信息不平等抑或不一致，信息不平等即在整个项目开发过程中开发人员获取的需求信息，需求变动信息，项目流程信息等比测试人员更广泛或者时间上更快捷，导致与最终测试人员的思想不一致而产生一些错报或者漏报的 bug，倘若再加上沟通不善，会对整个项目进度和流程产生很大的影响；信息不一致即开发和测试人员在理解需求上产生分歧，从需求阶段就产生的矛盾势必会蔓延至编码阶段和测试阶段，解决此种矛盾最后依然要花费很多时间而拖长项目进度。

在我看来，好的项目管理，良好的测试与开发的融合与合作犹如制造一双好的矛与盾，矛可以攻击盾，盾可以阻击矛，为何不可以同时造出优良的而且互相了解的矛与盾呢。测试如矛，开发如盾，造之初期，矛若知盾之强或弱点便可适时适地探测，盾若及早告知矛自身弱点何处须重点攻击，若弱点被攻破可及早修补，弱点也成为强点了。

测试和开发人员从项目初期也就是需求的萌芽状态就需要保持一个一致的状态，无论从需求信息，需求变更信息，风险信息，技术瓶颈信息，项目难度信息，都需要保持同等及时的认知，这就需要测试和开发人员互相沟通得当以及与外界沟通及时，还需要测试和开发人员自身技术的学习和提高才能跟上瞬息万变的市场。

在处理 bug 的问题上，好习惯的开发人员会根据自己的编码首先预测风险所在地及时的通知相关测试人员需要在该处重点狙击，防止重大问题发生；同样，好习惯的测试人员会在开发之前撰写 case 阶段就根据自身经验预测出开发人员容易忽视的逻辑从而预先告知防止错漏发生。这样在编码前和编码后都无形设置两道关卡，一可以减少重大 bug 的发生，二可以有效发现重大 bug。

此外，对 bug 的分析和争议，测试和开发人员应该较多从用户的可重现性，bug 的风险性，解决 bug 所需要的时间和价值性价比等方面去共同商榷，最后定夺出一个处理 bug 的最佳方案。一个被提出的 bug 其实并非只有解决，不解决，和无效这几种状态，可以根据整个项目的情况定夺它，目前不解决也不一定是一辈子不解决，有的可以治标有的可以治本，也如同蛀牙一样，可以补可以拔，也可以再植入新牙。最重要的，是它对最终的产品的影响力和投入比的估计要得当和适合，才能产出一个高性价比的产品，才能获得更好的用户口碑，最终获得收益的最大化，当然我们的开发和测试也获得了双赢的局面。

以子之矛，攻子之盾，在古代或许是绝对的对立，但是在今天，和谐社会，我们也可以和谐的测试与开发，马可思主义哲学说，矛盾可以促进社会的进步，那么我们共赢的矛盾可以可以放大进步的脚步。谁说一支好的矛一定要攻破盾，一支好的盾一定抵御得了任何矛，我们的好矛和好盾，不断的磨合和进步，终有一天会默契得强过任何矛盾，所向披靡。

利用云计算技术实现云测试

作者：丁志义

摘要：

随着云计算时代的到来，人们应用信息方式将发生改变，同样也会改变提供软件服务的企业交付模式、研发模式和软件测试方式。基于云计算技术的软件测试方式即是云测试。本文从云计算的概念引入什么是云测试，简单分析了哪些软件测试项目可以做云测试、为何要做云测试、如何进行云测试，是对未来测试方式方法的一种探讨。

关键词：云计算　云测试

一、云计算（Cloud Computing）简介

狭义云计算是指通过网络以按需、易扩展的方式获得所需 IT 基础设施的交付和使用模式。广义云计算是指服务的交付和使用模式，指通过网络以按需、易扩展的方式获得所需的服务。这种服务可以是 IT 基础设施、软件、互联网应用相关的，也可以是任意其他的服务。

云计算作为一个新名词，它既不是一项新技术，也不是一个新概念。云的含义绝不仅仅是针对计算，而是 IT 系统建设的一个总体方针和大势所趋。云代表的是一个崭新的 IT 应用时代。

2002 年，IBM 首次提出 On Demand 随需应变，随后 HP 提出了 Utility Computing 效用计算，接着 H3C 提出了 ITоIP。甚至在更早的上个世纪 90 年代中，全球各地就出现过一批以 ASP（应用服务商）、SSP（存储服务商）为运营模式的商业探索者，他们都是云计算的先驱和实践者。上述概念或商业构想与今天的云计算并没有本质的差异，都是对同一个 IT 发展愿景进行的不同角度表述。 这个愿景就是希望 IT 资源能够有一天像今天使用的电力、自来水一样"即插即用"，不需要关心“电”从何处来，“电”是怎样产生的，运输设备是什么。这些 IT 资源包括网络应用、软件、硬件设施等。

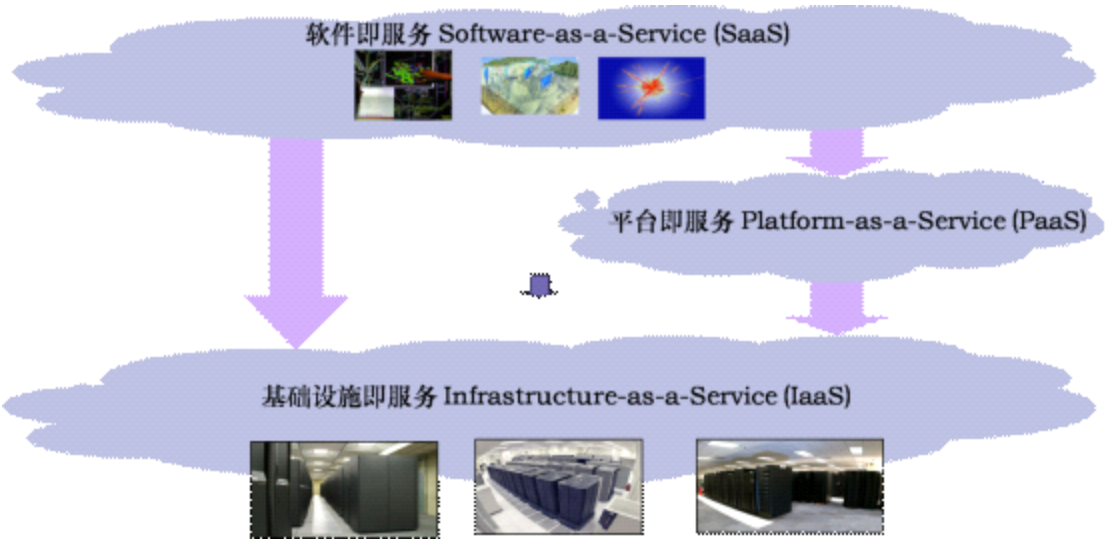
例如一家企业，他需要信息化办公，以往的模式是：企业花费大量资金采购硬件（机房、计算机）、布置复杂的网络、购买操作系统和办公软件、管理软件等、配置专业的 IT 管理人员等，有的设备或软件利用率还很低，实现信息化过程耗时、耗力、耗资金、更耗费社会资源；且日常使用还需要大量投入：例如设备保管、系统维护，软件升级等。而在云时代中企业只需要简单的培训，操作者通过简单的个人终端（显示器，手机等）接入云服务就可以实现系统化、自动化办公和管理需要，享受着更加质优、价廉、节能、环保的云服务。企业无须关心

数据存放在哪里、怎么实现，不再采购大量的硬件和软件，不再需要布置复杂网络，这些事情交给提供“云服务”的公司去完成。企业可以视同它们为躲在“云层”后面我们看不见的跑来跑去的“雨雾”一样，只关心落下的“雨滴”。也可以视同它们是在幕后的从没见过那些导演、化妆师等，我们只关心台上正在演出的这一幕和这熟悉的演员。

从上看出，提供云服务的企业应该为网络营运商、数据存储商（包括硬件）、软件提供商，或者是他们的组合。也许企业只需同以上一种云服务商打交道就能获得企业所需的 IT 服务，企业只需按流量或协议交付租金即可。例如用友软件的伟库网正是提供企业管理云服务的供应商之一，企业根据自身业务需要选择性地购买财务管理、进销存管理等服务，按年支付费用即可，价格低廉。

由于云服务要在网络环境下实行，所以软件 B/S 架构的应用方式必将是一趋势，否则很难提供云服务。

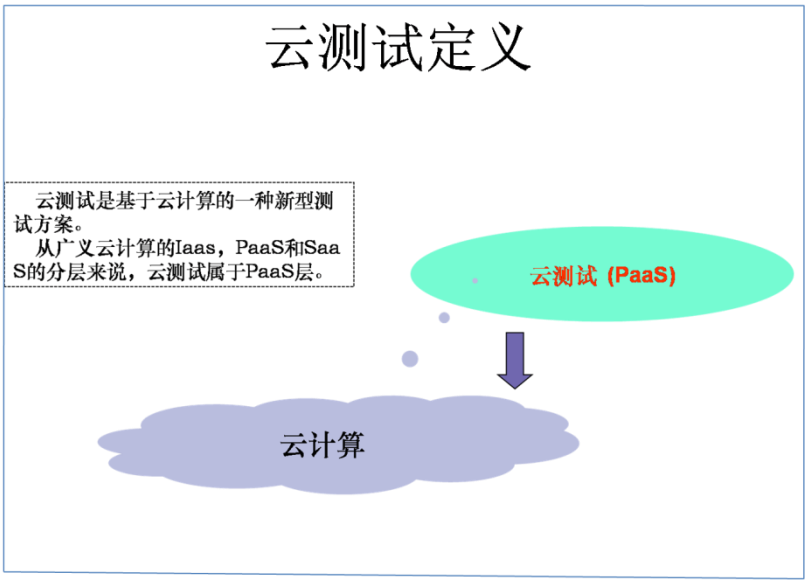
云计算基本概念示意图如下：



二、什么是云测试（Cloud Testing）

云计算时代改变企业 IT 应用模式，也会改变提供软件服务和云服务企业的交付模式、研发模式以及软件测试模式。

虽然对“云”的测试是必须的，但不是云测试的范畴。云测试不是去测试云，而是基于云计算的一种新型测试方案。简单的说云测试也是云计算技术。从广义云计算的 IaaS，PaaS 和 SaaS 的分层来说，云测试属于 PaaS 层（如下图）。它是提供软件测试工具（包括性能测试、功能测试工具等）的服务商，提供一个测试平台，软件开发企业在其平台上进行相关自动化测试、环境适应性验证测试等，企业不再是在本地计算机上安装和使用这些工具。这种无须本地安装和配置测试环境，在远程测试平台上进行适应性、性能、功能自动化等测试的方式就叫云测试。



三、哪些测试项目可以做云测试

通过云测试的定义我们看出：凡是测试中需要使用的软件工具和环境都可进行云测试，当前适合做云测试的项目或内容大概有：

硬件环境：测试软件在不同应用场景下对硬件环境的要求；

软件环境：操作系统、数据库、浏览器等，测试软件对不同运行平台的适应性；

适应性软件：防火墙及防病毒软件等，测试在安装不同防火墙及防病毒软件时，软件运行可靠性；

功能自动化测试：进行软件自动化测试；

性能测试：进行软件性能和压力测试。

随着云计算技术的发展，为软件测试服务的各种应用亦将得到发展。适合做云测试的项目也将不断增多。

四、为何要进行云测试

1、节约成本需要

每个企业都在追求成本最低和利润最大化。软件测试作为研发生产过程的一部分也有降低成本的要求，即使用最少的机器购买最少的测试软件来完成软件测试工作。利用云测试可实现巨大节省，不需要购买或准备很多的个人电脑，购买和安装各类测试用软件，也不再需要部署复杂的网络。只需要列出测试目的、环境的要求、虚拟机台数、何时间断租用即可，实现按需支付。例如购买一套自动化测试软件至少花 8000 元钱，测试中只需要使用 2 个月，但如果按 800 元/月租用该软件云测试平台，只需要支付 1600 元。同时随着企业软件版本和技术的发展，依赖的测试软件或环境亦需要升级换代，又会产生升级和维护费用。而在

云测试环境中这些因素都无须企业考虑，交由提供云测试服务的供应商完成即可。

2、提高测试效率

用云测试这种方式，极大地减少了测试环境搭建时间，如机器和网络准备、操作系统安装、各种测试工具软件安装等都将节省，只需提前将需要的配置环境告之云测试服务商，到时间直接使用即可。由于是基于网络上的应用，当测试中遇到软件使用上等问题时，亦可获得云测试服务商远程快速支持，而很少会出现停滞甚至停止测试现象。

3、性能测试更接近真实

在云测试平台上进行性能测试，可以开启更多的客户端，且本身就是外部网络应用而非企业内局域网模拟，这样的场景应用覆盖更接近真实。能够尽早发现和应对意料之外的流量高峰，让测试软件获得巨大的性能改善。

4、外部环境变化

随着云计算技术的进步，提供软件服务的企业将通过网络化方式（云计算）交付产品和服务，我们只能通过“租用”而非“购买”的方式使用这些测试软件时，只能进行云测试。这就好比你拿着电卡到银行支付电费，可银行不收现金了，你必须先开个银行账户并存入一定量的钱，然后通过账户自动代扣电费一样，你别无选择。

五、云测试缺点及规避

1、安全问题：包括企业信息安全和网络安全

使用云测试平台进行功能或性能测试中，企业软件如何实现相关功能的逻辑信息和技术手段等会部分体现在测试脚本中，软件的弱点和漏洞以及性能状况也将会体现在日志中，这些信息如果被泄露甚至落到竞争对手中，将对企业产生不利。

同时云测试基于网络，可能会出现网络中断、网速不够、病毒攻击等。

对于以上存在的缺点和风险，可以用对银行认识和态度来克服习惯性性保守思维问题：在当今金融服务发达时代我们不担心银行中的存款丢失，也不会担心到银行取钱出现断网而取不到钱的现象。虽然极小概率事件可能会发生：如存款少了、个人信息被泄露、出现排队现象等，但比起将钱存在家中，出入携带现金风险更小。我们亦享受着银联支付、电子支付的便捷好处。所以对安全的担心和对网络稳定性担心是不必要的，对企业技术信息和数据问题以及网络安全问题，也需同提供云测试服务的供应商签订合同,就好比你在银行开账户存款时，银行对你的账户信息有保密义务和保障义务一样。

将云计算类比为银行系统可能更为贴切：交易要以信任为基础，没有信任就

没有银行。云计算和云测试基础也是信任，如无信任，“天空”也将无“云”。

2、云测试适应范围限制

B/S 应用的软件（例如网站）利用云测试最为合适，而对 C/S 结构软件适应性较差，仍需在云测试平台中安装被测试软件，实现云测试稍微有些复杂。

对因保密等原因禁止接入外网的软件研发企业，也无法实行云测试，只能按原来方式测试或在企业内部搭建私有云环境进行软件测试。

六、云测试实现方式

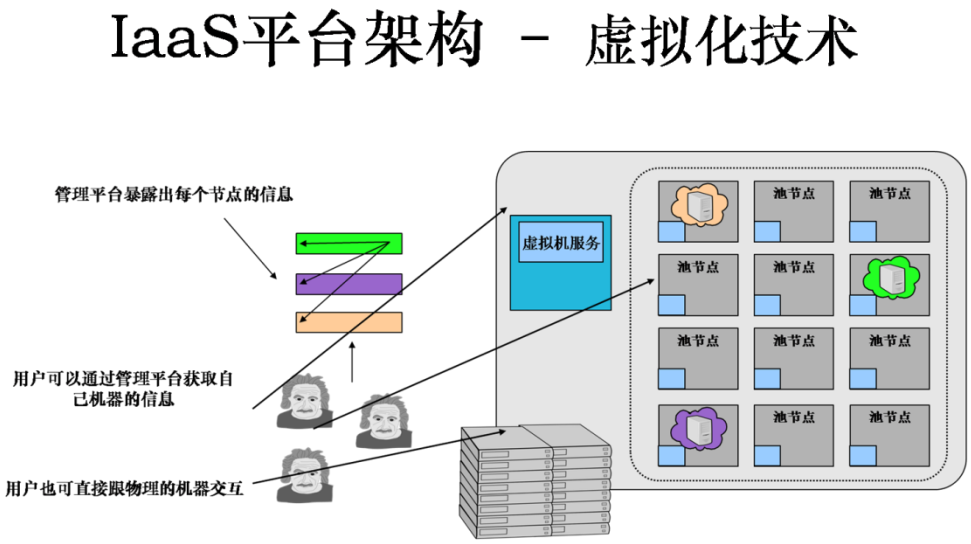
当前提供云测试服务的方式大概有以下两种：

1、利用 WEB 访问云测试服务：

在访问云测试服务商提供的网页中进行本企业软件功能测试和性能测试等，这个对 B/S 结构的基于 WEB 应用软件产品很好实现，嵌入公司软件环境即可实现功能自动化测试和性能测试。

2、利用虚拟机技术：

云测试服务商提供虚拟机 IP，客户通过远程桌面等方式连接虚拟机使用。通过虚拟机可以直接测试任意 WEB 网页，但对非 WEB 方式应用的软件，仍需要安装在该虚拟机上的才可进行相关测试。示意图如下：



实现云测试的主要过程大概如下：

1、用户登录云测试服务商网站注册使用者信息，作为管理和使用云测试平台合法认证标识，此过程可选，也可通过商务洽谈方式实现合法应用；

2、用户申请测试平台资源，申请中需要描述虚拟机环境配置要求，如操作系统版本、浏览器版本、内存大小、硬盘大小、硬盘转速、网络带宽、防火墙等以及使用时间范围并发数等信息；

3、云测试服务商审核申请并配置好测试平台；

4、根据租用协议，用户（网上）预支付服务费；

5、用户使用注册号或服务商指定的用户及密码通过远程或者 WEB 应用方式登录云测试平台，进行相关测试直到结束；

6、根据流量或时间结算实际费用，租用结束。

七、搭建企业私有云测试环境

借鉴云测试思路，在“大云”时代到来之前企业内部也可以先行搭建自己的云测试环境，以享受它的优点。因保密等原因而无法接入外网的软件研发企业亦可按此方法进行云测试。

自动化测试云环境：找一台性能好一点的机器或服务器安装自动化测试软件，如 Loadruner、Robot 等工具，各业务测试组分别登录这台机器进行各自的功能自动化测试即可，而不需要在本地安装独立的自动化测试环境。

多机器环境搭建：使用虚拟机技术快速在服务器上复制和启动若干个虚拟机，各测试端通过远程桌面或 VPN 等方式分别接入自己的虚拟机进行客户数据升级、环境、压力、性能、适应性软件等项目测试。

参考资料：

1、http://www.51testing.com，51testing 网站《云测试》，作者：qiaoqiaoni；

2、《从云计算谈 IT 建设的新思路和新方法》。

兼顾兼容性测试的冒烟测试和系统测试过程

作者：文青山

摘要：

如何在冒烟测试和系统测试的过程中，安排兼容性测试的方略，并在兼顾兼容性测试过程中如何安排测试策略；以科室现有经验为背景，详细介绍了前期就介入兼容性测试的指导思想、工作流程图、策略流程图等相关信息。

关键词：兼容性、冒烟测试、系统测试、策略、主浏览器

一、有关冒烟测试的策略和工作流程

1、原因和范围

我们先来看看对冒烟测试的定义，某书上写的是冒烟测试指对每一个新编辑的正式版本进行预测试,确定该版本可以进下一轮测试。从上面的定义我们可以得出，冒烟测试的特点在于重点功能的“把关”，所以我们在冒烟测试用例时只涉及到重点功能，而且需要建立一个统一的通过标准。另外，由于 Web 系统浏览器市场的广度，我们在冒烟测试时得选取一个主浏览做为一个平台。为什么要建立一个通过标准？其主要是防止当冒烟测试不通过时，测试组有能令人们达到共识的依据进行裁决。选取主浏览器，其主要是提高测试人员在工作过程中的专一性，以及保证重要功能在主浏览器上的实现成度，以前尝试过冒烟测试时多个浏览器同时操作，效果相当糟糕。

2、流程

2.1、主要思想

依据需求中的浏览器占有市场率，选取主浏览器；

高优先级的 Bug，需要优先验证，通过验证高优先级 Bug，预估开发人员修改 Bug 的质量；

在软件发布前就确定需求是否完全实现，避免需求丢失；

通过测试人员交叉测试，避免测试人员思维僵化；

依据冒烟测试通过标准，提高版本和测试的效率。

2.2、导入此流程的条件

有冒烟测试计划；

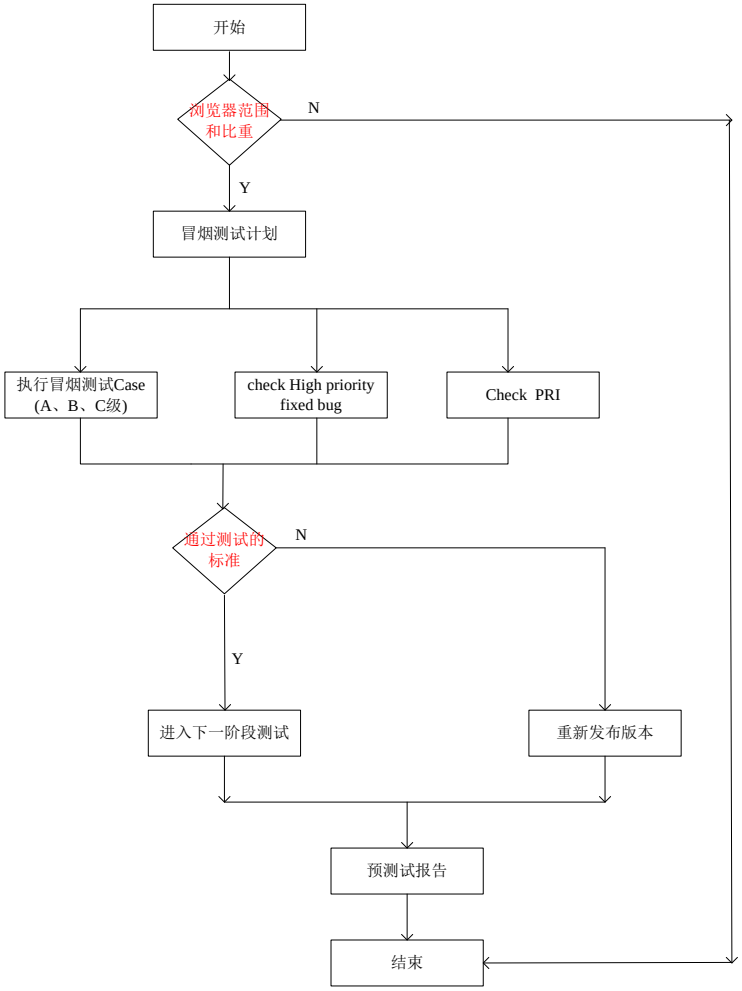
确定冒烟测试的主浏览器；

有冒烟测试用例；

建立了具有共识的冒烟测试通过标准。

2.3、流程图

冒烟测试的基本策略：
1、只选取需求中的主流浏览器进行测试
2、执行所有冒烟测试用例
3、验证fixed bug
4、验证所有PRI
5、测试人员交叉测试不同模块的用例(可在冒烟测试不同版本的计划中体现）
6、发现Bug后，需在需求规定的浏览器上做验证(由于IE浏览器不同版本在电脑中不兼容,所以至少要保证Bug在不同类型的浏览器中进行确认)
7、测试Leader确认Bug是否有效



二、有关系统测试的策略和工作流程

1、原因和范围

系统测试，是整个测试执行工作过程中的重点，在此阶段将对系统进行全面的测试。而浏览器兼容性测试的介入时间，有利于更快地了解 Web 系统的功能的对浏览器的支持效果，从而为兼容性 Bug 争得更多的解决时间。从以前项目的经验来看，我们在进行兼容性测试时都是选取主浏览器测试的同时，兼顾其它浏览器进行测试，从执行效果来看这种方式存在着一些缺点，比如测试人员工作的专一性不佳，往往忘掉还需要在其它浏览器中进行测试，另外在策略上体现得也不够明朗，并且在多人合作时有的测试人员由于个人使用浏览器的爱好，存在着避重就轻等现象。

2、第一阶段（初探）

2.1、主要思想

依据需求中的浏览器占有市场率，组成系统测试的先后执行顺序，按此顺序进行一轮全面用例的验证，立争通过此轮测试发现由于浏览器不兼容导致功能不能实现的 Bug，同时发现基于业务逻辑上的缺陷。

通过测试人员交叉测试，避免测试人员思维僵化。

依据第一阶段测试通过标准，预估功能在浏览器中的实现状态。

验证 Intest Bug，预估开发修改成功率。

通过发现的缺陷在其它浏览器上验证，排除是否为兼容性缺陷。

2.2、导入此流程的条件

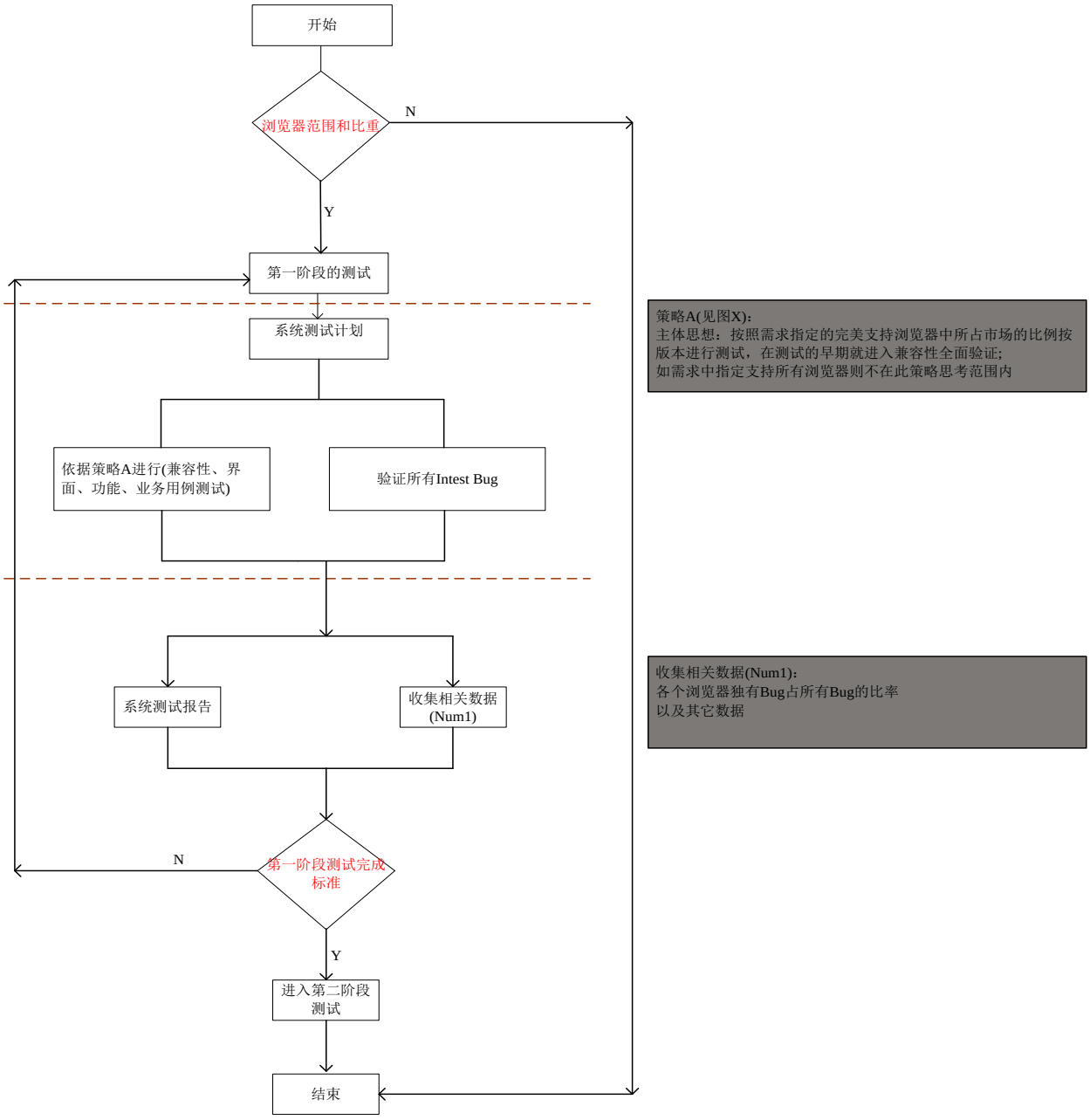
有系统测试计划。

确定每一次执行用例的主浏览器。

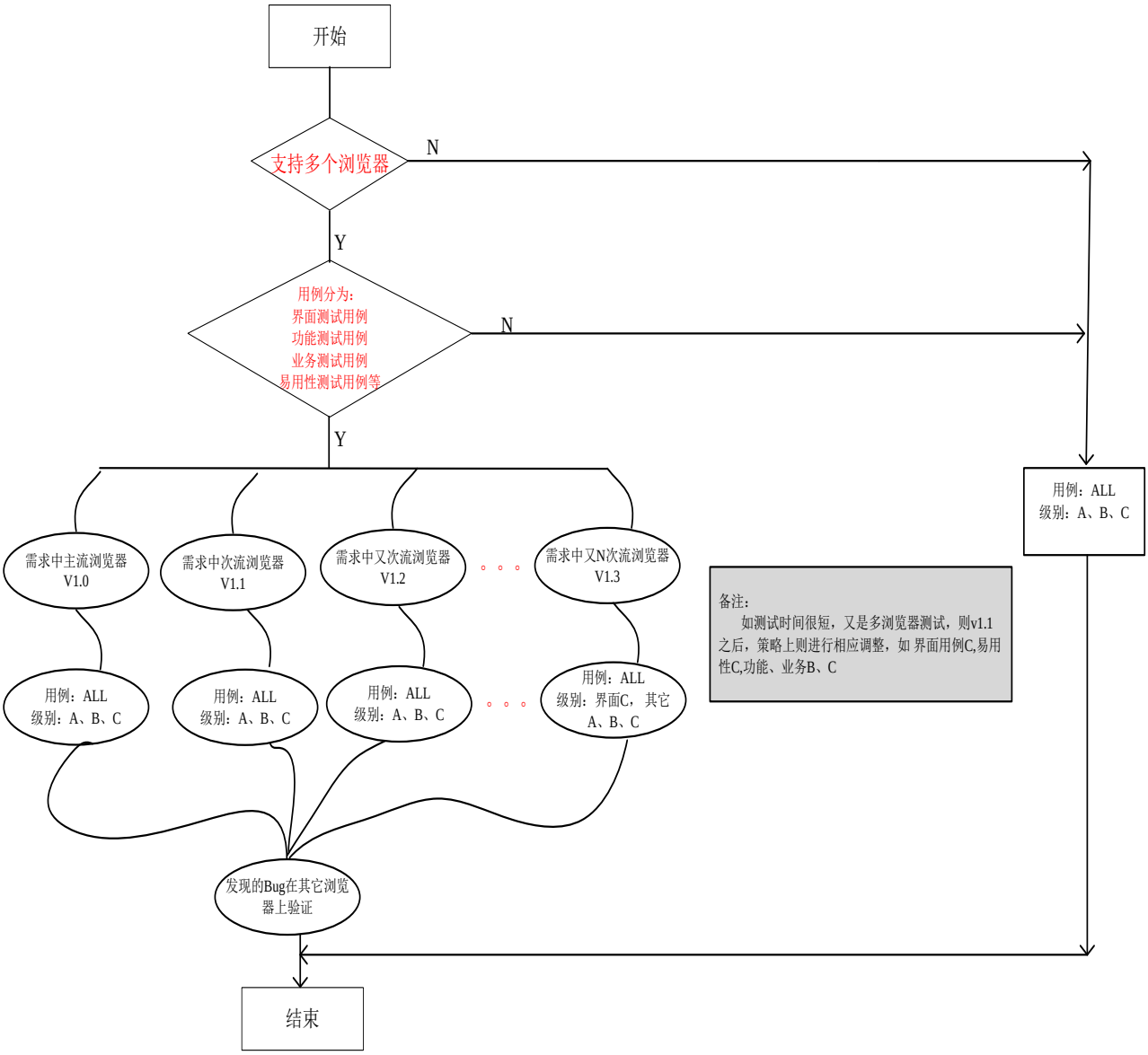
测试用例的组织架构上分为多种用例（如：界面+功能+业务+易用+控件等）。

建立了具有共识第一阶段测试完成标准。

2.3、第一阶段工作流程图



2.3.1、第一阶段策略 A 流程图



3、第二阶段工作（全面验证）

3.1、主要思想

当执行完第一阶段的测试后，关于兼容性的缺陷已呈一定比率的分布，按照 2/8 原则，我们认为兼容性缺陷较多的浏览器将会导致更多的缺陷，这时我们选取该浏览器为主浏览器进行测试，由于此阶段将会进行多次测试，所以每次的主浏览器都将不同，这样随着测试的推进，无形中将兼容性测试包容在多轮中，避免过去仅执行一两次浏览器兼容测试而导致发布后兼容性缺陷仍存在的尴尬。再配合系统或模块稳定性成熟度的分析，加以不同的策略覆盖，减轻测试组的工作压力，将测试组的工作重点由全面撒网向重点钓鱼推进。

通过测试人员交叉测试，避免测试人员思维僵化。

依据第二阶段测试通过标准，决定是否进行回归测试（即进入第三阶段测

试）。

验证 Intest Bug，预估开发修改成功率。

通过发现的缺陷在其它浏览器上验证，排除是否为兼容性缺陷。

3.2、导入此流程的条件

完成了第一阶段的测试工作

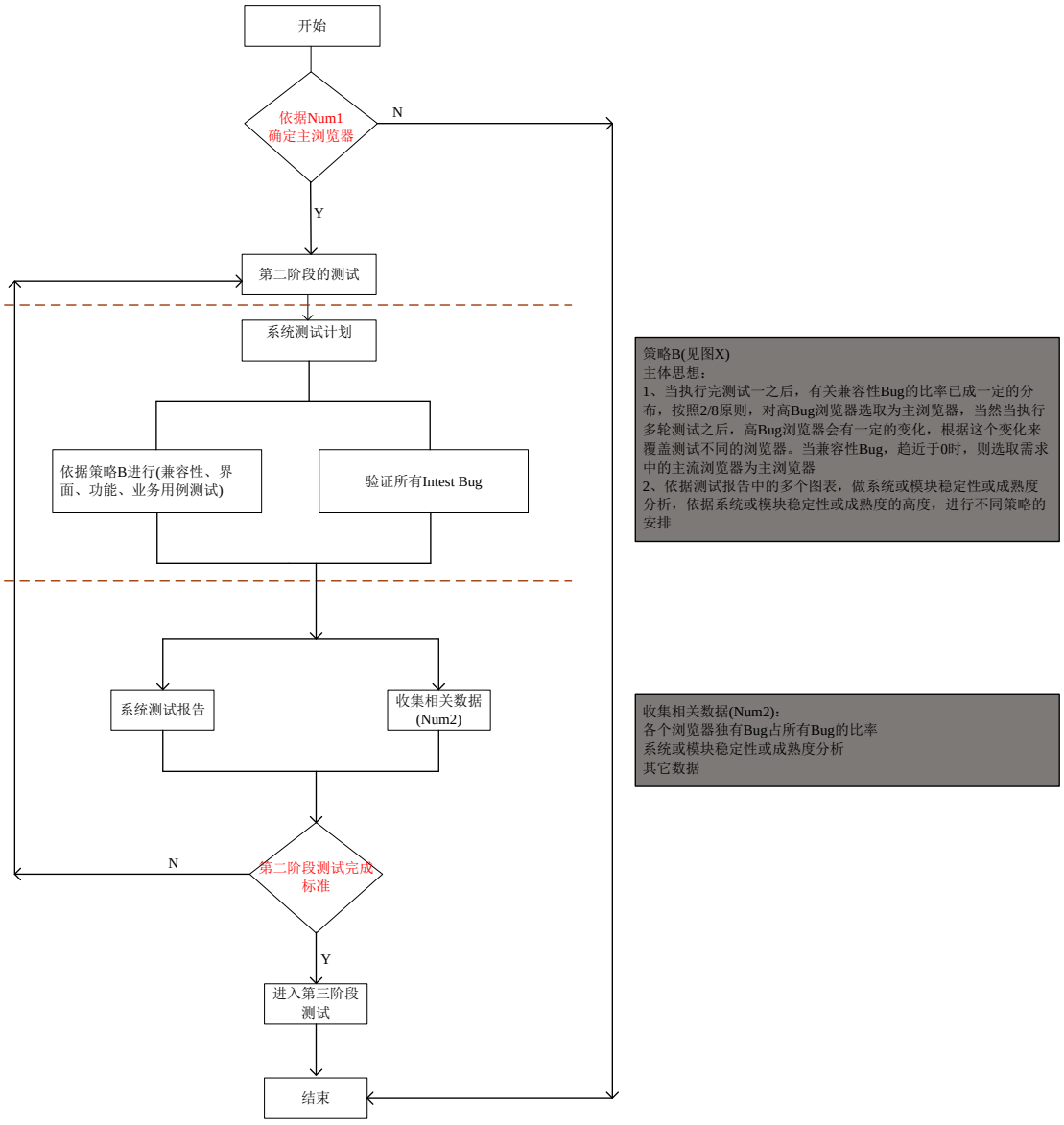
有系统测试计划。

确定每一次执行用例的主浏览器。

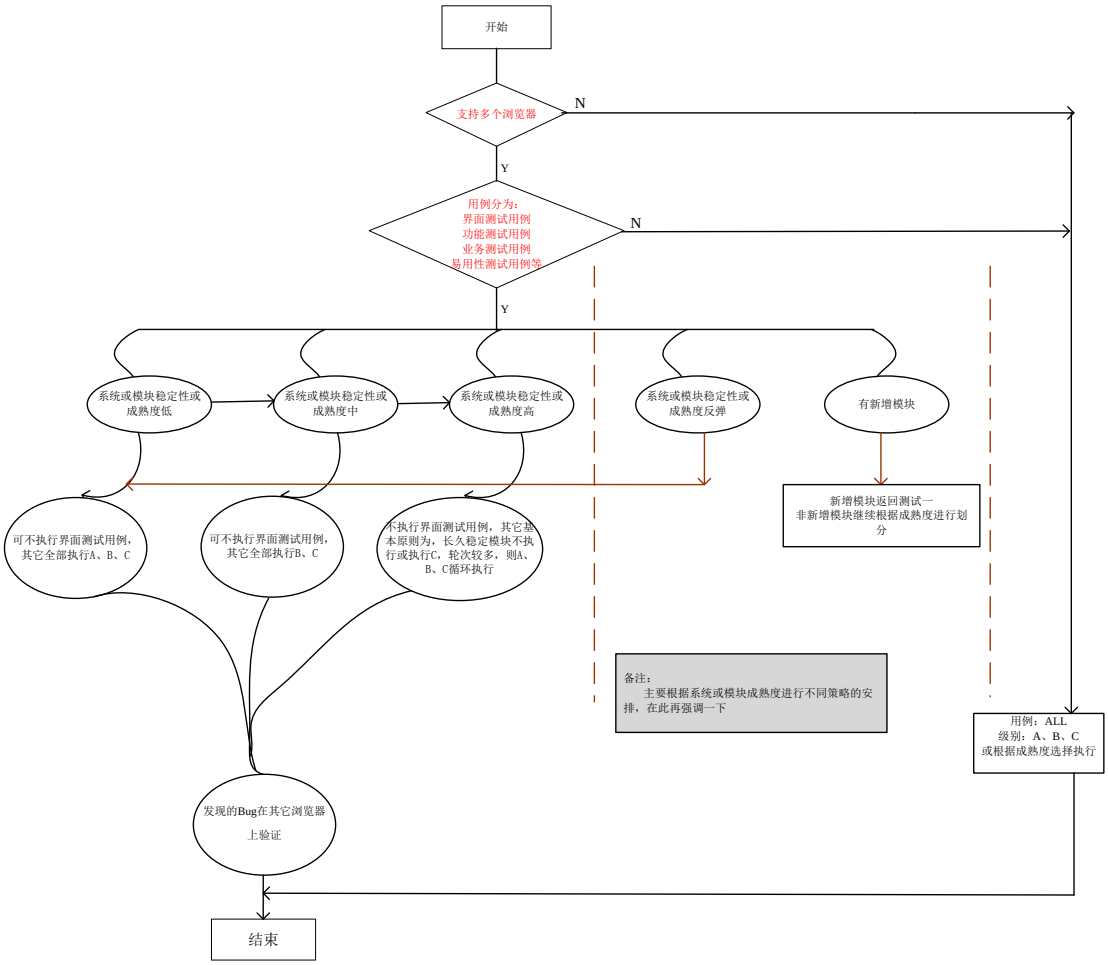
测试用例的组织架构上分为多种用例（如：界面+功能+业务+易用+控件等）。

建立了具有共识第二阶段测试完成标准。

3.3、第二阶段工作流程图



3.3.1、第二阶段策略 B 流程图



4、第三阶段的工作（即将消亡）

4.1、主要思想

当依据测试计划和系统或模块成熟度分析后，系统或模块已经达到了预期要求，那么系统测试就将至此停止。

依据回归测试报告或缺陷分析报告以及软件测试停止标准，决定是否停止测试，如否，则按相应策略继续进行测试，如是则提交系统测试停止申请或报告。

4.2、导入此流程的条件

完成了第二阶段的测试工作

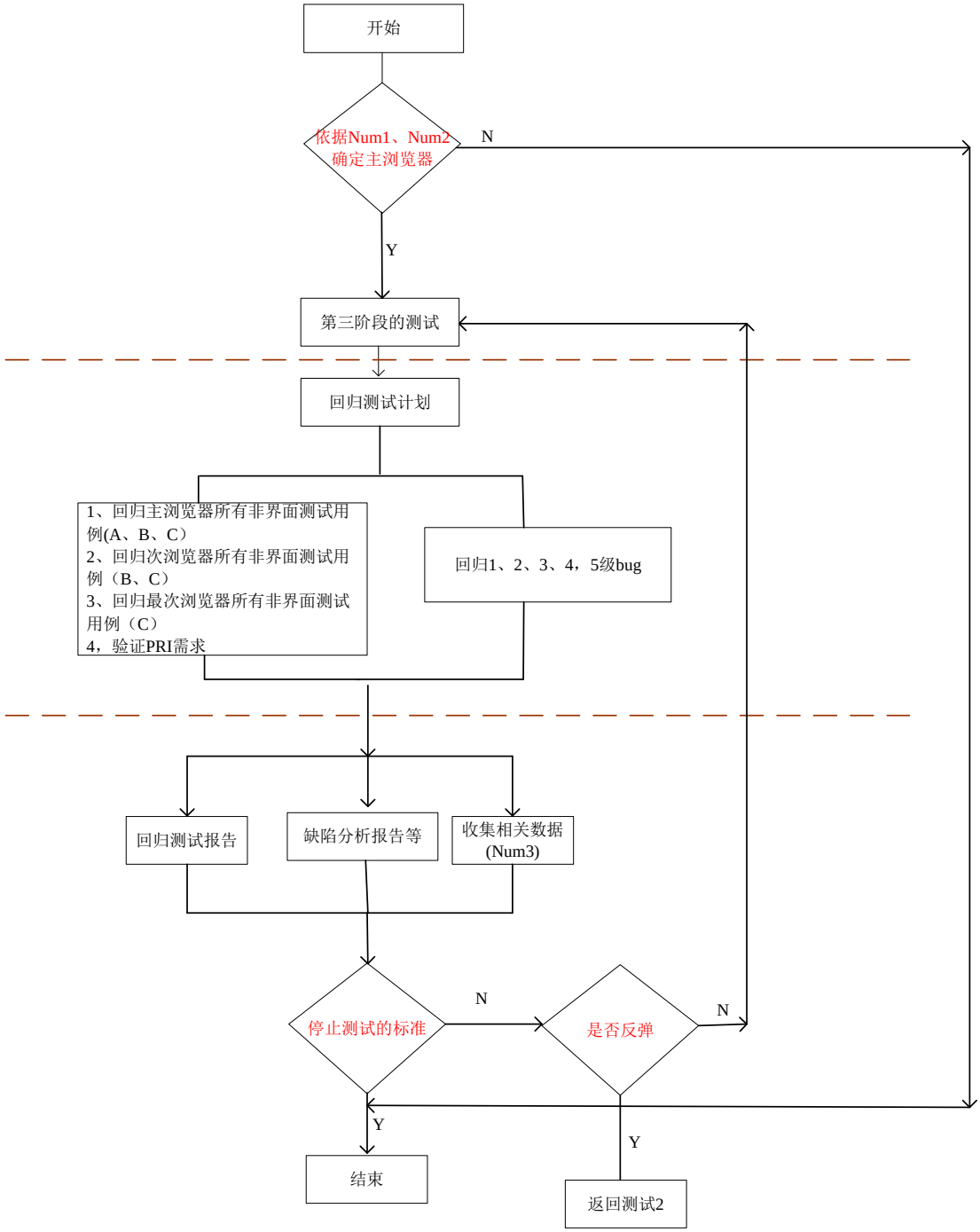
有回归测试计划。

确定每一次执行用例的主浏览器。

测试用例的组织架构上分为多种用例（如：界面+功能+业务+易用+控件等）。

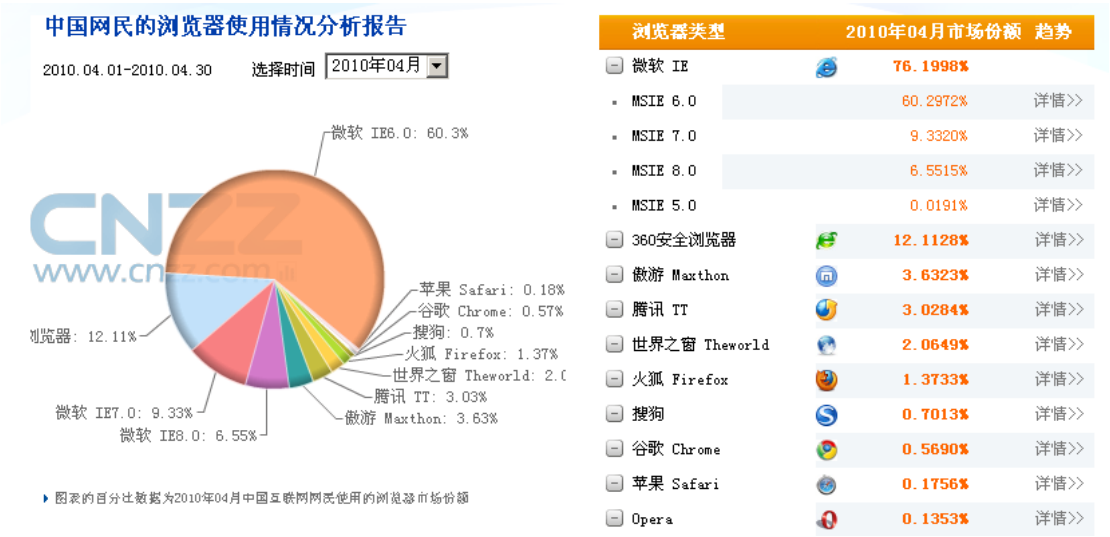
建立了具有共识系统测试停止标准。

4.3、第三阶段工作流程图

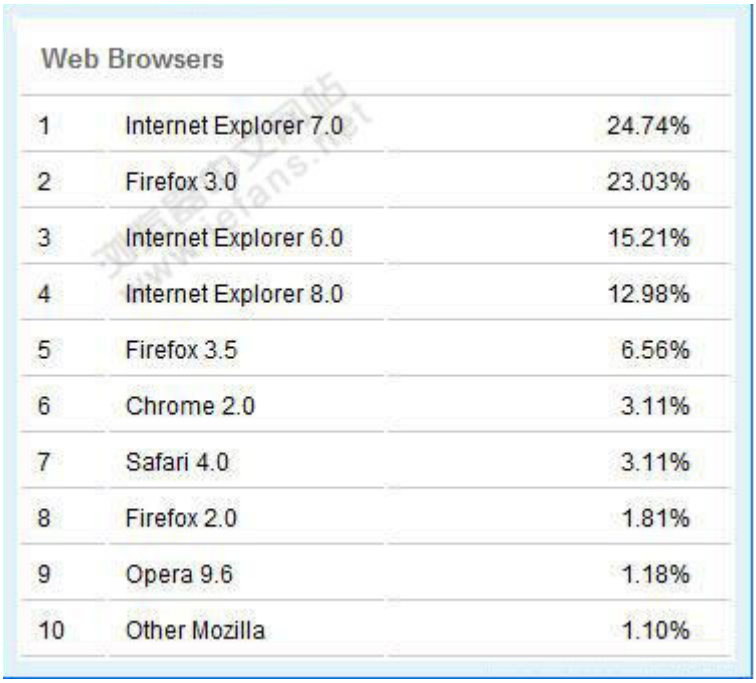


5、参考一：各个浏览器在市场上的比值

5.1、中国网民浏览器使用情况分析，详情请参考：http://data.cnzz.com/main.php



5.2、世界范围内的浏览器使用情况分布，详情请参考：http://www.iefans.net/liulanqipaihangbang/



Web Browsers

1	Internet Explorer 7.0	24.74%
2	Firefox 3.0	23.03%
3	Internet Explorer 6.0	15.21%
4	Internet Explorer 8.0	12.98%
5	Firefox 3.5	6.56%
6	Chrome 2.0	3.11%
7	Safari 4.0	3.11%
8	Firefox 2.0	1.81%
9	Opera 9.6	1.18%
10	Other Mozilla	1.10%

6、参考三：某公司内部浏览器的使用分布

项目名称	浏览器	公司内使用范围的比重
Patheon	IE6.0	略 40%
	IE7.0	略 30%
	firefox 2.0	略 5%
	firefox 3.0	略 15%
	firefox 3.5	略 5%
	其它	小于 5%

7、参考二：某公司通过冒烟测试的标准

A、冒烟测试模块用例 NT 率<10%，NT>10%的模块不进行系统测试，如

60%的模块 NT 率都超过 10%，则冒烟测试不通过（分析原因）

B、1、2 级 bug 修复率>95%，其它 bug 修复率>70%

C、冒烟测试 1、2 级 Bug 数<3 个

D、冒烟测试总 Bug 数<20 个

8、参考三：某公司第一个阶段测试通过标准

A、所有需求规定浏览器中都执行了相关用例

B、系统测试用例设计已经通过评审

C、其它行政要求

9、参考四：某公司第二个阶段测试通过标准

A、无 1、2 级 Bug

B、未解决 Bug 小于 10%

C、多个连续版本，系统或模块稳定性、成熟度分析后得到的数据都为高

D、无新增模块

E、缺陷分析中，缺陷数据在一段时间趋近于 0，并保持平衡

F、系统测试用例设计已经通过评审

G、其它行政要求

10、参考五：某公司停止测试标准

A、 系统测试用例设计已经通过评审

B、 按照系统测试计划完成了系统测试

C、 达到了测试计划中关于系统测试所规定的覆盖率的要求

D、 系统满足需求规格说明书的要求

E、 在系统测试中发现的错误已经得到修改，各级缺陷修复率达到标准

1、 一、二级错误修复率应达到 100%

2、 三、四级错误修复率应达到 80%以上

3、 五级错误修复率应达到 60%以上

F、其它行政要求

作者简介：姓名——文青山，毕业于一个不入流院校，金融危机时于千军万马之间杀入 BYD 至今，恍恍忽忽穿行于手机、Web、C/S、通信等诸多软件的测试工作之中，历经二年。

QTP 在性能测试中的应用

作者：张玉

摘要：

对于 CS 模式的软件，当界面要从数据库中读取数据时，计算数据量与读取数据的时间，数据量的多少与 UI 界面显示时间的关系，确定被测试软件的性能情况。通过自动化测试工具 QTP 在 LR 中来实现。

关键词：QTP，性能测试

引言

基于界面上的性能测试方法以及介绍比较少，本文提出一种基本界面上的性能的测试方法。主要是通过 QTP 对象的属性发生变化时，来判断 UI 上是否读取了数据。然后通过记录时间来确定性能问题。通过此方法对于 CS 模式的应用程序，如果 UI 上有数据库控件（Data Grid 等）都可以使用此方法来判断这些控件从数据库中读取记录到完全显示的时间情况。

1、QTP 对象机制及原理

QTP 主要采用的是使用 GUI 模拟人的操作。它在模拟人的操作时会记录操作的对象及所做的操作和顺序，然后在回放时按录制记录顺序操作这些对象。而在这个模拟的过程中，最重要的莫过于界面对象（控件）的识别，主要是通过对控件的一些属性和方法以及位置来确定记录对象，（例如：Button 推荐识别属性：Text；CheckBox 推荐识别属性：Name）然后通过 QTP 自带的对象库管理起来，加强脚本的灵活性。

2、QTP 基于 UI 的性能测试原理

由 QTP 对象机制可以知道，在自动化测试时，主要是对这些对象进行操作，从而实现模拟人来操作，实习自动化测试。而对于性能测试来说，就是判读数据对象的某一个属性值是否发生变化，以及发生变化所占用的时间。然后记录这些时间从而得出性能数据。 例如：现在界面上有一个 Button 按钮，它的属性 Text 的值是从数据库中读取出来的。那么我们只要记录到这个 Button 对象的 Text 属性值变为数据库中的值（期望结果）这之间的时间就可以了。现在对于 QTP 提供了方法 WaitProperty。

WaitProperty 的方法主要是当指定对象的值达到期望值时再做下一步操作。

object.WaitProperty (PropertyName, PropertyValue, [lTimeOut])，同样把此操作放入到性能测试中的事务中，就可以记录这之间的时间了。

3、方法的优缺点

3.1、优点

1.完全基于界面上的黑盒测试，对于部分实现方式不关注，只要测试程序的界面不发生变化，脚本不用维护。

2.实现起来比较简单，可能通过 Excel 制定出图表，方便性能分析。

3.2、不足

1.QTP 程序本身执行代码的时间也算在其中。

2.不能实习并发的操作，QTP 只基于单个客户端对性能的影响情况。

4、详细测试实例 1

4.1、性能测试需求

图 1

现在要得到当点击分类食品按钮后如图 1，随着详细食品数据的增加，完全显示所需要的时间情况，如**错误!未找到引用源。**：

食品数量 / 显示时间	1	10	20	30	40	50	60	70	…….
MS（毫秒）									……

表格 1

4.2、测试过程

4.2.1、测试方案

通过 QTP 基于 UI 的性能测试原理可知道，我们只需要判断当点击分类食品按钮后，来等待显示食品的 Button 按钮的 Text 属性是否等于我们预设置的值（期望结果）。使用 WaitProperty 方法。详细参见代码说明。

4.2.2、测试准备

数据准备

在数据库中新存储过程 IntoFoodMenu

```
BEGIN
    DECLARE StartNO int;
    DECLARE endNO int;
    set StartNO=4000000;
   set endNO=StartNO+ContRecord;
while StartNO<endno do
INSERT INTO `t_foodmenu` VALUES (StartNO, '4000271', 'M-01', null, null, 'No', 'Coffee', '咖啡', '3.0000', '3.5000', '4.0000', '6.0000', '8.0000', 'No', StartNO, null, 'Yes', null, 'liu hao', '2010-02-03 09:47:05', 'liu hao', '2010-02-03 09:47:05', image, null, '1');
set StartNO=StartNO+1;
end while;
END
```

4.2.3、测试脚本（部分）

```
‘进入到点餐功能
DeleData()    ‘删除出数据用于清空 UI 上的显示数据。
'RunAction "Action_IntoOrder", oneIteration
'选择操作.
For i=1 to 55    ‘用来确定食品数据的多少。
    AddData(i)    ‘添加食品数量的方法
    Services.StartTransaction "T_SelectFoodCategory"    ‘添加事务
    swfWindow("F1 Welcome Use WPOS").SwfWindow("F3 User By:liu hao").SwfButton("DRINKS 2").Click    ‘点击分类食品按钮
    '判断 Button 是否加完成
    Dim desc,descset
    set desc=Description.Create()
```

```
        For j=0 to s
    descset(j).WaitProperty "Text","Coffee",300‘等待属性性值的变化，超时设置为 300MS
Next
Services.EndTransaction "T_SelectFoodCategory"     ‘事务结束
DeleData()   '删除数据
Next
```

4.2.4、执行脚本统计数据

通过 QTP 自代的工具 Test Batch Runner，执行测试脚本。测试结果中把事务时间过滤出来，得到事务的时间。

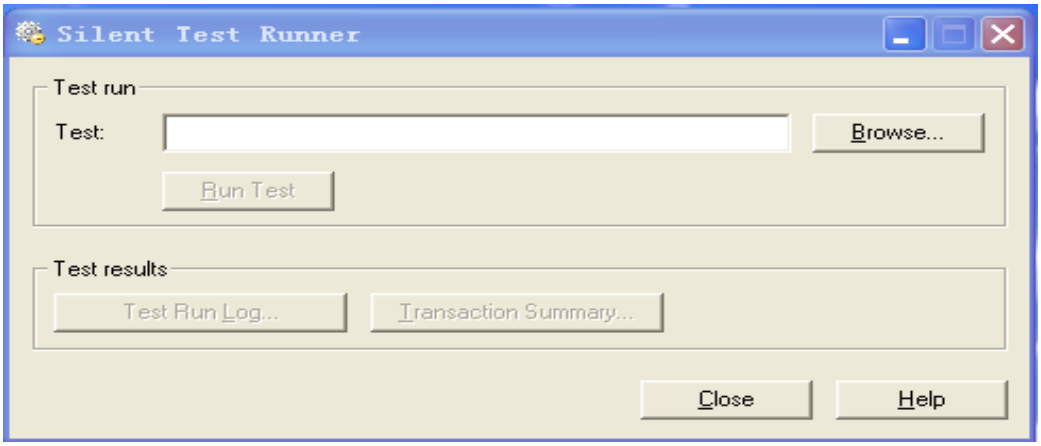


图 2

选择脚本文件，点击 Run Test，运行完成后，查看 Transaction Summary 中的事务时间，放入到 Excel 中做出性能趋势图，如图 3

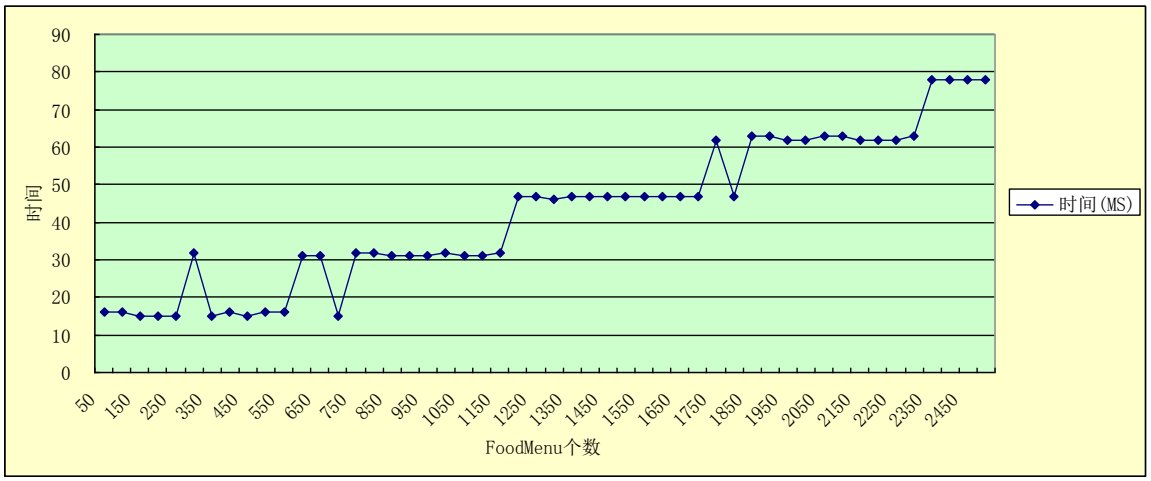


图 3

5、结论

通过此方法可以做所有 CS 模式下应用程序中基于 UI 界面数据库控件的性能测试方法。

参考文献

[1] QuickTest 领先技术研究专栏

[2] http://www.51testing.com/html/24/n-210024.html

一种基于行业特性组织数据的测试方法

作者：朱志敏

摘要：

用友产品是一个以通用产品+行业产品+个性化产品为主要特色的多层次立体式交叉体系，其客户群涵盖了多个领域。目前 U8 产品线可以满足大部分行业如机械、电子等离散制造业中型客户的通用需求，但是对于流程性行业及劳动密集型行业的匹配性还有待完善，行业产品正是基于此背景应运而生的。针对不同行业特性组织数据进行测试，可以更贴合企业应用场景，使得产品发挥更大效用。建立分层分行业的测试体系对于完善产品、提升测试队伍能力，充分发挥测试资源最大效能可以起到事半功倍的效果。本文就尝试从部分典型的行业特性出发探讨此类产品测试的方法。

关键词：行业特性 数据 场景测试

一、行业属性分析：

企业按照生产组织形式可以分为小品种大批量、小品种少批量、多品种大批量及多品种小批量等几种情况。

以服装行业为例，此类属于典型的劳动密集型产业，企业管理水平不高，主要分为自主品牌和 OEM 型；生产过程以订货会或者样品为起点，进行打版、排产、委外加工、入仓、备货、装箱、成本核算的全生命周期管理，往往伴随着按款色码组织业务，按款核算财务；按单品、单色混码、单码混色、混色混码模式排单装箱，按照色码配比制单，按照款式生产，车间生产存在工票扫菲计件，按照多尺码组备货发货等独特方式。此类产品的测试重点就是关注不同环节的数据输入输出是否匹配企业实际应用。

再以基础化工行业为例，此类属于典型的流程型产业，小品种大批量，对质量较为关注，常见于大宗原料采购业务，要求按质计价，工厂运输一般与 IC 卡射频机关联进行电子磅秤管理，根据实际计量结果与供应商或客户进行结算，按照流程进行质检，对质检结果中的数量、金额进行加权平均计算单价，按照明细汇总方式平衡物料需求，对领用物料按部门进行费用控制。 此类产品的测试重点是关注软硬件的接口正确性及多质检指标对计价规则、结果体现是否准确。

二、测试数据准备：

1）服装行业数据准备：母件存货一般启用颜色、尺码结构自由项，启用配码为非结构自由项，设置为销售、自制属性，MPS/LP 件，允许 BOM 母件；子件存货一般启用颜色结构自由项，设置为采购、生产耗用属性，MRP 件，允许

BOM 子件；半成品存货一般启用颜色、尺码结构自由项，设置自制、委外属性，允许 BOM 母件及子件。按照细分行业搭建 BOM 层级时，服装一般是单级 BOM，而鞋类需要搭建二至三级 BOM。启用不同单据上的颜色、尺码、配码等自由项字段及表体自定义项以便于单据数据的录入。参见图 1-1 表格的数据组织描述

存货属性	是否销售	是否自制/生产耗用	是否委外/采购	存货自由项、自定义项	是否MPS件	是否母件	需求跟踪方式
A	是	自制		颜色结构+尺码结构+配码非结构自由项	是	是	LP+订单号
A1		生产耗用	采购	颜色结构自由项			LP+订单行号
A2		生产耗用	委外	颜色+尺码结构自由项		是	需求分类号
A3		生产耗用	采购	颜色结构自由项			PE
A4		生产耗用	采购	颜色非结构自由项			LP+订单号

图 1-1 存货基本属性设置规则

2）化工行业数据准备：存货一般不涉及 BOM 结构，启用存货自定义项用来标识该存货是否参与流程质检，启用非结构自由项记录质检等级信息，一般为采购、自制\PE 属性。设置不同存货的检验指标、计价规则，磅房信息，质检规则。启用 U8 标准产品的质检管理模块，设置检验规则。

三、测试过程执行：

以 HY-01 为数据服务器建立两套账 001、002，分别具有服装、化工属性，HY-02 为应用服务器，各角色通过客户端访问应用服务器 HY-02 进行业务数据录入。

分配测试员 FQL 为制单及跟单员角色、测试员 ZSR 为采购主管角色、测试员 XHF 为库管员角色、测试员 LYB 为仓库主管兼质检主任角色、测试员 HM 为销售主管角色、测试员 SYJ 为车间计划员角色、测试员 CHJ 为车间主任角色、测试员 WD-1 为车间统计员角色、测试员 ZZM 为总经理角色，并赋予相应权限，以各角色命名客户端的机器名称便于查看，按照角色分配相应的功能权限及数据权限进行业务操作。

1）001 账套为服装行业账，数据维护如下：

FQL：以样品为起点，维护打样、制版、报价过程，并依据报价结果生成销售、出口订单；订单上记录存货自由项、自定义项及交期信息；

HM：审核、维护销售报价及发货交期信息；

XHF：根据订单检查仓库现存，符合出货要求安排发货、备货、出库,填制发货/备货单，若缺货则不发货；对采购到货、入库产品进行维护，录入批量出/

入库单

LYB：审核发货/备货/出库单，并修改相应存货的信息，更新存货单价及现存量信息；审核批量出/入库单据信息；

SYJ：以销售订单、出口合同、预测订单为源头维护计划，维护生产规划排产及多工序排程，将排出的三级生产计划报车间主任审批；维护存货的多级BOM，批量发布到物料清单中，批量维护订单 BOM、批量变更存货的 BOM，关联修改相关存货的 BOM 信息；

CHJ：根据插单情况修订生产规划已排产信息，审核月/周/日三级计划结果，生成采购、委外、车间订单信息；按照生产齐料展望分析结果进行采购、请购维护；

ZSR：维护、审核采购信息，针对多供应商进行比价询价最终定价；大额采购提交总经理进行审批；跟踪各批色采购物料的小样、大货期信息，方便把控；分析委外加工的成本差异，及时修正误差较大的订单信息；

WD-1：维护车间打裁床、车缝、工票扫菲统计，将统计结果进行计算，计件结果传递至薪资、成本模块；维护具体工作中心或车间的完成数量进行统计，反馈给车间主任完工、在产数据信息；

ZZM：对销售订单、出口订单进行跟踪分析，查看各环节—采购、生产、销售备货的情况，对重大异常业务重点关注。

2）002 账套为化工行业账，数据维护如下：

FQL：维护产品、材料的销售合同信息，对大宗物料采购及销售进行按质计价规则、存货维护；根据大修、基建、采购料定期进行各部门用料需求汇总，填制部门用料计划单；

HM：审核销售合同，参照销售订单、发货单过磅计量结果生成相应的销售发货、出库单信息；

XHF：对入厂、出厂的发运车辆进行过磅计量、符合要求的开具出门条，回收 IC 卡，将过磅计量结果写入地磅系统，回写到采购到货、销售发货、销售出库单的实际数量结果上；

LYB：对待发货、到货存货进行质检，按照抽样检验结果给出质量评级，传递至按质计价模块；对车间在制状况进行中控检验，合格品转入库存；严格控制各领料部门的费用，对超出费用预算的领料部门进行干预；分析中控、质检过程的台账信息；

ZSR：审批修订汇总平衡计划及追补采购计划，查看各材料的追补比率，及时掌握采购进度；

WD-1：按照部门用料计划进行领料申请，可以超领料申请进行领用；

ZZM：查看部门用料、产品质检、采购进展状况，对各流程环节进行管控；

四、测试结果呈现：

通过这种模拟行业特性及实际应用场景的方式进行测试的方法，极大提高了行业产品的业务匹配性及稳定性，使得测试问题数量、问题类型暴露更加充分，具体结果可以参考缺陷系统问题类型统计信息，如图 1-2 所示：

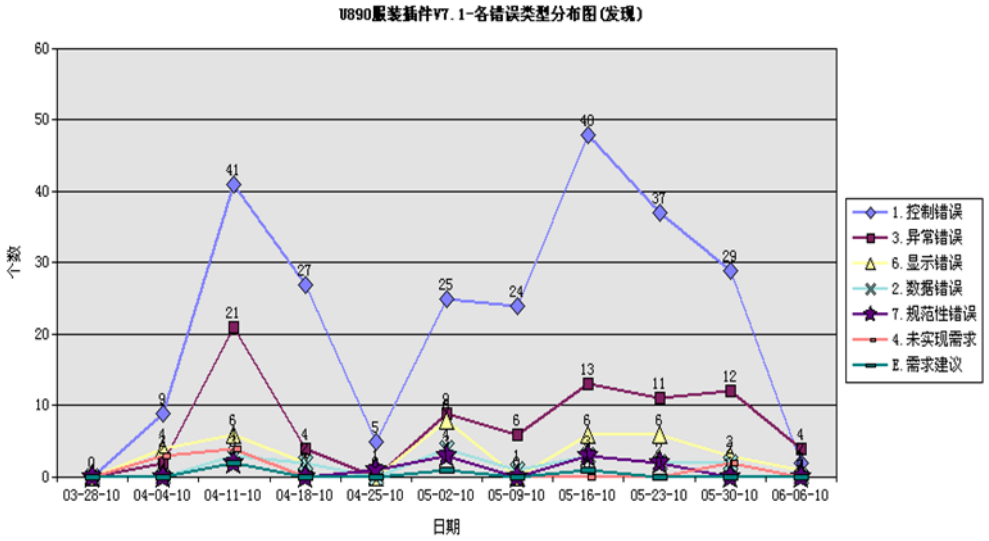


图 1-2 产品错误类型分布日程图

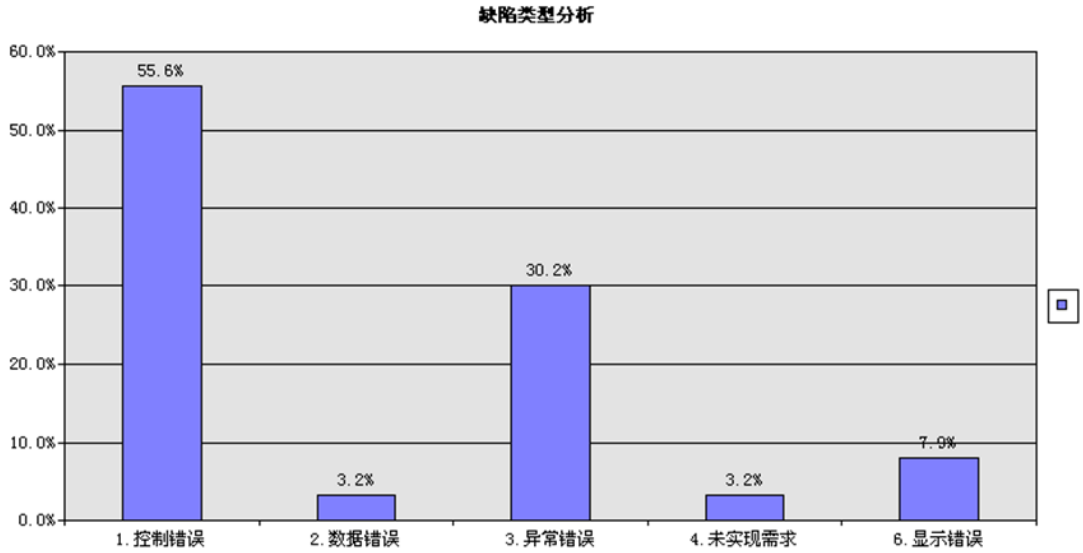


图 2-2 产品缺陷类型分布图

由上述图表数据我们不难看出，基于行业特性数据组织的测试往往更加关注业务匹配性，对于操作层的测试结果预期体现的更加充分。

综述

在应用行业特性数据时，对于测试人员的要求较高，除了充分了解产品功能外，还需要属悉各行业的特殊属性及业务模式，如服装行业跟单、插单，全生命

周期管理业务较为普遍，化工行业大宗物料采购对质量价格较为敏感，交通运输行业对于不同运输方式的运费计价比较关注，电子行业对于有限产能排产计算比较关注，汽车配件行业由于上游厂商的强势产生了供应商协同的业务需求等等。在分析了多行业特性后，需要抽象出相应的业务数据模型，有针对性组织业务数据的同时还要建立通用数据模型，以匹配各行业通用业务的要求。

基于不同行业特性进行数据组织，是 ERP 软件测试工作未来的一个趋势；建立分层分行业的测试体系对于完善产品、提升测试队伍能力，充分发挥测试资源最大效能可以起到事半功倍的效果。