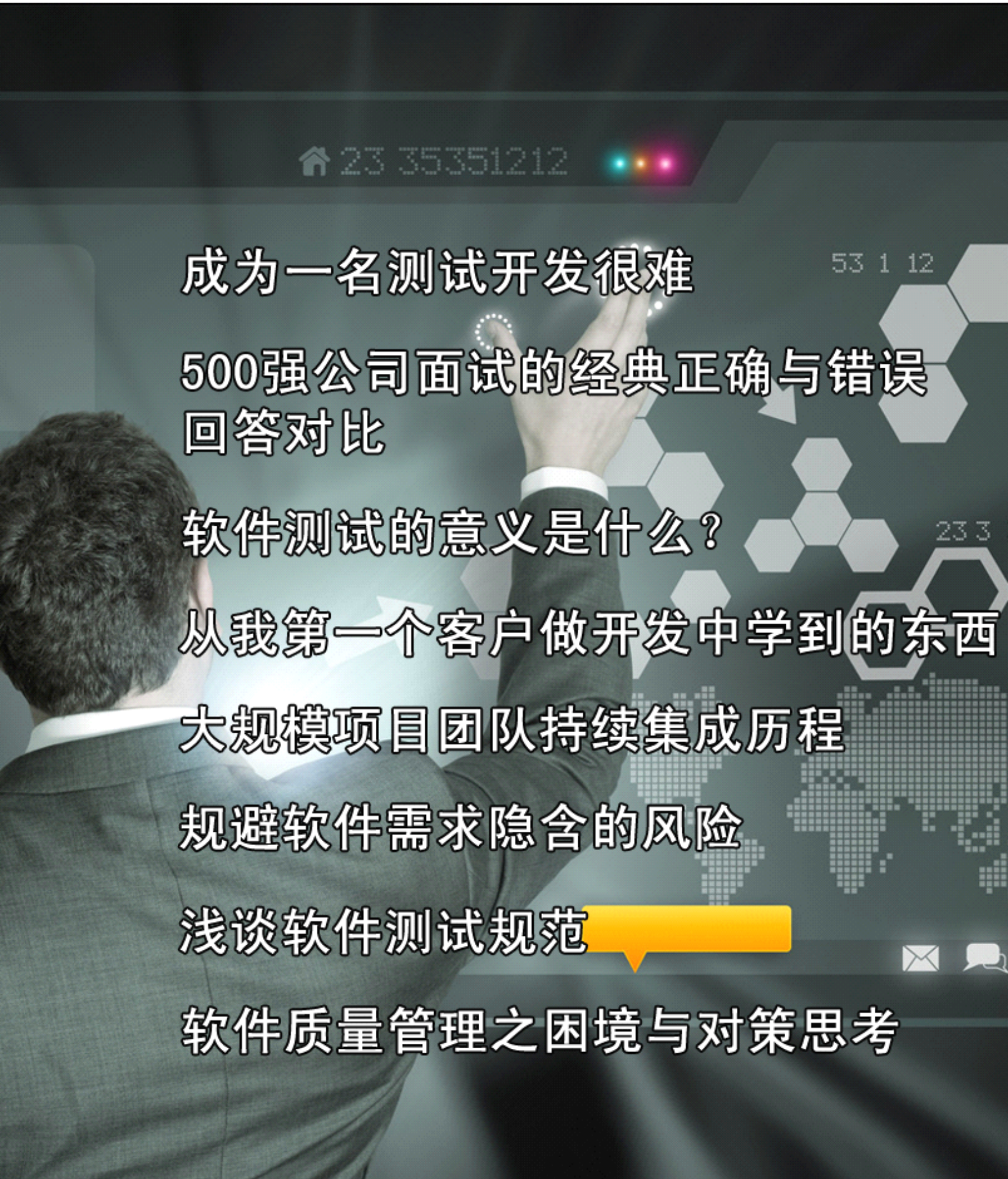
泽众软件测试网
电子期刊

SPASVO

第十二期
2012年12月



成为一名测试开发很难

500强公司面试的经典正确与错误回答对比

软件测试的意义是什么？

从我第一个客户做开发中学到的东西

大规模项目团队持续集成历程

规避软件需求隐含的风险

浅谈软件测试规范

软件质量管理之困境与对策思考

上海泽众软件电子期刊

2012 年 12 月 第十二期

主办单位：上海泽众软件科技有限公司

联系电话：021-61079698 传真：021-61079698 转 8017

意见反馈：fangmh@spasvo.com 投稿：wangmf@spasvo.com

公司地址：上海市普陀区曹杨路 450 号绿地和创大厦 18 楼 1801 室

邮政编码：200063

公司主页：www.spasvo.com 论坛：bbs.spasvo.com

目录

成为一名测试开发很难..... 4

500 强公司面试的经典正确与错误回答对比..... 5

软件测试的意义是什么？..... 10

从我第一个客户做开发中学到的东西..... 11

大规模项目团队持续集成历程..... 13

规避软件需求隐含的风险..... 18

浅谈软件测试规范..... 21

软件质量管理之困境与对策思考..... 22

泽众软件——专业软件测试门户
2012 年 12 月 第十二期 电子杂志期刊

成为一名测试开发很难

注：这篇文章说得也是实情。从测试转测试开发没那么容易。要牺牲业余时间，要付出大量努力。尤其是，当别人正在玩游戏、看网页、逛大街、看电影的时候，还要能耐住寂寞，真的很难。这就是代价。

所以我很喜欢一个同事曾经说得一句话，想要有收获，还不想付出代价，我最鄙视这种人。

其实，在其它行业，又何尝不是如此呢？

很多测试员从软件测试工程师开始。即他们执行测试，但不写任何（或者很多）代码。很多测试员梦想成为一名测试开发或者开发人员。同样，很多测试管理者则梦想把他们手下的测试员培养成测试开发。这能实现么？是有可能实现，但大多数情况下很可能实现不了。

很难自学成为一名开发人员。当我们考察计算机业内情况时，我们能够找到很多自学的程序员，看上去好像很容易。但是，一些人成功了，更多的人却失败了。为什么呢？我觉得有两个原因。第一，一些人不适合做程序员。第二，也许是更重要的，自学太难了。想成为一名好的程序员需要很多知识。这意味着大量阅读（网上或书上），大量练习。结果就是，做做梦容易，真的实施起来很难。

关于第一点原因，我在一篇文章中已经说了。研究表明，很多人，甚至是对程序感兴趣的，最终无法实现他们的梦想。我怀疑这和程序的抽象本质有关。这并不是说这些人根本无法编程，而是说他们不能写出好程序。而且，当难度加大时，越来越多的人也就放弃了。

第二个原因也困住了不少人。我见过很多人试图超越，但只有少数人成功了。成功的人花了很多自己的时间在上面。没成功的人则不愿在工作之外花大力气。期望从测试员成长为测试开发的人，如果仅仅是在工作时间努力则恐怕是要失望了。要成为一名称职的程序员要付出很多很多努力。在我早些的一篇文章中我给过一些建议。我建议不仅仅要学编程语言的语法，还要学习计算机科学的基础知识。当然，不学这些你也能编程，但做不到更好。而要学习这些，则要花费很多时间和汗水。

大部分时候，雇主们不会给你时间去折腾。他们想要员工们生产力高效，而正在学习编程的员工不会很高效。他们简单的事情都要做很长时间。当有事情要做的时候，团队里总是会有更有效率的程序员可用，因而事情就给他们去做了。并不是这些管理者不鼓励学习编程。他们也希望员工水平更高。但他们未必能预留足够的时间去让你实际去学习。

我已经阐明这其中的困难，那么测试员可以做些什么来提高他们的成功机会呢？测试经理那方面呢？我将在以后讨论这些问题。

* 请注意我这里说的。人们可以自学 C#来写个 ASP.NET 界面，或者自学 perl 来分析日志，这并不难。然而，这种程度离写出测试框架，分析性能，或者自动化测试 COM 对象，还差得很远很远。

500 强公司面试的经典正确与错误回答对比

问题 1 你为什么觉得自己能够在这个职位上取得成就？

分析 这是一个相当宽泛的问题，它给求职者提供了一个机会，可以让求职者表明自己的热情和挑战欲。对这个问题的回答将为面试人在判断求职者是否对这个职位有足够的动力和自信心方面提供关键信息。

错误回答 我不知道。我擅长做很多事情。如果我能得到并且决定接受这份工作，我确信自己可以把它做得相当好，因为我过去一直都很成功。

评 论 尽管表面上听起来这种回答可以接受，但是它在几个方面都有欠缺。首先，这种语言很无力。像“擅长做很多事情”以及“相当好”之类的话，都无法反映你的进取 心，而如果不能表现出足够的进取心，你就很难进入最好的企业。另外，将过去做过的所有事情同这个职位联系起来，这意味着求职者对这一特定职位没有足够的成 就欲望和真正的热情。

正确回答 从我的经历来看，这是我的职业生涯中最适合我的一份工作。几年来，我一直在研究这个领域并且关注贵公司，一直希望能有这样的面试机会。我拥有必备的技能（简单讲述一个故事来加以说明），我非常适合这一职位，也确实能做好这份工作。

评论这是一个很有说服力的回答，因为它可以告诉面试人，这个求职者拥有足够的技能和知识来完成这项工作。他所讲的故事表明了求职者的技能，也验证了他最初的陈述。最后，求职者表示了“做好这份工作”的愿望，这证明了他具备对这份工作的热情和进取心。

问题 2 你最大的长处和弱点分别是什么？这些长处和弱点对你在企业的业绩会有什么样的影响？

分 析 这个问题的最大陷阱在于，第一个问题实际上是两个问题，而且还要加上一个后续问题。这两个问题的陷阱并不在于你是否能认真地看待自己的长处，也不在于你是 否能正确认识自己的弱点。记住，你的回答不仅是向面试人说明你的优势和劣势，也能在总体上表现你的价值观和对自身价值的看法。

错误回答 从长处来说，我实在找不出什么突出的方面，我认为我的技能是非常广泛的。至于弱点，我想，如果某个项目时间拖得太久，我可能会感到厌倦。

评论这种回答的最大问题在于，求职者实际上是拒绝回答问题的第一部分。对第二部分的回答暗示了求职者可能缺乏热情。另外，基于对这一问题前两个部分的回答，求职者对后面的问题很难再做出令人满意的回答。

正确回答 从长处来说，我相信我最大的优点是我有一个高度理性的头脑，能够从混乱中整理出头绪来。我最大的弱点是，对那些没有秩序感的人，可能缺乏足够的耐心。我相信我的组织才能可以帮助企业更快地实现目标，而且有时候，我处理复杂问题的能力也能影响我的同事。

评论 这个回答做到了“一箭三雕”。首先，它确实表明了求职者的最大长处。其次，它所表达的弱点实际上很容易被理解为长处。最后，它指出了这个求职者的长处和弱点对企业和其他员工的好处。

问题 3 是否有教授或者咨询师曾经让你处于尴尬境地，还让你感到不自信？在这种情况下，你是怎样回应的？

分析 这个问题考查的是求职者在陌生领域工作的能力。通过这个问题，面试人可以了解到，当所给的任务超过自己目前的能力水平时，求职者解决问题的意愿和能力。

错误回答 我相信质疑权威是很重要的，但我不可能在学校里学到一切知识。很多人以为自己知道所有问题的答案，可实际上他们并不了解真实世界里发生的一切。你知道，那些都是象牙塔里的东西。

评论 这种回答的最大问题在于，求职者把问题的焦点从自己身上转移了。严肃的面试人并不关心你对高等教育的观点。他们想知道的是，当出现问题中给出的情况时，你将怎样处理。这种回答的另一个弊端是，它会使面试人对你是否愿意服从领导产生怀疑。

正 确回答 在我当学生的这几年中，我尽自己所能多学习知识，经常选择一些不熟悉的课程，因此往往会受到教授的质疑。不管什么时候，当我觉得自己对这个科目知之甚少 时，我就尝试预见一些问题，为回答问题做些准备。当我被难住时，我尽可能做出科学合理的猜测，承认我不知道的东西，并且从不懂的地方开始学习。（如果可 能，你可以举出一个例子……）

评论 这种回答的最大好处在于，它清楚地表明了求职者会积极面对艰难处境。它也显示了求职者有雄心和明确的态度，知道怎样处理离奇和模糊的问题。

问题 4 你是否曾经得到过低于自己预期的成绩？如果得到过，你是怎样处理这件事情的？

分析 通过对这个问题的回答除了可以揭示求职者的热情和进取心外，还可以揭示求职者是否愿意为某一事业奋斗，是否愿意为追求公平而奋斗。

错误回答 记得有一次，我觉得应该得 B 但却得了 C，我去找辅导员，他给我看了我在每个项目上的得分情况——我处在 C 级的边缘但很明显是 C.我很高兴能核实一下而不是接受既定的分数值。

评论 这个问题开始时回答得很好，但最后却不尽如人意。从最初的情况看，求职者似乎愿意追查到底。但是后来很显然，他（她）没有试图做出改变。

正 确回答 我曾经和一个研究地球科学的教授有过一段令人记忆犹新的经历。这个人一向以偏袒理科生而出名，而我偏偏又不是理科生。在我们班上，所有的非理科生都感到， 他对我们的知识基础有着非常不切实际的期望。由于他的偏见，这些非理科生大多都表现不好。尽管我表现还算不错，但我还是和其他学生一道向系领导发出了一份 声明，建议校方审查一下他的教学方式。

评论 这种回答能够表明，这名求职者有能力克服困难处境，而且能够脱颖而出并居于领先地位。这样的回答还可以表明，这名求职者高度重视公平感。同时也表明了求职者十分关心集体利益。

问题 5 出于工作晋升的考虑，你打算继续深造吗？

分析 这是一个简单的问题，它可以用来衡量你的雄心，也可以判断企业对你的重视程度是否会影响你对自己未来的重视程度。

错误回答 我不知道。我已获得了管理学学士学位，我认为自己已经受到了很好的教育。我觉得实际工作经验比在学校里学到的东西更有价值。

评 论 尽管求职者试图通过这种回答反映其积极的一面，而且这样回答从某种程度上也可以间接地讨好面试人（面试人就是“实际工作”的一部分），但是，它根本没有反 映出求职者追求上进的意愿。因此，根据求职者所表达的信息，如果碰上一个乐观的面试者，他（她）会认为你缺乏雄心，如果碰上一个悲观的面试者，他（她）可 能会认为你很自负。

正确回答 作为一名大学生，
“世界五百强面试题目及应答评点”版权归作者所有；转载请注明出处!

我学到了很多知识。如果有合适的机会，我当然会考虑继续深造。但是，我会认真考虑这件事情，我觉得很多人回学校学习是很盲目的。如果我发现自己所做的工作确实有价值，而且也需要获得更多的教育才能在这一领域做得出色，我当然会毫不犹豫地去学习。

问题 6 你曾经参加过哪些竞争活动？这些活动值得吗？

分析 通过调查你经历过的实际竞争场景，可以反映你对竞争环境的适应程度，也可以反映你的自信心。当竞争成为关键因素时，正是讨论小组活动或企业业务的一个绝好机会。

错误回答 从本质上说，我是一个竞争性很强的人。我认为，在所有我做过的事情中，我实际上都采取了一种竞争性的态度。毕竟，只有这样你才能在竞争激烈的企业界生存，对吧？

评论 这样的求职者阅读了太多关于鲨鱼和汉斯之类的故事，他这样回答让人感觉在企业界不是你死就是我活。尽管企业界是高度竞争的，但是企业中的人憎恨别人把自己看成是凶猛的梭子鱼。

正确回答 我喜欢小组运动，我一直都尽我所能参加这些活动。我过去经常打篮球，现在有时候也打。同小组一起工作、为实现共同目标而努力、在竞争中争取胜利……这些事情确实非常令人兴奋。

评论 这种回答表明，求职者能够正确看待竞争。这意味着他（她）能够利用竞争力量在竞争中取胜，而不会毁掉同事的工作成果。

问题 7 你怎样影响其他人接受你的看法？

分析 你的回答将告诉面试人，首先，你对影响别人有什么看法。其次，你影响别人的能力究竟有多大。

错误回答 一般情况下，这取决于这种想法的价值。如果这是一个好想法而且我所交往的人是通情达理的，那么，一般情况下，让别人接受我的想法不会太难。

评论 这种回答的问题在于，它并没有解决实际问题。这个问题实质上问的是你怎样对待那些不赞同你的看法的人。这个回答表明，你愿意在一种和谐的工作环境中工作，不喜欢不和谐的工作氛围。

正 确回答 这是多年来我一直非常努力探索的一个领域。对于好的想法，甚至是伟大的想法，人们有时并不接受。我现在认识到这样一个事实，那就是你表达想法的方式同想法 本身一样重要。当我试图影响别人时，我一般会假设自己处在他们的位置上，让自己从他们的角度来看待问题。然后我就能够以一种更可能成功的方式向他们陈述我 的想法。

评论 首先，这个回答表明，你理解人际沟通的复杂性，知道使别人改变看法具有一定的难度。其

次，这个回答还表明，你知道影响别人时运用策略很重要，而且也能够采用合理的方式说服别人。最后，这个回答还表明，你知道在沟通困难的情况下，沟通方式和沟通内容一样重要。

问题 8 在做口头表达方面你有哪些经验？你怎样评价自己的口头表达能力？

分析 这个问题旨在测评你的公共演讲能力，同时也可以了解你对演讲能力的自我评价。

错误回答 我认为每个人都会在做演讲时感到紧张，我可以做口头表达，但是说实话，人们并不总是愿意倾听。我认为，有时候，给人们发放纸面信息再回答他们的问题，这样做会更好一些。

评论 这种回答清楚地表明，你这方面的能力很欠缺，它不仅说明你不喜欢口头表达，同时也意味着你不愿提高自己的口头表达技能。

正 确回答 我曾经看到一篇文章，说公共演讲是美国人最害怕的事情。我认识到，如果大多数人都害怕做公共演讲，那么在克服自己的恐惧并掌握口头陈述技能之后，我就能够 在竞争中更胜一筹。因此我抓住所有的机会做演讲，而且我发现，做的演讲越多，就越对演讲感到轻松自如——当然也做得更好。

评论 这是一个很好的回答。因为它具体说明了你在这方面的能力，而且它也表明你正在继续努力提高自己的演讲技能。通过承认口头陈述是很复杂的，求职者同时表明了自己的诚实和正直。

问题 9 你怎样比较自己的口头技能和写作技能？

分析 这是一个暗藏杀机的问题。无论什么时候，只要被问及对两种事情做比较的问题，你就一定要小心。这样的问题通常是想让你说出自己的弱点。

错误回答 （任何表明自己的某一技能比另一技能好的回答）

评论 你中圈套了。

正确回答 从现在的情形看，企业越来越重视职员的能力，希望他们在口头表达和书面表达方面都能够做到清晰、明确。我总是利用机会提高自己的口头沟通和书面表达技能。我认为，这两种技能都是极为重要的，任何想要在企业界取得成功的人，这两种技能都应该具备。

评论 这种回答避开了陷阱，避免被别人认为自己在某一方面薄弱。同时，也可以表明，你理解高效沟通技能的重要性。更重要的是，它可以使面试人确信，在一般技能方面你拥有坚实的基础，而这些技能是无论什么企业都需要的。

问题 10 在写专业论文时你最不喜欢哪些方面？

分析 这个问题可以判断你是否愿意开展研究工作，是否愿意发现信息并找到困难问题的解决办法。

错误回答 我最担心的就是进行一个自己不感兴趣的研究课题。如果我对研究课题感兴趣，我不介意开展研究工作。但很多时候，研究所得的成果并不能在实际中得到应用。

评论 尽管很多读者可能会同意这种回答，但它却不能让面试人感到满意。很多工作任务都是单调和繁琐的，听到求职者表示不喜欢枯燥的事情，这会让人感到很不舒服。

正确回答 如果我认真工作的话，我会发现某一题目有无穷多的信息。我认为最难的工作就是判定什么时候才能获得足够的信息可以开始动笔写论文。

评论 这种回答表明，求职者理解研究的意义并愿意从事研究工作。它还表明求职者能够深入调查，也表明求职者能够胜任书面论文的写作。 这种回答也显示了求职者的雄心、热情以及动力。同时也表明，求职者具有与众不同的头脑，而且对重大职业决策非常认真。

软件测试的意义是什么？

昨天的一位同事和我讨论一个被测试对象。讨论之后觉得颇有所得，所以拿出来分享。

被测试对象是一个 DSL 描述的系统，通过输入一个 DSL 的输入文本，定义一个复杂的规则，使用这些规则来控制被测试对象的一个系统。

遇到这样的被测试对象，我第一反应时，首先，DSL 的语法实现应该是一个比较重要的测试点。然后就是要映射出 DSL 中所表述的每一个要点，是否可以完全被系统所接受。然后用几个实用的场景数据跑一下，就可以说测试过程结束了。

但是接下来我关心的事情是，这个程序是谁开发的。开发同学的能力和性格决定了被测试对象的稳定性。这个程序的开发同学我很熟悉，是典型的完美主义者，做事情非常稳健，所以我想当信任他写的代码。根据开发人员的情况，我第一反应是，DSL 语法解释引擎是不是使用了现成的框架。最后得到的情况是，果然会使用 antlr 作 Parser Generator。那么第一步 DSL 语法测试，应该就已经不是问题了。因为 Antlr 一定是已经经过测试的开源产品，可以用来直接使用的。

然后考虑第二步，如何验证 DSL 语句中的每一个表述都被正确的被系统理解。这个测试我的想法是把表述的意义归类，然后用过等价类划分的方式测试其中的一些，然后通过灵活的自动化测试框架，比如说数据驱动的框架，帮助测试人员做完整的测试。其实到了具体测试，就会考虑到，这种测试更容易在单元测试完成。所以我问了单元测试覆盖率。出于意料，代码覆盖率接近 100%。那这一步，其实已经覆盖了。而且开发人员的专业和谨慎让我非常放心。测试人员其实只需要对开发的单元测试进行简单 review，就可以确认覆盖。

第三步场景测试则是没有办法省略掉的。测试人员需要整理，制造和使用场景接近的 DSL 语句，提供给被测对象，进行测试。

分析到这里，也许你会开始质疑测试人员存在的意义了，如果只是简单的场景测试，我们测试人员还有什么存在的意义呢？其实当开发人员非常专业和谨慎的情况下，测试人员并不是必须的。Facebook 不是号称没有测试人员吗？遇到这样的开发人员，测试人员更应该庆幸，因为没有自己，整个产品的质量也是可以得到保障的。

那最后一步场景测试可以省略吗？我认为不可以，因为场景测试是站在用户考虑问题的，这是最后一道把关，所以是必须做的。而且如果在这一步做一点自动化，也可以提高之后的回归效率。

最后总结一下：

1、测试范围的确定要因开发而异。

2、测试范围的确定也要根据开发的单元测试覆盖情况而定。

3、专业且谨慎的开发时测试人员的福气。

从我第一个客户做开发中学到的东西

几个月前我的一个朋友的朋友在 Facebook 上联系到我，他在本地有一个推广公司。他得知我开发了 Thoughtback 这个 iPhone 应用程序，想问问我是否有兴趣为他做一些 iPhone 上的应用。我决定去他的办公室看看他究竟想要些什么东西。

在第一次会谈中，他告诉我 bd's Mongolian Grill，他的一个客户，想要一个 iPhone 应用，但不知道如何做。这就是叫我来的原因。我们谈了很多关于开发一个应用程序的过程上的事，以及如何能够尽可能的把这些东西应用到 iPhone 上。我本期望的是一场关于工作内容上的面试，结果吃惊的发现不是这么回事。他说他们会整理一下希望在应用程序里出现的功能特征，然后会告诉我怎么做。

数天后我又去见他，讨论了一下他们想要的功能，看看他们想要的都是否可行。此时我并不清楚他们是否要对我进行评估，看我是否能够完成他们要求的任务。我猜测他们早已想好了让我做这个东西了，因为在会上以及之后的邮件里，他们问我需要向我支付多少钱来为他们开发这个应用。

他们应该支付我多少钱?

应该要多少真是一个很难的问题。一方面，我希望他们让我来做这个东西，担心如果要的太多他们会找另外的人做。另一方面，我是一个合格的，专业的软件工程师，我的时间价格不便宜。我真的希望他们给我报个价，这样我就不必自己来琢磨这个数字了。

我在 Google 上查了一下，找到的数字让我吃惊。有人说出了 1 万到 5 万美元之间的数字。我估算了一下用多少小时能完成这个项目，然后乘上我上班时每小时能得到的工资数。得出来的数让我不敢接受。1 万美元不是个小数目。你要明白，这不是我挣钱糊口的工作，我并不是真的需要这笔钱，这全是外快。我最终选择了一个能让我舒舒服服旅游好几次的金额数，但仍然担心一个这样的小公司是否接受的起这样的报价。我感觉有点贱卖了自己，因为很显然需要开发很长的时间，超出你的想象。

你比你想象的要值钱!

如果让我给处在同种境遇的人提一些建议，我会说，选一个看起来稍微有点高的价位。你要明白，你是不可能得到你说出的价位的。不要因为以前没有遇到过这种事情而担心。对于你来说这是个好机会。我之所以认为我的报价有点低的原因是，我并不确定我能做出他们想要的东西。如今我真正的把它实现了，想起来有点傻，因为我能够做出来他们想要的任何东西(甚至更多)。

要多长时间能完成它?

当然，价格估出来了，他们要一个开发时间的估计。这又是一个让人头痛的问题。我以前从没有做过这么大规模的 iPhone 应用。他们要的很多东西我以前都没有做过，比如 Facebook 共享，Twitter 共享，iPhone 相机的调用。于是我历数每个功能点，依次指出它们需要多少的开发时间。然后还增加了多余的时间，因为事情总是会要比你想象的时间要长。感谢上帝，我的客户也理解这点，并给他们定的最后期限上又加了些多余的时间。

你讨厌估计!

也许你不是，但我确实是这样，而且大部分人也是这样，我想。当你第一次做评估时，总有一大堆

的情况你没能想到。在我这事情上，我最初估计每天在下班后花 2 个小时做这个项目。我没想到的事情是我有一个一周长的去佛罗里达的假期，我想好好享受这段时光(和女朋友一起逛街，玩游戏，看电影等等)。所以，到了项目的尾声的时候，一天 2 个小时变成了 3 至 4 个小时，周末有时会更长的时间。我还没预料到的是那些客户希望让我做的修改。因为他们想要一些不同的东西，我不得不在中途把一些功能做了彻底的修改。这引出了我的另外一个问题。

为一个客户工作和为一个卖软件的公司工作是完全的不同!

我任职的公司里没有这样的客户。我们把软件买给消费者。我们的时间期限是自我管理的，我们必须自己制定软件规格说明书。当你给一个客户工作时，他们给你设最后期限，他们告诉你他们希望应用里有什么。他们要他们想要东西，因为他们出了钱的。

你以后还会做这种事吗?

嗯，这很难说。这是一次很好的经验。你能得到一笔额外的零花钱，而且我学到了很多的关于 iPhone 开发的知识。这个客户非常的棒。整个过程中我们一直对此事进行交流，他们给了我很大的自由，在有些模块上允许我不按照原型开发。

尽管事情做得很棒，但付出了劳动也很多。我其实真的不需要这笔钱，就像是我很喜欢做开发一样，能把一件事情搞定是个很高兴的事。

我建议开发人员们在业余时间至少做一次这样的事情。我真的学到了很多，不仅仅是指开发知识上，同时也包括跟他人交流和合作的经验。

大规模项目团队持续集成历程

这篇文章是我在两年前写的，记录了一个 150+人的软件团队(最多时近 200 人)如何在一个庞大的遗留系统上，通过逐步建立一个持续交付部署流水线，从而达到频繁发布的状态。最终在该团队的持续交付基础设施中，共有 260 台服务器用于构建、测试和部署(几乎全部是虚拟机)。而这个产品也可以每六周发布一次。

在大规模项目团队中可能遇到的问题

对于小规模、短周期的项目来说，团队与持续集成会相处地非常融洽。而对于大规模、长周期项目的初期来说，也不会有太多的问题。此时常见的也是基本的持续集成模式就是：Build->test->package。然而，只要时间稍长一点儿，持续集成就会发出坏味道了。此时的症状包括：

1. 作为开发人员

要等很长时间才能知道是否可以提交代码了。如果你遵守“频繁提交”的原则，那么百人团队不间断的提交，会使集成服务器一直处于繁忙状态，而你不得不等待他人的 build 过了以后，才能看到自己提交的结果。

要等很长时间才能知道我的提交是否通过了;

如果 build 失败了，要花很长时间才能知道是否和自己的修改相关;

既使提交了 fix，也不知道自己的提交是否真的修复了这次构建;

构建经常处于失败状态。

2. 作为测试人员

测试人员不知道到哪里拿哪一次的构建产物来进行测试;

发布经理不知道当前各种各样的测试部署环境中，到底部署了哪个版本，包括哪些新功能或修改的 bug;

不确定在同一个构建里，所有组件的版本是否都是正确的;

3. 作为项目经理

不确定各个测试部署环境中的配置是否都与其上运行的构建相一致;

不确定测试人员测试的是否在正确的运行环境上运行了正确的版本;

4. 其他方面的问题

所有的安装部署都需要手工操作。

以上这些问题会给你的发布管理带来无限的问题和风险。那么，是否因为这种“持续闹心”就放弃持续集成呢?回答当然是否定的。Do it more if it hurts you. 不要因问题的暴露而放弃，相反，应该欢呼。因为这反映了发布过程中的问题与风险，是时候解决它们了。

如何解决大规模项目中的持续交付问题

由于大项目本身的复杂性，其解决方案也不能一概而论。下面以某大型项目为例，介绍其中的几个解决方法。

1、项目基本信息描述

该项目最初就试图建立一个好的持续集成环境和基础。由于是一个遗留系统，费了很大劲儿，才能够得到可工作的软件。然而，由于队伍不断壮大，而且环境也在不断变化，持续集成很快就无法达不到其预期目标了。怎么办呢?

项目背景:

项目是一个具有可配置性的 Web 门户产品，面向不同行业的市场，可自己定制门户。该项目有一个遗留的代码库，而且可以肯定的说，在今后的一年半之内是无法摆脱这个遗留代码库的。而且，很多紧耦合的、不必要的臃肿代码，同时根本不存在有价值的测试代码。现在我们在逐步地重写代码，但还是不能删除它们，因为某些网站还要依赖于旧代码。事实上，这是一个.NET 平台上基于 SOA 的网站。

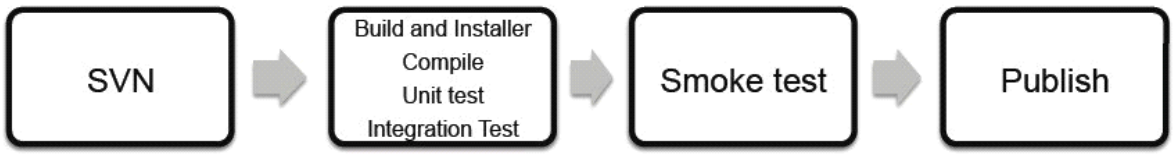
开发团队情况：

团队是一个敏捷分布开发团队(三地协作，均有开发人员，且有时差)。整个团队有 150 多人，分成十几个团队，每个团队都有一个完成的结构(BA/DEV/QA)，其中有一个是项目持续集成团队(项目之初，大约有五六个人，工作负荷很大，项目运行一段时间后只要两个人就足够了)。使用 SVN 做版本管理工具，在 Windows2003 上使用 NAT， MSbuild 和 batch 脚本进行构建管理，最初使用 CC.NET 做为持续集成服务器，后来使用 Cruise(Go)。

初始的持续集成环境：

上面所述的持续集成问题在项目一开始就出现了，因为该项目有一个庞大的遗留代码库，而且使用的基本持续集成方式(build->test->package)而且测试人员手工部署进行各类测试。

其初始的持续集成环境如下所示：



第一步目标：尽量减少团队之间影响

方法：先化整为零，再化零为整 ————根据团队划分代码(或者根据代码划分团队)。

手段：DVCS+私有持续集成服务器+全局持续集成服务器

每个团队都使用 GIT 做为中间源代码管理工具。这样，团队人员可以先提交到 GIT。每个团队有自己的持续集成环境。一但构建成功，触发将代码提交到中心的源代码库，并触发中心源代码的持续集成。有一个专职团队负责全局持续集成的结果跟踪。

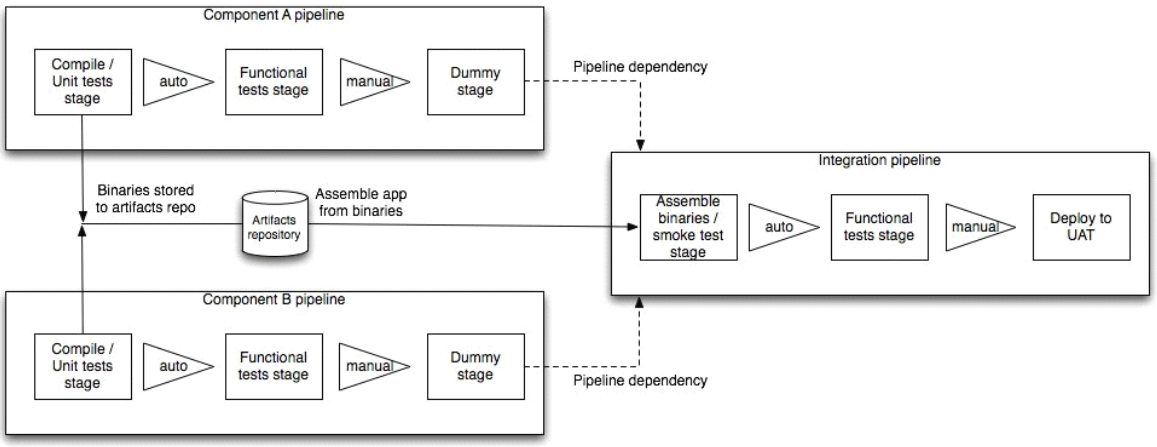
益处：

每个团队都可以天天提交代码 (如果这些代码没有让自己的构建失败，就说明至少能够通过初步检验)。

任何一个团队的构建坏了，并不影响整个项目，而只是一个团队。

有一个专职团队负责全局，不用每个团队都停下来。

如果全局持续集成失败了，不用所有的团队停下。



第二步目标：提高反馈速度

方法：化整为零，再化零为整————测试分组运行。

手段：并行化与中心仓库(Cruise 的并行化与中心仓库)

由于功能多且复杂，测试较多，运行很长。利用 Cruise 的并行化特点，每个团队将测试分成 28 组。一旦提交后，Cruise 会将其放在 28 台机器上并行运行。运行后，将所有测试输出和结果上传到同一处(Cruise Server 的中心仓库)。

益处：1. 反馈时间大幅度缩短(原来 30 分钟以上，现在 20 分钟之内)。

2. 易扩展：如果因测试增多而导致反馈周期长，可通过增加更多的机器解决。

3. 易维护：对于测试分组和构建机器的增减来说，在 Cruise 中都非常容易，只要在同一处修改配置即可。

4. 易追踪和 Debug：所有的信息(包括 artifacts)都放在同一处，能过 Web 访问。

5. Single view (在同一个 Web 管理页面上，可以监控所有团队的构建状态)。

第三步目标：减少手工操作

方法：一键发布————自动化部署

手段：使用持续发布管理工具 Cruise (Cruise 的 dependency + Story Tracker plugin + audit)

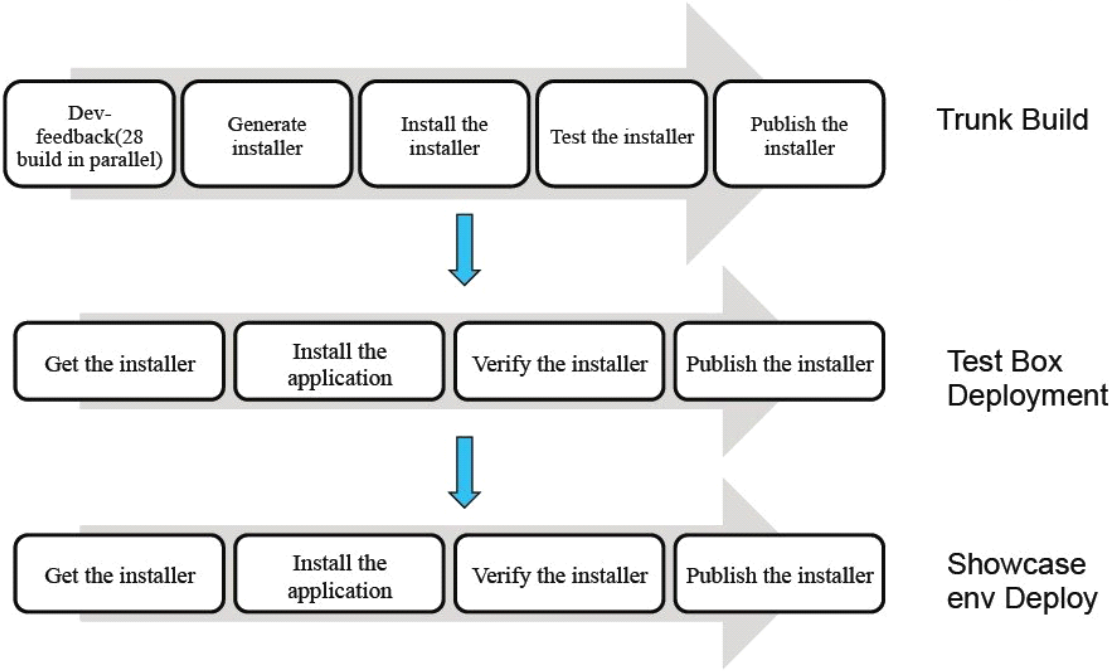
由于有很多个团队，每个团队都有多个测试环境。如果全部使用手工部署分花费很多时间。所以，每个团队建立三个构建管道，其目标分别为：(1)得到测试过的 installer;(2)部署到测试环境中;(3)将通过测试的 installer 部署到演示环境中。前一个构建成功后，就可以触发下一个(自动或手动)。

益处：1. QA 可以清晰识别需要测试哪个安装包，该安装包中含有哪些功能

2. QA 自己可能很容易地部署测试环境。

3. 易于追踪功能的历史版本(在同一个 pipeline 中，所有的 stage 同一版本.而且在使用 Pipeline dependency 时，版本信息会向下游传递)。

4. 易于掌握对各种部署环境的管理。



使用上述手段后，该项目的持续交付已入佳境。目前可以做到：

所有的构建和部署都是自动化的;

开发人员最多在 20 分钟内就会得到反馈。

对于每种环境来说，可以做到每天部署四次。

部署无差错：因为是自动化过程，每次的执行步骤都一样。

机器资源复用：Cruise 自动向 165 台机器上分发工作进行构建和部署。

所有信息都显示在一个屏幕上(Single dashboard view of everything!)

开发人员非常高兴：他们可以多次提交而不影响他人。

测试人员非常高兴：他们可以很快地得到好的 installer，通过自己点一下按钮就完成部署工作。

管理人员非常高兴：他们可以马上了解当前的项目状态(哪些版本在测试中，那些版本已经演示了)。

很容易了解每个版本里包括哪些功能和修复了哪些缺陷。

规避软件需求隐含的风险

在招标文件编写过程中，需求分析是非常关键的一步，特别是软件项目，前期需求的精确设定将直接影响到项目的后续实施，因此一定要避免一些可预见的风险。

在项目立项初期，需求分析占据了关键地位，它的准确与否直接关系到项目的成败。风险因素存在于需求获取、分析、验证和管理这一整个过程当中，要降低风险发生的可能性或减轻风险发生给项目带来的影响，必须在以下几个阶段着手采取措施规避风险。

需求获取阶段

● 绘制产品视图与范围。

如果团队成员没有对他们要做的产品功能达成一个清晰的共识，则很可能导致项目范围的逐渐扩大。因此最好在项目早期写一份项目视图与范围，将业务需求涵盖在内，并将其作为新的需求或修改需求的指导。

● 别挤压需求分析的时间。

紧张的工程进度安排给管理者造成很大的压力，使他们觉得不赶紧开始实施将无法按时完成项目，因而对需求一带而过。但项目因其规模和应用种类不同（如企业信息门户系统、邮件系统、网络管理系统）而有着很大的区别。粗略的统计表明： 需求开发工作应占全部工作量的 15%。

● 保证需求规格说明的完整性和正确性。

编制需求的往往是信息部门员工，而软件使用者来自业务部门，所以双方的沟通非常重要。设计者要以用户的任务为中心，根据不同的使用情景编写需求测试用例、建立原型，使需求更加直观，同时获取使用者的反馈信息，最后请专家对需求规格说明和分析模型进行正式的评审。

● 明确非功能需求和未加说明的需求。

由于使用者一般只强调产品的功能性要求，非常容易忽略产品的非功能性的需求。应该查明产品使用性、完整性、可靠性等质量特性，还有人性化的展示方式、查询方式，以此编写非功能需求文档和验收标准，作为可接受的标准。

另外，软件使用者可能会有一些隐含的期望要求，但并未说明，设计者要尽量识别并记录这些假设。提出大量的问题来引导使用者充分表达他们的想法和应关注的一切问题。如果不同的使用者对产品有不同的意见，那最后结果必将让有些使用者不满意。确定出主要的使用者，并采用产品代表的方法来确保使用者代表的积极参与，保证在需求决定权上有正确的人选。

● 别把已有的产品作为需求基线。

在升级或重做的项目中，需求开发可能显得并不重要。开发人员有时被迫把已有的产品作为需求说明的来源，认为“只是修改一些错误和增加一些新特性”，这时的开发人员不得不通过现有产品的逆向工程（reverseengineering）来获取需求。可是，逆向工程对收集需求是一种既不充分也不完整的方法，

因此新系统很可能会有一些与现有系统同样的缺陷。应当将在逆向工程中收集的需求编写成文档，并让专家评审以确保其正确性。

需求分析阶段

● 划分需求优先级。

设计者应划分出每项需求、特性或使用实例的优先级，并安排在特定的产品版本或实现步骤中。评估每项新需求的优先级并与已有余下的工作主体相对比以做出适宜的决策。

● 找到带来技术困难的特性。

设计者应分析每项需求的可行性以确定是否能按计划实现。成功好像总是悬于一线的，因此应运用项目状态跟踪的办法管理那些落后于计划安排的需求，并尽早采取纠正措施。

● 关注不熟悉的技术、方法、语言、工具或硬件平台。

设计者不要低估了学习曲线中表明的满足某项需求的新技术发展速度，明确那些高风险的需求并允许使用者有一段充裕时间用来从错误开始学习、实验及进行原型测试。

需求规格说明阶段

● 准确理解需求。

开发人员和使用者对需求的不同理解会带来彼此间的期望差异，将导致最终产品无法满足使用者的要求。对需求文档进行正式评审的团队应包括开发人员，测试人员和使用者。训练有素且颇有经验的需求分析人员能通过询问使用者一些合适的问题，写出更好的规格说明。模型和原型能从不同角度说明需求，这样可解决一些模糊的需求。

● 减少时间压力对未确定问题的影响。

将软件需求规格说明中需要将来进一步解决的需求注上 TBD（“待确定”）记号，但如果这些 TBD 并未解决，则将给结构设计与项目继续进行带来很大风险。因此应记录解决每项 TBD 的负责人的名字、问题是如何解决的以及解决的截止日期。

● 避免具有二义性的术语。

建立一本术语和数据字典，用于定义所有的业务和技术词汇，以防止它被不同的读者理解为不同的意思。特别是要清楚说明那些既有普通含义又有专用领域含义的词语。

● 不要在需求说明中包括设计。

包含在需求说明中的设计方法将对开发人员造成多余的限制，并妨碍他们进行最佳设计的创造性。仔细评审需求说明以确保它是在强调解决业务问题需要做什么，而不是在说怎么做。

需求验证阶段

● 未经验证的需求。

审查相当篇幅的需求文档是件烦琐的事，这就像要在开发过程早期编写测试用例一样。但如果在构造设计开始之前通过验证基于需求的测试计划和原型测试来验证需求的正确性及其质量，就能大大减少项目后期的返工现象。在项目计划中应为这些保证质量的活动预留时间并提供资源。

● 审查的有效性。

如果评审人员不懂得怎样正确地评审需求文档和怎样做到有效评审，那么很可能会遗留一些严重的不足之处。所以要对参与需求文档评审的所有团队成员进行培训，组织内部有经验的评审专家或外界的咨询顾问来做讲座，以便使评审工作更加有效。

需求管理阶段

● 减轻需求变更带来的影响。

设计者将项目视图与范围文档作为变更的参照基础，可以减少项目范围的延伸。使用者如果本着友好合作的态度，可把需求变更减少近一半。能在早期发现需求错误的质量控制方法可以减少以后发生变更的可能。而为了减少需求变更的影响，设计者要将那些易于变更的需求用多种方案实现，并在设计时注重其可修改性。

● 管理需求变更。

需求变更的风险来源于未曾明确的变更过程，或采用的变动机制无效，亦或是不按计划的过程来做出变更。设计者应当在开发的各阶层都建立变更管理的纪律和氛围，当然这需要时间。需求变更过程包括对变更的影响评估，提供决策的变更控制委员会，以及支持确定重要起点步骤的工具。

● 减少未实现的需求。

需求跟踪能力矩阵有助于避免在设计、结构建立及测试期间遗漏任何需求，也有助于避免因为交流不充分而导致多个开发人员都未实现某项需求。

● 扩充项目范围。

如果开始时就没有很好也定义需求，那么很可能隔一段时间就要扩充一次项目的范围。产品中未说明白的地方将耗费比预期更多的工作量，而且按最初需求所分配好的项目资源也可能要按用户的实际需求而调整。为减少这些风险，要对阶段递增式的生存期制定计划，在早期版本中实现核心功能，并在以后的阶段中逐步增加功能以实现需求。

浅谈软件测试规范

什么叫测试规范，大的来说是为了保障在测试过程中所做的一切是有序、有效、合理等；小的来说可以用“可控”二个字来概括，既然是浅谈测试规范，那就从小的开始讲。

例如测试负责人跑过来跟项目负责人说，这个产品我已经测试完成了，发现 XXX 个 BUG。好了，我们的项目负责人的可能存在几个疑问了：

1、是否按照产品需求的全部测试了？

2、测试所用的设备环境是否存在问题？

3、测试所使用的方法是否正确有效？

4、测试的时间是否超过了预期时间？

等等问题开始冒出来了。这里的问题其实就包含了人员技术的可控性、测试流程的可控性、设备环境的可控性、时间的可控性等因素。其实整个测试过程中包含了三种因素：人、设备和产品。

规范的做法是：测试人员首先需要经过系列的公司测试流程培训、测试方法及技术培训、公司设备维护与管理培训、产品缺陷分类培训、公司各种程序文件的培训等等后，再经过数个项目实际操作考核通过后。依据约束、规范和模板来设计测试计划，再根据测试计划来设计测试用例，依据测试用例进行测试实施测试产品并提交 BUG，完成后的测试报告提交这样的一个过程。 针对上面项目负责人的疑问，我们可以回头再去看：

针对第 1 个问题，根据公司测试流程，测试人员首先需要对产品需求进行熟悉后才进行功能和业务流程的提炼，然后通过 GB/T 所规定测试特性对产品功能进行全面的覆盖；

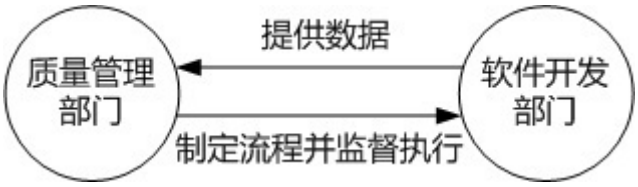
针对第 2 个问题，根据程序文件对公司设备的维护记录，只有合格的测试设备或软件环境才能给予使用，所以不会存在不合格的测试设备和软件环境；

针对第 3 个问题，根据公司测试方法及技术的培训和评审程序文件的约束，测试人员在编写好测试用例后，要交与测试负责人和其它评审组成员进行测试方法的评审，评审通过后才能按照测试方法实施测试；

针对第 4 个问题，一个合格的测试人员，会根据产品的技术特殊性，风险性，工作量进行合理的测试时间评估，以最大可能地减少误差。

所以要想保障测试的规范性，就要通过保障人员技术的规范性，方法的规范性，实施的规范性、保障设备环境及其维护管理记录的规范性，软件环境及其维护记录的规范性和保障产品开发的规范性。从而达到“可控”目的。

相信在不少与软件开发相关的企业内，质量管理部门与软件开发部门在日常运作中形成了如下图所示的“哑铃形”组织结构。

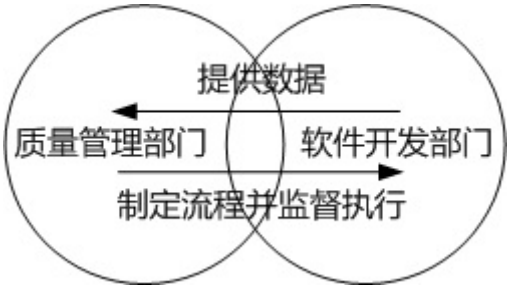


开发部门执行质量管理部门所制定的流程，通过提供证据的形式将各种流程执行后的数据反馈给质量管理部门（包括缺陷率和各种流程记录），质量管理部门根据这些数据监督流程的执行效果，并适时修订流程。联系两大独立部门的，是单薄的两条线和一些部门间的会议。理想情况下，在质量管理部门与软件开发部门间形成的是一个逆时针的良性质量管理环，理应获得良好的效果。但在我看来，事实却并非如此！

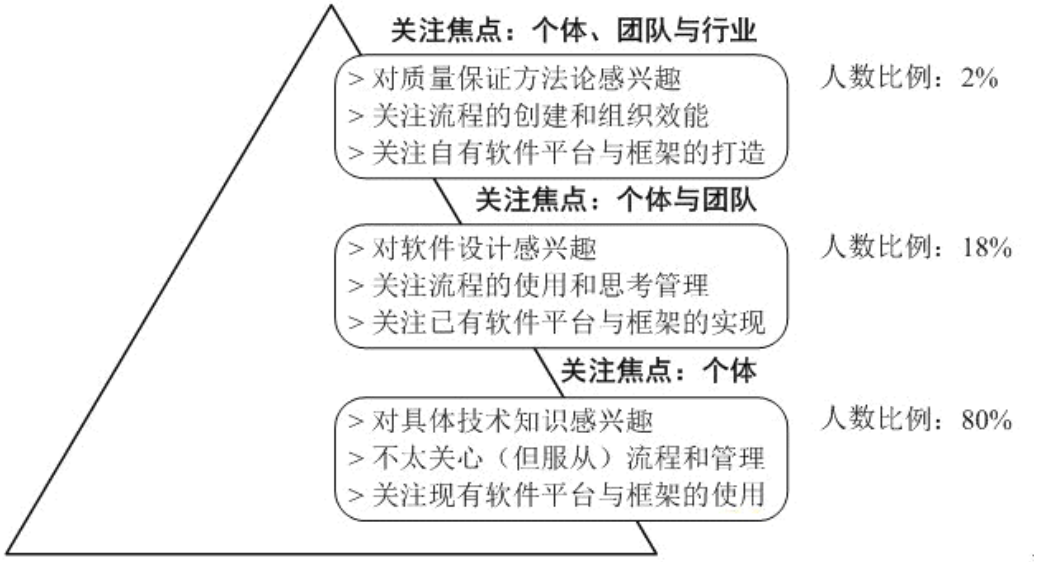
哑铃形组织结构所存在的前提假设有两个。其一，度量数据能真实地反映软件质量。显然，在软件危机仍四伏的今天，业内并没有找到完全能用于度量软件质量的指标，这一假设对于现实多少显得很是渺茫。其二，软件开发部门能诚实地提供度量数据。对于目前国内职业化程度不高的状态，这一假设也很难成立。

因此，哑铃形组织结构所带来的第一个困境是：将两个部门分别变成了“看数据的”和“造数据的”两大阵营。软件开发部门为了达到质量管理部门所制定的“质量目标”，不时需要考虑如何将数据“造”好，哪怕“造”的手法有点低劣；而质量管理部门由于只是通过数据去了解软件产品的质量状况，除了不能理解有些指标为何忽上忽下外，更无法督促开发部门就质量问题的根源进行根治。

克服这一困境的对策我认为需要从打破组织结构开始。真正掌握软件真实质量状况的并不是来自质量管理部门的人，因为他们根本没有触及软件源代码，而是来自开发部门的软件工程师。为此，两部门的人员应当存在交集才更有可能做好质量管理工作，或许下图的组织结构更有助于达到这一目的。



在新的组织结构中，两部门交集中的人应来自开发部门的、对软件质量管理有很好认识的技术专家，这些人来自下图“能力金字塔”的上面两层。他们除了帮助质量管理部门了解软件质量的真实状况外，还应帮助开发部门理解质量问题的根源和寻求技术解决方案。交集中的人可以考虑采用虚拟团队的形式进行组织与管理。



质量管理容易出现的另一大困境是：太强调流程与数据，而忽视质量管理很重要的内容是帮助工程师改善工作习惯（比如编程习惯）和提高开发环境的工作效率（比如项目的编译效率、单元测试的实施效率）。在这种困境之中，质量管理活动更多地表现为“钢性” — 达到设定指标或没有达到，而缺乏应有的“柔性”理解。虽然“产品质量源于过程控制”这一思想被业界广泛认同，但却仍容易忽视将工程师的工作习惯和开发环境的效率纳入到质量管理的范畴之中，这也是造成不少质量困境的关键因素。对于这两方面内容的重要性，无论如何强调也不为过。

最后，我认为质量管理应更多关注于实践，而非度量。由于软件开发的特殊性本质，我们难以寻找到有效的度量手段，与其在这方面毫无建树，不如花更多的时间去建立适合自己的实践方法，并将这些实践融入到工程师的工作习惯和开发环境中去。

本文出自李云的博客，请务必保留此出处 http://blog.csdn.net/hzliyun/article/details/8184538。

泽众软件工具使用技术支持

电话：021-61079698

Email：sales@spasvo.com

QQ：1404189128

MSN：spasvo_support@hotmail.com

产品租用		
下载	在线申请	详细

AutoRunner 是一款自动化测试工具。AutoRunner 可以用来执行重复的手工测试。主要用于：功能测试、回归测试的自动化。它采用数据驱动和参数化的理念，通过录制用户对被测系统的操作，生成自动化脚本，然后让计算机执行自动化脚本，达到提高测试效率，降低人工测试成本。

在线体验		产品租用	
企业版	免费版	在线申请	详情

TestCenter 是一款功能强大的测试管理工具，它实现了：测试需求管理、测试用例管理、测试业务组件管理、测试计划管理、测试执行、测试结果日志察看、测试结果分析、缺陷管理，并且支持测试需求和测试用例之间的关联关系，可以通过测试需求索引测试用例。

其他测试工具

Precise Project Management

Terminal AutoRunner

PerformanceRunner

有关培训、产品购买及试用授权方法等事宜

电话：021-61079698

Email： sales@spasvo.com

QQ：1404189128

MSN： jennyding0829@hotmail.com

SPASVO