你不知道的测试计划

话说测试计划就那么一些字儿，白纸黑字明明白白放那儿，应该不会有什么玄机。如果你是一个开发人员，这倒也不奇怪；如果你是一个测试计划制定者或审阅者，你还是觉得测试计划如此而已的话，那你可以好好看看我的见解，一不小心写得有点小多，希望不要看花眼。实在没耐心，如果看的起的话，可以直接把文档下了。

Part I 测试计划阅读的五重境界

在我看来，测试计划的作者和读者有以下五重境界。

第一重：什么都有用

对于一个测试新手来讲，好不容易找到一份测试计划模板，准备大干一场好好看看测试计划里面有哪些道道，看着看着发现很多东西都不知道，所以也分不清主次，自然也就觉得什么都很重要了。

第二重：什么都没用

当一个测试新手渐渐熟悉了测试的一些基础知识之后，回过头去看那些测试计划，发现里面什么“实质性”的内容都没有，没有他所关心的测试中的具体的方法，还没有一份测试用例来的有用。

第三重：仅部分有用

渐渐的，这个新手也不可避免地开始关注测试流程这一块的东西，再回过头看那个测试计划模板。这回感觉又不一样了，有些以前觉得没有多大用处的东西还是多多少少可以帮助我们更好的测试，比如测试计划模板中考虑到的要执行哪些类型的测试这部分内容应该就很有用，但是其他部分貌似还是很“虚”，一点实际用处都没有。

第四重：什么都有用

这是我自认为达到的水平级别~现在的我发现，一份好的测试计划模板中的所有内容都是有用的，包括风险分析这些我之前认为是用来凑字数的部分其实都是有着它的作用的，而且一份好的测试计划模板所包含的内容远远不止你从字面上读出来的那么简单，而这些也是我今天想要和大家一起分享的东西。

第五重：什么都没用

我还没有达到这种级别，所以这只是我揣测的一种境界。当某一类测试做的非常久非常熟了，对于这类测试的整个流程以及需要注意到的各个方面都已经烂熟于心，自然就不会把测试计划中的条条框框放在眼里了，或许这就是所谓的“随心所欲不逾矩”吧~（不过，我也想过，好记性不如烂笔头，或许这种达人级别的境界压根就没必要应用于实践吧。）

Part II 测试计划文档中容易被人忽略的部分

》 Project Goal & None Goal

说实话这是我之前认为测试计划里面最没用的部分，因此被我抛弃了很久时间，而据我所知这也是测试计划中最容易被人忽略的部分。不过，现在我却喜欢并且建议将这部分重视起来。作为一个项目来讲，尤其是产品类项目，整个 Team 需要明确自己应该做什么样的产品，不应该把产品做成什么样子，这个部分写在测试计划的第一部分，时不时瞅一瞅，提醒我们要向着正确的方向走。否则，在错误的道路上跑的越快，错的越远。

》版本历史信息和状态信息

这一部分容易被人忽略是因为几乎所有的文档中都有这一部分，或许因为这个缘故，这一块反而成了文档中最不受人关注的部分，大多数人一看文档直接跳到目录，甚至直接跳到内容的汪洋中大海捞针。版本变迁中最有用的部分是备注部分，一般这一部分介绍了文档最新更改的部分以帮助读者快速了解文档的一些基本情况。其次，其中的状态信息也会很有用，因为对于读者来讲，花费半小时看一份 Draft 是没有多大意义的。其他因为类似原因（因经常出现在各种文档中反而遭受忽略）而容易被人忽略的部分还包括“术语和缩略语”“引用”“文档介绍”“目录”，几乎所有的常见文档元素~

一份好的文档中这些部分都会恰到好处，读者阅读一份好的文档可能不会感受到欣喜，但是如果阅读一份没有或者写的很糟的文档则绝对会感受到痛苦甚至直接不看文档，这也从另一个方面导致了文档总是容易被人冷落，尤其是测试文档。

》测试接收标准和测试结束标准

这一部分主要是容易流于形式而被人忽略，对于很多项目来讲，根本没有所谓的标准而言，领导说开干，什么时候干好，ok，这就是开始标准和结束标准，而对于质量这些东西则早被抛到了最后。是的，或许有人会说，即使我们指定了一份好的测试标准，即使我们的领导也不会毫无理由的横加干涉，但是市场等原因也会造成产品在没有达到产品发布标准的时候发布出去。对于这个观点，网上通用的反对理由是：没有质量保证的产品最终会被淘汰，而且会累及公司的名誉。而我需要另外加一条理由：即使一个人系上安全带开车也会因为车祸挂掉，但系上安全带出事的概率要比不系要低很多吧。

》风险分析

我之前在写测试计划的时候，这一块一直是流于形式的客套话，写完了就完了，从此再也不去管它，没有把风险分析的作用利用起来。关于风险分析，文章后面还会专门提到。

Part III 测试计划文档中隐含的信息

》 优先级

或许文档的作者并没有直接标出那些计划事项是具有高优先级，哪些是低优先级的工作项，如果在这种情况下读者仍然能很清晰地知道自己先做什么后做什么——至少应该知道今天和明天

应该做什么吧——的话，那么测试计划的作者很可能把优先级隐含到了测试进度（Test Scheduler）安排这一部分了，一般来讲先要完成的事情优先级是最高的，而直接将优先级融入测试进度安排也是一种不错的选择。不过这种做法也有一些弊端，如果将工作项“写死”到进度安排中，当遇到某个工作项暂时延迟的时候会造成 Test Scheduler 的变化而影响其他工作项的执行时间。

》 Uncovered

在测试计划中，有一个部分叫做 Test Scope，而这一部分一般又会被划分成 Covered 和 Uncovered 两个部分。这两部分有什么玄机呢？大家应该知道测试的无穷尽特征，想到了这一点可能会有人马上反应过来：那 Uncovered 部分岂不是有很多内容？那为什么事实上 Uncovered 部分并没有洋洋洒洒几千字将我们没有做到的尽可能列出来呢？其实，一份测试计划只能表现出在特定项目中的测试（比如如果不需要 security test， 那么测试计划中可能就不曾提到 security test，甚至在 not Covered 部分也未曾提到。测试类型方法太多了，如果都在 not Covered 部分提到，那完全可以另外出一本书了），所以 Uncovered 部分提到的只是常见的测试类型或者方法，以及部分功能或者 UI 等内容，这部分是告诉读者，这一部分我们在测试里面不会——至少是不会专门设计相关的测试用例——测试的啊。这时候，我们一般会在 Uncovered 内容的后半部分看到关于为什么不覆盖到这一部分的“官方解释”。

》 文档的可读性

对于很多项目来讲，文档的目的是要让项目组成员读懂看清——至于目标读者是不是读了看了乃至懂了，那是另外一码事，所以作为文档的作者来讲，应该注意文档的可读性。之前提到的目录，术语和缩略词以及历史信息这些部分都有助于读者理解和使用文档。因此，保证文档结构的完整性是很重要的。另外，如果自己或者对方英语水平不太好，如无特殊要求，使用母语来写文档无疑有助于拉近作者与读者的距离。总之，作为作者应该始终记住：自己一个人看的文档和给所有人看的文档是不一样的。

》 标准是可以度量的

前面也讲了，标准这一块很重要，但是为什么很多标准成了形式主义呢？有一个很重要的原因就是标准是无法度量的，如果我们的标准定为“开发出一个很受欢迎的产品”，这就有点扯了，先不说你的“很”是怎么衡量的，单说“受欢迎”这个词儿，一千个观众眼里就有一千个哈姆莱特。因此，一个好的测试计划中是应该包含可以衡量的测试停止标准的。关于测试停止标准的话题，可以参见我的一篇文章[计划测试系列（七）——我们什么时候停止？](#) 浅薄之见~

Part IV 测试计划文档之外的东西

》 测试计划应该持续更新

这个观点早在 Ron Patton 的那本经典书籍《软件测试》中就已经提到过了，但是真正实践起来的却不知道有多少。没有实践的原因自然有很多（人总是喜欢并擅长寻找各种各样千奇百怪的理由来搪塞，而不喜欢花一点点时间和心思在本来就该完成的事情上），譬如没时间，反正没

人看等等。支持测试计划应该更新的最主要理由是，测试计划中有些内容需要实时更新。还记得前面提到过的风险分析么？项目的风险随着项目的推进有的被解决，也会有新的风险被发掘出来，而这些都应该通过文档记录下来，以保证任何人在任何时刻查阅都可以看到最新的风险分析信息。

》 测试计划之前应该做的事情

测试计划不应该是找到一份模板之后随便整整就成了自己的测试计划，那样的测试计划难免会成为案头一堆灰尘嬉戏的天堂。为了完成一份有价值的测试文档，除了上面提到的一些内容之外，我们还应该注意到以下几点：

-通读 Feature Spec 并理解需求

-了解开发设计文档

-分析风险和未决问题

-让 Team 的人了解你准备怎么做测试

-寻找一个测试计划模板　　最好是项目组或者公司已有的模板。如果你觉得模板不够好，也可以自己写一个并持续改进，但是我想为了保证测试计划的完整性，模板上的考虑应该还是比没有经验的测试计划设计者要全面的多。当然，我不建议不加筛选地随便在网络上找一个模板草草了事，模板要找但也要选，而且是精选。

》 测试计划之后应该做的事情

不是测试计划一完成就万事大吉了，除了前面提到的要持续更新之外，还有一些事情需要我们去做。

-召集相关人员召开测试计划评审会议

-根据评审的结果更新测试计划文档

Review 貌似是很多公司不存在的一个活动，不要说测试计划 review，甚至连 code review meeting 都是百年难得一遇的奇观。至于为什么我们应该 review，乃至为什么要有测试计划，甚至为什么要有测试，这样的问题不是本文的目的，我也不继续浪费大家的时间和自己的口水在这种问题上了。

Part V 总结

看似简单的测试计划，细细品尝起来还是别有一番风味。这篇文章本来在一开始的时候只是用来发表一下对于最近一个项目的感触和总结一下经验，没想到一下笔就写不完了。从开始写文章到今天落笔，前前后后花了一周的时间，所以衷心希望能给大家有所帮助。在第一部分讲五重境界的时候说了，我说过自己才刚刚上升到第四重，所以文中难免有纰漏，希望兄弟姐妹们可以指

出来并允许我添加到文中以方便后来的读者。另外，以上介绍的内容均来自实践经验和个人总结，由于项目千差万别，不同的项目的测试计划应该区别对待，切不可盲目照搬，以上观点仅供参考。

说了这么多，不知道对大家有没有帮助。以前看过我的《计划测试系列》文章的园子里的兄弟姐妹可能会发现这篇文章中讲到的内容似乎跟之前我提倡的“草根的实用主义”不一致，似乎违背了“够用就好”的原则，对此我想要重申（前文参见“杀鸡不用牛刀，杀牛亦不能用剪刀”）一次我的看法：草根的实用主义不仅仅在于够用就好，还要求我们要不断进步。如果，你刚刚加入这个行业，一下子灌输给你太多的规则只会适得其反，因此我们应该从简单的基本的地方着手，但是我们也要看到前面还有更好的方法和流程在等着我们学习，因此裹足不前，满足于够用就好的现状是不对的。我们不一定可以成为微软之流，但是我们可以有这份理想。尽管拥有了成为雄鹰的心，我们也不一定能成为雄鹰翱翔于蓝天，但是如果连成为雄鹰的心都没有，我们便永远成不了雄鹰。

PS：末段纯属闲扯，对于我的任何观点，所有读者都有权利发表评论。本博客言论自由，绝不删帖~

看到这儿了，如果对于文章或者观点有什么看法，点击在原文后留言，欢迎讨论~

原文链接：http://www.cnblogs.com/wuchaodong/archive/2009/03/12/1410095.html