目 录 (四十八期 下)

测试女巫之做个有灵魂的测试师

◆ 作者：王平平

摘要

将统计学_6 sigma 应用到测试管理中，需要使用的 6 sigma 工具如下：配对双样本检验，鱼骨图，柏拉图，假设检定，箱形图分析。

一、前言：

测试女巫今天非常感性，因为女巫刚刚看了朴树的小型演唱会，沉浸在淡淡的忧伤中，想起来女巫这么多年的工作，又想起来很久之前就一直听到，最近因为某位资深工程师的跳楼造成更火的话题：“中年危机”，所以更加深了这个忧伤感，但是现实不容许我们这种已经工作 10 多年的“资深底层管理人员或者工程师”沉浸在这个淡淡忧伤中，不是吗？女巫今年下半年有了本质的进步，所以终于开始有步骤有计划地脱离从事了 10 来年的“黑盒测试”，但是女巫就是这么感性，虽然要跟它 Say Bye 了，但是我还是希望能跟它好好地道别，因为它陪伴了我十几年，爱也好，恨也罢，我对它还是充满了感情，怎么 Say Bye，希望根据这十几年的工作经验，还是利用我的 6 sigma 工具，对它做个总结，在未来的日子里我以及我的团队不会再做黑盒测试了，但是公司或者看 51 Testing 杂志的兄弟姐妹还在做黑盒测试，这个就算我：一个做了 10 来年的黑盒测试老大姐（或者说老女巫）给大家留的一些“遗产”吧^^。黑盒测试也是需要灵魂，不管是测试开发，白盒测试，测试管理还是纯黑盒测试，无论哪种类型的工作，如果一直处于没有灵魂的工作，是早晚会被淘汰的……

这次我们学习新的“魔法”是什么呢？是我们学习外部的先进的测试思想，将其总结为测试工作流程，使用 6 sigma 工具将这个新的测试工作流程导入到我们的实际测试工作。使用的统计学知识如下：标准差和平均值的概念（此观念在第 39 期杂志上做了详细说明），柏拉图（此概念在第 38 期杂志上做了详细说明），箱型图(此概念在第 47 期杂志做了详细的说明)，配对双样本检验，鱼骨图

对于上述已经讲过的知识点，这里不再详细说明，亲如果忘记了，翻开对应期刊杂志看看吧，温故而知新，不亦乐乎嘛。

闲话不多说，我们开始正式学习吧！

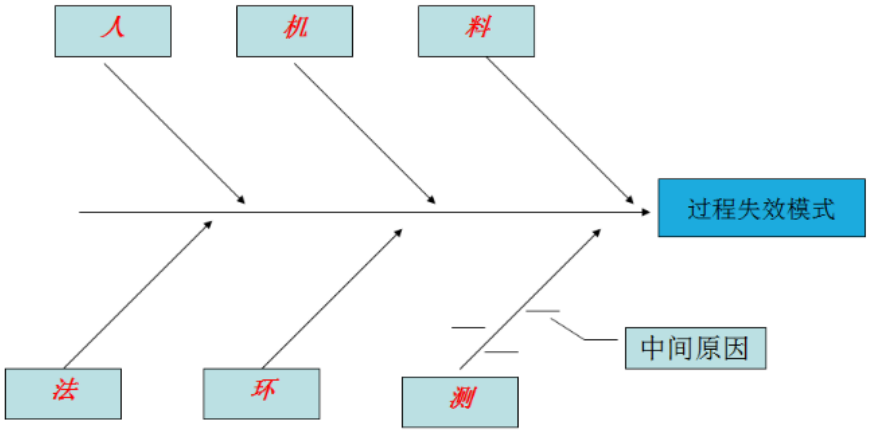
二、6 sigma 新常用工具基础知识介绍

1、鱼骨图

鱼骨图又叫鱼刺图或因果图，它是一种结构化的方法，它引导团队使用这个结构化的方法分析哪些原因影响结果，鱼头部分为问题的结果，鱼骨的分支为一层层的问题的原因。注意它只是提供一个分析问题的方法，具体的分析流程还是需要我们去一步步的做。

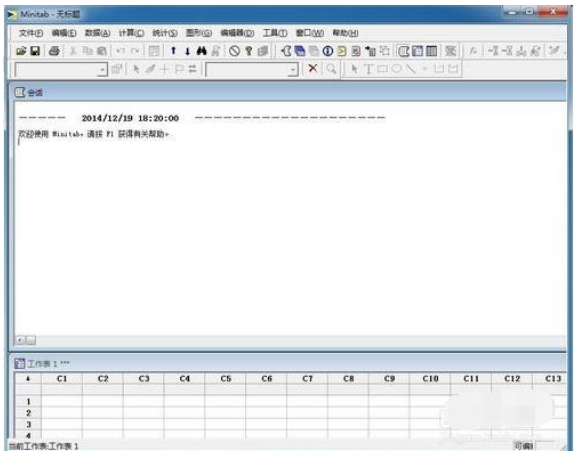
如下图，“过程失效模式”即就是“果”；“人”，“机”，“料”，“法”，“环”，“测”这六大方向是引导大家从这几个角度去分析导致这个“果”的因的分析方向；每个方向都需要我们自己根据自己的问题，与团队中的成员进行头脑风暴，将鱼的骨头，更加细化的罗列出来。

最后还是需要团队中的成员，开会讨论，哪些原因是影响“果”的关键原因。



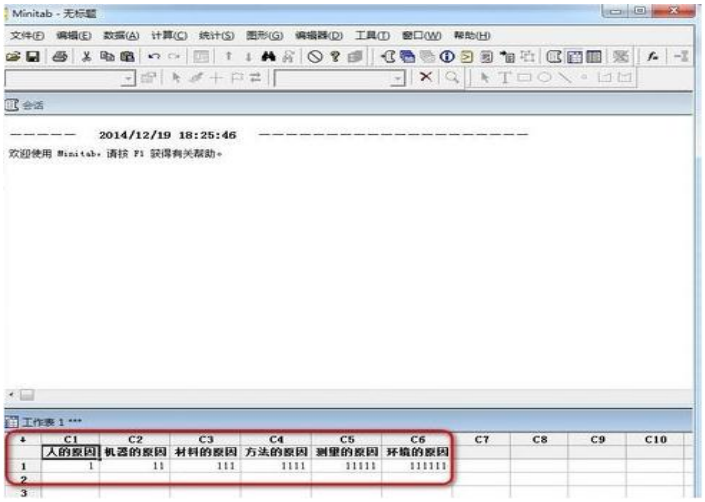
箱型图如何在 Minitab 上操作

1) 打开 MiniTab

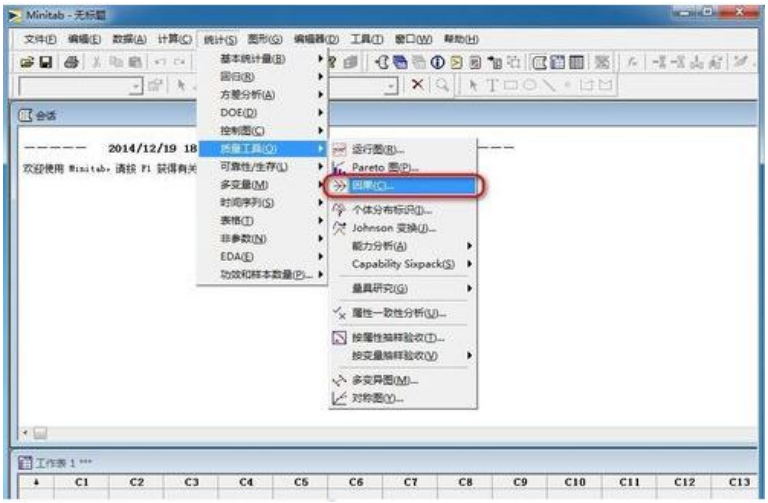


2) 在工作表中，写出鱼骨图的分析思路：人，机，料，法，环，测

注意这部分必须自己写，鱼骨图只提供这样的分析问题的架构。



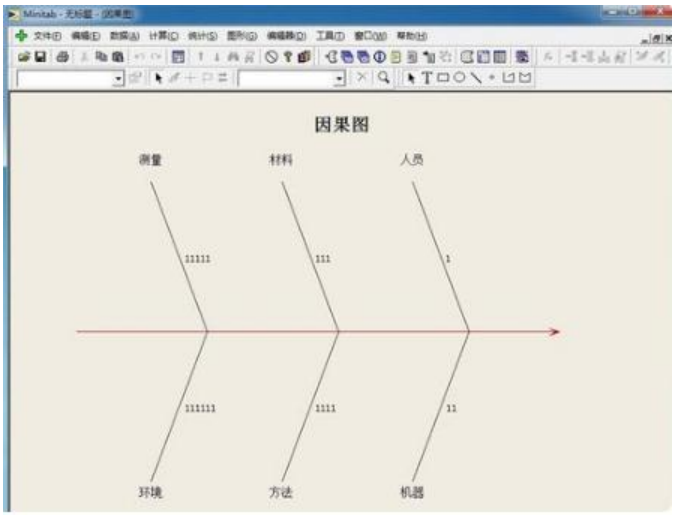
3) 选择“统计”->“质量工具”->“因果”



4) 注意“原因”是需要我们手动选择的，而标签是鱼骨图的架构，需要将你在工作

表中定义的原因与鱼骨图进行一一的对应。注意标签后面的“子原因”，就是大原因的分支，同大原因一样，需要现在工作表中定义，然后在此部分将相应的原因选进来。

5) 全部选择完毕，点击确认就可以生成你想生成的因果图如下:



2、双样本配对检验

它是应用假设检验的理论，我们曾经在第 38 期杂志介绍过双样本检验，我们的这个检验是与双样本检验非常类似，如果理解了双样本检验，那这个双样本配对 T 检验就非常容易理解。双样本检验就是检验两个样本是否有差异，例如两个产线生产出来的产品，我想知道针对某个参数，这两个产线是否有差异，就可以用双样本检定。而今天所讲的双样本配对检定就是，也是两个样本，只不过这两个样本是配对的，例如，一个人减肥前后的体重样本(样本数至少 30 个)，进行检验；这两个样本就是配对的样本，而我有两个样本是两个人减肥后的体重样本，这个检验就是双样本检验。

使用 Minitab 分析的方法:

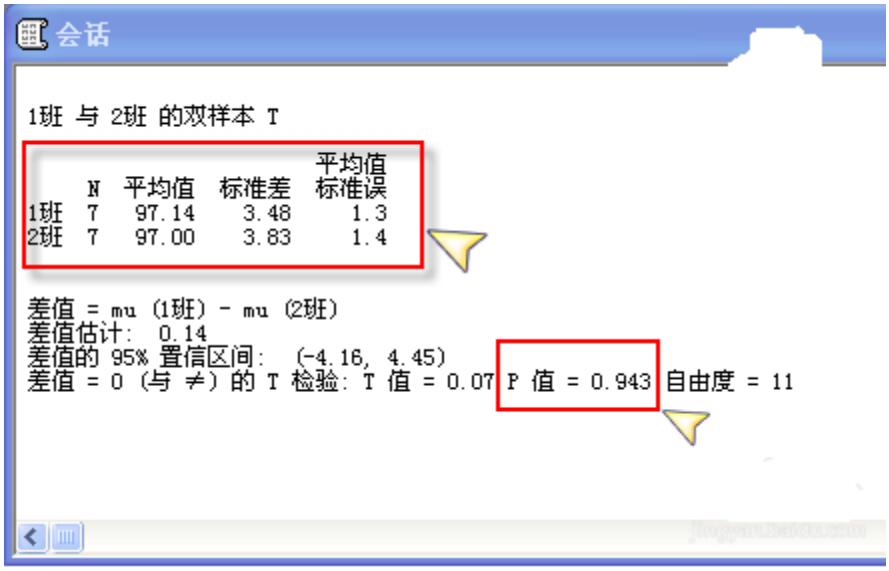
1) 打开样本数据

2) 选择统计(Stat)->基本统计量->双样本 T

3) 根据数据的形式选择样本的形式

4) 对产出结果进行分析，得出结论

这个结果机会给出平均值，标准差等基本统计学参数，还会给出统计学的结论：P-value（如何根据 p-value 得出最后的解决，此部分已经在假设检验中做了说明，这里就不再赘述）



三、如何将先进的测试工作流程引入到实际工作中

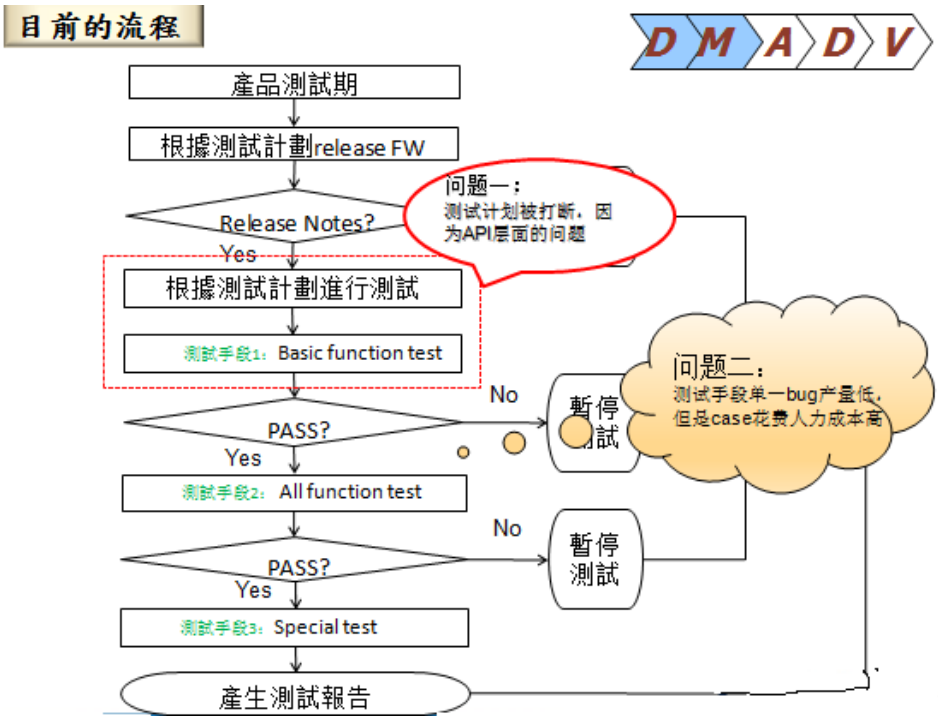
对于黑盒测试，我们一直做得测试工作就是根据需求准备测试用例，根据开发的时间表，制定我们的测试时间表，这个时间表就是：测试人员拿着极其详尽完备的测试用例进行一条条的测试，这个就是目前黑盒测试的状态，这个测试状态对吗？事实上，在实际工作中，作为测试主管深知，每一次测试如果都是按照“完备的测试用例”进行测试，你们产出的 bug 数量会非常的难看，我们会让一些灵活的，有经验的测试人员进行 Free test。我 2017 年初的时候在网上看到“淘宝的一个探索式测试研究”文档，知道了“探索式测试”的概念，进而搜索这个概念的出处和相关信息，所以我就希望运用 6 sigma 工具将这个 Free test 变为一个真正正规的可控的流程。

1、分析步骤

1）我们先看一下现有的工作流程是什么

现有流程就是在根据测试计划进行测试安排，测试总类也非常简单：Basic function test，All function test and Special test

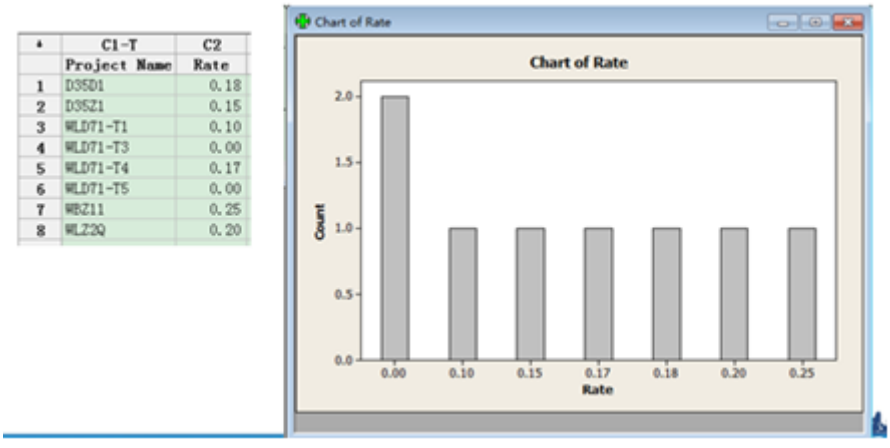
存在两个问题如下：



2）我们来看第一个问题：测试计划经常被打乱

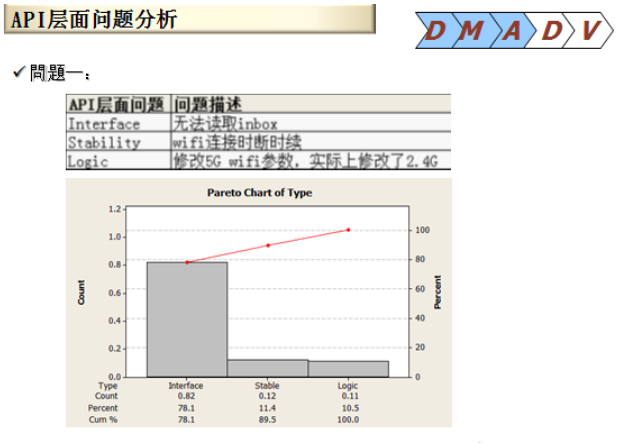
问题描述：

有时候 RD Released 的新版本会出现功能失效（经验证那些重大的问题有较多 API 层面的问题），我们统计了每个项目的退回率并进行分析。

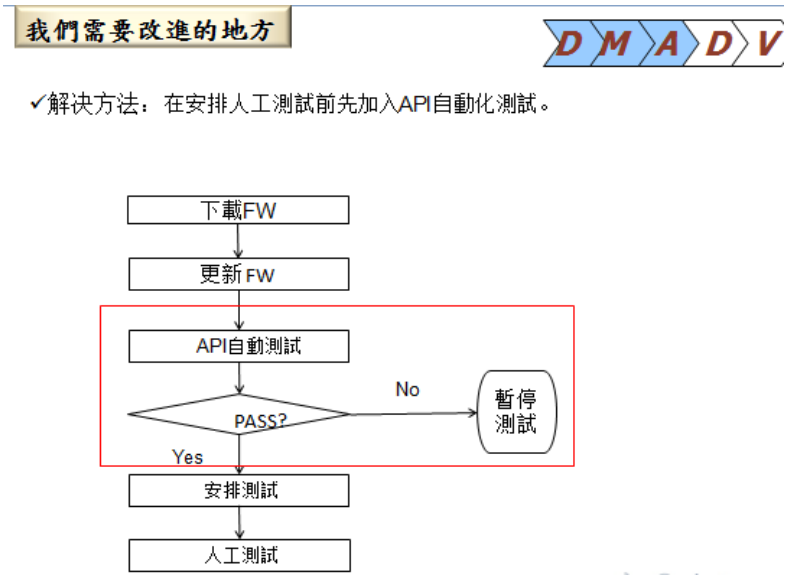


	C1-T	C2
	Project Name	Rate
1	D35D1	0.18
2	D35Z1	0.15
3	WLDT1-T1	0.10
4	WLDT1-T3	0.00
5	WLDT1-T4	0.17
6	WLDT1-T5	0.00
7	WRZ11	0.25
8	WLZ2Q	0.20

3）第一个问题的分析以及解决：



直接下对策：就是在人工测试前先将 API 进行自动化测试，保证 API 通过率满足大家制定的一个规则，才可以进行人工测试

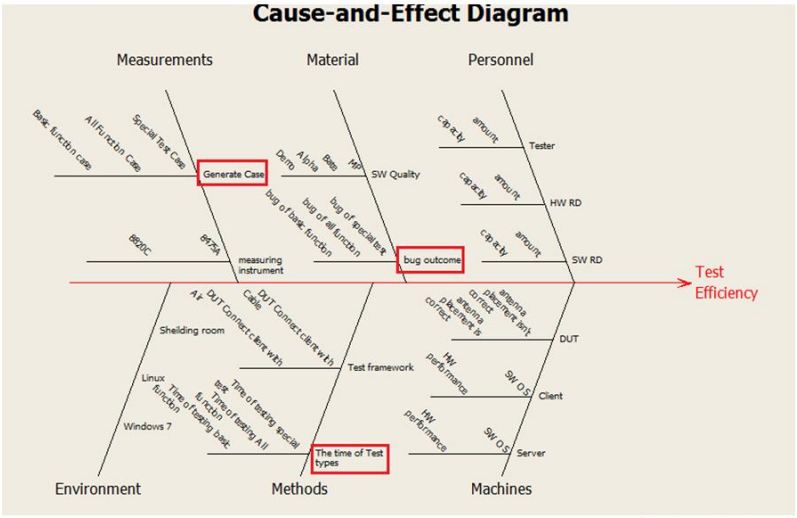


4）第二个问题的分析

这个分析就是使用鱼骨图，并召集相关的人开会，大家一起头脑风暴，根据鱼骨图的思考问题的方式，将关键因子找出来，下图中红框标出的因子就是大家认为的关键因子

問題二：影響手動測試效率原因

D M A D V



5）我们根据大家认为的关键因子制定出后续研究问题的方向：

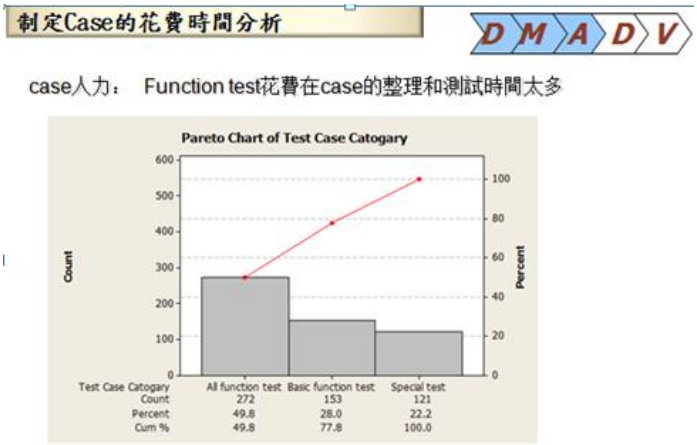
問題二：測試手法及產出分析

D M A D V

分析方向	說明
制定Case的花費時間	根據測試類型制定test case時間分析
測試類型執行的時間	分析一個完整執行的項目各個測試類型執行的時間
bug產量分析	分析多個項目，不同測試類型的bug產量

6) 分析研究方向 1：制定 Case 花费的时间：通过一个项目三种类型的 Case 花费时间进行柏拉图分析，分析结果如下：

说明：所谓 Function test 就是依据需求文文件，将每个功能的测试方法以及测试步骤一一描述清楚

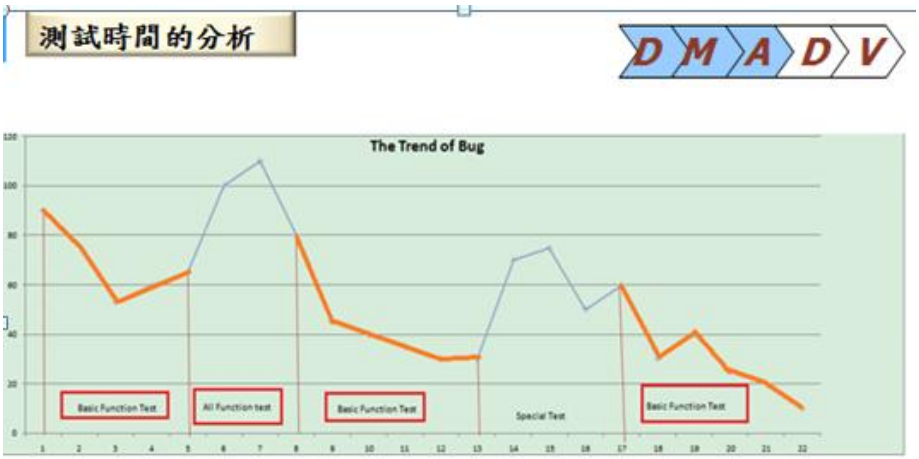


7) 分析研究方向 2：测试时间的分析

研究结果如下:

✓ 可以看出在 Basic function 测试周期占有很大的比率

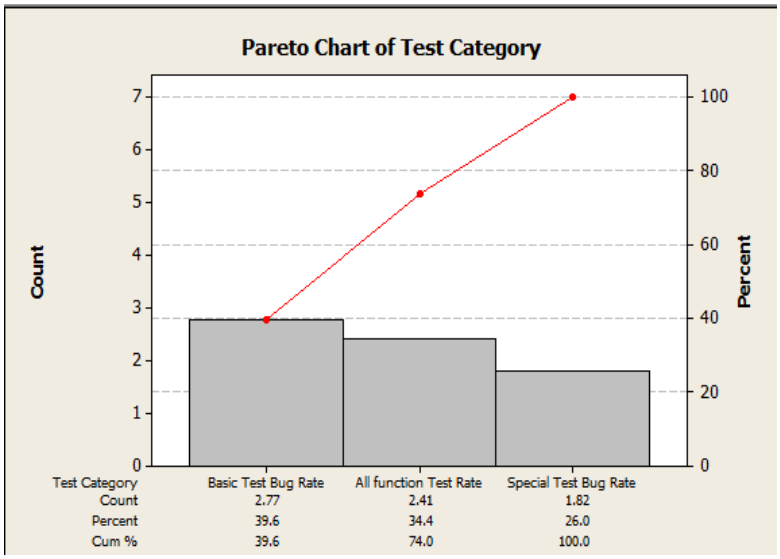
✓ Basic function+All function 这两种测试类型，几乎占了整个测试周期的五分之四



8) 分析研究方向 3：Bug 产量分析

三种测试手法在花费测试时间以及编写 Case 时间都不同的情况下产生的 Bug 数量相差无异。

结论：测试手法的单一性会无法面对系统对其的免疫性



9) 三个分析方向的总结:

a、过于依赖需求文档，花太多精力在制定 Function test case 上

(Function test 包括：Basic function test and all function test)

在测试的专业术语上，这叫做 Script-Based Test Management

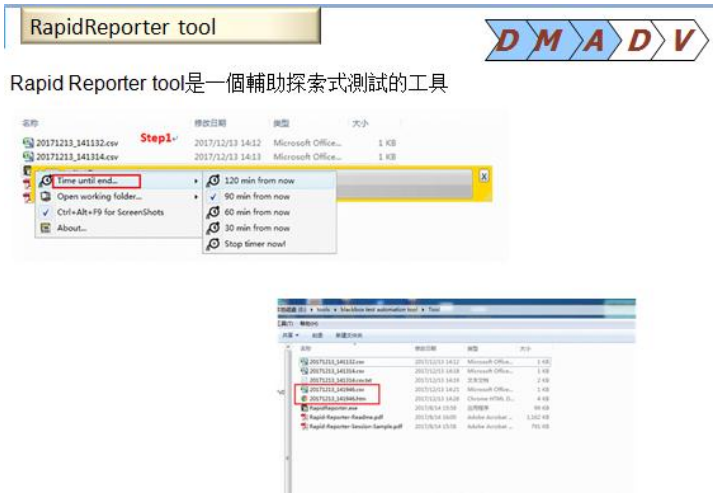
b、从产出看我们投入这么多精力在 Function test 上并没有非常亮眼的产出，Script-Based Test Management 太过于压抑自己的创造力和探索精神(感觉测试就是简单重复的执行测试用例)

我们为了解决这个问题在测试相关专业网站上找到如下信息:

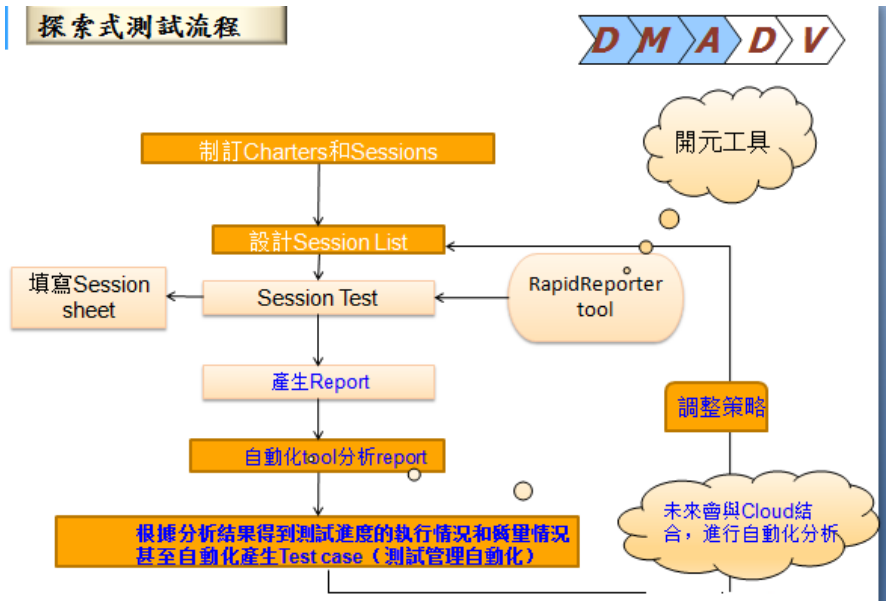
曾经任职为微软测试架构师和 Google 测试总监 James A. Whittaker 基于在微软的工作经历和积累，他撰写了《Exploratory Software Testing》一书，进一步扩展了探索式测试的概念和方法。Session-Based Test Management

10) 探索式测试有一个开源的免费的工具：RapidReporter Tool

目前还在从事黑盒测试的小伙伴们可以到网上找这个工具，非常小的一个工具，也非常容易使用



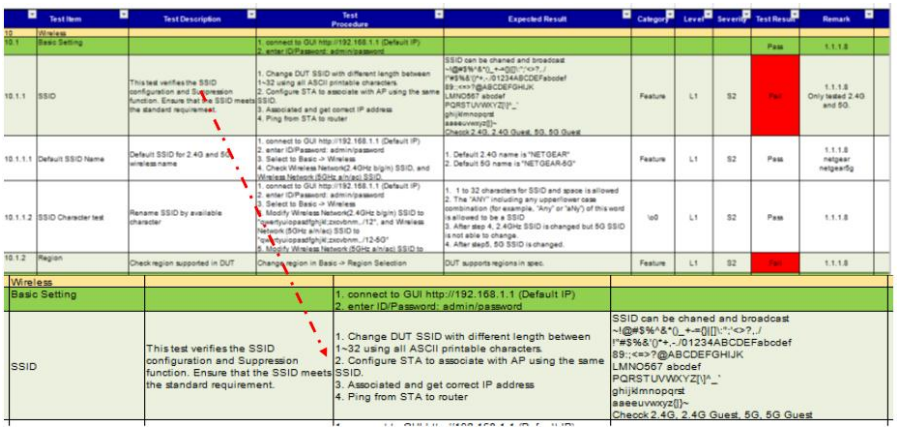
11) 根据探索式测试理念以及开源工具 RapidReoporter Tool 这个工具的研究，产生出如下新的工作流程如下:



12) 接下来我们对于传统式测试与探索式测试做一个简单的对比

第一个是测试用例的对比:

下图是传统测试的测试用例：我们要求对于待测物的测试流程要描述详尽，测试人员几乎可以不用思考，完全按照测试用例就可以完成测试传统测试即 Script-Based Test Management 设计的 Test Case Style



Test Item	Test Description	Test Procedure	Expected Result	Category	Level	Severity	Test Result	Remark
10 Wireless								
10.1 Basic Setting		1. connect to GUI http://192.168.1.1 (Default IP) 2. enter ID/Password: admin/password					Pass	1.1.1.8
10.1.1 SSID	This test verifies the SSID configuration and Suppression function. Ensure that the SSID meets the standard requirement.	1. Change DUT SSID with different length between 1~32 using all ASCII printable characters. 2. Configure STA to associate with AP using the same SSID. 3. Associated and get correct IP address 4. Ping from STA to router	SSID can be chaned and broadcast ~!@#$%^&*()_+-=[]{}\|\";'<>?,./ !"#$%&'()*+,-./01234ABCDEFabcdef 89:;<=>?@ABCDEFGHIJK LMNO567 abcdef PQRSTUVWXYZ[\]^_` ghijklmnopqrst aseeuvwxyz{|}~ Checck 2.4G, 2.4G Guest, 5G, 5G Guest	Feature	L1	S2	Fail	1.1.1.8 Only tested 2.4G and 5G.
10.1.1.1 Default SSID Name	Default SSID for 2.4G and 5G wireless name	1. connect to GUI http://192.168.1.1 (Default IP) 2. enter ID/Password: admin/password 3. Select to Basic -> Wireless 4. Check Wireless Network(2.4GHz b/g/n) SSID, and Wireless Network (5GHz a/n/ac) SSID	1. Default 2.4G name is "NETGEAR" 2. Default 5G name is "NETGEAR-5G"	Feature	L1	S2	Pass	1.1.1.8 netgear netgear5g
10.1.1.2 SSID Character test	Rename SSID by available character	1. connect to GUI http://192.168.1.1 (Default IP) 2. enter ID/Password: admin/password 3. Select to Basic -> Wireless 4. Modify Wireless Network(2.4GHz b/g/n) SSID to "qwertyuiopasdfghjkl;zxcvbnm,./12", and Wireless Network (5GHz a/n/ac) SSID to "qwertyuiopasdfghjkl;zxcvbnm,./12-50" 5. Modify Wireless Network (5GHz a/n/ac) SSID to	1. 1 to 32 characters for SSID and space is allowed 2. The "ANY" including any upper/lower case combination (for example, "Any" or "aNy") of this word is allowed to be a SSID 3. After step 4, 2.4GHz SSID is changed but 5G SSID is not able to change. 4. After step5, 5G SSID is changed.	\o0	L1	S2	Pass	1.1.1.8
10.1.2 Region	Check region supported in DUT	Change region in Basic -> Region Selection	DUT supports regions in spec.	Feature	L1	S2	Fail	1.1.1.8

Wireless			
Basic Setting		1. connect to GUI http://192.168.1.1 (Default IP) 2. enter ID/Password: admin/password	
SSID	This test verifies the SSID configuration and Suppression function. Ensure that the SSID meets the standard requirement.	1. Change DUT SSID with different length between 1~32 using all ASCII printable characters. 2. Configure STA to associate with AP using the same SSID. 3. Associated and get correct IP address 4. Ping from STA to router	SSID can be chaned and broadcast ~!@#$%^&*()_+-=[]{}\|\";'<>?,./ !"#$%&'()*+,-./01234ABCDEFabcdef 89:;<=>?@ABCDEFGHIJK LMNO567 abcdef PQRSTUVWXYZ[\]^_` ghijklmnopqrst aseeuvwxyz{|}~ Checck 2.4G, 2.4G Guest, 5G, 5G Guest

13) 探索式测试

探索式测试即 Session-Based Test Management 设计的 Test Case Style

- ✓ Session/Charters 字段是提供给 Tester 以保证测试方向以及覆盖率
- ✓ Test setup 以尽量简化的语言描述测试步骤，此字段提供了 Basic function test case 功能
- ✓ Reference SOP 辅助 Test setup，如果测试步骤复杂，则会整理一个 SOP 统一在部门内部分享
- ✓ Auto test script 对于基本且操作步骤简单的功能应该首先考虑自动化实现

Session List/Chartes		Test Step	Reference SOP	Corresponding Auto Test Script	Test Result	Notes
Wi-Fi Basic Setting(AP1)	Enable SSID	Enable SSID，无线搜索此SSID				
		Disable SSID，无线搜索此SSID				
	Edit SSID	任意编辑合法字符并保存，无线搜索此SSID				
	Broadcast SSID	开启Broadcast SSID，无线搜索此SSID				
		关闭Broadcast SSID，无线搜索此SSID				
		关闭Broadcast SSID，PC端手动添加该SSID并连接				
	Security Type	选择None，无线搜索并连接此SSID				
		选择WEP，无线搜索并连接此SSID				
		选择WPA，无线搜索并连接此SSID				
		选择WPA2，无线搜索并连接此SSID				
		选择WPA+WPA2，无线搜索并连接此SSID				
	Show Plain Password	勾选Show Plain Password，查看密码是否可见				
		不勾选Show Plain Password，查看密码是否可见				
	Beacon Interval	任意编辑范围内的值并保存				
		任意设置合法值，抓包查看Beacon包中Beacon Interval时间显示				
	RTS Threshold	任意编辑范围内的值并保存	\\172.30.100.1\njkr1500\KR1510\Project 匯集\WLD1&3\SOP\无线抓包测试SOP			
		任意设置合法值，PC1 Ping DUT's local IP -l (小于设置值) -t，PC2无线抓包，确认功能				
		任意设置合法值，PC1 Ping DUT's local IP -l (大于设置值) -t，PC2无线抓包，确认功能				
	Fragmentation Threshold	任意编辑范围内的值并保存	\\172.30.100.1\njkr1500\KR1510\Project 匯集\WLD1&3\SOP\无线抓包测			
		任意设置合法值，PC1 Ping DUT's local IP -l (小于设置值) -t，PC2无线抓包，确认功能				

14) 探索式测试即 Session-Based Test Management 执行范例

Tester 将会拿到的 Test case 范例

Session List/Chartes		Test Step
Wi-Fi Basic Setting(AP1)	Enable SSID	Enable SSID，无线搜索此SSID
		Disable SSID，无线搜索此SSID
	Edit SSID	任意编辑合法字符并保存，无线搜索此SSID

Tester 使用 RapidReporter tool 产生的 Report

Time	Reporter	Type	Content	Screenshot	RTF Note
2017/10/17 13:28:15	Mary	(Rapid Reporter version)	1.16.05.20		
2017/10/17 13:28:15	Mary	Session Reporter	Mary		
2017/10/17 13:28:15	Mary	Session Charter	Wi-Fi Basic Setting(AP1)-Enable SSID		
2017/10/17 13:28:40	Mary	Setup	PC Connect with DUT via USB cable		
2017/10/17 13:28:54	Mary	Note	Multi event		
2017/10/17 13:29:11	Mary	Test	SSID1 is enable		
2017/10/17 13:29:28	Mary	Test	plug out usb then plug in		
2017/10/17 13:29:51	Mary	Test	Client(NB) try to connect to SSID		
2017/10/17 13:30:24	Mary	Check	IF Client can connect to SSID or not		
2017/10/17 13:30:34	Mary	Bug	Mantis ID:21543		
2017/10/17 13:30:40	Mary	Question	no		
2017/10/17 13:31:32	Mary	Next Time	Maybe I can try other client for example Smart phone		
2017/10/17 13:31:47	Mary	Session End. Duration	00:03:32		

RapidReporter tool 产生的 Report 中对应的 Mantis Bug

Normal **Resolution** open

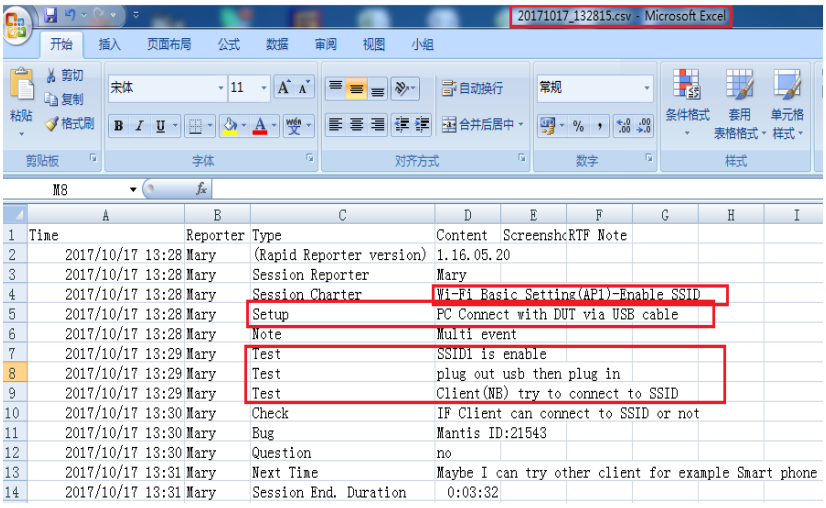
new

21543: A new Client can't connect to wifi after plug out usb.

[Step by step]:
1.PC connect with DUT via USB cable.
2.Wi-Fi->Wi-Fi Settings->SSID1 or SSID2->Modify settings->Apply
3.plug out usb then plug in
4.Client try to connect to wifi
[Actual]:
Step4:Client can't connect to wifi.
[Expect]:
Step4:Client can connect to wifi.

后续可以做的自动化: 因为是 csv 格式所以可以通过自动化，开源 Tool 中的关键词，为后续 Test case 自动化产生提供了前提条件。

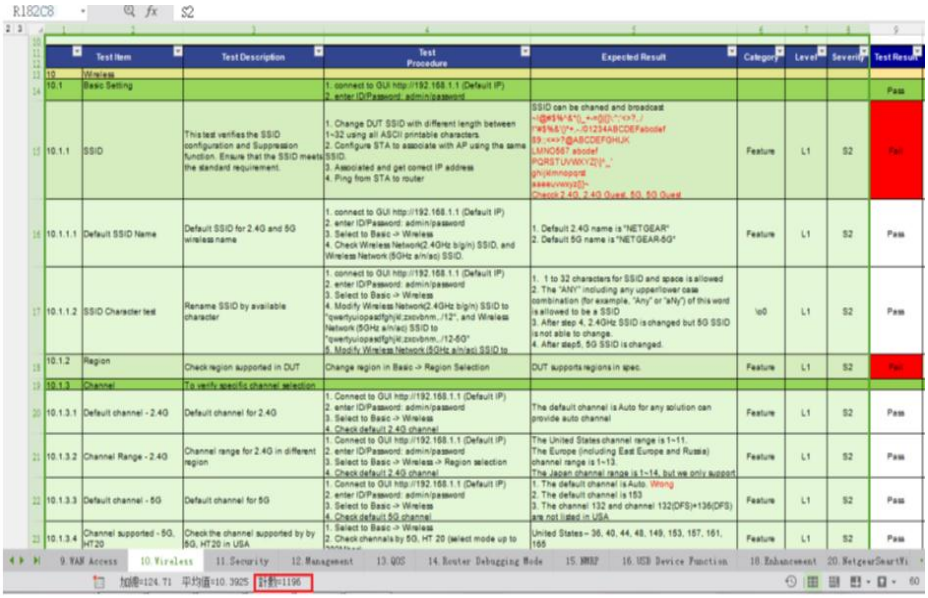
考虑将“测试系统生态化管理自动化”引入我们的测试流程中



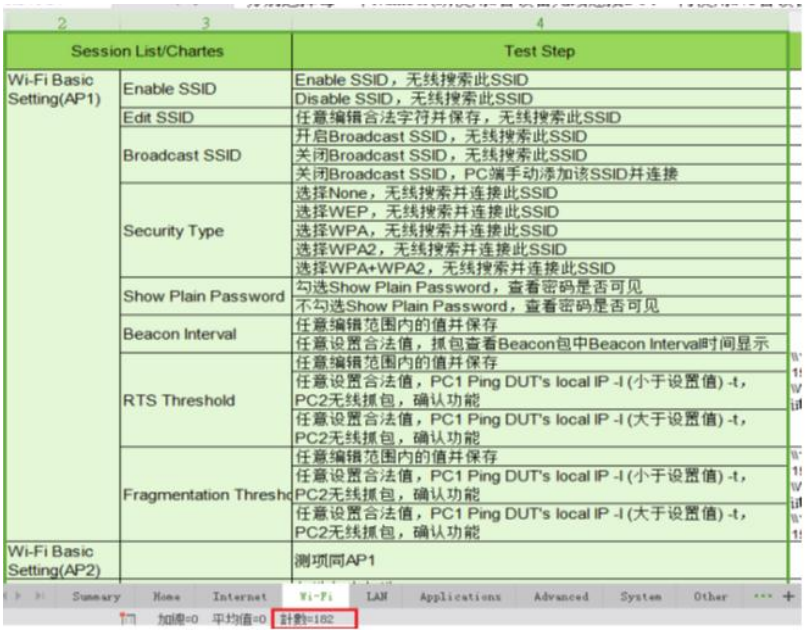
	A	B	C	D	E	F	G	H	I
1	Time	Reporter	Type	Content	Screensho	RTF Note			
2	2017/10/17 13:28	Mary	(Rapid Reporter version)	1.16.05.20					
3	2017/10/17 13:28	Mary	Session Reporter	Mary					
4	2017/10/17 13:28	Mary	Session Charter	Wi-Fi Basic Setting(AP1)-Enable SSID					
5	2017/10/17 13:28	Mary	Setup	PC Connect with DUT via USB cable					
6	2017/10/17 13:28	Mary	Note	Multi event					
7	2017/10/17 13:29	Mary	Test	SSID1 is enable					
8	2017/10/17 13:29	Mary	Test	plug out usb then plug in					
9	2017/10/17 13:29	Mary	Test	Client(NB) try to connect to SSID					
10	2017/10/17 13:30	Mary	Check	IF Client can connect to SSID or not					
11	2017/10/17 13:30	Mary	Bug	Mantis ID:21543					
12	2017/10/17 13:30	Mary	Question	no					
13	2017/10/17 13:31	Mary	Next Time	Maybe I can try other client for example Smart phone					
14	2017/10/17 13:31	Mary	Session End. Duration	0:03:32					

15) 同一个 DQA 在同一个项目三个 Function: WIFI Internet Firewall

传统测试建立 WIFI 此功能的 test case 需要书写 1196 个字体

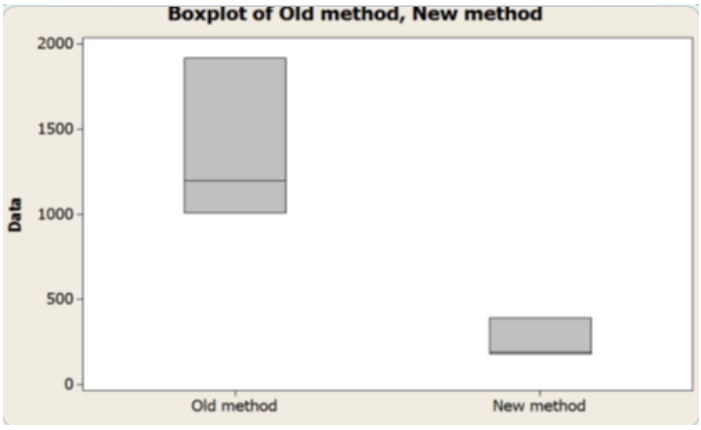


探索式测试建立 WIFI 此功能的 test case 需要书写 182 个字体

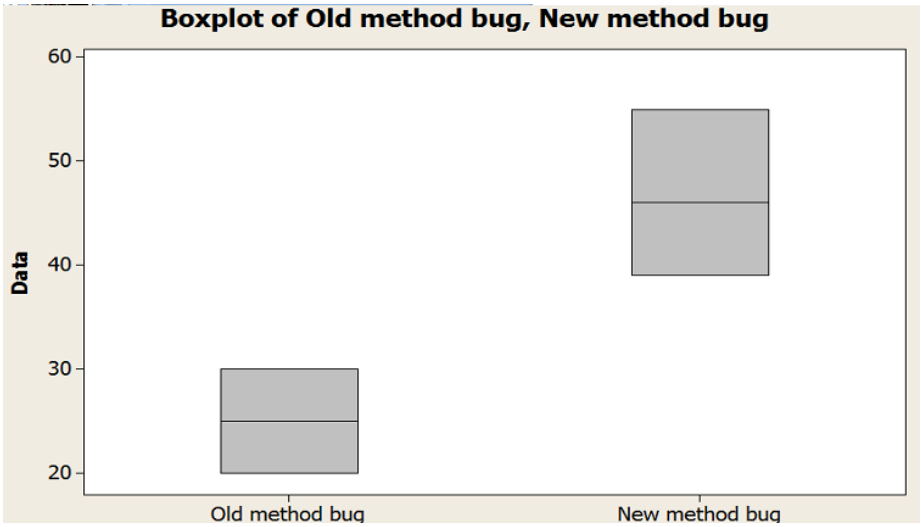


Session List/Charles		Test Step
Wi-Fi Basic Setting(AP1)	Enable SSID	Enable SSID，无线搜索此SSID
		Disable SSID，无线搜索此SSID
	Edit SSID	任意编辑合法字符并保存，无线搜索此SSID
	Broadcast SSID	开启Broadcast SSID，无线搜索此SSID
		关闭Broadcast SSID，无线搜索此SSID
		关闭Broadcast SSID，PC端手动添加该SSID并连接
	Security Type	选择None，无线搜索并连接此SSID
		选择WEP，无线搜索并连接此SSID
		选择WPA，无线搜索并连接此SSID
		选择WPA2，无线搜索并连接此SSID
		选择WPA+WPA2，无线搜索并连接此SSID
	Show Plain Password	勾选Show Plain Password，查看密码是否可见
		不勾选Show Plain Password，查看密码是否可见
	Beacon Interval	任意编辑范围内的值并保存
		任意设置合法值，抓包查看Beacon包中Beacon Interval时间显示
	RTS Threshold	任意编辑范围内的值并保存
		任意设置合法值，PC1 Ping DUT's local IP -l (小于设置值) -t，PC2无线抓包，确认功能
		任意设置合法值，PC1 Ping DUT's local IP -l (大于设置值) -t，PC2无线抓包，确认功能
	Fragmentation Thresh	任意编辑范围内的值并保存
		任意设置合法值，PC1 Ping DUT's local IP -l (小于设置值) -t，PC2无线抓包，确认功能
		任意设置合法值，PC1 Ping DUT's local IP -l (大于设置值) -t，PC2无线抓包，确认功能
Wi-Fi Basic Setting(AP2)		测项同AP1

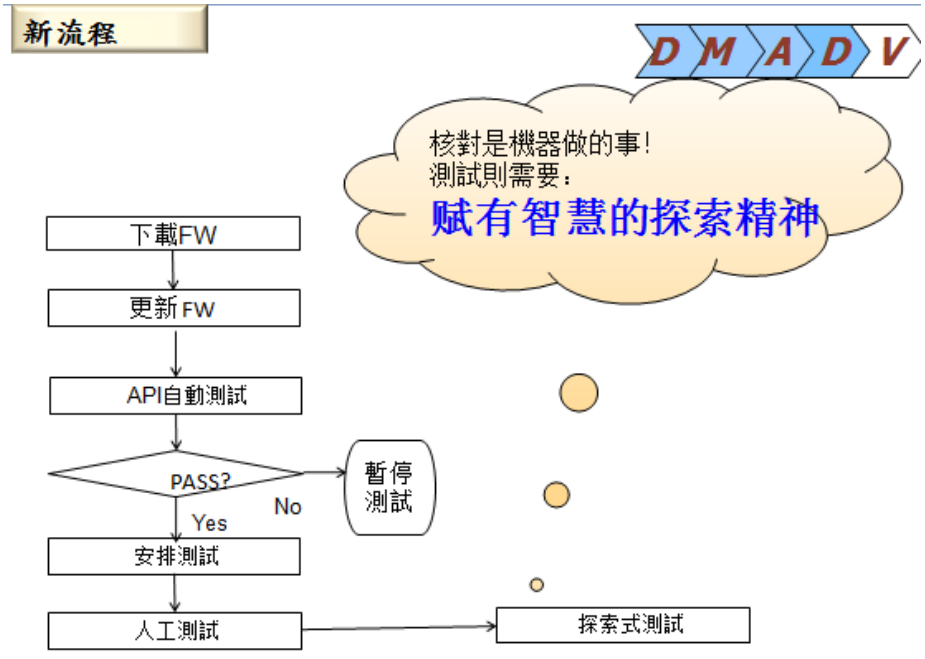
16）两种测试方式手动编写 test case 的字体数量的对比

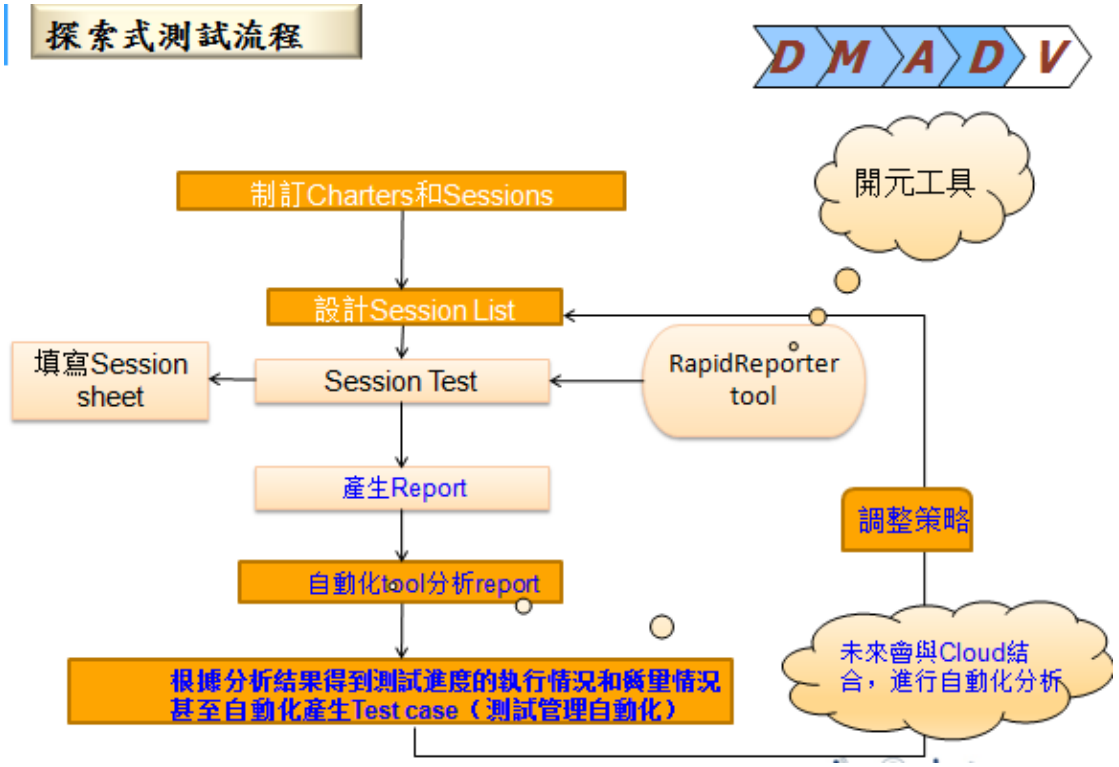


17) 两种测试方式发现 bug 数量的对比



2、改进的流程





3、验证新流程

同一个 DQA 在 4 个项目同三个 Function： WIFI， Internet，Firewall 针对 Test case 花费时间以及 bug 产出进行对比分析验证

C1	C2	C3	C4	C5
Test case_New Method	Test case_Old Method	Bug_New Method	Bug_Old Method	
2.0	17.1	9	12	
2.5	8.3	10	6	
4.2	16.8	21	11	
4.3	12.7	24	6	
3.1	12.4	15	7	
3.2	12.9	16	8	
2.4	8.8	10	6	
2.6	16.9	11	10	
4.6	17.2	23	12	
3.8	8.2	14	4	
3.2	8.6	16	6	
4.7	12.2	23	7	

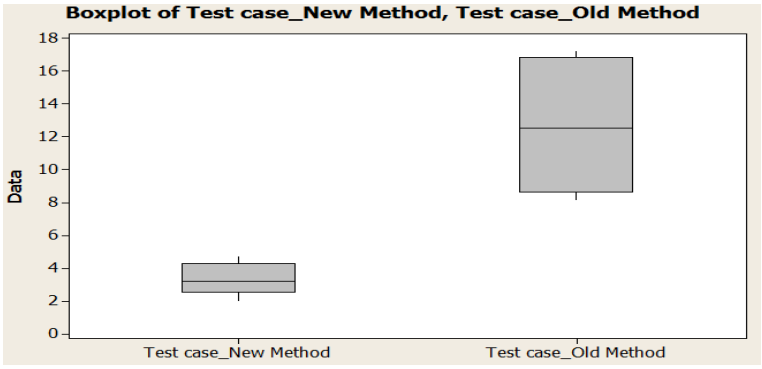
1) 针对 Test case 的验证结果

Paired T-Test and CI: Test case_New Method, Test case_Old Method

Paired T for Test case_New Method - Test case_Old Method

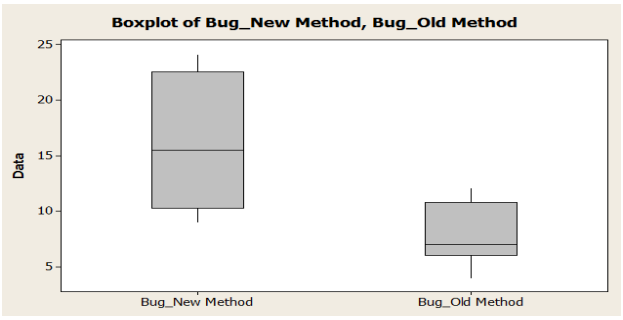
	N	Mean	StDev	SE Mean
Test case_New Me	12	3.3833	0.9203	0.2657
Test case_Old Me	12	12.6750	3.6440	1.0519
Difference	12	-9.29167	3.62453	1.04631

95% CI for mean difference: (-11.59458, -6.98875)
T-Test of mean difference = 0 (vs not = 0): T-Value = -8.88 P-Value = 0.000



2) 针对 bug 的验证结果

```
Paired T-Test and CI: Bug_New Method, Bug_Old Method

Paired T for Bug_New Method - Bug_Old Method

                    N     Mean    StDev  SE Mean
Bug_New Method     12  16.0000   5.5432   1.6002
Bug_Old Method     12   7.9167   2.6785   0.7732
Difference         12  8.08333  5.93079  1.71207

95% CI for mean difference: (4.31509, 11.85158)
T-Test of mean difference = 0 (vs not = 0): T-Value = 4.72 P-Value = 0.001
```



四、总结

我们在这一期中主要介绍了统计学中推论学上的重要的知识点：鱼骨图的基本概念以及如何使用 Minitab 画鱼骨图；配对双样本检验以及如何用 Minitab 进行配对双样本检验

这一期的重点是如何改进我们的黑盒测试流程，我们一直习以为常的工作流程不一定是对的流程，现在我们处于一个知识分享的时代，我们需要经常到 51testing 之类的专业网站上，多多吸收其他人，其他公司的资讯。只有自己的眼界宽了，看到的东西多了，才会有不断改善我们的工作的想法，现在大家都在讲 AI，人工智能。其实对于我们测试人员来讲，我们还是要着眼于我们手头的工作，每天都想着如何改善我们的工作，多学习各种工具，另外测试人员必须要学习一到两个编程语言，女巫推荐 Python，毕竟它适用于高大上 AI 的开发，我们即使目前对于 AI 一无所知，我们可以先尽量接近 AI。有了一定的编程基础，我们回过头来审视我们的测试工作，就会发现不少工作都可以用自动化实现。作为测试管理者，更要多学习一些工具，例如流程改善的工具：6 sigma 的一系

列工具，相信我，有了这些工具的加持，你看这个世界的眼光都变了，你会对于大家习以为常的工作流程，产生很多质疑，然后搜集工作中的一些数据，使用这些工具研究这些数据，产生高效的新的工作流程，这难道不是一种工作累计方式吗？

回到我们开头的引言：所谓的“中年危机”，女巫也工作了 10 多年了，没有感觉到所谓的“中年危机”，并不是我所在公司非常安稳，我所在公司，经历大大小小裁员，针对我团队的裁员不下 3 次，不可否认每次都很痛苦，但是你不能说，公司很无耻，公司的寒冬来了，如果它不断臂，它就会无法生存。我们不如把指责公司“无情”的手收回来，想想自己，自己的价值是什么，如何才能增值自己，让自己增加年龄的同时，也增加了别人无法企及的能力！

还是那句话：对于 6 sigma 可以带来的奇妙旅程，我将不懈的坚持探索，怀着赤子之心去探索如何使用这些工具去改善我的工作，反之在不断的使用这些工具的过程中又大大加深了我对统计学原理的认识，所以“路漫漫其修远兮，吾将充满欢喜的上下而求索”！

参考文献：

1、有关标准差基本概念介绍的网站：
http://baike.baidu.com/link?url=4v_m3jJhHUKllt1lA0tRe_I-pVurV1tNNrztJ6_PbZf0Me5rJr-bxIdp4fRCdsrsSIRd-hSFBglCEyZCkunk6K
2、有关介绍统计过程控制的网站：
http://baike.baidu.com/link?url=M9BLOK736USXqiqCHa2uhW624zYJpD-UXRAotTwo9PyTn-AWZv3GDHofg2Gz_uOqC42RK1GoKdLf-Wr30fWJR0_sgWYIj_ItTG-JSdB1Amm

定时重启进程的 Shell 脚本实例

◆作者：咖啡猫

题记:
笔者是一枚软件测试从业者，最近在一次任务中，需要编写定时重启 java 进程的脚本。由于之前只会一些 vim cd tailf mkdir cp scp rm ll 之类的简单指令，所以这次决定借此机会入个门。特此把这段经历记录下来，给广大同行参考。

一、如何实现定时执行任务

Linux 中通过 crontab 来运行定时任务。

1、安装 crontab: yum install crontabs

2、查看 crontab 的状态: service crond status

3、启动 crontab 服务: service crond start

4、编辑配置文件 crontab -e

配置文件格式为: minute hour day month dayofweek command

例如每天十二点执行重启脚本就配置为:

0 0 * * * /scriptpath/test.sh param

其中 scriptpath 为脚本路径 test.sh 为脚本名称 param 为脚本参数，此例中有 4 个参数: start stop restart status ， 分别代表开启、结束、重启、状态查询。

编辑完成后，记得: wq! 强制保存退出

5、在/scriptpath/路径下创建 test.sh 脚本

二、编写 test.sh 脚本

话不多说，直接贴代码:

第一段定义一些全局变量，running_user 指运行脚本的用户，APP_HOME 指 JAVA 程

序存放的路径，APP_JAR 指应用程序依赖的 JAR 包的位置，APP_MAINCLASS 指 JAVA 程序的主类，JAVA_OPT 配置的是 JAVA 虚拟机的内存配置参数，JAVA_HOME 是 jdk 的安装路径，psid 是 JAVA 进程号。

```
#! /bin/sh
source /etc/profile
running_user=root
APP_HOME=
APP_JAR=
APP_MAINCLASS=
JAVA_OPTS=
JAVA_HOME=/usr/java/jdk1.7.0_75/bin
psid=0
```

第二段 checkpid()函数用于检测 java 进程的进程号。使用 jps -l|grep $APP_MAINCLASS 获取 java 进程的状态，如果改命令返回字串长度大于 0，则使用 awk'{print $1}'将进程号赋值给 psid，否则 psid 为 0。

awk 命令的格式 awk [-F field-separator] 'commands' input-files 它的工作流程是读取有\n 换行符的一条记录，按指定的域分隔符划分域，$1 代表第一个域。

```
checkpid()
{
javaps='jps -l|grep $APP_MAINCLASS'

if[-n "$javaps"];then
 psid='echo $javaps|awk '{ print $1}''
else
  psid=0
fi
echo $psid

}
```

第三段的 start()函数，用于启动 java 进程。当检测 psid 不为 0 时，给出提示。检测 psid 为 0 时，进入 JAVA_HOME 路径，使用 java -cp 命令启动进程，启动完成后，再调用 checkpid 检测一下是否启动成功。

```
start()
{
if [$psid -ne 0]; then
echo "================================================================"
echo "Warning:$APP_MAINCLASS already started!"
echo "================================================================"
else
  echo -n "Starting $APP_MAINCLASS ..."

  cd $JAVA_HOME
  java $JAVA_OPTS -cp $APP_JAR  $APP_MAINCLASS &
  checkpid
  if[$pid -ne 0 ]; then
    echo "(pid=$psid) [ok]"
   else
    echo "[failed]"
  fi
fi
}
```

第四段的 stop()函数用于停止 java 进程。首先使用 checkpid 检测 java 进程的进程号，如果进程号不为 0，则使用 kill -9 杀死它。然后使用根据$?（$?判断上条命令是否执行成功，执行成功为 0）判断一下是否杀死成功并给出提示。再用 checkpid 检测进程号，如果不为 0 继续调用 stop，直到杀死该进程为止。

```
stop()
{
checkpid

if[$psid -ne 0];then
echo -n "Stopping $APP_MAINCLASS...(pid=$psid)"
su -$running_user -c "kill -9 $psid"
 if[$? -eq 0];then
 echo "[OK]"
else
 echo"[failed]"
fi

 checkpid
 if[$psid -ne 0];then
 stop
fi
else
  echo"================================================================"
  echo"warn: $APP_MAINCLASS is not running"
  echo"================================================================"
fi
}
}
```

第五段的 status()函数，根据 psid 值获取进程的状态。

```
status(){
checkpid

if[$psid -ne 0];then
 echo "$APP_MAINCLASS is running!(pid=$psid)"
else
 echo "$APP_MAINCLASS is not running"
fi
}
```

最后一段使用 case 语句，将脚本的入参做了定义。共有 start，stop，restart，status 四个参数。

比如定时重启可以在 crontab 中定义 0 0 * * *　/scriptpath/test.sh　restart

```
status(){
checkpid

if[$psid -ne 0];then
 echo "$APP_MAINCLASS is running!(pid=$psid)"
else
 echo "$APP_MAINCLASS is not running"
fi
}

case $1 in
 'start')
   start
    ;;
  'stop')
   stop
   ;;
   'restart')
    stop
    start
   ;;
   'status')
   status
  ;;
 *)
  echo "Usage:$0 { start|stop|restart|status}"
  exit 1
esac
exit 0


}
```

至此，一个定时重启 JAVA 进程的任务就完成了。在完成整个任务的过程中，参考了网络上的一些文章资料，也请教了有经验的同事，在学习编码过程中交流是很重要的，希望未来大家一起加油共同进步！

从测试开发到 Delivery Manager

◆ 作者：晴 空

写在前面:

2017 年，新的征程，自己不仅身为人父，而职业上的角色也有了比较大的转变，总结下自己从测试开发负责人到 Delivery Manager 转变过程中遇到一些事儿~ 也算是自己的年终总结。

从 0 到 1 是很难很难的，而从 1 到 N 更是不易。一句话“大道至简，知易行难”再合适不过了，希望我遇到的问题以及应对策略对现在或者以后职业生涯中遇到类似问题的同学们有些许参考意义。

--------谨以此文，献给那些热爱软件质量保证工作的亲们!

一、那些转变

1.1、工作内容

作为自动化测试负责人时，我管控的工作任务项是下面这些:

1: 开发自动化测试框架。

2: 设计自动化测试用例。

3: 实施监控自动化执行。

4: 培训扩散自动化技能。

而转变为 Delivery Manager 之后，之前的所有任务项就变成了任务的子集。有同学可能会问什么是 Delivery Mananger? 我个人理解是产品质量负责人，这个角色对版本能否发布上线有这最终的决定权，当然~权利越大，责任越大!

Delivery Manager 负责的工作任务项包括下面这些:

1: 发布流程规范化及流程推进。

2: 分层自动化实施(当然包括测试框架开发，用例设计，测试执行这些)。

3: 测试资源安排协调。

4: 测试执行。

5: 版本质量评估。

6: 招聘质量保证同学。

7: 新人培养。

8: 生产环境故障跟进。

其实，总结起来就一句话："只要可以保证版本质量的事儿，Delivery Manager 都需要去尽力尽心的去做"。

1.2、心态和手段

自动化测试负责人对每个版本中自动化测试的执行结果负责，而 Delivery Manager 操心更多(自动化覆盖的场景之外，手工测试验证的场景也是非常重要的，此外，跨部门沟通在有些团队里不是易事)。

心态上：从局部到全局的变化，要协调的资源多，肯定会出现各种幺蛾子，肯定会有很多变化在预期之外，自己更多的学会了如何权衡全局，如何对部分因素妥协。

手段上：有人的地方就有江湖，有江湖的地方就有恩怨纷争。

虚与委蛇，软硬兼施是必须的；我尝试过一直很强硬，然而，强极易折，很可能会导致团队和个人都很累，到最后疲于应对各种突发事件。道理大家都懂的，只是没经历过，很少人能体会从自身出发强行转变自己的风格的痛苦。在此鸡汤一回吧~ 痛，说明你在成长，勇敢的去汲取知识和各种优秀的姿势，在痛苦中成长吧，多年后的暮然回首，你会感激曾经奋力拼搏的自己!

二：流程规范化推进

2.1、为什么要规范产品交付流程?

优秀的过程不能保证结果，但是，没有优秀的过程，要求产出好的结果是难上加难，或者说几乎是不可能是事情。

就好比学习，对大多数同学来说平时不用功考试临时抱佛脚，能考出好成绩那才叫见鬼。

对一个研发团队来说，如果没有规范化的流程，想要出成果，想要交付优秀的产品 天

方夜谭!!!

个人认为：一个适合团队的，规范性的交付流程是持续交付优秀产品的必要前提条件。

(PS：可能有同学会和我扯敏捷开发，别急，在最佳实践部分，咱们好好聊聊~)

2.2、什么是最佳实践?

首先，个人愚以为：适合自己的，才是最好的；适合自身团队的才叫最佳实践。

其次，我想写写大部分同学说起敏捷开发所想到的，嗯！如下观点很常见：

极限编程

(以为极限就是指工作时长，你 TM 考虑过效率么？个人承认，延长工作时间会在一定程度上赶赶进度，但，当加班成为一种常态，呵呵~ 国内有些巨头强制 996 美其名曰团队文化)

每天只工作 8 小时？还没到你'极限'嘛，工作量严重不饱和，来来，多给你分配点任务;

拥抱变化

(这个需求很简单，怎么实现我不管！需求有变动可以理解，问题是需求变动的频率和变动之后的应对策略，貌似需求变动之后的唯一应对方案就是加班)

产品、策划需求频繁变更，常常自打嘴巴，昨天说好的今天又变卦。

"我的需求有一个很'简单'的改动……怎么要花这么久才搞好？你的代码架构一点都不灵活，根本就不够敏捷嘛！你懂得拥抱变化不？"

迭代式开发

(天真地以为敏捷不讲发布质量，"又不是不能用"不应成为伪敏捷外衣下研发团队的质量底线)

"做完了就赶紧先上线，出问题了就再修复一版更新上去"

持续集成

(没有快速的自动化测试实施的前提下,别跟老子扯持续集成,你那叫断点集成！哦，不对，你那叫集成)

哥天天提交几百行代码，不用编译，不用自测，没有 Test Case，哥就是这么自信。

测试驱动

(测试驱动强调的是从可测性出发编写代码，当然前提也是强大快速的单元测试实施)

“我点了两下，没发现什么明显问题，可以发布出去了”

Sprint 冲刺

(临时抱佛脚，阿弥陀佛~ 代码注释里贴佛祖保佑永无 bug 不无道理)

“版本明天就要上线了，今晚大伙儿来个通宵大作战，努力冲刺一把。”

迭代回顾

(所谓复盘，要么不做，要么说些不痛不痒的话，放点不香不臭的屁！)

“那谁，版本刚上线问题肯定很多，这两天辛苦一下，手机 24 小时开机待命”

结对编程

(这个。。。嘿嘿 还不面向对象编程，至少没有对象还可以 new 一个)

“开玩笑，一份事情两个人轮流干，每人只领一半工资可以不？”

除此之外，我想问问同学们，

你看过几本关于敏捷实施的书？你经历过哪些成功的敏捷团队？

敏捷模式最初的提出是解决什么问题的？

敏捷开发有没有要求说需求文档可以没有？

敏捷开发有没有说压缩测试执行时间？

敏捷开发有没说可以不顾一丁点儿质量 随意发布？

一本相关的书都没完整的看完，就 baidu 下敏捷宣言敏捷模式，然后开始鼓吹，我们要敏捷，敏捷大法好，敏捷大法秒，自从实施了敏捷，腰不疼了，腿不酸了，有精神了!!!

~~~同学，醒醒 ！ 工头喊你写 bug 啦~~~

最后，我们回过头来看下什么是最佳实践，每个团队都出来显摆自家的最佳实践，那是说明每个团队的最佳实践是不同的，当然，优秀团队的最佳实践都是相似的；而披着伪敏捷外衣的渣渣团队的最佳实践各有各的不同！
~~~

2.3、如何把握流程中的关键点？

(这部分和同学们聊聊从需求到发布各个阶段，作为测试负责人应该注意的事项，仅做参考)

2.3.1、需求介入

1.1、产品需求需列示以下内容:

1) 需求提出单位（内部需求/外部需求，外部需求需注明客户全称、行业），用以帮助 PM 了解需求的行业背景和可能的业务逻辑;

2) 客户的联系方式（如有），用以 PM 回访需求提出方，挖掘需求背景;

1.2: 项目 kick off 会议之前项目经理预估项目时间，需要在项目 FBRD(最终用户需求文档)评审阶段给出项目周期时间段预估，PM 在项目 kick off 之前申请好开发，UI，测试资源组成项目组。

1.3: 测试负责人跟进 kick off(项目立项) 之后的项目所有流程，其中当概要设计评审之后需给出一份完整的带测试策略的测试计划。

FBRD 必须包括以下内容: 项目背景，项目目标，项目详细功能及 Demo。

2.3.2、项目计划

1) 统一书写成甘特图，放置在石墨文档上(创建项目名称目录而不是版本号)。

2) 项目计划包含内容: 时间 deadline，开发/测试资源。

3) 计划变更

a) 如发现影响项目时间或任务安排的一定要在计划和进度安排上体现出来。

b) 发生偏差但不影响里程碑点和项目上线时间的情况下，无需调整进度安排文档，但需要做相应的跟进。

c) 生较大偏差，影响到里程碑点或发布计划的情况下，需要调整进度安排文档，由项目经理负责跟进项目上线计划。

4) 计划变更影响到项目里程碑日期的需要邮件通知项目组，抄送 CTO

2.3.3、测试计划

1）测试范围及测试策略是测试计划中很重要的部分

但去除一些无用的部分，如测试退回后工作安排等

2）测试进度安排

如较小的项目直接用很简单的表格显示关键时间点即可，稍大的项目，测试工作量超过 10 人日或者测试人员 3 个及以上，需要进行进度安排

注：小型测试项目定义：工作量<=10 人日，不涉及跨产品线，参与人数 3 人以下(不含 3 人)

3）测试策略

增加可选项：项目中测试的重难点，最耗时的部分的测试策略

4）人力资源

去除非测试的其他角色，项目较小如一个人的时候可以不用填写，角色较多可进行填写

5）环境资源

直接写上负责人及环境绑定即可，公共环境作上标记

6） 环境部署方案

有特殊情况的可以写一下

7） 风险列表

注意要精简，不是越多越好。

2.3.4、测试分析

a) 数据准备

b) 数据库检查要细化到表中对应字段的更新

c) 页面展示数据需要有数据的计算逻辑或对应的数据源

d) 页面布局检查，页面展示，文案。直接附上 demo，对比 demo 即可，不用写步骤。

e) 对于应用的通用功能可以不写用例步骤，常规表单---提交，编辑，删除，查询

f) 应用中开发框架中通用及固定的内容，如分页，置顶，置底，分页等，使用原有的实现方式，可不写步骤，复用原有的测试分析即可

2.3.5、功能预演&冒烟测试

1）冒烟测试通过率要求达到 100%，即每个冒烟功能点都要通过;

2）隐藏冒烟测试点:QA 从分支流程中随机选取 20%的用例进行测试且事先不告知开发。隐藏冒烟点也必须 100%通过。

3）小需求的测试分析和用例执行时间大于等于 5 天，需要进行冒烟;

4）冒烟测试不可裁减。如遇到特殊情况，必须得到测试 M 的书面批准。

注意事项

1) 冒烟点尽量覆盖项目的主干功能，冒烟点需要明确指出通过的条件;

2) 小需求的测试分析和用例执行时间大于等于 5 天，需要进行冒烟;

3) 冒烟点在 TC 评审时，与项目组确认达成一致意见;

4) 冒烟测试执行时间建议不超过 2 个小时;

5) 冒烟测试执行的当天(如冒烟测试完成时间过晚，也可以第二天发送)，以邮件的形式发送冒烟测试结果邮件，列举具体通过情况，包括开发首次提交代码时间、冒烟开始时间、冒烟结束时间、冒烟点和冒烟点通过情况。

6) 若冒烟测试不通过，测试负责人先线下和 PM 沟通，分析原因，确定下次提交时间，再发出测试退回通知邮件给项目组成员、项目组成员的经理。开发修改代码后，重新提交冒烟测试。

7) 根据以往团队成员的质量情况，灵活选择冒烟点的个数，冒烟点可以不局限于主干流程。

2.3.6、测试执行

1) 测试执行过程中，确认执行过程所遇到的问题，及时记录缺陷的情况，注明用例的执行状态;

2) 测试执行过程中，对于自己无法把握的问题，需及时告知项目经理;

3) 测试执行过程中，若发现用例需要修改或补充的，应及时修改测试用例;

4) 测试执行过程中，若得知有需求变更，应主动跟进变更情况;

5) 测试执行过程中，需及时跟进 Bug 修复情况，对已修复的 Bug 进行验证，同时检验 Bug 处理是否符合 Bug 处理原则;

6) 测试负责人跟进项目 TC 执行进度和 BUG 处理情况

7) 测试中的异常以及对项目的风险评估第一时间反馈并记录在案，在编写质量报告/个人周报时发送出来

8) 测试过程中，提醒测试成员将任何有价值的东西都要记录下来，形成测试手记

9) 更新环境后如主干功能不可用，与项目经理沟通后执行测试退回

2.3.7、预发布验收

1) 项目发布计划评审前，测试负责人完成预发布发布验证计划，内容包括

（1） 账号、支付卡

（2） 预发布发布环境其他需求，如需开发协助查询日志，运行定时器

（3） 预发布验证功能点、验证人员安排及时间

（4） 正式环境验证功能点、验证人员安排及时间

2) 在发布评审中，与项目组成员对预发布发布验证点和范围达成一致

3) 预发布验证遇到问题，及时找开发定位问题;

4) 预发布验证完毕，及时响应流程;

5) 预发布使用账号密码不可告知非测试人员使用;

6) 预发布正式环境不可做对线上用户造成影响的操作，如：给线上用户发反馈。

2.3.8、发布后的项目复盘

项目发布后，及时做项目分享。大型项目或有亮点有问题的项目必须做项目分享。项目分享文档形式不限，包含项目业务范围、项目过程碰到的经典问题和解决方案/建议、测试方法、经典 BUG 等。分享邮件发送给全体项目组成员。

三：如何把控发布质量使之符合产品发布标准

3.1、版本能否发布上线由谁 or 什么来定?

毫无疑问，版本质量需要有人去评估(Delivery Manager)。

那根据哪些信息来确定能否发布呢？

1: 回归测试场景覆盖率。

2: 回归测试过程中出现的 bug。

3: 接口测试用例是否全部通过。

4: UI 自动化测试通过率。

5: 性能测试(若需要)是否通过。

6: 安全测试是否通过。

能否发布不是个人意志而决定，而是回归测试的结果来评估是否满足版本发布质量，简而言之：以测试结果评估。

3.2、产品交付过程中的那些 checklist

(这部分的 checklist 是 Delivery Manager 的工作，去检查测试负责人是否能很好地把控了过程中的关键点)

3.2.1、测试计划 checklist

检查项	检查标准
测试范围和重点	以项目的角度分析测试范围和重点（包括接口、浏览器、操作系统的兼容性等等）
测试策略	是否考虑可复用 TC（以前有类似项目、公共用例库）
	功能测试区分手工测试和自动化测试，是否需要做性能测试：对于每种测试，都应提供测试说明，并解释其实施和执行的原因
	自动化脚本相关： 1）内审前需要与自动化小组成员确认可实现自动化的功能点并写明;如该项目不实行任何自动化,需写明原因 2）是否考虑自动化脚本新增\调试\执行时间 3）是否考虑本项目会引起的现有自动化脚本的维护时间
	对于每一名测试成员安排一些机动任务，一旦项目中出现异常情况(如测试退回、提交测试延迟、项目暂停等)，可及时调整工作内容（如写自动化脚本、做总结等）；也可回退到上一个版本继续测试
	列明不能测试的功能点，并说明是否有替代的解决方案（比如开发自测、开发提供测试工具、服务人员协助测试等）
	对于多浏览器/分辨率测试，是否写明不同测试执行阶段的浏览器测试方案
	是否可以利用工具或编写代码来进行验证

	预发布测试计划（可选）：人员安排，检查点，检查方案，是否包含美国\杭州\香港(针对国际站,SE)
测试进度	需求了解时间/测试分析时间/测试步骤编写时间/测试执行时间的工作量估算是否合理
	与开发工作量比例是否合理，如果测试工作量超过开发工作量的一半以上，需要说明原因
	需包含的时间点：需求了解、测试用例设计、TC 评审、冒烟测试、功能测试、回归测试、预发布测试、发布
测试资源	测试的资源是否确认，包括人和机器
	明确部署时间，人员分配，环境更新频率时间
风险列表	需要考虑项目中存在的风险，并分析其内容、影响和预防方案，以测试计划编写规范中风险列表的形式体现,见风险列表
project 计划	按功能模块划分,任务分解细化至少到 3 天以内，正常应细化到 1 天,除以下情况:1) 如部分功实无法分割的,可放宽至 3 天以内
	可用的工作量是否按每天 6.5 小时计算，是否预留了 TC 评审会议、项目周会和发布计划评审的时间
	是否考虑安全测试的时间
	是否考虑环境搭建时间（包括功能测试阶段和回归阶段）
	是否考虑验证 bug 的时间（主要用于提醒测试负责人需要考虑这块工作的时间）
	自动化需求分析、脚本编写及执行任务是否在计划上有体现
	性能测试部分在计划上是否有体现
	资源不是百分之百投入的需要在 open project 的详细计划资源中加上百分比
	体现不同浏览器和分辨率的测试任务分配
	回归阶段有无 bug 分析任务（可选）：回归测试阶段介入 bug 分析的工作，但需要统一一个标准:bug 收敛率+bug 等级加权，召集项目组开发，架构做一个 bug 分析
	预发布测试计划（可选）：是否包含美国\杭州\香港(针对国际站,SE)
其他	测试计划需经过产品线负责人或导师 review
	对外承诺的计划必须与产品线负责人确认过并征得主管同意
	功能演示目前 QA 是不要求参与的，但如有特殊情况(涉及环境比较特殊,开发无相关环境无法造数据等)需要 QA 配合，需要提前跟 PM 确认并在计划中体现
	如果为合作性项目，测试负责人在测试计划评审之前需与合作方测试负责人一起确认项目进度时间点

3.2.2、测试分析 checklist

一	业务需求是否覆盖	
1	入口是否完备	对于需求中实现的功能，有多个入口可以到达，是否考虑全面
2	会员类型和权限是否完备	针对不同的业务类型，要测试的会员类型是不同的，会员类型是否考虑全面 是否考虑了权限控制测试

3	数据准备是否完备	1、数据准备是否完整正确 2、数据准备是否覆盖到线上环境的所有情况 3、数据准备是否标识出业务流程上处于什么条件，如:OFFER 处于审核不通过状态 4、数据准备是否写明数据表字段值,如:OFFER.status=TBD 5、是否有选择数据准备的原因，即数据准备的分析 6、对于复杂的数据准备，是否有 SQL 写上(要自己写 SQL，不要用开发的 SQL)
4	主干和分支流程是否完备	1、用例产生前是否有测试分析过程 1）因果图法：罗列每一种条件的不同情况，对条件进行排列组合,形成矩阵，矩阵精炼,去掉不可能存在的情况 2）场景测试法：流程图是否完备 2、最终得到的各种情况是否都组合考虑全面
5	异常流程是否完备	1、根据具体业务，是否检查操作违反正常流程 2、是否考虑数据本身异常时，系统的处理方式 3、是否考虑了并发测试
6	后置流程	1、是否考虑了操作成功或者失败后的页面提示以及展现 2、是否考虑了存储检查：包括日志检查,图片存储检查，数据库 3、是否考虑了后续功能检查:如：邮件、贸易通，短信，查询、接口应用等
7	页面检查是否完备	1、每一个元素的不同情况是否按等价类，边界值划分法有用例覆盖到。(注意特殊字符的显示) 2、页面上的数据值是否进行了正确性检查 3、页面检查是否附上 demo 4、页面跳转是本页面，还是新开页面，页面元素确认修改后，页面是关闭还是保持打开状态 5、是否检查链接正确性，破页，重叠，JS 报错，美观性等
8	查询检查是否完备	1、查询字段是否包括正确数据的不同等价类，边界值的检查（如：最短，最长，不同数据类型） 2、查询字段是否包括非法数据（非法类型、不符合要求的数据、为空，溢出数据等）的检查 3、是否考虑了组合查询 4、是否考虑了精确查询还是模糊查询 5、是否做了查询结果是否正确的检查 6、查询结果是否检查翻页，查询后再翻页 7、查询结果是否检查排序情况 8、查询结果是否考虑 0 结果，多结果，大数据量结果
9	表单输入检查是否完备	1、是否包括正确数据不同等价类，边界值的检查（如最短，最长，不同数据类型） 2、是否包括非法数据（非法类型、不符合要求的数据、为空，溢出数据）的检查 3、正常异常输入是否有对应的提示。 4、是否有表单默认值的检查

10	数据检查是否完备	1、引起数据新增，修改，删除的功能点，是否都检查到数据库层 2、对于统计数据值的检查不止是页面显示和数据库是否一致，是否还包括数据库的值计算正确 3、是否考虑了与其它环境的数据同步 4、是否考虑了大数据量的情况
11	接口检查是否完备	1、数据的传入传出是否接口定义正确，是否正确，包括长度，类型，值 2、每一个字段数据不同等价类是否都检查到（包括最短，最长，边界值，不同数据类型<考虑特殊字符组合>,是否必填,枚举值等）。 3、接口链接失败时调用到该接口的功能是否正常处理
12	脚本检查是否完备	1、是否考虑了脚本的先后执行顺序 2、是否考虑了单条数据，多条数据，大数据量时，脚本的运行情况 3、是否考虑了脚本数据要求之内的数据处理后的相关检查 4、是否考虑了脚本数据要求之外的数据是否也被处理 5、是否关注到脚本日志
13	回归的范围是否完备	1、对于一个功能模块里的一个小功能点做了修改，或者在一个功能模块里增加了一个小功能点，不是只是测试这个小功能点，对于这个功能点需要作回归。 2、对一个页面里的一个元素作修改，需要对这个页面的相关功能元素作回归。 3、对于一个功能点的部分入口做改动，需要本次不需要修改的入口，也要回归验证是否没有被改动到 4、需要检查与需求上的功能点相对立的情况，如：TP 实现某功能，需要检查 free 是否没有实现某功能。
14	测试范围是否有定义	1、测试范围是否大于等于需求范围 2、是否写明不在测试范围内需求点，无法测试的需求点,如:数据迁移是否在测试范围内 3、兼容性测试是否考虑周全:如:要测试哪几个 IE 版本，WINDOWS 版本，贸易通版本 4、易用性测试是否考虑周全 5、url 拼装测试是否考虑周全 6、本地化测试是否考虑周全
二	用例结构书写是否合理	
1	用例是否具执行性	1、异常检查，是否可执行，如网络传输异常，接口调用失效，需要在用例里写出怎样才能达到这种异常情况， 2、是否在在用例里出现待确认的问题 3、是否直接拷贝 UC
2	用例是否具可读性	1、用例语言是否精准，是否会存在歧义：以下字眼不要出现在用例里：保持原有逻辑，正确的数据；错误的数据；如果；或者；系统不处理；按规则匹配；"最多 XX 个，不足 XX 个有多少显示多少"，“检查 URl 是否正确” 2、测试分析是否按照统一的模板格式写。 3、测试分析是否按照书写要求写。 4、是否附上参考文档：如 UC；环境说明：如 host 绑定，参数修改
三	其它原因	

3.2.3、测试用例 checklist

一	业务需求是否覆盖	
1	数据准备是否完备	1、步骤中是否有数据准备 2、多种情况的数据准备对应一批相同的步骤，要分开来，按一种情况的数据准备后续跟一批对应的步骤来编写 3、数据准备是否完整正确 4、数据准备是否标识出业务流程上处于什么条件,如:OFFER 处于审核不通过状态 5、数据准备是否写明数据表字段值,如:OFFER.status=TBD 6、对于复杂的数据准备，是否有 SQL 写上(要自己写 SQL，不要用开发的 SQL)
2	入口是否完备	1、步骤中是否写明入口 2、对于步骤中要实现的功能，有多个入口可以到达，是否考虑全面 3、多个入口对应一批相同的步骤，要分开来，按一个入口后续跟一批对应的步骤来编写
3	会员类型是否完备	1、针对本次步骤的会员类型是否考虑全面 2、多种会员类型对应一批相同的步骤，要分开来，按一种类型的会员后续跟一批对应的步骤来编写
二	步骤结构书写是否合理	
1	步骤的 description	1、操作描述清晰，如：在什么页面，点击什么链接或按钮；页面入口、链接、按钮名称都要写清楚 2、操作和结果是一一对应的 3、操作中不要包含结果的检查
2	步骤的 expected result	1、结果不能为空(数据准备和入口可以没有结果) 2、结果中只能包含结果，不能有步骤 3、一个结果有多个检查点时，检查点完整
3	用例模版	1、模版用例是否已经调用，且调用正确 2、模版用例中的参数是否都已经实例化 3、模版用例不能调用模版用例
4	步骤整体结构	1、从上至下是否按照执行的顺序写步骤的，连贯性是否强 2、步骤是否有可执行性，比如：build 操作，后台审核，都要写到步骤里去
三	其它原因	

3.2.4、需求变更跟进

若出现需求变更，主动跟进变更结果，将变更结果第一时间分发测试人员；并要求需分发送变更邮件，修改变更列表/需求文档，根据最新变更估计对测试的影响，及时更新测试计划。

执行过程中，若得知有变更，应主动跟进变更情况，尽早将变更结果告知测试人员；要求需分发送需求变更申请邮件，通知到项目组全体成员，并根据实际情况修改测试计划。

3.2.5、如何对测试执行质量做到“心里有谱”？

项目质量报告发送的频率(最好每天发送姓名质量报告)和包含的信息。

质量报告形式不限，但必须具备内容说明中 5 个要点（进度说明、质量状况、项目中遇到的问题和重大遗留问题、质量风险、deffered bug 说明）

质量报告中的风险点和严重问题，用红色粗体字标记。

测试进度说明：进度汇报要详细，要说明原计划，实际情况，产生的结果或影响

质量状况：内容包括 Bug 统计情况、Reopen bug 数量、一个 bug 引发多个 Bug 的数量。

项目中遇到的问题或大遗留问题，内容包括问题描述，解决人，预计解决时间及解决方案。在书写质量报告前必须跟项目经理进行充分的沟通，明确表达自己对问题的见解，自己对解决这个问题的意愿或要求，并协商出可能的解决行动方案

质量风险评估，尽量使用表格的方式，描述风险和解决方案。

项目发布后的跟进:

项目发布后有 deferred 的 bug，在发布当天的质量报告中，需要在质量报告中说明，并附上相应的附件。

四、团队管理

4.1、你真的适合做 team leader 吗？

什么是管理？

之前，我以为管理是赋权和督促；后来被 CTO 指正。刷新了管理观念。

管理的本质是责任，还是那句话：权力越大，责任越大。

说下我现在对管理的理解吧

1：责任，责任心若的同学不适合做管理角色，为什么？自己去想。

2：团队赋能，这块儿分为授权和激励，简单点儿说要带给团队满满的正能量。

3：监督，只有激励是不行的，有些同学会蹬鼻子上脸，更严重的情况是没有团队荣辱观。

4: 引导，不管是技术还是其他方面的成长，当你遇到一个好老师你的成长加速度会变大，职场中也同样的道理。只是好老师可遇不可求，好 leader 也一样。

4.2、你的管理方式是哪种风格?

常见的管理风格: 管家婆式，强硬式驾驭者，和稀泥式，教练式。

管家婆式:

管理事无巨细，亲力亲为，好处是可以让团队的每个成员做到强即时反馈收集团队的信息，坏处是这样做可能会限制团队成员的成长空间(三国时诸葛丞相是典型的管家婆式管理吧)

强硬式驾驭者:

强硬督促的好处是团队执行力会明显提升，不好的地方是有些同学确实不喜欢被催的感觉。

当然，强极必折，把握不好度会把团队成员搞的很疲惫。

和稀泥式:

对中层 leader 来说，这种方式最要不得，跨部门协作时，这种方式会让你的团队话语权重变轻，该争则争，当辩则辩，该出手时就需要你为团队站出来发声。

教练式:

我个人最推崇的管理方式，告诉小伙伴们团队目标，给团队争取该有的资源，给团队成员示范正确的姿势，然后以最终目标为导向反推小伙伴们的成长方式。

五、新人招聘和培养

5.1、招聘不是份儿容易干好的活

团队发展需要持续补充新鲜血液，那么问题来了，如何把控好新补充的小伙伴的质量?

聊聊我自己对招聘的理解吧~

招聘过程中需要考虑的事情:

1: 薪资，第一要务，一分价钱一分货，给不了那么多米，请不来优秀的小伙伴。

2: 技能要求：对 QA 来说，测试基本功得扎实吧 测试用例设计，研发流程关键点把控，SQL，脚本开发能力。现在终于理解为什么好多公司要求做一份面试题了，减少招聘对团队的耗能啊，如果面试题都做不好或者不愿意做，那还有个 J8 可聊的

3: 沟通能力，这点在面试过程中可能不会发现有什么大问题，个人建议是试用期 3 个月内不定时考察。

4: 学习能力：跳槽到一个新环境，技术和业务上很可能会有大大的不同，能否在短短的 3 个月时间内熟悉业务掌握团队 QA 技能是必须的考察点，如果不行，趁早各回各家各找各妈。

5: 新同学入职后的那些事儿：新同学对团队的第一印象除了面试那天的感觉外，入职第一天是非常重要的，一定要热情欢迎新同学，让新同学感觉自己被团队需要。

5.2、新人培养

5.2.1、成长计划如何做?

成长，是一个至关重要的环节，不管对谁。

对不同的同学指定不同的成长计划，提出更高的要求，让其对团队贡献最大化是作为一个 leader 必不可少的课程。

对 QA 同学来说，个人建议可以从以下项中选择:

1: 测试用例设计能力。

2: 脚本开发能力。

3: 业务熟练程度。

4: 沟通表达能力。

5: 文档能力。

6: 总结能力。

7: 学习能力。

有些是软技能不好评估，但是可以以一个季度或者半年的维度来考核。

考核以最终结果为最重要的依据。

5.2.2、怎样有效的去评估成长？

前面也说到这个点儿了，成长是一个过程，需要时间，我这边是以一个月为维度来考核各位小伙伴，就是这个月中交付过程中关键点把握，自己跟进的任务上线后是否有故障，自己制定的学习计划是否完成来评估。

完成评估后我会告诉每个小伙伴他们做的优秀的地方和他们的不足。

六、绩效考核

6.1、为什么要有绩效？

无规矩不成方圆，团队中只有激励是不行的，一定要做到赏罚分明，这些才可能做到令行即达。另外，在大多数团队里，绩效关乎到加薪升级等等，所以绩效推行的重要程度不言而喻。

6.2、研发团队中哪些“然并卵”的 KPI

人人认为以下 KPI 考核项是过时的或者是“然并卵”的

1.执行的测试用例的数量

2.Bug 数

3.通过率百分比

4.单元测试代码覆盖率

5.自动化测试百分比

因为这些因素是保证发布质量的必要非充分条件，而作为 leader，应该从最终结果去反推影响因素，比如一段时间内版本发布质量非常高，上线后几乎无故障出现，这时候就应该反推发布过程中小伙伴们做了什么，做了哪些，至于怎么做的需要留给各个成员自己的成长。

6.3、推行 KPI 还是 OKR

什么是 KPI 关键绩效指标？

KPI（key performance indicator）意即关键绩效指标，是指企业宏观战略目标决策经过层层分解产生的可操作性的战术目标，是宏观战略决策执行效果的监测指针。是用来衡量某职位任职者工作绩效的具体量化指标，是对任职者工作任务完成效果最直接、客

观的衡量依据。通常情况下，KPI 主要来源于两个方面，一方面来源于企业的战略目标，另一方面来源于部门和岗位的职责。KPI 的主要目的是明确引导任职者将主要精力集中在对职位贡献最有成效的职责上去，并通过努力及时采取提高绩效水平的改进措施，因此它是最能影响企业价值创造的关键驱动因素。

KPI 是衡量企业战略实施效果的关键指标，其目的是建立一种机制，将企业战略转化为内部过程和活动，以不断增强企业的核心竞争力和持续地取得高效益。使考核体系不仅成为激励约束手段，更成为战略实施工具。

KPI 的精髓，或者说是对绩效管理的最大贡献，是指出企业业绩指标的设置必须与企业的战略挂钩，其“关键”两字的含义即是指在某一阶段一个企业战略上要解决的最主要的问题。例如处于超常增长状态的企业，业务迅速增长带来企业的组织结构迅速膨胀、员工队伍极力扩充、管理及技能短缺，流程及规范不健全成为制约企业有效应对高增长的主要问题。解决这些问题便成为该阶段对企业具有战略意义的关键所在，绩效管理体系则相应地必须针对这些问题的解决设计管理指标。

什么是 OKR?

OKR 全称是 Objectives and Key Results 即目标与关键成果法，OKR 是一套定义和跟踪目标及其完成情况的管理工具和方法: 1999 年 Intel 公司发明了这种方法,后来被 John Doerr 推广到 Intel 和 Oracle Google,LinkedIn 等逐步流传起来，现在广泛应用于 IT、风险投资、游戏、创意等以项目为主要经营单位的大小企业。

OKR 的指导原则:

1、OKRs 要是可量化的（时间&数量），比如不能说“使 gmail 达到成功”而是“在 9 月上线 gmail 并在 11 月有 100 万用户”

2、目标要是有野心的，有一些挑战的，有些让你不舒服的。一般来说，1 为总分的评分，达到 0.6-0.7 是较好的了，这样你才会不断为你的目标而奋斗，而不会出现期限不到就完成目标的情况。

3、每个人的 OKRs 在全公司都是公开透明的。比如每个人的介绍页里面就放着他们的 OKRs 的记录，包括内容和评分

两个不同:

1、O 和 KR 的不同：O 要是有挑战性的，如果是板上钉钉的事情就是不够的；KRs 能很好的支持 O 的完成，是要明显可量化的，便于评分的。

2、个人、组、公司 OKRs 的不同：个人 OKRs 是你个人展现你将会做什么；组的 OKRs 不是个人打包，是组优先做的事情；公司 OKRs 是高层对整个公司的展望。

对技术团队来说，两种考核原则孰强孰弱，相信各位一目了然。

七、写给新入管理坑的同学

7.1、空降兵的注意事项

1: 刚入团队一定不要大放厥词，要先了解团队现状。

2: 先和部门内同学们打好关系，不要一来就特立独行表现得特别另类(这无关性格，只关乎教养或者素养)

3: 强推是最下下策，最好的方式是先和小伙伴们沟通了解他们的痛点，再争取大 boss 们的支持，这样一来流程规范化的推动阻力会小很多。

7.2、不要以为 IT 界不需要“套路”

很多外界同学以为 IT 同学们很闷骚，其实我想说闷骚的只是一部分，退一步讲，他们闷是因为和你没话题聊。

日常时候一起吃饭打游戏这些大家都会，部门内部还可以搞搞团建。

成立兴趣小组(技术，游戏，户外运动，玩乐队，吃货趴)是很不错的形式。

其实程序猿大多数都是很单纯的，天天面对这机器想不单纯都难！

但是这并不表示和他们沟通相处不需要套路，如果你真的这么认为 那就 too young，too simple 了

7.3、技术&管理两手都要硬

最后，我个人表示还没见过不懂技术并可以很好地带研发团队的 leader!

不懂技术，你没办法有效地和研发团队小伙伴们沟通，更难真正地融入研发团队。

不懂管理，你会被一些刺头和事儿妈搞得头晕脑胀，虽然你可以果断地开掉他们。

以一句戏谑性的结尾吧“不懂代码的 QA 不是好产品!!!”

软件测试行业是一座围城

◆作者：非比君

摘要:
软件测试越来越热门，一方面很多测试从业者进入这个行业。另一方面，一些在职测试人员对测试工作感到迷茫、乏味。软件测试行业有自己的优势，也存在不尽如人意的地方，它就像一座围城，城外的人踊跃想进入，城内的人乏力地想走出去。

近年来，我国的软件测试人才缺口越来越大。随着互联网+、移动互联网、物联网、大数据等新兴 IT 产业的迅猛发展，企业用人需求连年上升。软件测试越来越热门，有些人毕业后直接从事软件测试工作，也有人通过报班培训、开发转行测试、自学网络课程和测试书籍等方式涌入这个行业。

即便如此，软件测试行业就像一座围城，城外的人想进来，城内的人又想出去。

在测试行业待久一些，在相同部门一直测试同一个项目内容，该项目可能迭代版本已经有十几版甚至几十版，却依然要求测试人员去执行测试。这让测试人员变得没有激情。

在整个项目过程中，开发人员有一个优势，就是他们的工作产物是每个人都真正关心的。开发人员编写代码，发布能为公司赚钱的应用。代码是项目过程中产生的最重要的文档。至于测试人员，当项目上线的时候，其实没有人真正想去了解测试到底做了什么。

如果产品深受人们喜爱，大家就会认为这是测试人员理所应当做的；如果产品很糟糕，大家就会质疑测试人员的工作。大家的关注点始终集中在不断增长的代码库中。软件测试工程师的地位还远远不及开发人员，虽然近年来软件测试已经越来越被重视，但是依然会存在一些问题。比如，公司层对于软件测试的概念停留在“点点点”上，把软件测试工程师当作“背锅侠”。

再者，软件开发过程中，需求变更太快，计划永远赶不上变化。测试人员今天刚写

好的测试用例或者测试脚本，需求变更又得另起炉灶。测试人员加班加点顶着压力赶进度……

为此，许多测试人员跳槽去到另一家公司，改变工作环境。也有测试人员转行当产品人员、销售人员、开发人员等。

软件测试行业是一座围城，有人跳进来，有人跑出去，也有人一直待在里面，打怪兽晋级。对于在测试行业想要不断往前的测试从业者，鄙人与大家分享一些经历和想法。

一、创造更多价值

软件测试人员的职责在于协助开发人员尽可能地在早期发现并提出问题，提高产品在用户体验、安全性、稳定性、功能性等各个方面的质量。优秀的测试人员可以为公司创造更多的价值。

但是一个项目测久了，我们原先对它的兴趣和激情会慢慢地消失。对于一个稳定的版本，即或有新功能增加，也要求测试人员在原先的基础上进行一轮又一轮的回归测试。面对每天几乎一样的重复流程、重复操作，测试人员会产生思维定势，很难去发现隐藏的 Bug。

该如何去创造更多的测试价值？我们需要通过不断地学习来使自己更有价值：学习需求知识、学习软件基础、学习测试工具、学习一切可学习的内容。测试人员与其花时间抱怨测试工作中的难处，不如多学习一些相关的测试内容。

比如，你从事 Web 端的功能测试。你除了需求分析，编写测试计划、测试用例，执行用例，提交 Bug，验证 Bug 等必要的工作，你还可以学习 Web 端的 HTTP 协议、学习自动化测试内容、学习和产品相关的行业内容……测试人员需要学习很多东西，从而提升自己，在测试过程中提高产品的质量，为公司创造更多的价值。

二、扎实测试基础

鄙人在职场中遇到一些新人，她们刚刚从事测试工作不久，对于每天的功能测试感到不满足，嚷着要学习自动化测试、性能测试。新人有学习的心愿和目标，是很好的事情。但是如果没有做好本职工作，仅仅认为自动化测试或者性能测试比手工的“点点点”更高级、更热门，这是一种很危险的想法。

盖一幢高楼大厦，首先得打好扎实的地基，从事软件测试工作也是这样。地基若不

牢固，任凭你盖多高的楼层，也是一幢危楼，时刻有坍塌的风险。软件测试工程师，不管是进行功能测试、接口测试、性能测试、自动化测试等，首先都得以“测试”这个地基为准，不断培养测试思维，扎实测试基础。

当然，扎实测试基础，不是说让测试人员每一天都手工操作重复的测试工作，操作一个月、一年、几年甚至更久，而是告诉测试人员不要好高骛远，应该关注当下，把该尽的本职工作、该学习的测试功课，一步一步地完成好。

三、提高代码编写能力

从一些招聘网站发布的测试工程师招聘内容中，我们会了解到，企业对于软件测试工程师要懂开发语言、有编程经验的情况，变得越来越普遍。在今后，测试人员不得不提高代码编写能力，从而提高自己的竞争力，尽管写代码比纯粹的“点点点”测试来得不容易一些。

人若有心去学，肯定是能学会的。

在《Google 软件测试之道》书中讲到，Google 将软件测试工程师分为软件测试开发工程师（SET）和测试工程师（TE）。从某方面来说，SET 就是开发，他们以测试的角度进行开发，直接负责很多的功能特性，如可测试性、可靠性、可调试性，并且开发测试工具给项目团队使用。而 TE 是真正的产品专家、质量顾问和风险分析师，他们把用户放在第一位来思考，代表用户的利益。有时为了更好地测试，TE 也自己编写测试脚本去执行测试。

会编写简单的脚本，这对测试人员来讲是非常有必要的。

每个行业都是一座围城，都有各自的特色，有优势也有劣势。我们应该正确地对待测试行业，做一行，爱一行~!

参考文献:

[1] James A. Whittaker， Jason Arbon， Jeff Carollo. Google 软件测试之道[M]. 美国:人民邮电出版社，2013.
[2] 51Testingt. 2016 软件测试现状调查报告[R].

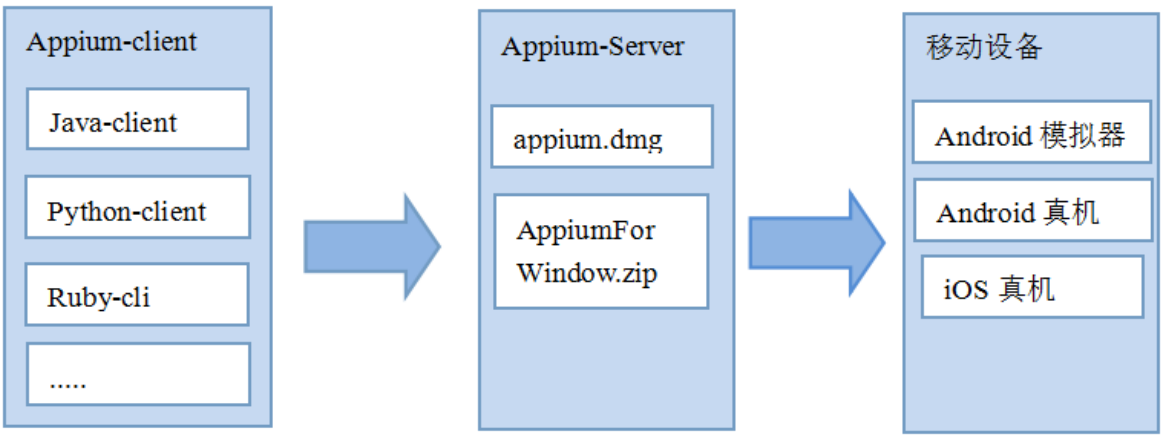
Eclipse+Appium+夜神虚拟机环境调试

◆ 作者：桃 子

最近在学习 Python，发现教程几乎都是连接真机实测，但是目前手里没有可测试的真机，所以想到用虚拟机练习。网上关于这部分的内容真的是非常少，好不容易找到一个遇到的问题也是卡了很久才解决，所以总结一下测试过程和遇到的问题，重新回顾一下。这部分环境搭建确实挺复杂，会遇到各种各样的问题，大家需要有点小强精神哦！

一、Python 自动化实现原理

首先先来了解一下 Appium 工作原理，以及如何进行交互，方便我们了解自己需要搭建什么样的环境



如上面图所示，appium 自动化环境包括三个部分：客户端，服务端，移动设备，那么这 3 部分是如何工作的呢？

1) 客户端编写好脚本后，运行代码，这里的客户端支持各种语言（python，java 等），通过 webdriver 协议调用 appium 服务器

2) 服务端首先建立一个会话，通过 4724 端口和移动设备通信

3) 移动端 bootstrap.jar 接受到请求后发送给 Uiaotomator

4) Uiaotomator 执行脚本命令，进行自动化测试

二、搭建环境所需工具安装

通过上面的过程，我们了解到只要从这 3 端中挑选出适合自己的工具就可以了

这里给大家介绍一个很好的搭建网址，供大家参考，介绍的非常详细，相信比我写出来的会好很多：http://www.testclass.net/appium/

这里附上我自己下载的安装包路径，环境是 win7 64 位：https://pan.baidu.com/s/1hsxT9FQ

序号	依赖环境	安装	备注
1	Jdk Eclipse	安装包安装，注意 eclipse 和 jdk 版本号要一致，否则很容易启动 eclipse 时报错	安装完成后配置环境变量 新增 JAVA_HOME 为 jdk 安装路径 C:\Program Files\Java\jdk1.8.0_111 Path 中增加 %JAVA_HOME%\bin;%JAVA_HOME%\jre\bin CLASSPATH 增加值 为：;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;
2	Androidsdk	解压即可	配置环境变量，同上 Path 中增加 sdk 的 tools 和 platform-tools 路径
4	Appiumdesktop	安装包安装	下载后直接安装就可以了
5	Appium-Python-Client	pip install Appium-Python-Client	进入命令行直接安装
6	夜神模拟器	安装包安装	下载后直接安装就可以了
7	Node.js	安装包安装	在 Windows 环境下，打开命令提示符，然后输入 node –v 可查看安装版本号

三、环境调试

在这一部分，我将按照实际过程的步骤进行讲解，大家按照我这个过程一步一步来，分别为测试包安装-》虚拟机连接电脑-》启动 appiumdesktop-》建立回话-》运行脚本-》脚本运行成功

3.1、模拟器测试安装包安装

我这里安装的测试包为公司正在测试的一款软件‘飞凡’，将安装包直接拖拽到 cmd 后 enter，可以看到模拟器安装包已经开始安装，安装成功如下图所示（如果不支持拖拽的话，在 cmd 中输入 adb install app 所在目录，如输入 adb install　D:\chrome 下载 \com.wanda.app.wanhui_422000000.apk 也可以安装成功）

3.2、模拟器 adb 命令连接到电脑

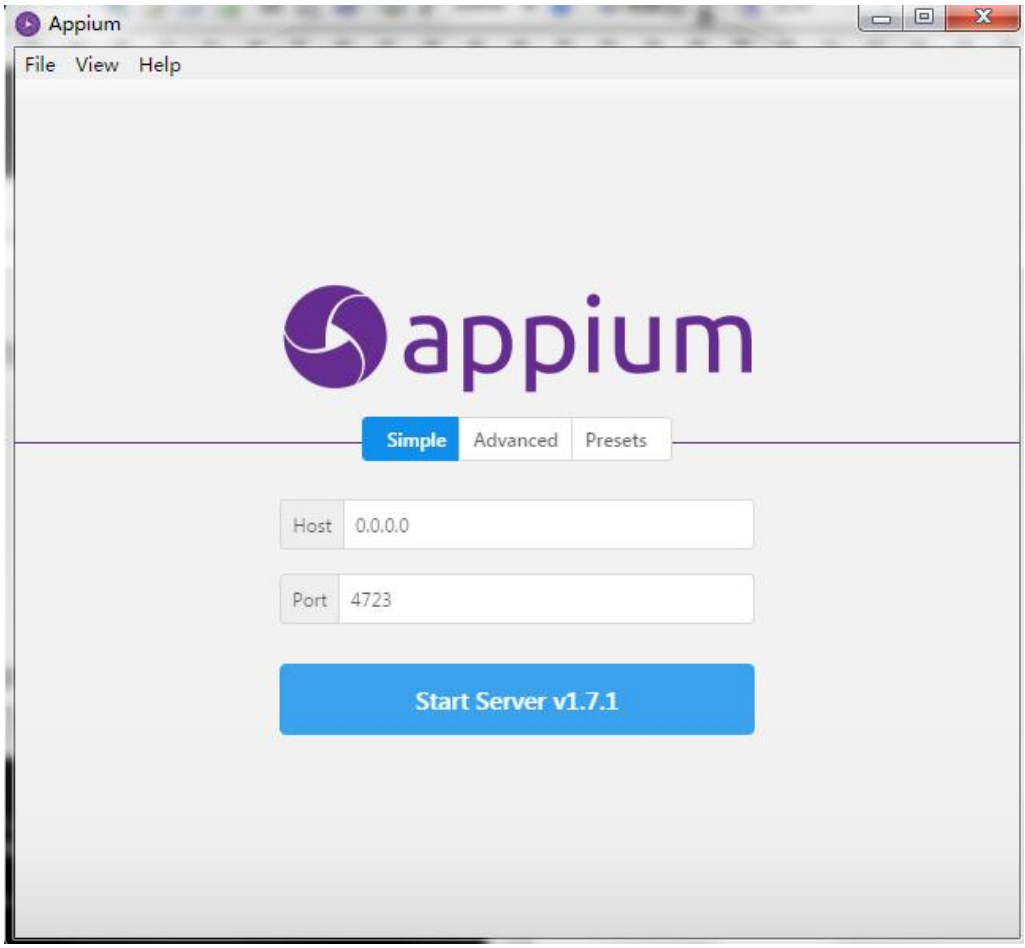
打开夜神（）虚拟机，在 cmd 里输入命令 adb connect 127.0.0.1:62001（如果环境变量没有配置 ANDROID_HOME，cmd 需进入 Android sdk 下的 platform-tools 目录下再运行 adb connect 127.0.0.1:62001），出现 connected to 127.0.0.1:62001 提示说明连接成功。

```
C:\Users\admin>d:

D:\>cd android-sdk

D:\android-sdk>cd platform-tools

D:\android-sdk\platform-tools>adb connect 127.0.0.1:62001
adb server is out of date.  killing...
* daemon started successfully *
connected to 127.0.0.1:62001

D:\android-sdk\platform-tools>
```

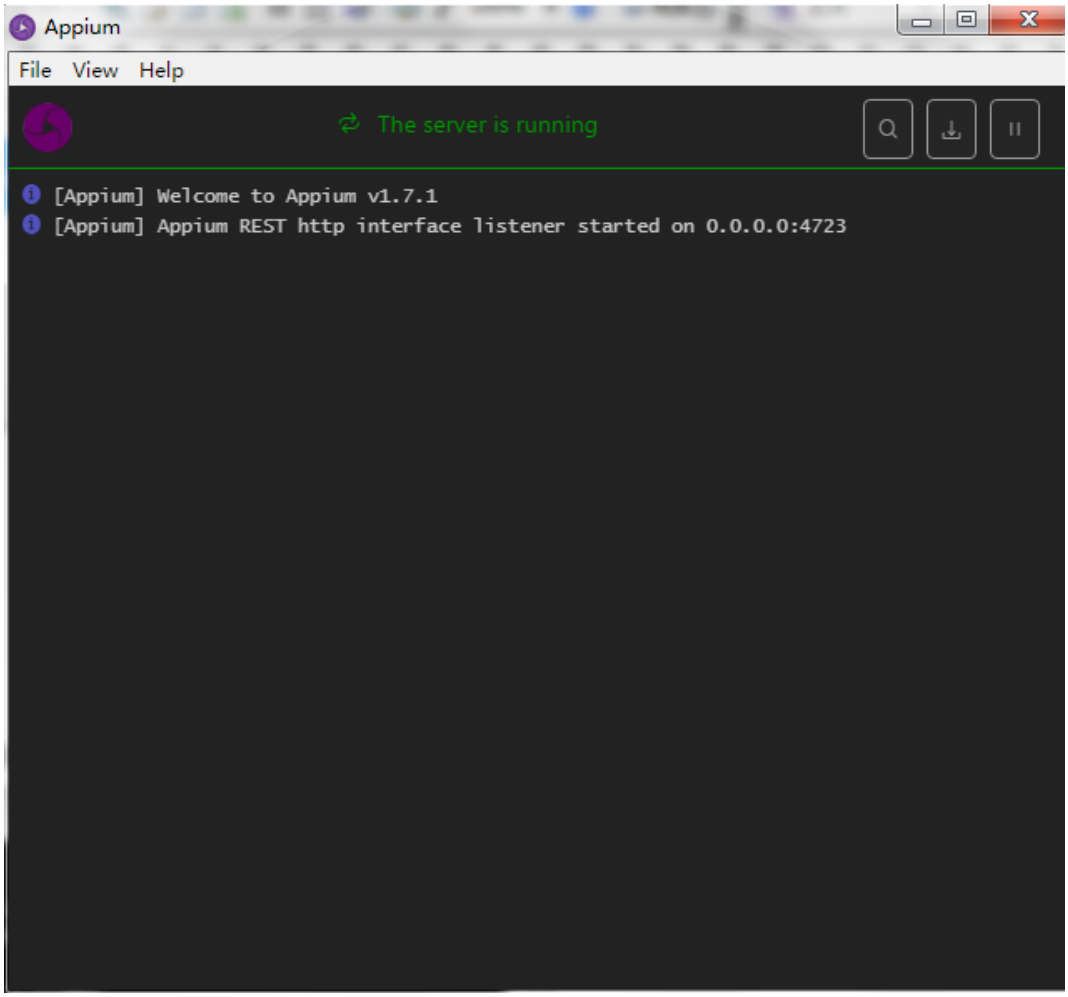
3.3、启动 appiumdesktop，建立 session 回话，确定 capability

1、点击图标 

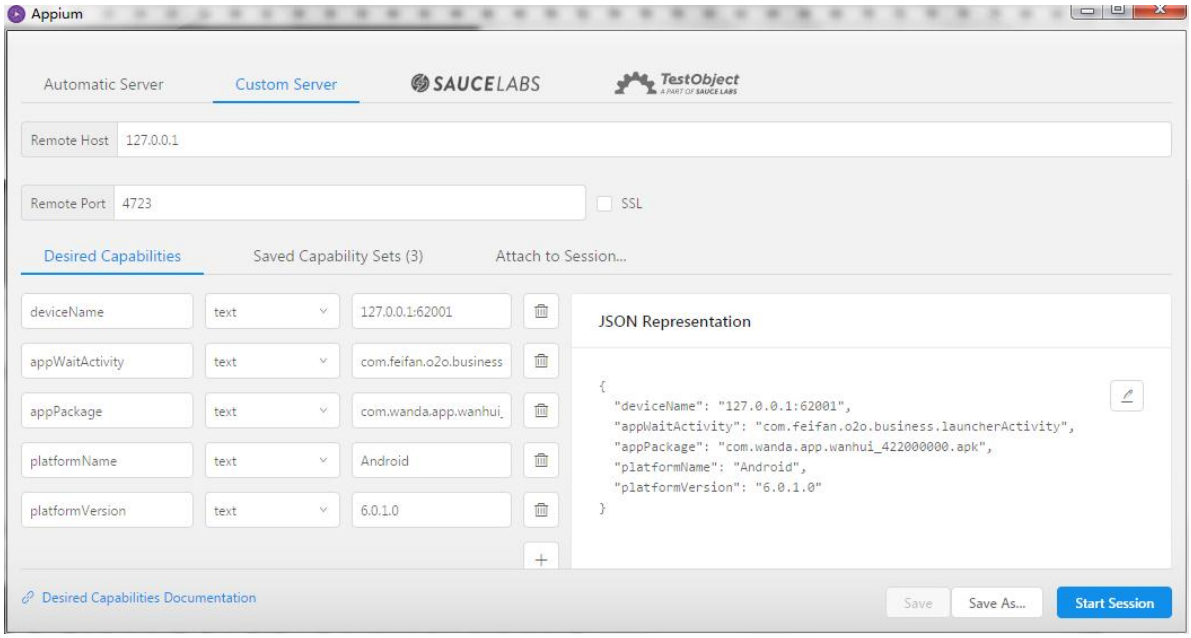
启动 appiumdesktop，启动界面如下图所示



2、点击 start server 按钮，出现 server 运行成功界面



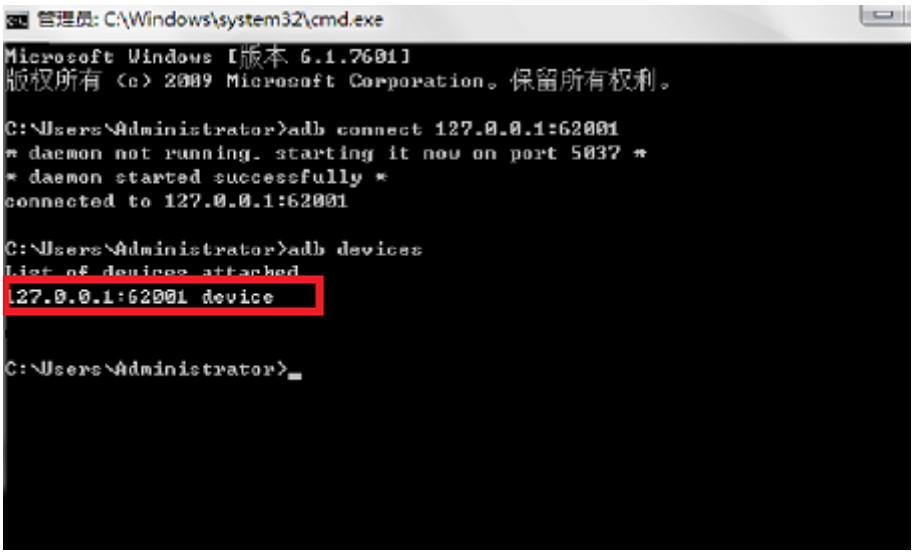
3、点击右上方搜索图标，在这里很关键，有几个需要我们确定的 capbility，下图是我已经确定好的



4、那么如何确定这几个关键因素呢:

- Devicename（设备名称）
- appWaitActivity（测试包的首个等待页）
- appPackage（测试包的首个等待页）
- platformName（测试平台名称）
- platformVersion（测试平台版本号）

Devicename 虚拟机连接电脑成功后，在 cmd 里输入 adb devices，device 前的就是设备名称，因为是虚拟机，所以出现的是 ip 地址端口号形式，如果要是真机的话就不是这样的

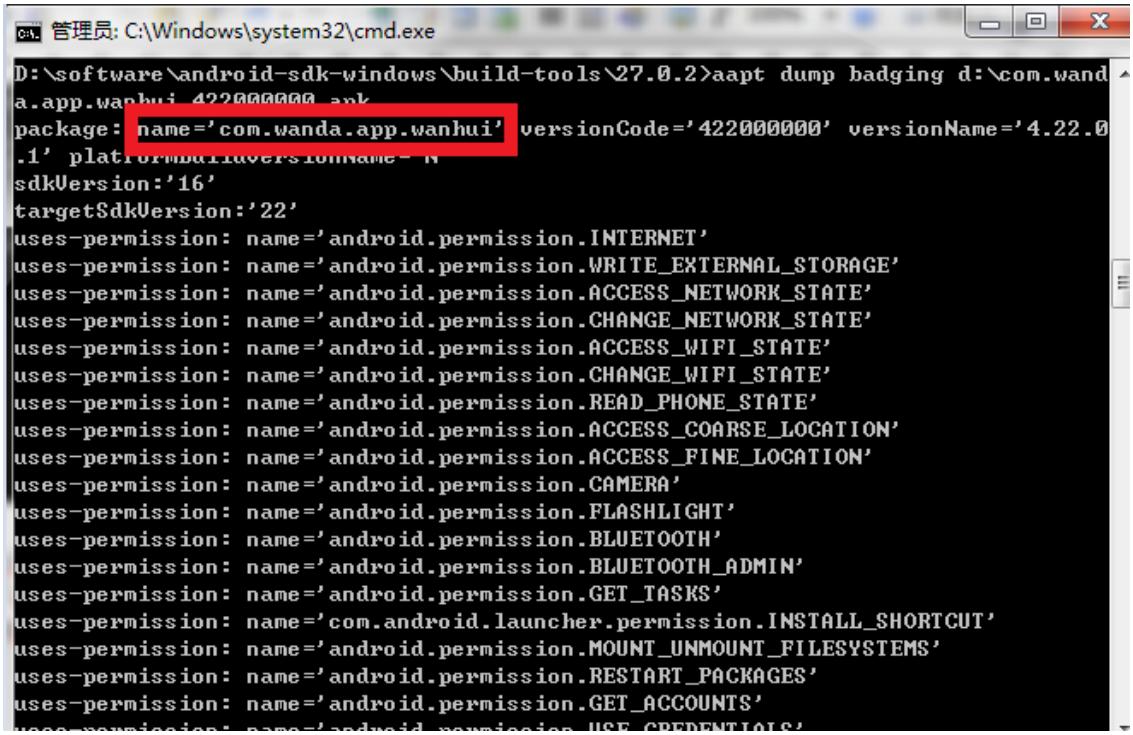


appPackage、appWaitActivity 这两项我试用了很多方法，比如 adb logcat 命令，但我发现查找功能项 appWaitActivity 的时候查找不到，最后推荐大家使用我下面的方法：aapt

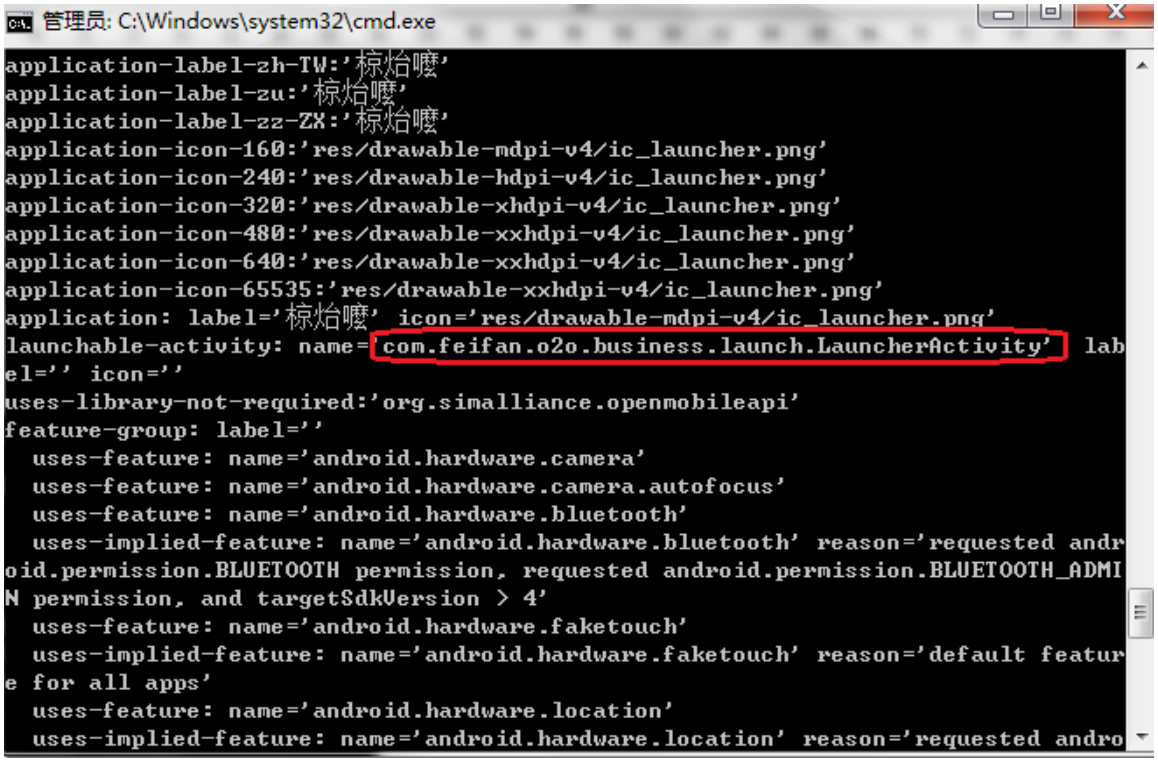
1、cmd 进入 Android SDK 下 aapt 目录：我这里目录是 D:\software\android-sdk-windows\build-tools\27.0.2

2、运行 aapt 工具：aapt dump badging 路径+名称，如 aapt dump badging d:\com.wanda.app.wanhui_422000000.apk，

包名称就是:com.wanda.app.wanhui,在 Android 系统中是判断一个 App 的唯一标识，这里要注意以这里显示的为准，不要直接写上面安装包的名称如 com.wanda.app.wanhui_422000000，否则会报错



页面下拉找到 appWaitActivity 为：com.feifan.o2o.business.launch.LauncherActivity



3、platformName 就是使用哪种移动平台，我这里就填 Android

4、platformVersion 是测试平台版本号：虚拟机可以进入设置->关于查看，我这里是 4.4.2

OK，到这里我们把确定好的 capbility 都添加到 appium 中，点击右下方的保存按钮，appium 的虚拟环境设置工作就完成了，接下来我们在 eclipese 里运行脚本

3.4、eclipse 运行脚本，调用 appiumdestop 中 session 回话

这里我把建立项目的过程说一下:

在 eclipse 里运行脚本前需要导入 python 项目，参考配置网址: https://www.cnblogs.com/Bonker/p/3584707.html，如果觉得配置麻烦，用 python 自带的 Idle 运行也可以，简单方便

将下面的代码拷贝 eclipse 模块中，点击运行按钮，如果一切正常，就会看到夜神虚拟机中飞凡 APP 成功启动运行

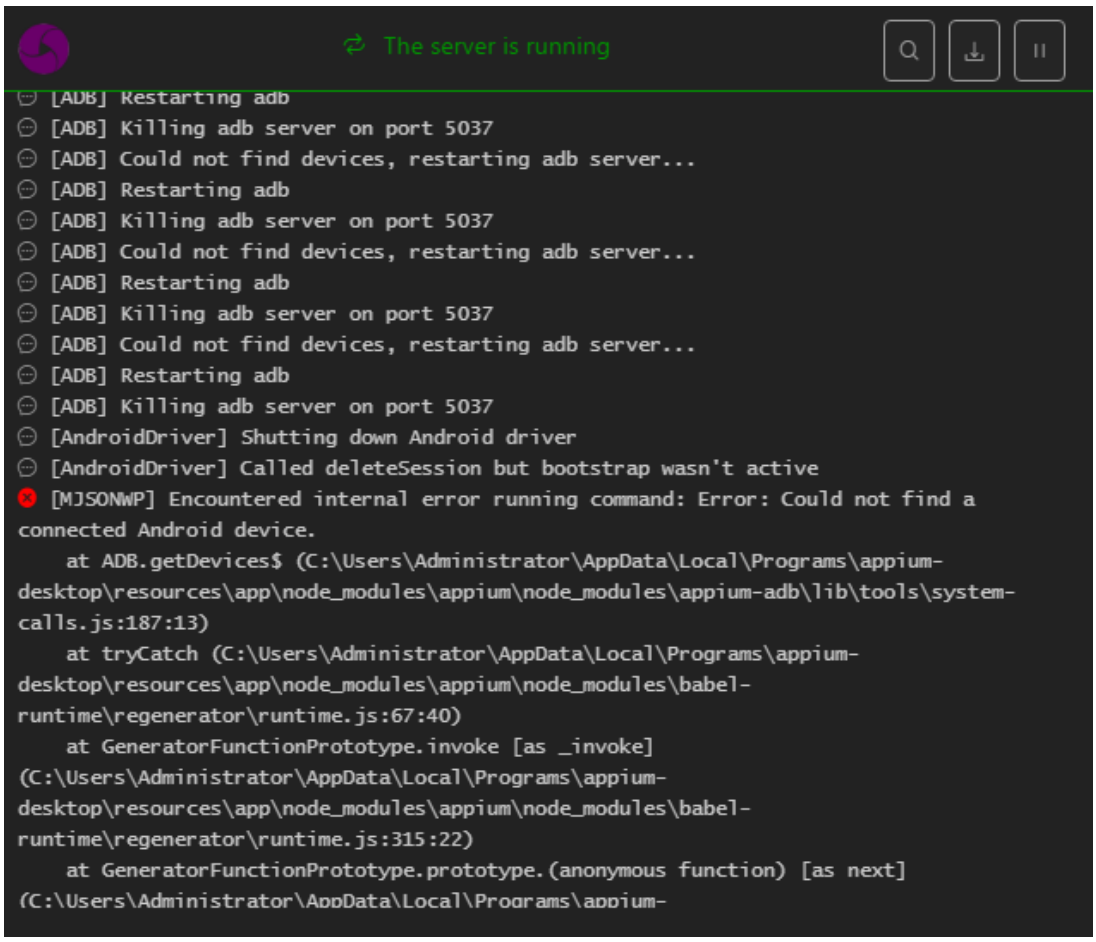
@author: Administrator

```
'''
#coding=utf-8
from appium import webdriver

desired_caps = {}
desired_caps['platformName'] = 'Android'
desired_caps['platformVersion'] = '4.4.2'
desired_caps['deviceName'] = '127.0.0.1:62001'
desired_caps['appPackage'] = 'com.wanda.app.wanhui'
desired_caps['appActivity'] = 'com.feifan.o2o.business.launch.LauncherActivity'

driver = webdriver.Remote('http://localhost:4723/wd/hub', desired_caps)
driver.find_element_by_name('黄商超市八里商贸中心店').click()
driver.quit()
```

四、在运行过程中会遇到的错误

实际上，刚刚接触自动化环境，搭建完成后都会遇到各种错误，下面举例说明一下

1、运行代码后 appium desktop 启动，提示“could not find a connected Android device”，这个问题困扰了我好久，明明知道是没有找到设备，但就是想不明白为什么找不到，这时我就开始各种怀疑人生，难道是我环境配错了，进行了各种试错过程，后来想了想 adb 经常重启，就尝试着在运行代码后，在 cmd 里重新执行 adb connect 127.0.0.1:62001 命令，ok 问题解决，当用真机的时候就不会有这样的问题

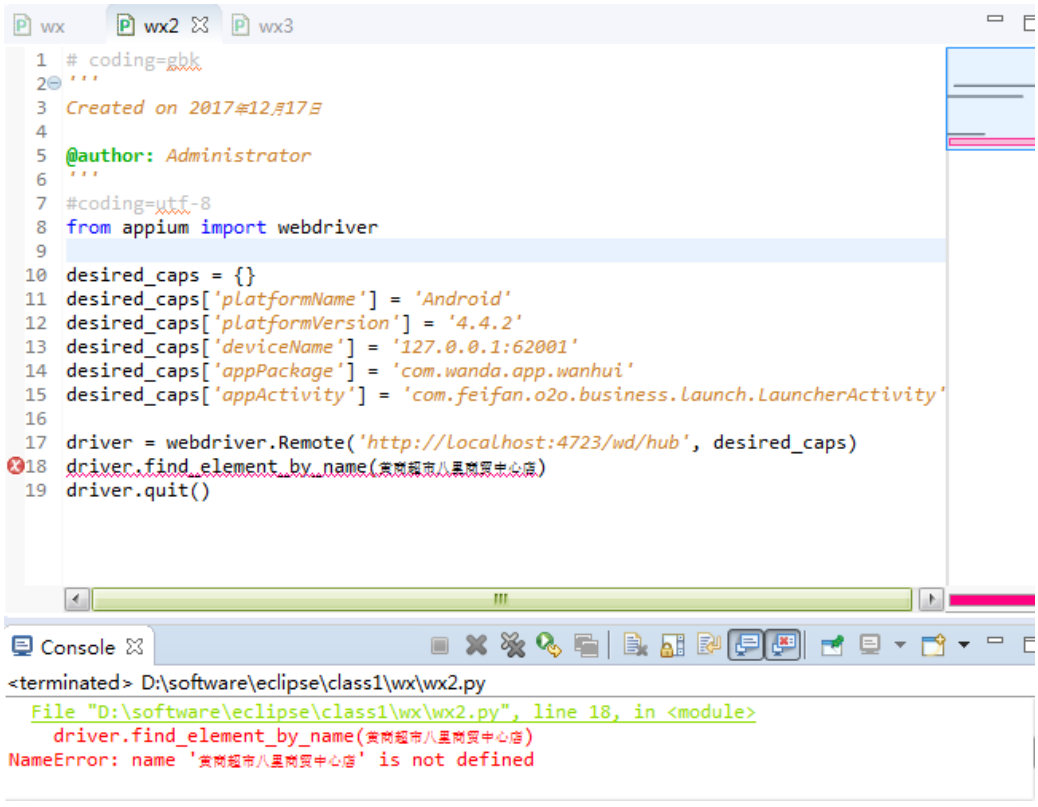


2、问题 1 解决后出现如下提示 error running command: Error: Unable to find an active device or emulator with OS 4.4.4.. The following are available: 127.0.0.1:7555 (4.4.4)

原来 4.4.4.，后面多了一个“.”，这种错误同志们不要尝试犯这种低级错误

3、继续运行一遍代码，这回提示错误“activity used to start app doesn’t exist or cannot be launched！make sure…”，提示的意思是启动 app 时没有找到 activity，可以确定是我们的 launchable activity 写错啦，重新查一遍 launchable activity 具体是什么就好了，回到上文 aapt 命令翻阅

4、eclipse 运行程序，报错“name '黄商超市八里商贸中心店' is not defined”



查看上面的代码发现（）里的内容没有加 ‘’ 导致，添加即可

综上，其实在 eclipse 运行代码后，夜神虚拟机中指定 app 启动成功就说明你的环境已经配置成功了，接下来就可以进行自动化脚本编写啦，在下一篇文章中我会给大家讲解一下具体的功能操作。

职场小白的职业探寻之路

◆作者：徐佳丽

摘要:
我从没有想过自己会从事软件相关的工作。因为作为一名医学院毕业的应届生，从一开始选择在医学院就读时，就有一颗成为医生去治病救人的想法。然而事与愿违，其中的坎坷艰辛和不能成为一名医生的失意让我觉得人生是如此迷茫。直到，我遇到了我从事的第一份工作：软件实施工程师。至此，我找到了人生事业大体的方向，软件从业人员。从最开始的软件实施，到软件开发，到现在以及以后的软件测试，在这一年多的时间里，我在这个行业里摸索探寻，虽然每个岗位的时间都不长也对每一个职业没有更深入的了解，但是， 通过这不长时间的体会了解，我选择了软件测试，作为我以后长期发展的职业。

2016 年 6 月，大三的我开始了为期一年的实习生涯，我选择了去做医疗软件的公司，那时候对于医疗软件没什么概念，只认为是医疗技术与计算机技术的结合。由于缺乏专业的计算机知识，医学知识也只是浅薄的那么一点。于是，我被公司安排的职位是软件实施工程师，是针对医院的挂号、收费等流程的软件。而最开始作为实施人员，我也只是提前将这一套软件用测试账号使用熟练，之后去现场教医生和护士使用，仅此而已，这是一项对任何人都没有技术难度的工作。做了三个星期的软件功能教学之后，我渐渐对于这项工作失去了耐心，觉得这是一份随时都能被任何人替代的工作。于是我开始寻找更加有技术含量的方式去做这份工作，就在这时候，将学校学到的数据库知识用上了（文章末尾会告知我的专业）。我尝试着将数据库表中的数据与前端页面显示的数据进行比较，以核对数据的有效性（数据库的增删改查）。这种方式，跟现在做的测试其实是类似的，因为测试也是要判断数据的有效性问题。就这样，对于这份工作最深入的了解也仅在于有一个数据库的参与。最终，在一个契机下，我被调回了公司，做软件测试。这是我从事的第一份软件测试工作。

在公司做测试期间，测试用到的技术就是手工点点点，作为一个菜鸟，我连测试用例都不需要写，只需要跟现场用户类似，不断的进行功能操作。在这个期间，我找了不少学习资料，都是软件测试的理论方法，冒烟测试、边界值分析、因果图分析都是在那个时候了解到的。也是在这个时候了解到，要做好一个软件测试人员，要学的东西有很

多，理论知识要学，方法技术也要学。此后，对于软件测试的看法，再也不是以前认为的那样：没有技术的点点点了。

由于公司很小，我作为一个不是很忙的菜鸟测试人员，也会参与一些产品设计的工作，像用 Axure 绘制软件原型，测试环境的搭建，产品说明书的编写。除了技术，还学习到一些工作思考的方式。虽然那时候软件测试技术并没有一个很大的提升，但是这段经历让我对于软件测试的认知有了一个质的变化。就这样，我在第一家实习公司待到了年底，也是在年底我辞职了。辞职的原因是这家公司所有的测试人员的测试方式都是点点点，而我不觉得自己能够在这种环境下去学习更深入的测试技术。

第二年，我找到了我的第二份工作，测试开发。说到这儿，我很感谢这一家公司，因为他们接纳了我这个技术小白。测试开发，先学的是开发技术。由于公司做的是 Web 端的印刷检测系统，所以我要从前段开发的基础知识开始学。首先，便是 Html、CSS，然后是 Jquery、JS。当然这其中会包括许多技术，而我这期间只是学了其中的基础知识，因为对于一名技术小白来说，缺少实际的项目经验，学深了也是徒增疑惑。就这样，我花了 2 个月学习，到第三个月的时候，参与公司新项目的开发，负责的是平台所有的验证模块。也是此时，我开始学习测试方面的知识，包括测试理论方法以及单元测试的技术。那时候，第一次成功用 QUnit 完成了职业生涯的第一个单元测试。

到了 7 月份，我毕业了，离开了第二家公司，拿着毕业证回到了老家的省会城市。这个时候我依然很迷茫，因为我觉得我学习到了开发的技术，不去做开发很可惜，于是，我拼命的找前端开发的工作，然而没有名牌大学的光环，没有专业技术的加持，很多公司不愿意收留一个对于他们而言在前期不能创造任何价值的应届生。所以，我很无奈的选择了软件测试。看到这里，你会觉得我对软件测试不是真的热爱，其实那个时候，我是不热爱，也不讨厌。但是现在，我深爱我的工作。

回到正题，回老家的第一份工作，也就是我的第三份工作是手机端测试，主要是安卓端软件测试。公司里这个岗位基本都是今年毕业的应届生，资格最老的也只是比我早毕业一年而已。部门老大对于我们测试人员的要求就是通过重复的手工劳动去发现功能错误。于是，我又开始了我的点点点生涯。但是，我也不甘将之前学习到的知识放弃，所以我尝试将开发知识用于测试。于是，我开始学习手机端自动化测试工具，第一个就是 monkey,但是 monkey 是向待测应用发送随机按键消息，只是验证待测应用是闪退或者崩溃，因为它不支持条件判断，也不支持读取界面的信息来验证操作，所以只能作为生

成一些随机事件的工具来使用。于是我学习了第二个测试工具---Monkeyrunner。Monkeyrunner 在我做手机端测试期间一直在学习使用。因为其测试用例是用 Python 写的，所以在这期间也简单学习了些 Python 的一些基础知识。然而，由于每天都有要更新的软件，所以基本上每天都是在不断的重复手工测试，留给自己在上班期间学习使用工具去做测试的时间根本没有，因为一款软件当天更新就要当天通过测试当天上线。于是，我辞职了。辞职原因有很多，最重要的还是在这个环境下学不到我想学的测试技术。

我的第四家公司，也是我目前就职的这家公司。面试的时候公司领导觉得我有朝气就让我通过了面试。目前正在学习 LoadRunner 和 UFT，版本分别是 11 和 12。值得一说的是，这家公司软件部刚刚成立不久，目前软件测试就我一个。这种环境让我觉得自己责任很大，所以要更加努力去学习。作为一名应届生，要学习的知识真的还有很多很多。我觉得只要按部就班的学习，在实践中找测试方法，那么一定能够出色的完成工作任务，成为一名合格的职场人。

最后补充，我的专业是信息管理，理想是作一名纯粹的软件测试工程师。

《51 测试天地》(四十八) 上篇 精彩预览

马上阅读