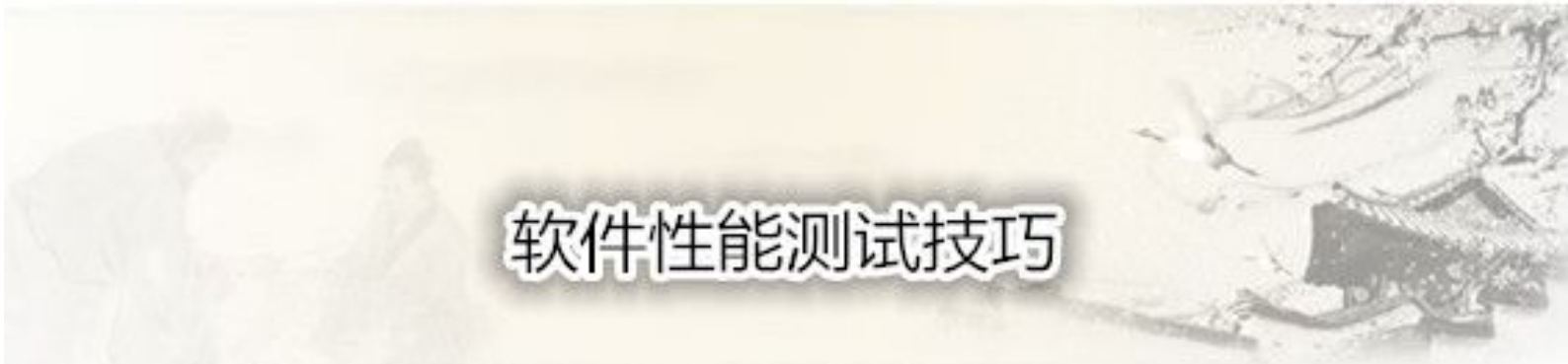
泽众软件测试网
电子期刊

SPASVO

第三十一期
2015年3月



软件性能测试技巧

怎样做好一个软件测试工程师

国际化测试和本地化测试

云测试中QA团队的作用

以不同的观点进行测试设计——六顶思考帽

不是所有测试工具都一样

多平台移动开发背景下的自动化测试和QA

上海泽众软件电子期刊

2015 年 3 月 第三十一期

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：lizf@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

目录

软件性能测试技巧

偶尔我会看见一些关于性能测试员所需技巧的热烈讨论。看起来似乎大多数专家都赞同：性能测试需要更多技巧和知识，而不仅仅只要通过使用特定负载测试工具来创建并运行脚本就足够了。但仍是有可能想象大型企业的一名性能测试员只创建脚本并机械地运行它们而由其他性能专家监控系统并分析结果的，我对这个人或这种方法都不抱希望。系统现在已变得很复杂了，所有特定专家的意见都无法完全地描述性能。性能测试所需的技巧，除了负载测试外，我们至少该想想以下几点:

——系统是什么？

——监控和性能分析。

——我们发现一个问题，该怎么办？

——诊断，调整并进行系统性能管理。

——调整不管用，是不是应用程序出问题了？

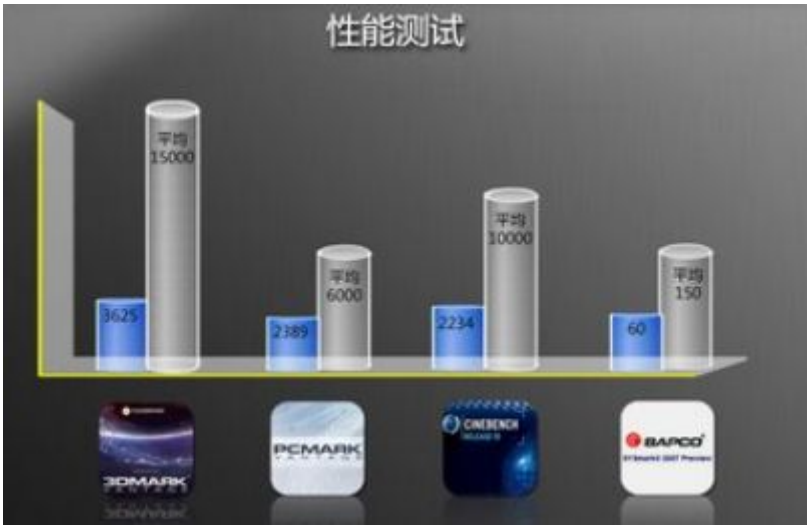
——进行软件性能管理。

——要是应用程序出问题了怎么办？

——建模并做出性能规划。

——我们该怎么完成这一切呢？

——沟通，呈现并进行项目管理。



要想成为一名优秀的性能测试员（通常在这方面更资深的专家是性能工程师或性能设计师），你或许该对上述内容稍作了解。你不必成为一名比如数据调整方面的专家——大多数公司都有这方面专门的DBA——但你确实需要能够和一名 DBA 用他/她的语言交流以便有效地协调工作；或开始关注当前应用程序设计的性能的重要性。很遗憾，这并不简单，你需要知道的够多以了解到底是怎么回事并进行有效沟通。问题是该如何获取这样的技巧。通过不断地自学并一步步地累积经验？当然没错，但是那要花上不少时间。更何况，要从头学习某些领域相当困难。

你需要在适应独自深入学习前做一些基本了解。报个班如何？肯定要的——报一个教性能测试和你的主要工具的班。但是你在用的许多其他不同的产品怎么办呢？这或许意味着要上几周的性能相关的每种产品的班。但是这些是专家通过调整这些特定产品以谋生的法子，你没有时间报所有这些班，通常也不需要研究的这么深入。和一名专家谈谈？当然可以，如果你身边有一名专家的话。性能专家是稀缺而忙碌的，所以你最好准备好有质量的问题，这一点很难做到如果你对这个话题了解甚少的话。

当你沿着这条路走的够远时，又会陷入另一个困境。你已经充分了解到基础训练没有用，但是对性能测试员来说基本是没有高级班的。当你超越入门阶段时，像环境、工具、系统、应用程序等的细节就变得很不一样，这样一来为特殊组合创建一个班就会毫无意义。你知道你需要更多信息，你需要确认你

与其他专家的方法和做法，你需要更高级的技巧和窍门，你需要找到可以和你共同探讨你的问题的人。我觉得适用于两种情况的解决方案是一个好的会议。一些人将信息消化然后再回过头将之呈现给你。它不是绝对理想的，因为所呈现之物的质量和呈现者一直在变。但当你面对许多不同话题时它可能仍是最有效的方法。然而对于一名性能测试员来说并没有完美的活动。我觉得最接近的应该是 CMG (www.cmg.org)举行的性能和容量会议——一个实用的致力于有着较强性能测试轨迹的性能工程和容量规划会议，尽管比起测试，会议的重点更放在性能上。性能和可靠性研讨会（WOPR）可能是唯一一个一心致力于性能测试（以及一些相关方面）的了，但是由于其格式，它仅在 20-25 个人之间，只限于受到邀请的人。有许多很不错的测试会议比如 STAR 会议（www.sqe.com/Conferences），Agile Testing Days（www.agiletestingdays.com），或 CAST (www.associationforsoftwaretesting.org)，在这些会议中，你或许可以找到一些关于性能测试的报告——但实在是少之又少。在软件测试教授（STP）会议上可以找到一点性能相关的材料——但是它的重点却是测试，基本和性能工程没多大关系。

Velocity 会议(velocityconf.com)对于 web 性能是一个重要活动。在 Velocity 会议上，你会看见许多性能测试员和厂商展示他们的性能工具，但很少真正涉及经典的性能测试。Surge（surge.omniti.com）是另一个很好的 web 性能和可扩展性会议——但是在那儿你可能听不到关于测试的信息。还有不少更专业更学术的跟性能的不同方面相关的会议，如果你对性能的一个专业领域有兴趣的话就可以考虑看看，但是通常测试是不包含在内的。当然，有很多厂商活动覆盖了他们的特定产品，这或许会引起你的兴趣如果你正在使用他们的产品。

怎样做好一个软件测试工程师

测试工具：人们总是认为测试工具是每个测试难题的解决方案。有了工具实施，测试就会进行地很快，质量更高，自然也更便宜…… 可惜现实却是，测试工具实施要花上不少钱，而且投入还不一定有回报。到底为什么测试工具实施经常失败呢？

二十五年前，我刚刚成为一名年轻的软件工程师，那时候专业的测试员还很少。只有大型的重要项目才会有测试团队。对于多数项目，有首席分析师的软件测试只在发布前检查系统。偶尔，会在接受阶段与顾客一起进行测试，这会导致不可预测的结果。在经历了尤其痛苦的顾客接受阶段后，经理会召开会议，宣告：“下一个项目，我们一定要在发布前进行测试。”“但是谁来测试呢？”项目负责人问道。“Bob 和 Jackie 不是很忙，他们可以在发布前两周进行测试。让他们尽可能多地找出 bug。”“好的，那就试试吧。”于是下一个项目中，所有测试成员都在最后两周都埋头苦干。我们新晋的测试员 Bob 和 Jackie 尽管经验不足但仍是竭尽全力。但是 Bob 并不想以测试为职业。Bob 对整天测试没兴趣，他终会离开这个项目。Jackie 找到的 bug 比 Bob 多，她坚信该过程，她会将她所学都贡献到以后的项目之中。这是一个十分典型的场景，让我想起了 2010 年 3 月出版的 Dilbert 漫画，Dilbert 的老板在一个新的软件版本的质量测试中寻求他的帮助。Dilbert 找了一堆的蠢借口不想当质量测试员，还用 binder 打老板的脸，总结了他对老板的不满。这件事中，很明显 Dilbert 对测试不感兴趣，更何况他很可能根本就不具备一名优秀测试员的技能。当然，这只是讽刺，但它和现实很接近。近几世纪，开发方法和测试过程都明显地发展了。但是 IT 界仍有人相信任何人都可以成为优秀的测试员——但是真的每个人都能正确地测试吗？想当然地认为任何人都可以做好测试是不正确的。我个人认为要想成为一名优秀的测试员你需要一些遗传特性，在这里我们讲的是什么特质呢？

一个天生的测试员：

1.需要技术知识和深奥的分析能力创建极其复杂的测试。这些特点，伴随着一个将事物分解的本质特点，增加了终端产品的力量和可靠性。简单的测试可以找出最明显的 bug，比如格式错误或丢失边界验证。但是需要更具体的测试场景来揭开逻辑错误或级联效应。比如，将一个状态图的所有例子都过一遍，尤其是从一个状态到一个禁止状态，常常会有令人惊讶的结果。对于复杂的例子，将要执行的测试记录下来很重要。使用过时的 Excel 表总比什么都没有好。

2.拥有学习能力。测试员可能会被要求在很短的周期内从有限认识产品到掌控该产品。他们必须能

够在总体概述产品时记下细节并了解每个模块的概念。测试员必须要愿意通过学习技术资料并把时间花在重要分析师上来检查并学习预期系统行为。我记得一个十分复杂的用于铝冶炼厂的应用程序。管理层不太确定测试团队是否能充分测试。但是通过读了所有资料并提问我们，我们干的不错。想要了解应用程序的细节，尤其是说明不够明确的时候，绝不要因为害羞而不提问题。

3.你能打破常规，并将假设与具体事实考虑在内吗?并不是所有状态都一定在功能说明中。这就像你买一辆车的时候，你下意识就知道很容易打开引擎罩检查汽车。汽车功能中并没有提到这条准则，但是每个人都这么希望的。测试员应该试着测试未写的功能。一些未写的特点可能对终端产品有重要影响。因此需要体会言外之意。比如，系统可以支持一些要求的功能，但是如果我尝试一些不同的东西会发生什么呢？系统支持吗？会崩溃吗？会破坏数据吗？

4.培养敏锐的洞察力并留心小细节。很不幸他们的完美主义可能惹恼编程员和开发员，但是优秀的测试员可以在最不可能的情况中找到最大的 bug。如果用户知道系统操作的顺序，为什么他们不能操作呢？为什么屏幕上用不同的字体标注？没有正确对齐或大小写不一致的报告是对产品质量产生不利影响的小细节的其他例子。一些人更多地注意到这种错误。它们很有可能在他们的日常生活中。

5.深切关注终端产品。他们坚信他们的任务：保护公司的声誉。他们喜爱测试并以找到 bug 为荣。找到一个 bug 很令人满意，找到一个特别棘手的 bug 尤其使人开心。

6.有组织且灵活。他们很注意说明并系统地构建测试。这对重现 bug 很重要。为了重现而不能详述的 bug 是无法被修复的。他们也能适应项目中的变化且如果必要的话愿意一遍遍重复测试。一个 bug 修复后，或许需要改善测试用例并重新执行以验证系统的质量。

即使有了这些特质，如果一个人无法给开发团队带去积极的影响的话，他们仍不算是优秀的测试员。测试员必须提供积极的反馈，能够激励团队成员改善他们的工作质量，且在一般情况下管理每个成员的自尊。

测试员的角色在不断地变化。为了在如今的市场上占有一席之地，公司必须以更少的成本更快的速度生产更复杂的软件解决方案。测试管理工具，系统仿真以及自动化测试用例执行如今势在必行。我们必须通过开发我们的编程能力或与开发员紧密合作来适应这些变化。和开发员一起尽可能快地推动更完整的单元测试并进行测试可以大大地帮助在测试早期减少错误。即使是高效的测试员也不能保证一个产品完全没有 bug。但是选择正确的人担任测试员将通过减少遗漏 bug 的影响而带来最佳结果。

总之，你的下一个项目不要选择一个 Dilbert 去进行质量测试。选择一个测试开发员时，你是在试着选择正确的人来做你的项目。你想要最好的一个。只需在选择一个质量测试员时使用同样的原则。一个高效的软件测试员将帮助你收获最大的投资收入。

国际化测试和本地化测试

简介

有多少公司真的了解将目标瞄准其他国家的价值呢？如今的数字化市场与 15 年前的市场很不一样。尤其是自 20 世纪 80 年代起移动电话和因特网极大地推动了市场的复杂度与软件的开发。可笑的是，30 年后仍然还有一些公司和知名品牌因为没有好好研究过去其他公司挣扎失败的原因而无法锁定这些“新”市场。这就造成了巨大的经济损失且很可能负面影响了不少的品牌信心。这些日子，零售国际化也已变成不少学术研究的话题。知名品牌比如耐克试图通过 Lebron James 和功夫大师瞄准亚洲青年市场。但是由于中国市场觉得被侵犯而失败了。Puffs 面纸试图进入德国市场，却发现“Puff”一词在德语中表示“妓院”。Gerber 试图在亚洲销售婴儿食品并使用美国一样的包装——标签上是一个白人婴儿。问题在于：在亚洲，公司更喜欢在标签上印上包装内物品的图片，因为很多人会看不懂里面是什么。经常，软件项目开始时没考虑瞄准国际市场，因此无法相应地分配资金。因此，许多项目计划在开发生命周期后期考虑这些测试活动毫不奇怪。开发周期后期引进本地化肯定会给管理者带来关于预算，资源，流程和技能方面的伤脑筋的 “惊喜”。一般利益相关者意识不到本地化不仅仅只是测试支持的语言。国际化要在考虑本地化之前实施，除非团队使用从一开始就支持这种形式的方式来定制系统。许多团队

不知道不理解，但是国际化和本地化之间有一个很明显的区别，而且将被确定的缺陷种类是不一样的。

i18n 和 l10n 说明

首先，我们来讲讲国际化（即全球化）和本地化的区别并了解一下每一个将呈现的缺陷类型。

A.国际化/全球化（i18n）

缩写“i18n”被广泛使用，18 代表首字母“I”和尾字母“N”之间的 18 个字母。国际化就是设计并开发一个产品保证其能适应用户的文化，语言，更甚是区域。系统是设计来支持通用字符集，统一的字符编码标准以及其他技术以便支持更多可以被垂直写入的有挑战性的语言如日语，韩语，汉语和蒙古语。大多数这些缺陷，开发员和测试员不用通过翻译理解外国语言文化就能找到。本地化的产品中出现的典型性问题不该只以一种语言出现，比如没有正确显示的不同的字符集和特殊字符；由于 UI 限制而缩短的字符串；字符串串联；错误的日期，时间，货币，数字等的格式。下图，一个展示了日期的例子，另一个中没有正确展示字符串。

B.本地化（l10n）

缩写“l10n”被广泛使用，10 代表首字母“I”和尾字母“N”之间的 10 个字母。本地化就是将实际内容转化为另一种语言。一个特定语言中的主要问题基本是语法错误，未翻译的字符串，缺少本地化比如错误的 URLs 展示，误译，或术语错误。

参考，标准和服务

不是每个人都会遵循标准，想遵循标准，或甚至了解标准。有时候没法选择，但是尽管如此它们仍是学习和更好地了解的上好资源。使用服务是不错的选择，因为要为每种语言都雇佣翻译或许相当费钱。期待一个测试员能覆盖所有的配置和语言是不实际的，使用一个在线翻译 app 比如 GOOGLE 翻译也绝对不是解决方法。许多开发网络很好地覆盖了国际化。一些常见来源有 IBM Globalization 网站，Microsoft’s MSDN Go Global Developer Center [3]和 Oracle Globalization Support [4]。本地化行业标准协会（LISA）2011 年的时候关掉了，如今在那儿的是语言术语/翻译和创作协会（LTAC）。有兴趣的话，看看 TBX（用来交换已被 ISO 通过并出版为 ISO 30042 的结构化术语数据的 XML 标准，管理术语，知识和内容的系统）。

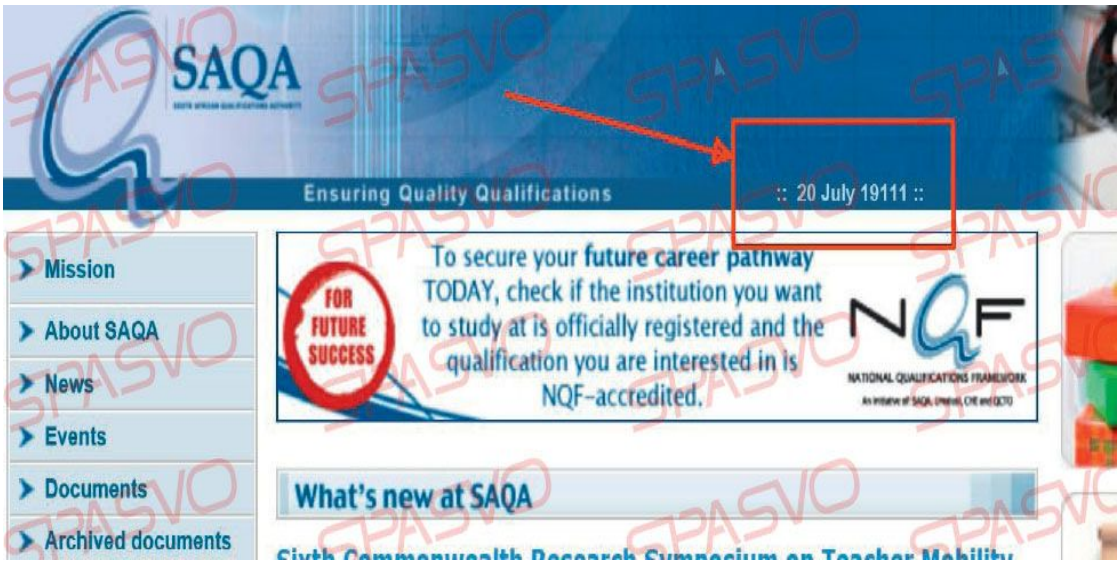


图 1. SAQA 网页上的日期格式

还有 W3C 国际化活动，它与 W3C 工作小组合作；它支持其他组织，使得 web 技术对各种不同语言、脚本和文化都行得通。

提到服务，我们就来看看国际化和本地化协会（GALA），它是一个为全球企业提供资源，教育，宣传和研究的非盈利组织和语言行业协会。还有 TerminOrgs，它是一个推动术语管理成为大型企业的企业标识，内容开发，内容管理和全球交流的重要组成部分的术语学家的共同体协会。

好的做法和技巧

好的做法就像当大家都发现你即将为人父/母的时候，突然，每个人都成了迎接小宝宝，照顾她/他，

等她/他成年后怎么指导她/他选择大学的专家。众所周知，兼听总没错。涉及国际化和本地化时，下面几点可以给予我支持。

1.从一开始就考虑全球战略，并优化你的语言集和语言。

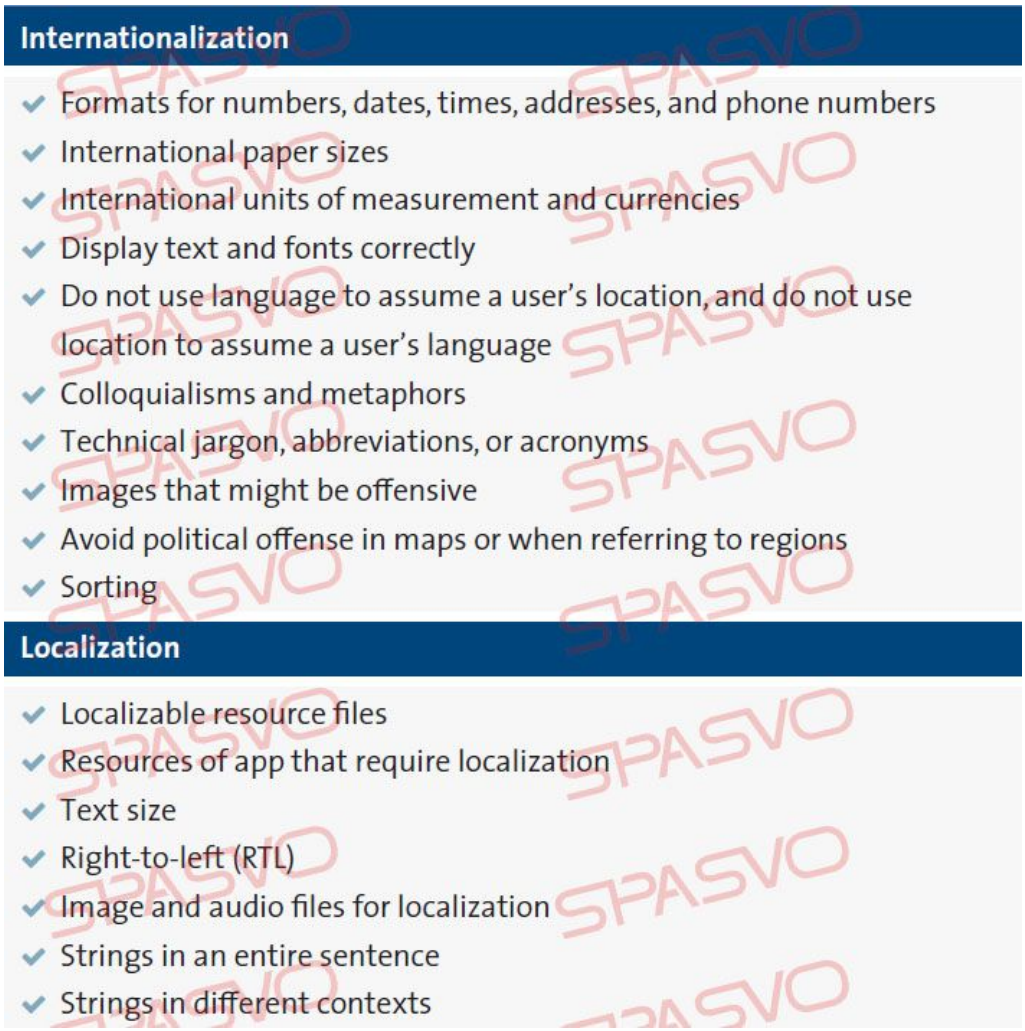
2.不要想当然，因为如果事情和你想的不一样，你会出丑。

3.让区域利益相关者参与其中。

4.使用第三方专业机构的技术和服务。

5.早点并经常进行测试。需要有一个明智的测试策略，因为支持多个浏览器，操作系统和设备将快速增加所要求的超出初步预算的测试工作。

6.使用免费的清单和 bug 分类。只要搜索“本地化清单”。这里有一个简单的清单样例：



7.找到字符串的硬编码资源并将它们放到一个资源文件夹中。

8.如果有任何联结的字符串，比如描述环境的形容词或更详细的，找出来。很多时候英语行得通但一些其他语言却不行。

9.找出任何硬编码日期，时间或货币格式。还要找出一周的第一天是星期几。一些国家是周一，一些是周五，一些是周六，还有的是周天。

10.如果产品需要支持亚洲语言，要确定使用了 UTF-16。如果不需要，就要使用 UTF-8，除非你有充分的理由可以不用。对 UNICODE 了若指掌，因为写的时候，它们就已经在起草 7.0 版本了。

11.找出任何字符串基本无法很好地适应的地方。如果产品需要支持德语或汉语，就会有许多像这样的地方且很多 UI 设计师需要想出聪明的方法来解决。

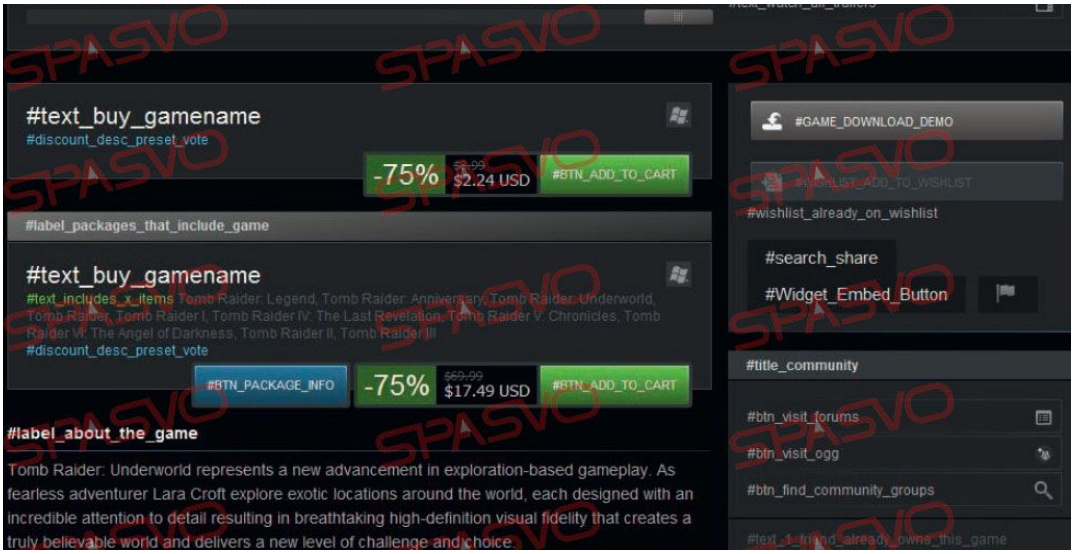


图 2. 游戏古墓丽影的转换失败：冥界页面

尝试伪本地化测试以阻止常见的国际化缺陷。这是创建一个使用与英语相同的人工语言的本地化产品的过程。除了一点：各个字符是由一个不同的视觉上类似英文字符的字符写入的。这应该完全由机器生成以便节省时间，伪本地化应该用与建本地化完全一样的方法去建。甚至单语英语软件开发者和测试员都可以读取伪本地化文本，事实证明这是在开发早期发现国际化问题的一个很好的方法。开发早期，通过重点关注包括英语在内的五种语言优先找出国际化缺陷。经验表明我们最有可能找到以下语言的特定缺陷：

（1.）德语：它含有很长的单词，比起其他语言，可以更好地揭示对话长短和对齐方式的缺陷。

（2.）日语：数万字符，多个非拉丁脚本，非正统的输入方法引擎以及一个相当复杂的拼字法。日语是一种很好的找出影响许多东亚语言的缺陷的方法

（3.）阿拉伯语：它是从右写到左的，是上下文成型法，就是一个字符的成形依赖于相邻字符。

（4.）印度语：该语言可以帮助找到遗留的，非统一字符编码的会影响所有语言的缺陷。

12.如果你知道用来开发产品的库，那就对他们的论坛和 bug 追踪系统做些调查以找出限制和问题。这些是设计测试创意想法的良好来源。

13.找出有分类功能的区域。使用网络浏览器来分类往往成问题。

14.测试自动化是你的朋友。尽可能使用数据驱动的测试自动化，但要注意：测试自动化框架能支持的内容或许会有限制。

管理缺陷

如果根据（现在已被代替的 ISO/IEC 25010）ISO/IEC 9126-1 和 ISO/IEC 9126-2，按质量特点类别来记录缺陷，那么国际化可能会归类到轻便，而本地化可能会归类到实用。将缺陷分类就为质量保证（QA）团队提供了有价值的统计资料以帮助他们评价整个软件开发生命周期中现有流程的成功，并根据所学在将来建立项目。同时，确保团队使用的缺陷追踪系统支持所有其他系统支持的语言。开发到一半切换到电子表格只能增加不必要的手动管理和困难。此外，还建议让所有支持的语言使用单一的管理系统，因为这样就可以简化并提升报告效率。一旦各种语言的测试报告蜂拥而入，要有效管理这个就相当困难了。别尝试通过检测单个 bug 并使用手动流程处理多语言 bug 报告，接着由拥有受影响组件的功能团队手动一个接一个地翻译，实时跟进。这是一个耗时且容易出错的事儿，不适合大型且多变的项目。为缺陷追踪工具创建一种（如其报告的一样）能够自动检测用户缺陷语言的语言检测 API。或只要让报告者可以选择语言就行。如果缺陷追踪工具能够让一个特定的团队或个人根据特定领域的特定值提前提出缺陷，这样的话这也能有所帮助。必须要存在并明确规定缺陷影响（严重性）来帮助快速优先处理缺陷。一般，项目使用四种严重级别。明确规定严重等级的例子可以很简单，就如下表中阐明地一样。

Severity	Description
1 – Critical	Crash or hang, complete loss of essential functionality with no workaround, copyright issues, offensive text, serious financial loss, loss of confidence and reputation in organization, losing >$100k/day
2 – Major/High	Essential functionality not exactly as specified but workaround exists, critical text not visible, loss of reputation, losing between $50k/day and $100k/day
3 – Minor/Medium	Less essential functionality, mainly cosmetic, losing between $1k/day and $50k/day
4 – Trivial/Low	Punctuation, cause irritation, losing between $0/day and $1k/day

总结

如你所见，一个瞄准国际受众的产品并不总像一些人认为的那么简单和便宜。犯错的话会严重影响品牌，使你们在接下来的几年成为一个笑柄。我们作为测试员就要试着尽早参与其中，并说服利益相关者仔细规划和分配充足的资金使之顺利成功。

云测试中 QA 团队的作用

公司转向云部署模式或使用云测试服务来测试一个 app 时有不少好处。本文将介绍高水平的云概念并讨论云测试中质量工程师的作用。

我们来看看云测试时需要考虑哪些吧。

什么是云计算？

根据一份 Gartner 报告，云计算是“一套规则，技术和用于提供 IT 功能（软件，平台和基础设施）并将之作为一项按需扩展弹性服务的商业模式”

云计算的五大特征

▪它使用动态的，共享的和虚拟的基础设施

▪它是弹性的，可扩展的（根据负载增加和减少）

▪它根据消费定价

▪它可以用在各种常用网络上

云部署模式

基于部署模式，有公共云，私有云，混合云和社区云。

为云用户提供的云服务模式

高水平的有三种模式：SaaS， PaaS， and IaaS。但是这些已经被最近开发的更多服务模式扩充了。主要云服务模式有：

▪软件即服务（SaaS）- 例: QuickBooks Online accounting software on the cloud，Yahoo mail，Gmail

▪平台即服务（PaaS）- 例: Google AppEngine， Amazon Beanstalk， Saleforce’s Heroku， Cloud Foundry (开源 PaaS)

▪基础设施即服务（IaaS）- 例: Amazon Elastic Compute Cloud service(s)， S3， Windows Azure Storage

▪数据库即服务（DBaaS）- 例: Oracle Cloud database service， Amazon Relational Database Service - MySQL， Microsoft SQL Azure， Amazon DynamoDB， MongoDB database as a service

▪分布式计算即服务（big data）– 例: Hortonworks， Cloudera， MapR， AWS

▪测试即服务（TaaS）– 例: Soasta， HP， Keynote， Perfectomobile， Gomez

▪内存内缓存服务– 例: Amazon Elasticache

▪后端即服务– 例: Parse， Stackmob， Cocoafish

▪集成平台即服务– 例: Mule Software

云计算和云测试的好处有？

A）云计算

▪业务敏捷性和快速上市时间模式云计算提供即消应用云的资源如 IT 基础设施，软件平台和商业应用.

▪优化 IT 成本——云可以减少你的部分 IT 运营成本。

▪现收现付模式——云计算的一个重要特征就是它的按需功能。计算，存储功能或 app 资源只在需要时用，且你只需支付你使用的。

▪资产高回报——云计算通过卸载数据中心帮助减少 IT 成本，IT 运营人员和相关成本。

▪创新推动者——云提供一种方法来创建更多业务和 IT 组织内部的创新。

▪业务试验推动者——云为业务试验和减少风险提供一个平台

B)云测试

让我们将测试分为高水平的两类：

1.测试云中的 app。

2.使用云测试服务测试 app。

测试云 app

使用之前提到的一种部署模式将 App 部署在云中。测试员测试功能、集成、端到端、负载、压力、和安全等测试的各个方面。

例子：小业务要用的 QuickBooks Online (SaaS) cloud accounting software， Gmail， and Yahoo Mail。

Web app：app 用户界面是通过浏览器和 web 服务（SOAP， RESTful）并使用客户端工具（如 Chrome Postman， SOAPUI）或使用基于 java 的单元测试框架来测试的。

移动 app：app 下到智能机上，并用真实设备和/或模拟器测试。

例如：iPhone 的 QuickBooks Online，Android 的 QuickBooks Online，和 iPhone/Android 的 Yahoo Mail.

使用云测试服务

他们交互式地自动地为测试 web 和移动 app 提供基于云的测试平台。云测试服务供应商需要维护测试基础设施，提供持续集成工具和自动化工具。这种服务模式也称为测试即服务(TaaS).有时，term 这个术语也可以指外包测试模型。但是本文的重点是云测试服务模式而不是外包模式。

例如：Soasta， HP， Keynote， Perfectomobile， Gomez

云测试的优点

一些优点和之前提到的云计算的优点一样。以下是一些不一样的优点：

▪复杂性和资源：app 在增加在改变，先在还多维。例如，一个 webapp 需在不同的操作系统上用不同的浏览器测试。如果是一个移动 app，那么就需要在不同的移动设备，移动网络并按照如 3G， 4G，Wi-Fi 和 Wi-Max 的标准来测试。创建很复杂的测试实例需要资金和资源。云为测试提供所需基础设施和工具。

▪成本效益：组织不需要投入时间，资源和资金来维护测试实例。我们只需要为在云中的使用付费。如果你在进行负载测试，你就不需要等测试实例准备好了。工具许可证和维护费用被取消了。

▪更快的交付周期：因为测试员基于其要求的配置快速地得到测试实例，一个云测试服务可以被集成到我们的持续集成系统中，所以它积极影响了上市时间。这也支持使用多台虚拟机器的测试平行执行。

▪稳定的测试实例：如果没有服务（储存空间用完，资金用完，服务器宕机，等），测试工程师通常会记录事件。因为云测试服务是 SLA 驱动的，多数时间测试实例在增长和运行。

▪按需自动化服务：测试实例是从大范围的虚拟环境以自助服务的方式来按需提供的。测试可以手动也可以自动进行。

▪多地执行测试：对于负载测试，可以通过选择实例，服务器和服务器位置来按需形成负载。

QA 团队的职责

1.云就绪

▪你的 app 准备好要在云中测试吗？如果没有，建议 QA 团队花时间考虑安全和防火墙等问题以便让云就绪。

▪确保测试，测试框架和工具准备好使用任何一个云测试服务供应商而非锁定一个供应商/工具来进行测试。

2.云测试的策略和发展蓝图

如果你的组织计划使用云部署模式，那你就需要确保测试的可测试性和各种类型包含在发展蓝图内。

▪了解云供应商提供的云部署模式，模式和技术。

▪想出一个云测试策略。这也应该覆盖将被执行的测试，手动或自动化测试，以及测试执行的持续时间。

▪列出你的 app 平台/技术的组件和它的版本，比如 app 服务器，web 服务器，函数库，数据库和操作系统。

▪与你的团队和主题专家重新评估策略。

3.列出相关服务/系统

比如，如果你的 app 被集成到内部/外部相关服务中，那么当你的 app 部署到云上或用云测试服务测试时确保相关的组件工作。

▪如果你的 app 被部署到云中，确保云部署服务提供商和测试服务提供商能够彼此交流一下。

▪如果你的 app 被部署到你那儿，要确保你的云测试服务提供商可以接触到 app 和集成组件。

4. 确定测试（手动和自动化的）

确定将在云测试环境中执行的手动和自动化测试。因为测试服务成本是基于使用，花时间计划测试执行很有效。

#	Attribute	Cloud-testing service
1	Application's technology stack	Check whether they support your application's technological stack.
2	Mobile testing	Make sure the vendor provides real mobile devices with the required versions – this is important for testing native applications such as camera or certain dongles. Make sure they provide emulators. If you have both web and mobile testing needs, make sure they support both platforms.
3	Tools	Check whether the vendor provides industry-standard tools for continuous integration, and automation tools.
4	Monitoring	Check whether they provide monitoring of the test execution during and after the run, and monitoring of the resource utilization.
5	Performance testing	Check whether they support performance testing instances with different geographical regions.
6	Time	Compare the time taken to provision the test instance and the time taken to execute your test.
7	Debugging	Check whether they provide access to the log files, screen shots, and video of the test execution.
8	Cost analysis	Analyze and compare the cost of different vendors for your organization's specific requirements.
9	Service level agreements	Understand the vendor's service level agreement and identify the responsibilities of your organization and the cloud service provider.
10	Customer support	In the event of issues, check the quality of their customer support based on other customers' experiences or references.

图 1. 选择云测试服务提供商的准则

5.为在云中测试移动/webapp 选择正确的供应商。在选择供应商前，彻底了解你公司的具体需求以

及服务提供商提供了什么。使用他们的免费试用机会去了解。有效地花时间和金钱在此任务上很好。

6.安全：QA 团队应该带上白帽子。维基百科上说术语“白帽子”在网络俚语中指一个有道德的电脑黑客，或是一个电脑安全专家，精通渗透测试以及其他测试方法，确保组织信息系统的安全。了解资产的安全责任，比如物理安全，网络基础设施，虚拟机器，传输中的数据，静止数据，操作系统，凭证，策略和配置。一直要在你的手动/自动化云测试中使用非敏感合成数据。云安全联盟（CSA）是一个非盈利组织，它的使命是促进最佳实践的使用以在云计算中提供安全保障。云安全联盟是由行业从业者，企业，协会以及其他关键利益相关者组成的大联盟。根据 CSA 2013 报告，以下是前九大威胁：

a. 数据泄露

b.数据丢失

c.账户劫持

d.不安全的应用程序接口

e.拒接服务

f.恶意的内部员工

g.滥用云服务

h.不够关心

i.共享技术问题

7.规定遵守

我们应该要知道云供应商规定的用以维护安全并保护数据的限制。遵守责任基本大家都知道——云供应商提供基础设施的保障，且你的公司负责在基础设施之上的 app 的遵守。我们需要了解如 Sarbanes-Oxley， ISO 2001， ITIL， SAE 116， SAS Type I and II， HIPAA， PCI，等由云供应商基于你们公司具体要求而规定的标准。我们可以基于我们的业务需求要求审计员制作的报告和证明。政府组织有特殊的监管需求。比如，亚马逊网络服务（AWS）GovCloud(美国)是用来解决美国政府机构，教育机构，其他用户和合作伙伴的特定监管需求的。

8. 了解服务水平协议

了解云服务提供商的服务水平协议很重要。它指出了服务不满足服务承诺时的正常运行时间，以及申请信贷请求的规定。

9.成本使用

确保云服务被正确使用，因为会按你的使用收费。同时，确保服务能被充分利用。大多数云供应商提供资源使用报告。验证报告以确保你为你已经使用的服务付费。

10.了解云供应商的位置和可用性区域

云供应商的位置对于网络延迟和性能测试很重要。比如，世界各地都有亚马逊的弹性计算云（EC2）。这些地方包括域和可用区。每个域都是一个独立的地理区域，都拥有多个独立的可用区。

11.移动云计算的未来

除了成功的业务云如电子商务网站，还有更多云可以解决全球贫困，农业问题和健康问题。我们看到医生使用社交媒体来和其他医生联系以获得建议和第二意见。移动和云计算都在不断发展，移动云计算同样也如此。这样就导致了更多该领域的标准。

移动手机的使用增加了。IEEE 将移动云列为 2014 的尖端技术趋势之一。IEEE 说到：“移动和云计算在融合，创造一个新平台——它有可能提供无限的计算资源。移动设备被其内存，处理能力及电池寿命所限制。但是结合云计算，数据进程和存储可能会在移动设备外发生。IDC 宣称“第三平台”能更好地同步数据，提高可靠性和可扩展性，使集成以及随时随地使用商业应用程序和协作服务更容易，增加用户体验和对新服务的探索。

举例：一个移动应用程序监控设备的位置。该移动设备不断基于设备的移动将其位置信息发送给云上的数据库。一个汽车传感器即时将数据发送到云。分析数据，发送的结果可以提高燃料效率。

作为适量保证工程师，我们需要学会它并享受云测试带来的好处。

以不同的观点进行测试设计——六顶思考帽

不只有一种观点

作为一名测试员，最重要的是一个观点，一个你自己的观点。这个观点是基于大量经验并源于你最近的项目和体验积累的知识。它也基于你最近常变的心情和你对软件，开发员，团队，客户等的个人态度的。你最近的观点也决定了你在测试设计上的能力和创造力。但是一名测试员需要做的不仅仅是以他们的个人观点来评估软件。我不认为这对于一名好的测试员客观地检查软件测试规格和/或一组预先定义的测试集足够了。你或许会错过许多关于软件的重要的项目信息。评估一个产品的质量要比计算已找到且修复的 bug 或已执行且通过的测试集更重要。有很多方法可以提升你以不同观点收集额外信息的测试设计技巧，这远不只是“通过”或“失败”了。你不必重新使用它们，你只需要在执行你现在的测试集或章程时把这些方法加到你日常的工作中。有很多方法帮你做到。今天我想为大家介绍两种方法，这两种方法是一个好测试员应该有的或应该加到他或她的工作中以便收集额外信息以及对简单节时方法的见解，并找出问题，bug 和观点，与建筑师，分析师和利益相关者探讨探讨。

六顶思考帽

Edward de Bono 的六顶思考帽最初是作为小组结构讨论的一个创新技术。目的是至少在一次讨论中引进六种不同的观点。思维导图软件 XMind 2013 在其模板中引入了六顶思考帽（见图 1）。

该方法也极适合软件测试。你可以将这六顶思考帽分给你团队中的一些人或者你也可以自己一个一个戴。使用颜色编码的元素有助于集中注意，比如列出了每个帽子（颜色）最重要特性的有色基帽或有色卡片。这样可以帮助你在带不同帽子时进入并保持最佳的心境。

现在说说不同的颜色吧。蓝帽子是客观且应该能够帮助带帽者专注讨论。如果你独自使用该方法，你就要带上蓝帽子，这样你才不会迷糊。如果要把六顶帽子分给团队成员，就可能会给测试经理和测试领队。

白帽子代表客观的信息和分析思考。这顶帽子的重点是需求和如何实现它们。在测试设计中，白帽子帮助创建 app 的模型。戴白帽子就要如预期地执行一个测试集并专注于事实。这个人的任务就是收集事实以向正在进行中的讨论告知价值中立。

红帽子代表感性思考，积极和消极都有。这顶帽子应该可以帮助你观察你自己的情绪。测试时，你建立了对被测软件的感情。依我之见，在很大程度上这也包含难以衡量的“魅力”特性。我喜欢使用这个软件吗？用起来很麻烦吗？或者很难？这样的信息通常很难放入一份 bug 报告中吧，但是至少得告知利益相关者，这样他们就有机会做出反应了。使用时令你头疼的软件或许功能和技术上都是正确的，但用户却不会觉得它有多好。

黄帽子代表一个乐观的回应。一切都围绕着最佳用例。这顶帽子只看得见软件中好的方面和益处，所以它是一顶快乐的路径测试的好帽子。黄帽子是为了体验明媚的一天，但是如果黄帽子没有其他信息，你就应该要小心了，因为这是不好的征兆！

黑帽子完全就是关于识别能力的批判而悲观的思考。这顶帽子是你肩上的小恶魔，它很擅长识别缺陷和风险。黑帽子是怀疑的，批判的。好好听黑帽子所说的，因为它可以找到许多新的错误场景或未知的风险。

绿帽子，最后但同样重要，它代表创造性思考。这顶帽子创建新想法且以不同的方法角度思考。测试中，绿帽子可以找出新方法去测试或使用功能。绿帽子可以创造性地帮助优化软件，你也可以用它找到解决方法。我建议试着像个孩子一样思考。孩子会用多种大人想象不到的方法去使用事物，因为大人受限于他们的固定思维。试着使用绿帽子来摆脱你根深蒂固的思考习惯。这很困难，尤其是刚开始的时候，但你会遇见很多有趣的想法。

Figure 1. De Bono's thinking hats, template from XMind 2013 (www.xmind.net)

其中一些想法你一开始会试着放到一边，但最好是把它们记下来之后再回顾。使用六顶思考帽子时，你为收集信息创建了无数的可能。你的项目环境应该要准备好接受不仅仅关于 bug 的信息，否则就是对创造力和反馈的浪费。测试执行时可以同时使用几顶帽子。比如，红帽子在积极输入时可以组合使用黄，绿帽子。如果红帽子的输出很消极，那么它就该与黑帽子组合使用以找出更多的风险和问题。将它们与蓝帽子组合对于将信息资源分开并在你的流程中拥有一些结构一直很重要。

你可以在思维导图中收集你的信息（参见 XMind），帮助改进结构并将所有信息一并呈现出来。

Personas

“质量对重视它的人是重要的。”——选自 Jerry Weinberg，由 James Bach 扩展。“Personas”是一种通过创建虚构代表来定义几组软件用户的方法。这种方法不只是角色测试或使用用户故事。你关注的重点不是工作或任务而是作为一个人的人，并创建一个可以抓取用户尽可能多面的样本用户的配置文件。这与向演员描述并创造一个电影角色相类似。该方法对于测试（将被很多不同用户所使用的）软件的测试员尤其好。在商业软件中，给用户做了培训或至少向他们简单介绍了一下系统。这对很多种软件来说是不可能的，因此软件必需直观并提供简单的帮助文本或不解自明的形式和流程。作为一名测试员，你已经花了几个礼拜的时间在那个产品上了，你了解每一处细节说明。你发现了许多方法，提示和伎俩。对你来说，测试那个软件很简单。但是你该如何摆脱你知道的一切？酒精和毒品没有任何帮助，因为你不该完全失去你所知道的，你只需在一两个场景中将它放到一边。那就是 Personas 试着帮忙的地方了。你扮演一个角色，你试着尽可能多地将知识放到一边，你试着完全改变平时的态度，这样你就可以看到并学到软件的新方面。你第一个发现的可能是你希望你的用户所拥有的基本知识。

停在或返回你测试期间每个环节的角色很重要。比如进入 Frank 的角色，67 岁，一名退休技工，他有点近视。过去工作的时候他用电脑，但那是好几年前了，现在他家里一台都没有。想想看：一个显示屏上，你接下来要做的不明显或没有任何描述说明。别按下面的按钮，因为你知道那是到下一页的按钮。Frank 会怎么办呢？是不是缺了什么显示按钮在哪的东西呢？将你的用户分类不容易，处理你所有用户的问题不可能。你必须找出正确的角色组合并尝试你 Personas 的定义的必要深度。这里商业软件有一系列不同于如通勤使用的售票机器上使用的软件的要求。

尤其是最后一个例子，它是看看你为何应该使用 Personas 的好机会。去火车站，观察售票机器的用户。那些是什么人？他们背景如何？他们看出下面要去哪简单吗？有人在看屏幕上显示的大量文本吗？使用正确的 Personas，你就会发现超时设定或许太短，因为你没有足够的时间读完页面上的所有帮助文本。那个超时设定场景或许在说明以及一些测试用例中有描述。但是那种情况通常都是分秒必争的，是否有可能缓慢并完整地看完屏幕上每个信息就不一定了。当 DHL 被引入德国，你可以在那些大黄盒子那儿随时寄出你的包裹并接受包裹，我个人认为其用户菜单是我所见过的最好的之一。但是当你排队等候并观察其他系统的问题时，你就会思考你有哪些要改进的地方来创建一个更好的用户体验，这样大家就会喜欢用那个盒子了。

总结

不要只从你自己的角度去测试，这一点很重要。像我刚刚描述的方法是否能帮助你设计测试并收集新且重要的信息取决于项目背景。但是知道那些方法并在正确的背景中使用它们，应该是每个测试员工具箱中的一部分。项目怎么使用你找到的信息，当然 bug 除外。但是收集和呈现信息是测试员的任务之一。

不是所有测试工具都一样

“节省时间和金钱！”“使用真简单！”“在哪都能测试！”这些只是云测试工具热潮的几个成功之处，就是这样！云工具给了我们机会可以扩大我们的测试性能且不必再一直被绑在台式机上了。既然有不同的测试方法，也就有不同的测试工具。千万别愚蠢地认为所有云测试工具都是用来做同样的事情的——不是的。

下面我们一起来探索一下三种主要的云测试工具，以及每一种的优缺点。你一看完，就知道如何选择最适合你需求并最节省时间金钱的工具了。

第一种：手动测试管理器

你还在电子表格中追踪你的手动测试和结果吗？准备为自己省下一堆时间吧。云手动测试管理器可以让你在网页上创建，管理并使用你的所有手动测试和结果。

手动测试哪里比电子表格好？对于新手，放弃本地电子表格转用云管理器意味着：使用你的测试和结果再不用受限于你的电子表格了。有了云手动测试管理器，你只需简单地登录工具的网页（一般通过电脑或移动设备），就可即时使用你的测试和数据了。另外，这些工具允许多个用户同时使用你的测试和结果。因此，你团队的每个人都可以轻易看见测试是如何进行的而无需任何人花时间收集每个测试员的输入，添加信息到电子表格中并传播结果。

优点：

▪通过任一 web 浏览器轻易使用你的手动测试和结果。

▪基于结果创建图表和报告。

▪自动与其他用户分享测试和结果（无需通过 email 寻找信息）。

缺点：

只让你管理手动测试和结果；不管理自动化测试或连续集成。

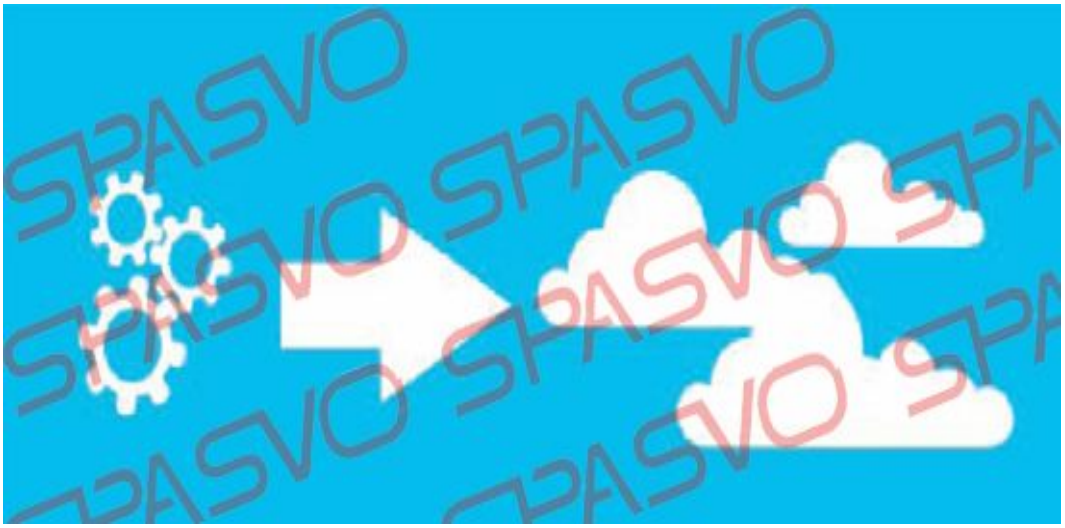
这一类的样例 app：

▪qTest——qasymphony.com/qtest.html

▪TestWave——testwave.co.uk

第二种：自动化测试扩展器

你有为自己的网站写的自动化测试或使用类似 Selenium 的 web app 吗？自动化测试扩展器通过让你可以在它们的云服务器上平行运行它们帮助减少花在运行那些测试上的时间。如图所示：



首先，你注册并创建一个账户来使用服务。然后你准备好要运行你的自动化测试时，你可以选择使用工具的服务器或在你的机器上运行你的测试。

为什么在自动化测试扩展器的服务器上运行你的测试？一个词：可扩展性。多数公司并没有需要的基础设施和资源来购买和维护（用以充分测试 app 所需的）上百台机器或虚拟服务器。通过签约使用自动化测试扩展器服务，你就能平行运行你的测试了，在多个浏览器上，多个设备上，有时甚至是在世界多地，完全不需要购买额外的机器和服务器。

优点：

▪良好的可扩展性来运行你的现存自动化测试。

▪允许你在任一设备上使用任一浏览器浏览哪些测试运行的结果。

缺点：

▪无法用来管理你的云测试

▪不能节省你的自动化测试，所以你可以稍后重新运行它们。

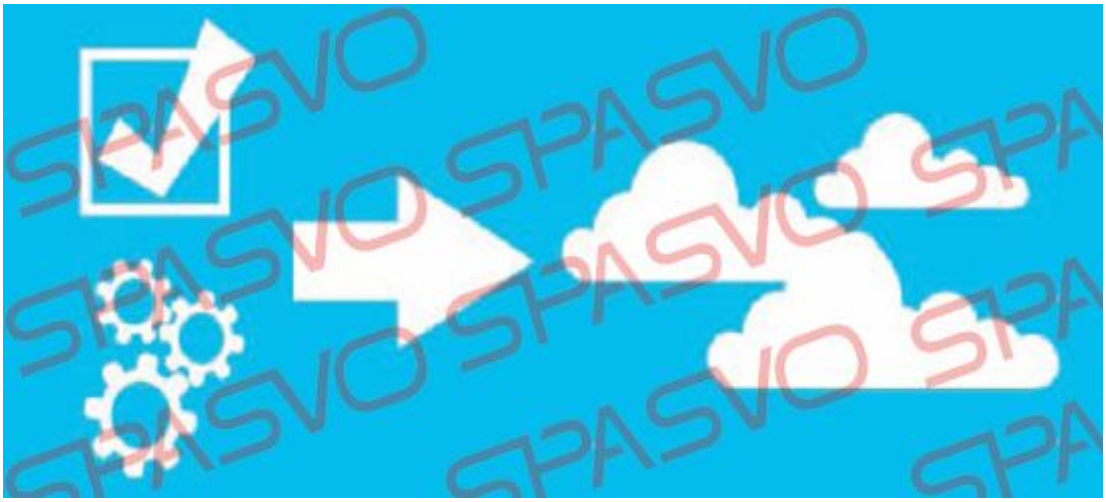
▪不支持手动测试

这一类的样例 app：

▪Sauce Labs——saucelabs.com

第三种：自动化测试和手动测试的运行器和管理器

第三种云测试工具是“多面手”。自动化和手动测试运行器和管理器可以让你通过网页创建，运行和管理自动化测试和手动测试。



如果你对手动测试感兴趣，你可以登录工具网页并创建，运行并报告你的手动测试就像第一种里你用工具所做的一样。如果你偏爱自动化测试的话，你可以登录工具网页，创建，运行并报告你的自动化测试和结果。这种测试工具不仅拥有前两种工具的所有优点，还可以让你创建，节省和管理你对工具本身的测试。这样，你的自动化测试流程就完全脱离了桌面，且可以从任一计算机或移动设备运行。

优点：

▪包含其他两种的所有功能

▪让你可以在任一设备任一浏览器创建并管理你的自动化测试。

▪安排自动化测试来自动化运行并通过 email 发送结果。

缺点：

▪可以输入一些现存自动化测试，但不是全部。

这一类的样例 app：

▪Tellurium——te52.com

这只是云必须提供的测试工具的一个样例。每一种工具都有其优缺点，所以要确保周全地衡量这几类工具的优缺点以便确定哪种工具最适合你。

多平台移动开发背景下的自动化测试和 QA

“app”一词表示我们在处理“小的应用程序”。尽管在一些情况下这或许是真的，但本文中它是指用于远程监控一个机器不同部分（比如：灯，气流和位置）状态的相当大的应用程序。机器使用一个可用后端服务器访问的（我们的 app 通过因特网访问的）移动通信网络。总之，其复杂程度和一个桌面 app 相同。app 的一个重要方面体现在不同的管理上。不同的客户群接受不同的功能设备，而不同的机器类型需要特定的数据陈述。这就形成了一个充满变数的 app——构建和运行期间其组件都是如此，这取决于我们想用哪一种机器。因此，这绝对不是一个“小”项目。它不是一个伴随现有业务应用程序的移动 app，它是这个业务流程唯一的解决方案。会有一个更长的维护阶段来支持产品改进，新功能及更多的版本。App 是机器的一个内在组成部分，且必须要有同样好的质量，实用性和用户体验。本文提供了一份该项目的概述，以及我们关于 QA 和测试自动化，持续集成和项目管理的决定和经验。

项目设置：一个概念，两个 app，两个团队

项目使用 SCRUM 敏捷软件开发框架。一次 sprint 要花上两周，包括一天的 sprint 综述，回顾和规划。sprint 综述是由整个开发团队，一两个客户代表，有时还有来自 QA 或基础设施部门的利益相关者执行。综述后客户会将规范细化。在 sprint 规划的第一部分——用户故事选择中，一名客户代表也要参加。这样开发人员可以就规范细节提问，有时提出规范变化以允许 app 更多的“本地”行为。

最重要的开发阶段计划要花上九个月。期间，团队规模会在 8-13 人之间变化，包括产品负责人和 Scrum 专家。波动一部分是跟了这个项目一段时间的学生，部分是因特殊原因暂时加入团队的专家。我们的目标设备是 iPhones 和 Android 手机，尤其是 iPhone 3GS 及以上 - (iOS 6+)，还有 Android 4 及以上。机器将被控制，服务器后端已存在，因此，我们唯一的任务就是开发 app，包括用户界面，后端通信，变量管理以及特定平台服务（比如推送通知，地图或社交网络）的集成。为了保证最佳用户体验，我们不会使用跨平台工具包；反之，我们正在开发两个独立的本地 app。

开发人员被分到子团队中，子团队中都有各自的专家和平台。为了促进沟通，两队在一个网站上进行合作。因为这两个团队，产品积压由多数用户故事构成了两次，一个版本用于一个支持的平台。对于大多数用户故事，两个版本都是根据与 iOS 和 Android 不一样的开发流程而在同一次 sprint 中计划的。

实施了一个故事时，其结果就会被拿来与其他平台的 app 相比。在 sprint 概述中，我们更愿意为 iOS 和 Android 平行呈现一个功能。通过这么做，我们就可以确保我们为两个平台都实现了功能相同的 app 和相似的用户体验。除了用户体验，Android and iOS app 还有相类似的软件结构，尽管是分开进行的。一个常见的软件结构文件中详细规定了数据模型，分层设计，屏幕流管理，变量管理以及特定域算法。因此，为第二个平台执行一个功能时，很容易理解模板，因为它是在同样的基础上执行的。因为不同部分和用户集成概念，这对视图实施却行不通——在这儿，开发是有特定平台的。

图 1. 从移动设备到机器的通信设置

自动化测试

我们测试 QA 的挑战是：对每个 app 进行多个级别（单元，集成，验收，UI 测试）的测试。因为我们想尽可能地自动化，所以我们有一个 QA 顾问，他是团队一员，推动我们的测试自动化。他负责测试规范和测试执行的审查。自动化测试的实际执行是由开发者完成，由一个为每个用户故事默认生成的测试任务触发。根据执行功能，，还有验收测试（自动化 UI 测试），单元测试和集成测试。测试总是特定平台执行的。在 UI 测试中，他们同步使用一样的验收标准。对于一个（UI 相关的）用户故事中定义的每一个验收标准，都至少有一个 UI 测试。为了实现所有这些不同的自动化测试，我们使用特定平台框架。我们低水平的测试是分别使用 SenTest 或 JUnit 实现的。关于 ios，额外的函数库像 nocilla 和 JRSwizzle 则被用于模拟。对于 UI，我们 iOS 用 KIF，Android 用 Robotium。Robotium Recorder（商用产品）已被证明可以帮助获得更稳定的 Android 测试并消除“假阴性”结果。尽管很重视 app 功能相同，但 iOS 和 Android 的导航和用户体验间的区别表明：取得并使用每个功能所需的步骤是不同的。不像在桌面领域，只是理论上有可能使用一个 UI 测试来覆盖超出简单概念的跨平台 app。这有不利之处，会增加技术和精力；但是也有好处，能够使用特定工具解决特定问题。关于比例，人们常说 UI 测试应该是测试中最小的一块。这部分是因为执行时间，也因为它们仍被视作最难写和最难维护的。我们的经验是：最好要重新评估特定测试等级在每个项目中的比例。随着对客户验收越来越重视（敏捷项目和移动领域中都是），UI 测试工具的稳定性越来越强，UI 测试和 GUI 逻辑测试不该被忽视；确实，测试中单元测试的比重要高于 UI 测试。对于这个项目，我们有 10%的单元测试，40%的集成测试以及 50%的 UI 测试。因为我们从独立后端供应商那接收的产品中的质量问题（很差的接口规范），所以集成测试的数量只低不高。

持续集成

我们使用带有一个基本的分支模型的 Git 作为我们的版本控制系统。它明确了一个主分支，发布分支，以及每个功能和 bug 修改的分支。用户故事完成后，开发者将一个功能作为一个整体合并到主分支中。为了保证不把不完整的功能整合到住分支中去，每个用户故事都生成默认任务。有验收（由产品经理和 QA 顾问审查）和代码质量（代码审查，静态代码分析，还有自动化测试）的默认任务。持续集成

的基础是主分支，因为它应该始终包含一个“随时准备发布”的项目。每次提交（整合功能）都触发一个完整的开发周期，包含以下工作：

▪更新/建立依存关系

▪建立 app（现在有三个不同的构建版本）

▪静态分析

▪单元测试

▪UI 测试

▪通过内联网进行 app 分配

图 2. 测试级别比例（最好的，典型的移动开发项目）

iOS 和 Android 我们使用 Jenkins，因为它是公司默认值，且由 IT 部门支持。尤其是在 IOS 开发中，我们遇上了(如果我们能选择 Xcode 服务器，就可以避免的)Jenkins 的初期问题。但是，Jenkins 中的额外插件最终使得将我们的 IOS 系统集成到 CI 中成为可能，比如，Clang Analyzer 和管理环境变量或共享工作间工作空间的插件。

最后的思考：自动化在哪儿不起作用

和描述的一样，我们的流程包含各种帮助我们实现我们客户的高质量期待的因素。整个团队都参与质量流程，每次 sprint 中的项目都能保证高质量。这多亏了自动化测试的帮忙。但是有的地方必须手动保证质量。团队进行桌面开发时不一定能立即想到这些，它们都列在下面了：

▪实用性和用户体验：

这是人们广泛接受的必要的手动测试，但是移动 app 的客户很重视质量。随着指令的难度增加和方向的变化，我们发现我们必须更加重视质量——测试由团队及客户代表手动执行。此设置中，是由客户来确保不同设备作为其手动验收测试被检测的。我们自动化测试被限制为一个平台一个版本。

▪国际化：

多语言桌面 app 中的正常流程是：让讲本地语言的人检测字符串，在 app 文本之外。由于显示屏尺寸变小，我们计划的超过 15 种语言的国际化比桌面 app 的更耗时间。我们的转换是由外部员工执行的，每次转换都必须手动测试以确保其在显示屏上使用恰当的空间。我们使用我们的 UI 测试通过创建可以被转换团队审查的自动截图来支持这个。

质量最重要——即使（尤其）是对于 app

本文表明：app 开发要求至少要有与桌面商用 app 同级别的测试策略及质量保证工作。在某些方面，因为为两个平台开发一个 app 的挑战，对于高质量工作的要求更高了。从许多方面来说，我们可以看到移动开发不会改变要求的质量工作。不过能改变的是特定工作的相关性或重要性。对我来说，这在我们单元，集成和验收测试的分配以及在（一个非移动项目中并不重要或并不花费时间的）手动测试中很明显。我们的结论？质量最重要——知道你要测试什么，为什么测试以及如何测试很重要。即使是对于移动 app 来说。

泽众软件工具使用技术支持

电话：021-61079698

Email：sales@spasvo.com

QQ：1404189128

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

SPASVO