目录

[一、敏捷测试有何不同？ 1](#_Toc71743043)

[二、测试自动化 2](#_Toc71743044)

[2.1测试金字塔 2](#_Toc71743045)

[2.2什么是测试自动化 2](#_Toc71743046)

[三、有策略的测试自动化才会更高效 4](#_Toc71743047)

[3.1测试人员的绩效考核标准 4](#_Toc71743048)

[3.2过度关注测试自动化，也会适得其反 4](#_Toc71743049)

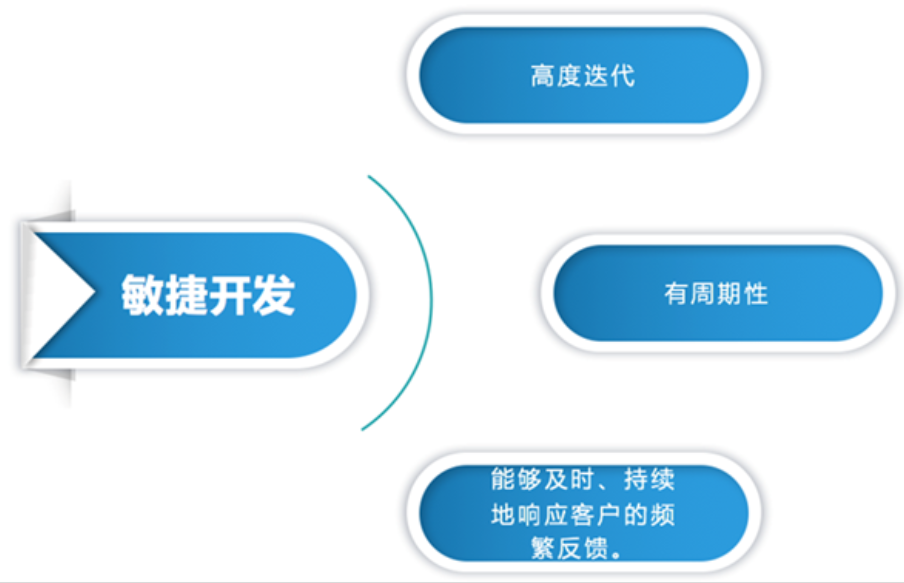
[3.3并发性能测试 5](#_Toc71743050)

[3.4自动化测试的要点 5](#_Toc71743051)

# 一、敏捷测试有何不同？

在传统项目中，我们往往更习惯于去严格定义软件开发生命周期中的各个阶段。例如从制定发布计划和需求定义开始，最终以测试和延迟的发布结尾。对于测试来说，在传统项目中往往被破担任门卫的角色。

对于团队的领导，或者大部分项目干系人来说，测试往往被寄予期望承担项目质量控制的职责。然而在传统项目中这点很难做到，因为测试既不能控制代码如何编写，也不能控制开发人员测试他们的代码，但所有的质量把控却都被希望能压缩在开发之后的测试阶段圆满完成。



在敏捷项目中，测试人员不再坐在那里等待工作的降临，而是主动寻找在整个开发周期中都贡献价值的方式：与用户一起编写需求的测试用例，与开发人员一起寻找程序中的漏洞，聚焦使用覆盖面更广、更灵活的测试方法。在敏捷中，开发人员从来不超前于测试人员，因为一个功能在被测试之前处于“未完成”状态。

简单的总结出敏捷测试的几个特点：

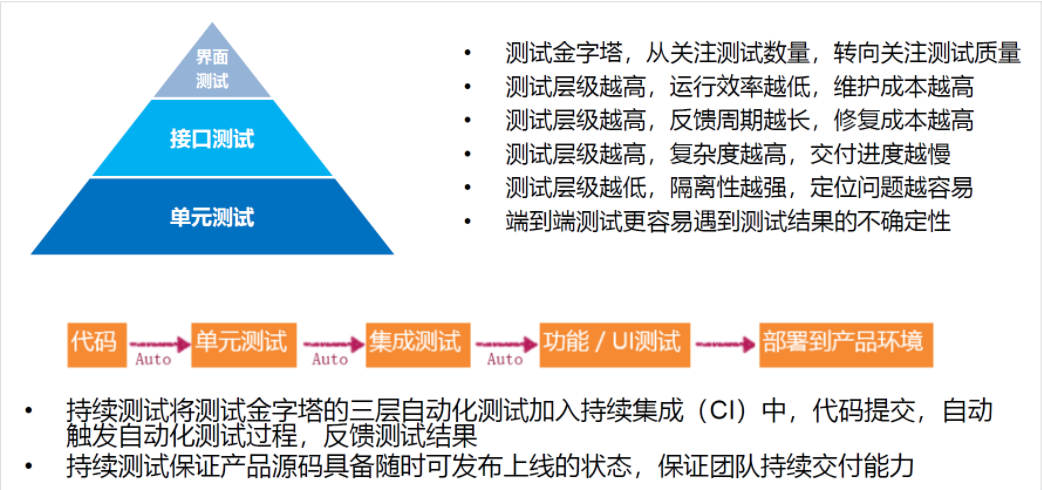
* 强调从客户的角度，即从使用系统的用户角度来测试系统。
* 重点关注持续迭代地测试新开发的功能，而不再强调传统测试过程中严格的测试阶段。
* 建议尽早开始测试，一旦系统某个层面可测，比如提供了模块功能，就要开始模块层面的单元测试。同时随着测试深入，持续进行回归测试保证之前测试过内容的正确性。

# 二、测试自动化

在敏捷和DevOps中，测试的自动化是必需的。我们需要用自动化的手段去管理关键的测试活动，并为开发提供必要的反馈。以下介绍测试自动化都包含哪些内容，以及如何做好测试自动化。

## 2.1测试金字塔

一个经典的测试自动化的模型—测试金字塔。



测试金字塔模型的目的是，指导团队从测试自动化中尽量以最低的投入获得最大的价值。金字塔展示了3个不同的自动化测试层次。

最低的一层是基础，主要有单元测试、组建测试等面向技术的测试所构成，这一层也代表了大多数的自动化测试。在这一层中，测试用例的单元隔离性最好，定位分析问题最容易，使用的代价也最低。

金字塔的中间一层包含了大多数用来支持团队的自动化业务测试。这些功能测试只在验证我们在做正确的事。

金字塔的顶层很少使用自动化，因为他的运行效率最低，开发复杂度最高，测试ROI最低。

## 2.2什么是测试自动化

有多自动化测试的手段，例如单元测试、API测试等等。这些是测试的执行部分，也就是把一些测试执行的人工测试手段通过工具做成自动化的测试过程。但是测试自动化不仅仅是只是执行部分，还包括了从环境的获取到生成测试数据、执行自动化测试、最终生成结果并提供反馈。如果测试结果有问题，系统会自动推给相关的人。最终自动生成测试报告，测试人员可以直接拿到测试结果。这整个闭环的过程才是测试自动化的最终组成。

以华为DevCloud为例，以下是完成测试自动化实践的工具：

* 在测试管理上提供了包括上文提到的整体测试流程管理、测试的用例和需求、虚线能够双向可追溯。
* 在自动化方面，提供了移动应用测试、API测试和性能测试。
* 移动应用测试提供了对应用软件包进行系统化的兼容性测试，检测软件包是否有兼容性的问题，能够涵概多少用户。



* 接口测试提供自动化的API测试工具，通过编写测试用例实现对API的自动化测试。



* 性能测试可为用户模拟一些大并发的场景、提供多种加压策略，能够在测试过程中对于用户的吞吐量、响应时间、负载能力，整体进行结构分析。在测试完成后还提供多维度可视化的看板，能够详细了解测试执行的情况。



# 三、有策略的测试自动化才会更高效

## ****3.1测试人员的绩效考核标准****

衡量测试人员的最佳方式就是每个测试人员发现的Bug数目。因为你可以从Bug数据库中直观地得到这些数据，这也会使你更容易、更客观地对每个人做出评估。但完全通过这个方式来衡量是非常不公平的。无论一个测试人员有没有发现Bug，都不能说明他有没有好好工作。测试人员的职责更多的应该是“保证质量”（quality assurance），而不是“控制质量”（quality control）。一个好的测试人员应该是在问题出现之前，防止其成为Bug。

如果是传统的瀑布模型，开发人员把做好的代码给测试人员，就是期望测试人员能够发现问题，预防又从何谈起呢？

但对于一个敏捷团队，测试人员已经从项目一开始就参与进来了。测试人员需要参与功能说明评审，描述用户使用情形，评审设计，有时也会评审代码，最终才进行测试。测试人员还帮助团队统计代码覆盖率等。从这个意义上来讲，他们的职责已经不单单是测试本身了。一个好的测试人员可以发现开发过程中的各种问题，帮助改善团队流程，帮助提升开发质量。如果整个过程运行良好的话，测试人员在代码提交后，应该不会或很少再发现Bug，因为他们从一开始就跟大家协作，预防了Bug。因此，对于一个敏捷团队而言，再单纯以其发现的Bug数量，作为衡量其绩效的唯一标准，是非常没有意义的。

一个好的测试人员跟差的测试人员相比，最终可能会发现更少的Bug。

## ****3.2过度关注测试自动化，也会适得其反****

测试自动化本身是非常好的，是值得做的一件事情。可以让我们在不需要人工干预的情况下，自动完成测试。没有自动的回归测试做保证，我们每次对产品的改动，所需的测试如果全由手工来做，根本完不成的。

其实，单元测试也是一种形式的自动化测试，这应该是每一个开发人员都必须做的事情！单元测试用例越多，覆盖的代码越多，效果越好。但更多的基于用户使用情形的自动化测试，并不总是好的。如果有一个核心测试集，能够覆盖用户使用一个产品的常用情形，会更有价值，没有必要对所有用户的使用情形都做自动化测试。

一个自动化测试用例，无论如何也不可能模仿真实的用户行为，即使你在测试中引入了一些随机因素。过度依赖自动化测试，会造成一些测试黑洞 ，有些问题只有在产品发布后才能发现。

如果自动化测试的基础架构非常脆弱、不稳定，产品的每次更改，都需要对自动化测试本身做很多的修改。测试人员花太多的时间去保证自动化本身的正常工作，而忽略了对产品进行真正的测试，那就南辕北辙了。过度关注自动化，最终会让测试人员落后于开发人员。如果开发人员所做的设计是具备可测性的，那不会有什么问题的，测试人员可以很快地据此开发出自动化测试用例来；反之，如果开发人员已经在做下一个产品特性了，而测试人员还没有开始测试，根据Scrum流程的定义，这个特性是属于“未完成”的。当测试人员发现Bug的时候，开发人没有切换回来，重新修改代码，甚至设计，如果间隔太久，这个修复效率就会很低，毕竟开发人员需要重新思考当时的设计是如何做的，代码逻辑是怎么写的。

## ****3.3并发性能测试****

并发性能测试的过程是一个负载测试和压力测试的过程，即逐渐增加负载，直到出现系统的瓶颈或者不能接收的性能点，通过综合分析交易执行指标和资源监控指标来确定系统并发性能的过程。负载测试（Load Testing）确定在不同工作负载下系统的性能，目标是测试当负载逐渐增加时，系统组成部分的相应输出项，例如通过量、响应时间、CPU负载、内存使用等来决定系统的性能。负载测试是一个分析软件应用程序和支撑架构、模拟真实环境的使用，从而来确定能够接受的性能过程。压力测试（Stress Testing）是通过确定一个系统的瓶颈或者不能接收的性能点，来获得系统所能提供的最大服务级别的测试。

测试的基本策略是自动负载测试，通过在几台高性能PC上模拟成千上万的虚拟用户同时执行业务的情景，对应用程序进行测试，同时记录下每一个事务处理的时间、中间件服务器峰值数据、数据库状态等。通过可重复的、真实的测试以度量应用的可扩展性和性能，确定问题所在及如何优化系统性能，通过了解系统的承受能力，为客户规划整个运行环境的配置提供依据。

## ****3.4自动化测试的要点****

* 无论测试人员有没有发现Bug，都不能说明他没有好好工作。测试人员的职责更多的应该是“保证质量”（Quality Assurance），而不是“控制质量”（Quality Control）。一个好的测试人员应该在问题出现之前，防止其成为Bug。
* 对于一个敏捷团队而言，单纯以测试人员发现的Bug数量作为衡量其绩效的唯一标准，是非常没有意义的。
* 如果有一个核心测试集，能够覆盖用户使用一个产品的常用情形，会更有价值。对所有用户使用情形都做自动化测试是没有必要的。
* 如果想提高测试效率，让测试敏捷起来，一定要参照科恩（Mike Cohn ）提出的“测试金字塔”。
* 测试越往下面测试的效率越高，测试质量保障程度越高。
* 测试越往下面测试的成本越低。、

