32 更高更快更强 - 谈精准测试

更新时间：2019-10-25 10:28:31

> 宝剑锋从磨砺出，梅花香自苦寒来。——佚名

上次的 chat 呢，我们聊了聊虽然让大家热血沸腾，但是很难实现的 AI 人工智能测试，这次，我们聊一聊更接地气更容易实现的“**精准测试**”。

相比较 AI 测试，精准测试这套理论已经在很多公司中落地了，在聊它之前，我想先聊聊我自己发明的一个概念：**预精准测试。**

其实所谓的预精准测试大家也在这个专栏中听了N多次了，简单一点说就是结合代码的测试。由于现在我们的互联网行业过于敏捷，迭代的非常频繁，甚至经常出现多个研发同时编写不同功能的情况，这让我们在做回归测试的时候没办法想清楚到底变动的范围有多大。全量回归吧，太耗时，万一你面对的是像淘宝、京东这样的庞然大物根本不可能实现；按照功能周边回归，又担心把圈子画小了有所遗漏，导致线上问题；有时候想着问问开发，往往开发也不太能说的清楚，那

在精准测试的概念和思想出现之前，最好的办法是什么？

最好的办法就是：我们通过比较 GIT 代码的版本变化，结合自己对于代码、业务的清晰了解，分析出所修改代码会影响的范围，然后基于这个范围来挑选相应的回归测试用例，通过或自动化测试或手工测试的方式进行回归验证。

这基本可以算是结合我们自身代码能力和业务能力最好的方式了，所以我把它叫做预精准测试。其实我们已经做了精准测试中的部分工作，但是，是人就会有疏忽，你的代码能力再强、业务再熟悉，都难免有所疏漏。所谓千里之堤溃于蚁穴，往往就是不小心的疏漏，带来了严重的生产问题。

所以我们就想，能不能通过技术的手段来解决这样的问题？

于是，精准测试的理念应运而生。概括的来说，精准测试相比较传统测试、我们刚刚的预处理测试，最大的特征就是融入了“技术”。通过技术手段来对程序进行无死角全方位的观测，将代码变更、自动化测试覆盖与代码之间关系、命中的测试用例、通过程序生成海量的原生态测试数据融入一套可持续集成的框架之中，并且将后续的自动化测试过程和结果展示出来。

这样说似乎有一点难以理解，我们来找一个比较现实的精准测试的链路来说明一下。

例如，现在有开发在 Git 上进行了代码更新，那么接下来我们一起看看优秀的精准测试下会发生什么？

STEP 1： Git 代码更新

STEP 2： 持续集成监控到代码更新，开始启动新一轮精准测试

STEP 3： DIFF 比较当前版本与上一版本代码差异，精确到代码行、方法名

STEP 4： 结合覆盖率工具，分析和计算精确到每条自动化测试用例覆盖到的代码行、方法

STEP 5： 将 3 与 4 步骤中的统计数据进行集合，确认本次版本修改命中的代码行与自动化测试用例

STEP 6： 通过代码分析及配置好的测试策略，为每个自动化测试用例生成大量测试数据

STEP 7： 自动化测试执行

我们可以看出来，之前预精准测试过程中，我们手动进行的很多操作，在这里都通过技术手段得以了实现，并且实现了完全的持续集成和自动化，对于测试人员来说，回归测试显得并不难，只需要对测试结果进行观察和分析，就可以很准确的评估当前版本的影响。

在这里我们聊几个细节：

1. **针对覆盖率的统计**：如果你用过覆盖率统计的一些工具，那你就会知道，静态的将自动化测试与代码关联是不太现实的。所以往往这时候会将历史的结果做一个留存。方式类似于：

   a. 第一次执行，进行覆盖率插桩，全量测试用例自动化，统计出当前每条测试用例覆盖到的代码行和方法，存储到本地文件或数据库。

   b. 后续每次执行都先按照数据库的结果进行统计关联，并且每次精准测试执行完成后再次统计，更新本地文件或数据库（命中执行的更新，未命中的不进行更新）。

   c. 每隔一段时间或每日闲时（凌晨），触发全量自动化执行，重新统计。

   这样，就可以保证在大多数状态下，覆盖率的统计是相对准确的。

2. **结合覆盖率的考量我们大体可以想到**：自动化测试更多以单元测试和接口测试为主，UI 自动化为辅。所以实际上精准测试更多的候应用于后端测试，针对页面的测试，精准测试未必能起到太大的作用。毕竟修改了某个页面会影响的测试用例命中很难评估。

3. **如何生成大量的合理的数据呢？**这里其实我们大约有两种方式：一是通过测试策略，对一些边界值和常用数据进行生成，但是在精准测试下，其实并不常用。第二种就是通过自建的数据工厂，对历史留存的甚至是生产上的一些数据进行清洗加工，使之成为原生态的数据，引入我们的自动化测试执行。

相信我们大家也发现了，这样的精准测试中所需要的技术沉淀、架构设计都是比较宏大的，目前也只有一些一线互联网公司才开始关注这个领域，因为他们对于回归测试、代码修改影响评估的需求最高。同时，我们也发现了，完善的精准测试体系不仅仅对于测试有着十分重要的意义，同时也对开发有着很大的帮助。比如，我可能要去修改一部分代码，但是我并不清楚修改这部分代码会影响哪些场景，那么只要我在 Git 轻轻的进行提交，通过精准测试的结果马上可以分析出相关联的功能点，也让后续开发和修改过程有了更大的底气和更全面的考虑。

除了对技术、架构的要求，精准测试的理念同样要求我们有比较完善和全面覆盖的自动化测试用

一套完整的精准测试体系，带来的效果是显而易见的。它能够最大程度的减少由于测试人员人为评估不足带来的项目风险，也能够大大减少后期回归测试过程中的成本，还大幅度的帮助我们提高代码质量，与此同时，也能让我们更加深入的去了解被测系统和架构。

不幸的是，由于精准测试的特殊性，很难有某些精准测试的体系是可以完美适应各种公司和代码场景的，所以目前也没有非常完善的精准测试开源框架。真的想要深入去做精准测试，仅仅这样一篇文章当然还是不够的，有兴趣的同学可以更多参考下腾讯的精准测试设计。也可以把技术分块分步骤的着手研究，也许下一个留言让我到他的开源框架项目中 star 的就是你呢?

← 31 黑科技：有没有一天，AI（人工智能）会取代软件测试

33 打破传统壁垒：容器化与 TestOps →

精选留言 1

欢迎在这里发表留言，作者筛选后可公开显示

zhouqiangsheng

😠🤔💪🏽

0 回复 2020-11-25

千学不如一看，千看不如一练