

本文为内部培训文档，意在从基础到深入分享单接口测试及系统接口测试内容。  
此文 借鉴和引用了多位同行的文章内容，均已标注内容来源。大家可通过链接查看相关内容原文。



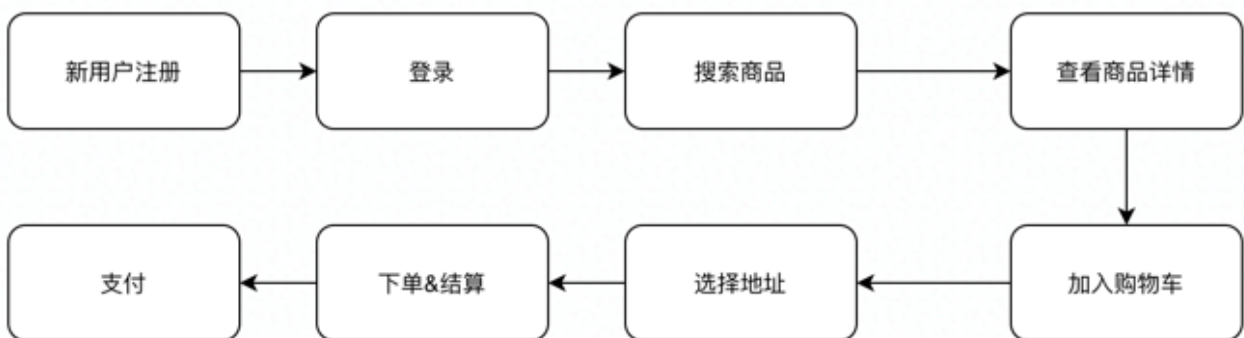
测试的第一目标是质量保障，所以我们从质量保障的纬度，来了解接口测试。

## 1、了解接口

### 1.1 接口做了哪些事？

首先，从功能角度理解。

比如，从一个用户购买一个商品的业务流程来理解：



- 新用户注册：通过**注册接口 新增**一条用户数据（Create）；



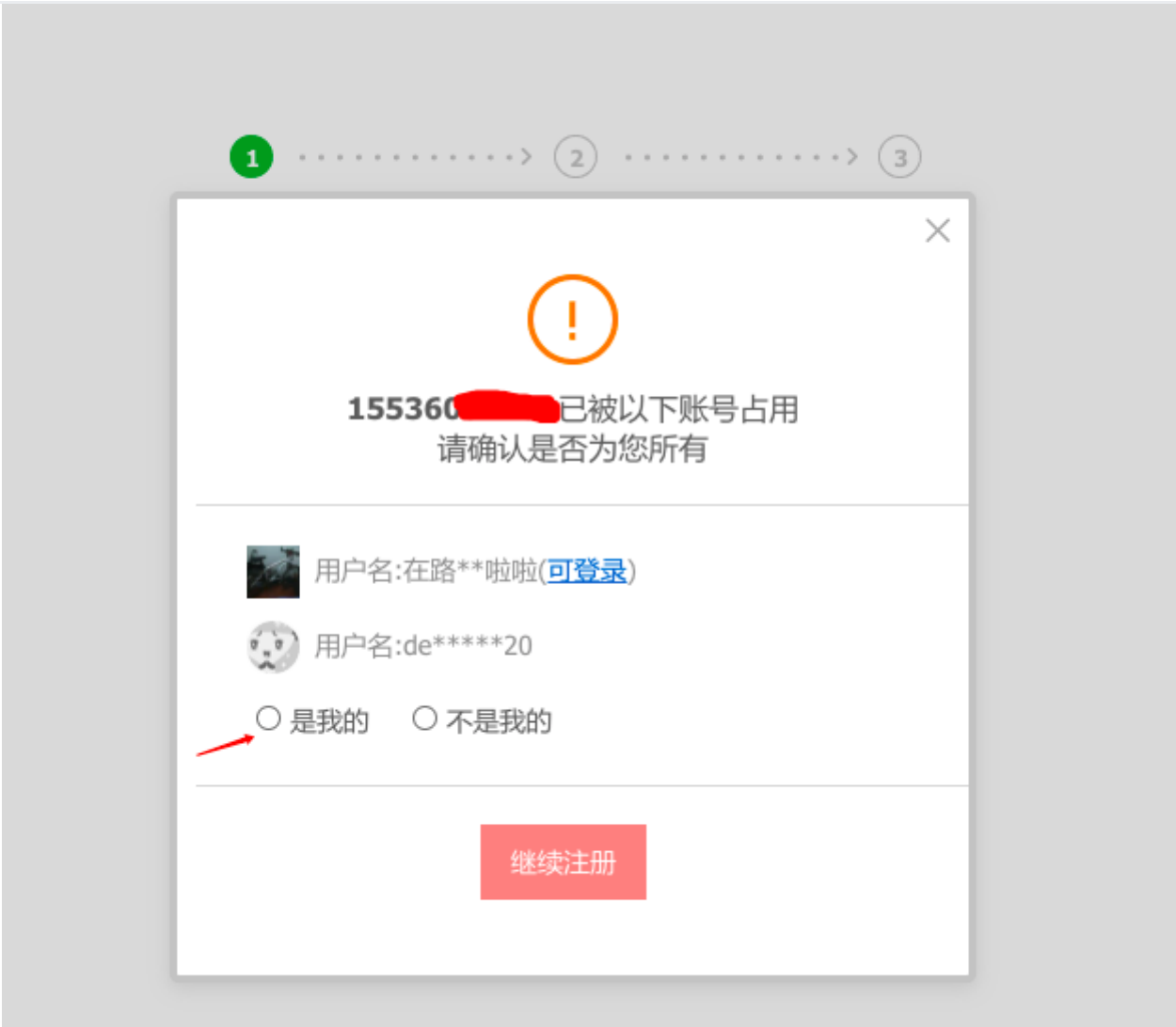
- 搜索商品：通过**商品搜索接口**搜索目标商品，本质是从商品数据库中进行**条件查询**（Select）；
- 查看商品详情：商品ID + **商品详情接口**，**查询**商品详情（Select）；
- 选择商品加入购物车：**添加购物车接口**，**更新**购物车数据，库存数据减一（Update）；
- 创建地址并选择地址：**新建地址接口**，**新增**一个新的地址（Create）；
- 下单&结算：**下单接口**，**新增**一条新的订单数据（Create）；
- 支付：**支付接口**，如果支付成功，**新增**一条支付信息，订单状态**更新**为已支付，**新增**一条电子发票，**新增**一条物流信息。

接口的功能主要是客户端和服务端的数据交互，即通过接口对后端数据的增删改查，来实现用户和产品的交互。

## 1.2 如何保障接口质量

从京东网站的注册接口来看，我们需要从哪些纬度保障质量。

### 分析注册接口



验证手机号

填写账号信息

注册成功

用 户 名

您的账户名和登录名

①

支持中文、英文、数字、“-”、“\_”的组合，4-20个字符

设 置 密 码

建议使用两种或两种以上字符组合

确 认 密 码

请再次输入密码

邮 箱 验 证

请输入邮箱

邮箱验证码

请输入邮箱验证码

获取验证码

立即注册

Http 注册接口:

▼ General

Request URL:

https://reg.jd.com/p/regSubmit?ReturnUrl=https%3A//www.jd.com/

Request Method:

POST

Status Code:

200 OK

Remote Address:

120.52.148.41:443

Referrer Policy:

strict-origin-when-cross-origin

Request Header:

1

2

3Accept: \*/\*

4Accept-Encoding: gzip, deflate, br

5Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7



```
8 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
9 Cookie: shshshfbp=sJZnAsUTcZJjuNedVMhztBA%3D%3D;
10 Host: reg.jd.com
11 Origin: https://reg.jd.com
12 Referer: https://reg.jd.com/reg/person?ReturnUrl=https%3A//www.jd.com/
13 sec-ch-ua: "Google Chrome";v="89", "Chromium";v="89", ";Not A Brand";v="99"
14 sec-ch-ua-mobile: ?0
15 Sec-Fetch-Dest: empty
16 Sec-Fetch-Mode: cors
17 Sec-Fetch-Site: same-origin
18 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 11_2_3) AppleWebKit/537.36
19 X-Requested-With: XMLHttpRequest
```

Request Body: (忽略部分无法解读参数)

```
1
2 uuid: 68455e864c894bda845de413849204d0
3 authCodeStrategy: 1 (验证码策略)
4 phone: +00861553605xxxx (手机号)
5 mobileCode: 116547 (手机验证码)
6 regName: demo83520 (注册的用户名)
7 email: xxx@163.com (注册邮箱)
8 mailCode: 661591 (邮箱验证码)
9 pwd: MvaEqtzkZ4/R4P3wMoRIuZpA4egWYBmz7bikspIwRYwozJg0HJQlQW8P0p8elFhi70Xchoz1C
10
```

接下来，从接口开发设计角度，分析注册接口。因为是纯黑盒角度，所以从UI交互和接口参数两方面分析注册接口的设计逻辑。

### UI交互角度分析

- 同样手机号、不同邮箱最多可以注册3个账号；
- 用户名：
  - 支持中文、英文、数字、“-”、“\_”的组合，4-20个字符，不能是纯数字；
  - 用户名不能重复；
- 密码：长度只能在8-20个字符之间，建议使用字母、数字和符号两种及以上组合；
- 不能频繁注册；



### 接口参数分析（取几种主要参数）

- uuid：用户唯一ID？uid生成规则？
- phone：注册手机号格式校验？
- mobileCode：手机验证码位数、字符类型校验？
- regName：用户名规则校验？
- email：注册邮箱格式校验？
- mailCode：邮箱验证码 位数、字符类型校验？
- pwd：密码加密后字符串

## # 2、如何测试接口

接口测试，主要分三步：

1. 准备测试数据（可能不需要）；
2. API测试工具，发起请求；
3. 验证返回结果的Response；

### (1) 测试数据

测试数据生成方式：

- 基于API生成数据；
- 数据库直接构造数据；
- UI操作生成数据；

生成时机：

- 实时创建：测试用例执行过程中生成（会导致测试用例执行时间变长）；
- 事先创建：测试执行前，批量生成所有测试数据（可能事先创建好的数据已经被修改而无法正常使用）；
- 测试环境不稳定，导致测试数据的顺利创建；

脏数据：

- 概念：脏数据是指，数据在被实际使用前，已经被进行了非预期的修改。



## 测试数据分类：

- “死水数据”：指相对稳定，不会在使用过程中改变状态，并且可以被多次使用的数据。这类数据适合事先创建。
  - 注意：“死水数据”是相对稳定的，是否稳定由测试目的决定。比如用户数据，在测试非用户相关测试时基本稳定，但是对于专门测试用户账号的测试用例来讲，往往涉及到用户注销之类功能，所以此时就是不稳定的。
- “活水数据”：指只能被一次性使用，或者经常会被修改的数据。例如 优惠券、订单之类的数据。

## (2) 测试用例设计

用例设计从两个维度来设计，功能性需求和非功能性需求。

### 功能性需求

用例设计法参考：[软件测试基础---流程和用例设计方法-piecesof](#)



### 非功能性需求-安全纬度

- 敏感信息是否加密：密码前后端传输是否加密
- sql注入？（结合用户的输入数据动态构造 Sql 语句，如果用户输入的数据被构造成恶意 Sql 代码，Web 应用又未对动态构造的 Sql 语句使用的参数进行审查，则会带来意想不到的危险。）
- 逻辑漏洞：
  - 批量注册重复消费问题？（比如，同一时间，同样的参数，高并发请求，是否只有一个 Http 请求注册成功）
  - 同一手机号，不同邮箱，是否可以注册超过3个用户？

### 非功能性需求-性能纬度

- 基准性能是否达标？比如单请求要求小于500ms？
- 高并发性能评估：通过性能测试手段，评估注册接口性能。具体查看：[服务端性能测试-入门指南](#) 和 [服务端性能测试-工具篇](#)



非功能性需求纬度-性能纬度：2条用例。

### (3) 如何做接口断言？

- Http Response断言：
  - Http 状态码
  - Response Body 字段、结构 校验
  - Response Header
- 数据断言：
  - 对数据库中的数据断言
- 响应时间满足要求吗？

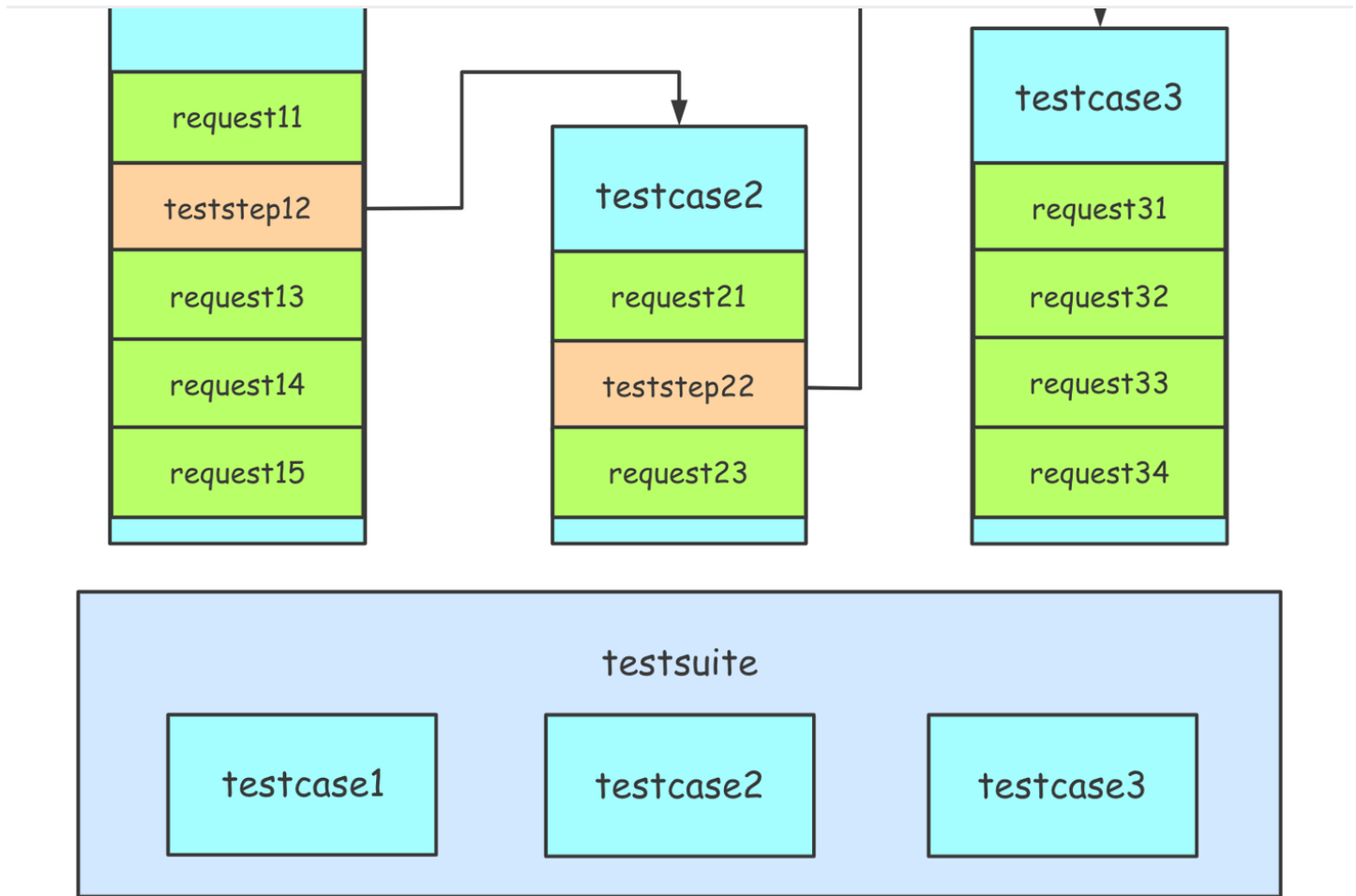
## 3、如何用接口测试一个系统？

### (1) 复杂系统测试用例结构

参考：[HttpRunner之step/case/suite](#)

测试步骤(testStep) -> 测试用例(testCase) -> 测试场景/测试用例集(testSuite)





测试步骤(testStep)

对于接口测试来说，每一个测试步骤应该就对应一个 API 的请求描述。

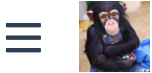
测试用例(testCase)

测试用例（testcase）应该是为了测试某个特定的功能逻辑而精心设计的，并且至少包含如下几点：

- 明确的测试目的
- 明确的输入
- 明确的运行环境
- 明确的测试步骤描述
- 明确的预期结果

测试用例设计原则：

- 测试用例应该是完整且独立的，每条测试用例都应该可以独立运行；
- 测试用例由测试脚本和测试数据两部分构成。



- 测试数据：是对应测试的业务数据。
- 测试数据和测试脚本分离：方便实现数据驱动测试。通过对测试脚本传入一组数据，实现同一业务功能在不同数据逻辑下的测试验证。比如：购买商品接口，会员和非会员的商品价格是不一样的，优惠券逻辑也不一样。所以通过不同的用户数据，可以测试会员和非会员购物逻辑。

## 测试用例集(testSuite)

测试用例集是测试用例的无序集合，集合中的测试用例应该互相独立，不存在先后依赖关系。

如果确实存在先后依赖关系，例如登录功能和下单功能。正确的做法应该是，在下单测试用例的前置步骤中执行登录操作。

## (2) 测试数据管理

来源：孙高飞-测试框架中的数据管理策略

数据的两个属性：

- 作用域：共享数据（适用于testSuite级别）、隔离数据（适用于testCase级别）
- 创建方式：调用开发接口、使用Sql、独立开发数据模版

### 测试数据的作用域

**共享数据：**所有case或一部分case共同使用的测试数据

- 优点：速度快，数据只需要创建一次就可以给很多 case 使用。
- 缺点：
  - 数据为很多 case 准备的，你很难分清哪些数据是给这个 case 准备的，哪些数据是给另一个 case 准备的。case 的可读性低
  - case 之间互相有影响。因为待测的功能本身就会对数据库造成影响。很可能一个 case 的失败或者成功就会造成一批 case 的 fail
  - 数据本身不能扩展，稍微一改动，影响就很广泛。数个甚至数十个 case 的失败是很常见的。维护脚本的成本比较高

**隔离数据：**每个case都有独享测试数据，case之间互不影响，即每个case都做setup和teardown的操作。case执行前创造数据，执行后销毁数据。



- 缺点：速度慢。。。灰常慢。。。因为每个 case 都有很多的磁盘 IO 操作。。。维护数据的时间比调用功能的时间都要长的情况并不奇怪。OK，这种方式其实是在测试中运用的最多的方式。虽然它很慢，而且对很多人来说实现起来也比较难。但是它带来的可维护性实在太诱人。我再也不用整天维护那些不稳定的脚本了。慢点就慢点吧。反正我们做接口测试和 UI 自动化测试的持续集成策略也是定时运行的。跑个 10 几分钟，几十分钟的我也不在乎。只要不是做监控代码变动的策略，一切好说。

**敏感数据：账号、密码、key等敏感信息，设置为有权限限制的环境变量。**

- 敏感信息不能公开的主要原因：
  - 加强权限管控：参与项目的开发人员可能会有很多，大家都有读取代码仓库的权限，但是像 key 这类极度敏感的信息不应该所有人都有权限获取；
  - 减少代码泄漏的危害性：假如代码出现泄漏，敏感数据信息不应该也同时泄漏。
- 推荐的解决方案：
  - 对服务器进行权限管控，只有运维人员（或者核心开发人员）才有登录服务器的权限；
  - 运维人员（或者核心开发人员）：在运行的机器上将敏感数据设置到系统的环境变量中；
  - 普通开发人员：只需要知道敏感信息的变量名称，在代码中通过读取环境变量的方式获取敏感数据。

## 如何构造数据

### 调用开发接口

- 优点：在脚本中实现起来相对简单，不用深入理解后台数据库。
- 缺点：
  - 耦合性太高，依赖产品的其他接口创造数据的方式注定了 case 是非隔离性的。注意隔离性是 case 质量的一个重要指标。一旦创造数据的接口 bug 了，你说得有多少个 case 失败。而且在真实环境中，需要调用 N 个接口去创造你需要的数据。无法判断到底哪个接口的 bug。这已然变成了端到端测试了。能够快速定位 bug 位置，也同样是 case 质量的重要指标。
  - 如果做隔离数据，产品中的接口往往很难满足你销毁数据的需要，举个最常见的例子。这个世界存在一种删除机制叫做逻辑删除，也就是产品的接口并不是真正的删除了数据库中的数据，而是用一个逻辑标示位，标示这条数据被删除了。不要再反馈给用户了。这样其实就做不到“隔离数据了”
- 使用建议：不建议使用。虽然该方式脚本维护成本低，但是造成用例耦合度高、隔离性差，问题定位成本高。一个被调用开发接口的BUG，可能会导致大量用例失败。



**直接使用 sql：就是直接写 sql 创造和销毁数据。**

- 优点：隔离性和bug 追踪都很好。
- 缺点：如果交给测试人员在脚本中写 sql 的话，难度，可读性都不太乐观，而且太依赖测试人员本身的能力，出错率较高。不过好在我们可以测试框架上做一些手脚，解决这个问题。
- 使用建议：除查询sql外，增删改sql 慎重使用，因为实现成本高、操作风险高。需要非常了解数据库表结构和业务逻辑，删改数据很可能影响实际业务或其他同学测试。

**数据模版：为核心业务测试数据，创建独立的数据模版，专人独立维护。**

参考：测试能效平台的诞生-国际化商城智能物料平台 · [TesterHome](#)

- 实现思路：
  - 对于数据构造较复杂，且数据关联业务多、异常数据风险高的数据（比如电商的物料数据），建议基于开发接口封装通用函数，且针对性的做好异常处理和定位。
- 优点：
  - 专人开发维护，极大降低构建复杂数据成本和风险
  - 降低功能测试构造数据精力，突破相关测试人力占用瓶颈。
- 缺点：开发成本高，使用重量级业务系统

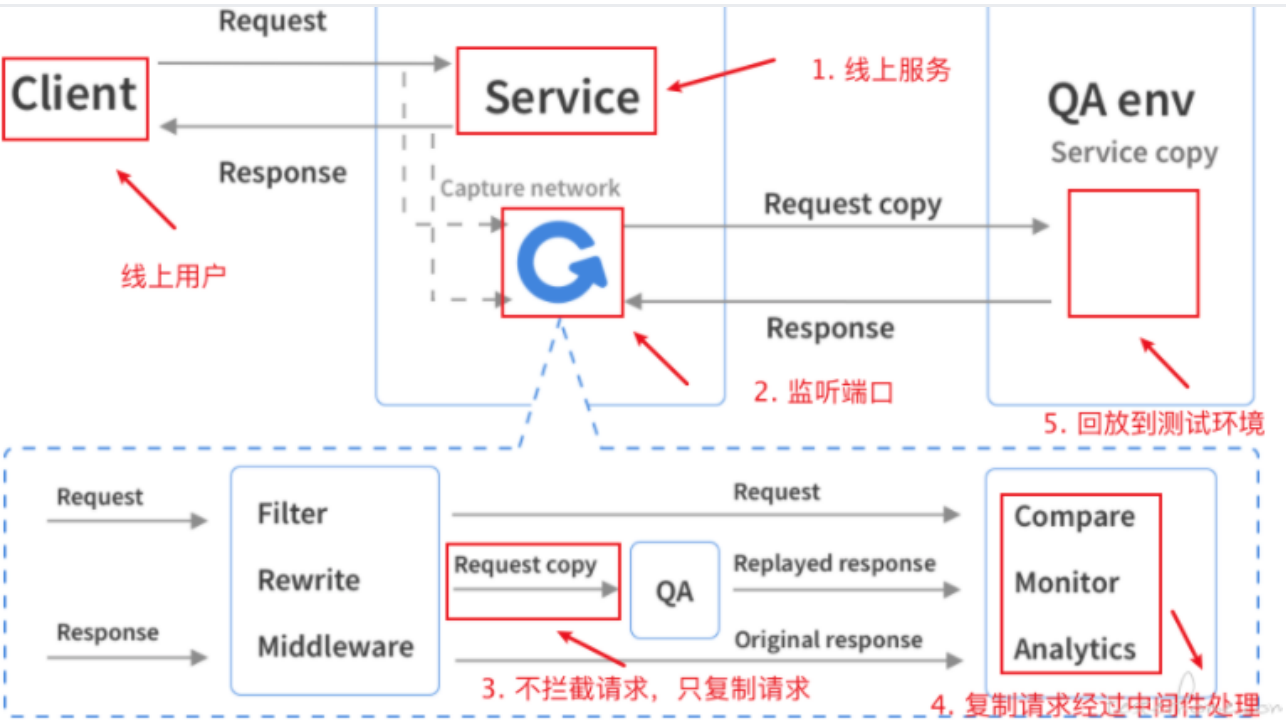
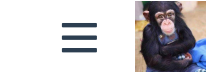
## 4、接口测试进化

回顾一下前面接口测试的内容，发现几个问题：

- 复杂系统动则几千个接口，回归测试工作量大；
- 用例编写成本高，参数多的接口

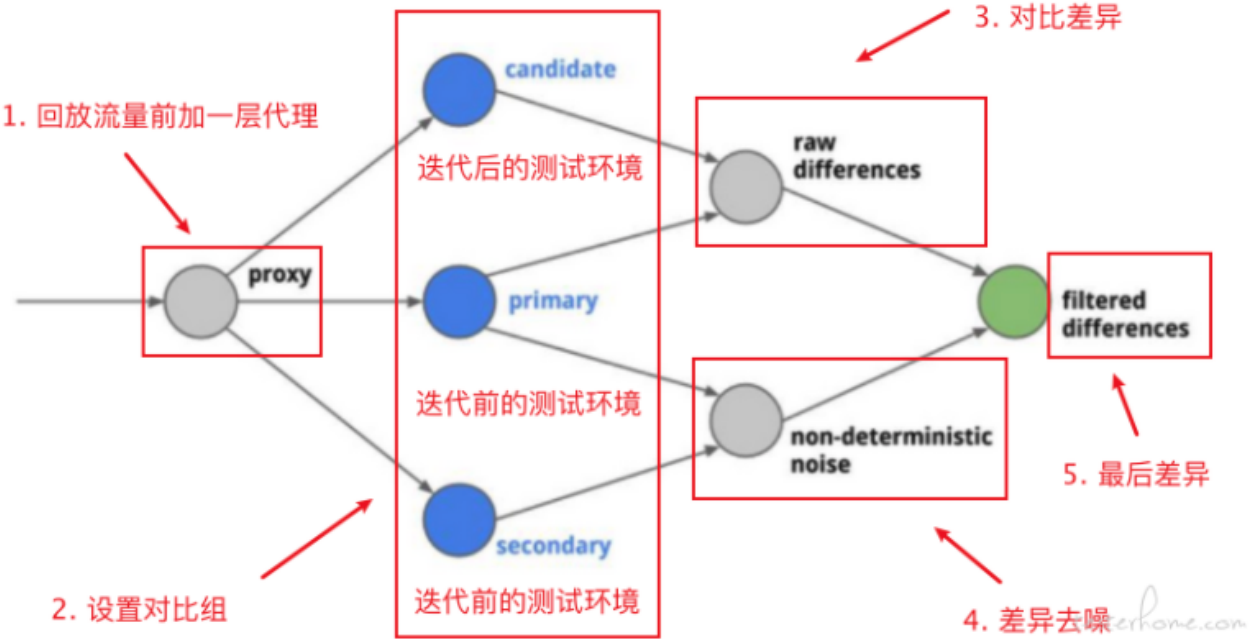
### (1) 回归测试工作量大？录制线上流量回归

参考：录制线上流量做回归测试的正确打开方式 · [TesterHome](#)



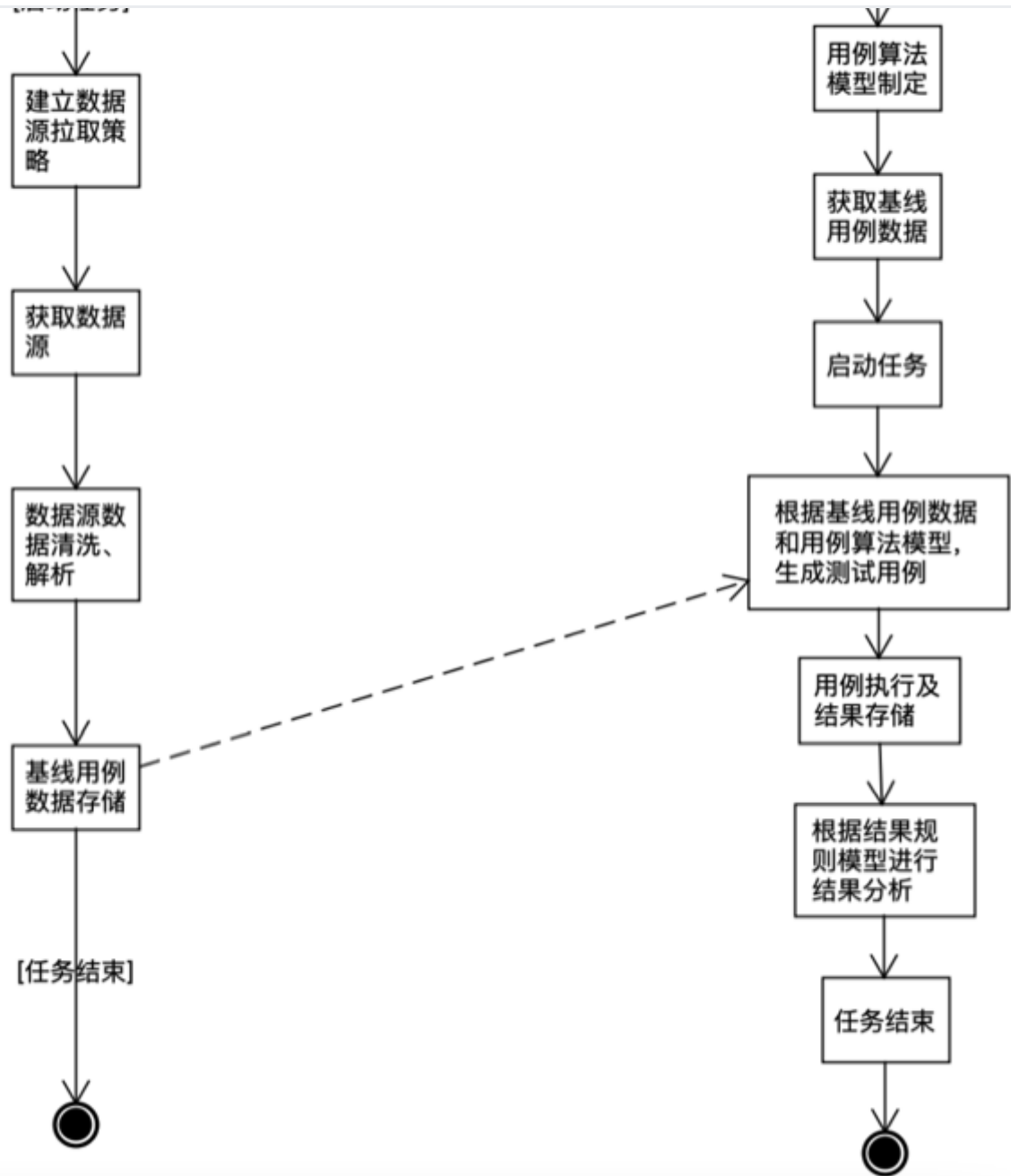
回放差异 diff

diff 实现对比和去噪



(2) 用例编写成本高？通用接口自动测试方案

参考：通用性接口健壮性扫描方案-有赞



### (3) 快速校验接口数据结构变化？ 接口自动化全量字段校验

来源：接口自动化全量字段校验 · TesterHome

实现：自定义接口返回数据格式（【契约定义】）-实际响应数据格式校验（【契约校验】）的功能

校验原则：

- 实际返回字段名要严格等于或者含契约定义字段名(根据不同匹配模式来确定)



服务端性能测试入门指南 →

评论

Powered by **GitHub** & **Vssue**



登录后才能发表评论 | 支持 Markdown 语法

使用 GitHub 帐号登录后发表评论

使用 **GitHub** 登录

登录后查看评论

昵称

邮箱



网址(http://)

Just Go Go



提交


11 评论



2022-03-29

回复

赞一个




在路上

Chrome 99.0.4844.84    macOS 10.15.7

2022-03-29

@防水材料加盟

, 这个广告可以




Anonymous

Chrome 97.0.4692.71    macOS 10.15.7

2022-01-17

回复

hhhhh




Anonymous

Chrome 97.0.4692.71    macOS 10.15.7

2022-01-17

@Anonymous

, dddd 🤔🤔🤔




Anonymous

Chrome 96.0.4664.45    Linux

2021-12-29

回复

hi boy



在路上


Chrome 96.0.4664.110    macOS 10.15.7

2022-01-04

回复

@Anonymous

, Hi Girl




Anonymous

Chrome 96.0.4664.110    macOS 10.15.7

2021-12-24

回复

订阅新能源车使用体验 🤔



Anonymous

Chrome 96.0.4664.110    macOS 10.15.7

2021-12-24

回复

https://ontheway.cool/skills/server/interface-test-guide.html#\_1-2-如何保障接口质量

16/18





如果不开空调，续航可以达到90%



在路上

Chrome 96.0.4664.110 macOS 10.15.7

2021-12-24

@Anonymous，可以找车试驾一下，我试驾了好猫和秦Plus，果断买了秦。

特斯拉驾驶体验应该是最爽的



Anonymous

Chrome 95.0.4638.69 Windows 10.0

2021-11-26

回复

想起了张敬轩的on my way 😁



在路上

Edge 95.0.1020.53 macOS 10.15.7

2021-11-17

回复

欢迎大家留言 😊😊



Anonymous

Chrome 95.0.4638.69 macOS 10.15.7

2021-11-15

回复

太强了



Anonymous

Chrome 92.0.4515.159 Windows 8.1

2021-09-10

回复

为什么找不到您之前写的财务自由的文章了,是删了吗,还想推荐朋友看



在路上

Edge 93.0.961.38 macOS 10.15.7

2021-09-13

回复

@Anonymous，最近没时间弄，过一段时间更新出来哈。



神乐

Edge 111.0.1661.44 Windows 10.0

2023-03-24

回复



**Anonymous** Chrome 92.0.4515.159 macOS 10.15.7  
2021-08-21

回复



**在路上** Edge 91.0.864.70 macOS 10.15.7  
2021-07-21

回复



加载更多...