《23\_微服务化的C2C电商社会化治理平台如何进行全链路测试？》

nacos + dubbo整合，原理，以及案例代码的开发，大家都已经有了一定的了解了，手把手带着大家去做了，spring cloud alibaba去开发自己公司的项目的话，整合框架，背后的原理，设计、开发、测试、上线

（1）系统设计（概要设计、详细设计 / 简化设计）

概要设计：业务架构、技术架构、业务流程、技术方案、部署方案，都需要做一个大致上的设计，架构师

详细设计：接口定义、数据库表结构、核心类建模、各个接口被请求时的系统运行时序图、技术方案细化

简化设计：画一些系统运行流程图、技术方案、接口、表、时序图

设计完毕之后，会有一个设计评审的过程，会找相关的其他同学过来评审，比如说给人家确认一下你设计的接口，是否满足你的调用方的需求

（2）开发代码

每个人可能都是维护自己负责的子系统、服务，微服务框架，spring cloud alibaba里面的nacos + dubbo，用dubbo定义各种你需要对外提供的接口，按照你自己的设计文档以及技术方案去进行代码的开发

如果仅仅是一些crud的话，此时基于数据库就可以搞定了

但是如果说涉及到一些复杂的技术方案，使用中间件系统，es、redis、mongodb、hbase、rocketmq，等等

（3）本地自测

服务本地跑起来自己测试各个功能，直接通过dubbo服务调用，浏览器的http请求，直接请求你的接口，测试一下自己的各个功能，你自己一个人维护一个java web系统，不依赖别人的接口

也可能跑不起来，那就单元测试，单元测试其实是一个很专业的领域，跑本地单元测试的时候，需要把你的spring容器跑起来，然后对各种bean的注入可能需要打桩，接着再测试各个接口，这里不详细展开了

说句题外话，国内很规范做单元测试的公司也不多，大多公司的单元测试做的第一不规范，第二不完善，第三形同虚设，所以基本可以忽略，如果要把单测做好了，写单测的代码的时间跟你写系统代码的时间可能甚至是1:1，1:2

更多的还是写完代码，自己本地跑起来，想办法简单测试一下罢了

有的时候跑起来需要有一些其他人负责的服务的配合，这个时候有一些方案可以做到，比如说本地可以跑起来一个服务注册中心，然后其他人的服务你手头没有，那团队可以做一个统一的本地服务模拟工程，工程跑起来就自动往本地注册中心去注册，接口的返回结果都是mock的

然后你的服务跑起来，就可以跑通了，包括数据库，缓存，MQ这些东西，都可以在本地部署，有一个本地的maven profile，放这些配置

小项目，协作方不是太多

或者是在公司内网环境里，提供几台机器，作为dev环境，部署了数据库、缓存、MQ等中间件，服务可以部署，一台机器可以多部署几个服务，然后当你笔记本电脑在公司内网的时候，就可以访问到那几台机器，那么你本地启动，就可以直接访问到测试环境里的其他服务了

maven profile，spring boot profile，百度搜一下，非常的简单，都是很对不同的环境可以去放一套不同的配置资源文件

（4）持续集成：可选

很多同学可能都听说过持续集成，但是不太清楚到底是什么

有的公司会做持续集成，意思是你每天开发好的代码和功能，都必须有对应的单元测试可以进行自动化的测试，然后你本地单元测试跑通了，就可以提交代码到git仓库以及进行代码合并，直接触发jenkins这种集成工具，他会拉取你的最新代码，然后跑你所有的单元测试，确保你的代码在所有测试下可以正常运行

甚至可能是多个人维护一个系统，每个人每天/隔天，都要提交自己的代码+单测到git仓库进行代码合并，集成的概念，单人维护一个独立的工程/服务，每天不停的提交最新代码到你的git仓库里，你在不停的把自己最新写好的代码集成到已有的完整的代码里去

持续集成，提代码

多人维护一个系统，你本地写好的代码，本地跑单测是ok的，但是你提到git仓库合并进去，此时可能别人也会提代码合并进去，此时你们俩的代码集成在一起了，此时到底集成好的代码能不能正常工作呢？

jenkins持续集成的工具，如果发现你有提交代码以及合并的操作，此时jenkins会触发一个任务，拉取你的代码到本地，自动运行所有的单元测试，用你的单元测试自动运行和检查，来确保你现在最新的集成后的代码都是可以运行的

有的时候还会专门写特定的自动化集成测试代码，就是说你代码提交之后，然后可能会把你完整代码跑起来，此时所有代码是一个集成在一起的状态，接着就是运行集成测试的代码，把你集成在一起的代码都跑一下，检查是否正常

但是这个比较麻烦，搞持续集成，在工具上要求git、jenkins之类的整合，你要做很多配置，同时要求你每天的代码都有对应的自动化测试代码，所以真的把持续集成做好，做的很标准的公司，其实不多

（5）联调测试/功能测试

一个人维护一个java web系统，对其他人没有依赖，太low了

比较正常的，就是你写好了代码，自己简单自测完毕了，然后部署到一个联调测试/功能测试的环境里去，这个环境，是可能团队内部各个服务之间联调，或者甚至和其他团队的系统进行联调的地方

这个环境最好是独立的一套环境，部署好之后，QA会进行大量的手工测试，各种点击系统，也可能会有自动化测试，不过说实话，能玩儿自动化测试的公司不多，最后在这个环境，会有一个PM功能验收测试的过程

这个环境重在联调，把各个系统和服务跑通，确保功能运行没问题，一般机器都是低配置的，而且一个服务就一台机器，甚至是一台机器几个服务都有可能

（6）预发布测试

接着一般是预发布环境的测试，这个环境一般是模拟线上环境，可能会在这里做压力测试、全链路压测、性能测试、可用性测试、稳定性测试，都可能会在这里做，一般机器配置跟线上一样，每个服务都是独立机器，甚至每个服务给多台机器

比如说线上高峰QPS是1w+，线上机器是4核8G的，部署20台，那么预发布环境可能就是模拟每秒QPS是1000+，每个服务部署2台机器，做一个低压力测试，把全链路都压一下，测试性能，QPS，机器负载

有时候也可能会跑一些可用性测试，比如设计一些特殊故障之类的

在这个环境，通常流量是从线上获取回放过来的，有一个线上流量回放的过程，很多公司其实没这个环节，此时可能也就是走个过场，但是正经来说，是要做流量回放的，不是靠人力来测试，而是回放线上流量，看系统的功能是否正常，压力是否ok，性能是否还行，机器负载如何，全链路表现如何

有时候也会在这个环境让QA做一个全量功能回归测试，这可能是大版本变动的时候要做的

如果一切正常，那么就可以上线了

（7）线上部署

生产环境必须是一套独立的机器，直接进行部署即可，上线之后要通过各个机器的重要日志、请求是否正常、机器负载等是否正常、然后PM线上验收，一切正常，上线成功