《26\_互联网公司部署方案：蓝绿部署、灰度发布以及滚动发布》

设计、开发、测试、部署，流程都讲过了，微服务技术栈，服务注册中心，nacos，RPC框架，dubbo，设计就要把各个服务拆分完毕，包括你的业务逻辑，需求，接口，数据库，类，功能的时序图

每个人就负责开发自己的服务就可以了，nacos+dubbo

用dubbo开发一些接口，只要定义一些接口和dubbo注解，更多的还是写java代码

不同的环境之下，你的服务注册的namespace必须是不同的

（1）滚动发布

这是最常见的部署模式，一般就是说你一个服务/系统都会部署在多台机器上，部署的时候，要不然是手动依次部署，最low的比如就是每台服务器上放一个tomcat，每台机器依次停机tomcat，然后把新的代码放进去，再重新启动tomcat，各个机器逐渐重启，这就是最low的滚动发布

中小型公司现在稍微好点的话，都会做自动化部署，自动化部署用的比较多的是jenkins，因为jenkins是支持持续集成和持续交付的，之前说过持续集成，那么持续交付就是比持续集成更进一步，简单来说，就是你每天都提交代码，他每天都自动跑测试确保代码集成没问题，然后可能每隔几天，就把一个生产可用的小版本交付到线上

这个时候，就需要一个自动化部署，jenkins可以自动在多台机器上部署你的服务/系统，过程其实也是类似的，只不过把手动改成自动罢了，你自己部署tomcat/基于spring boot内嵌容器，其实都行

中大型公司，一般发布系统都是自己研发的，你在上面指定对一个服务，指定一个git仓库的代码分支，然后指定一个环境，指定一批机器，发布系统自动到git仓库拉取代码到本地，编译打包，然后在你指定环境的机器上，依次停止当前运行的进程，然后依次重启你新代码的服务进程

这都是典型的滚动发布

但凡发布，都要考虑两个问题，一个是验证，一个是回滚

验证就是说，你怎么确定你这次部署成功了？一般来说，要观察每台机器启动后处理请求时的日志，日志是否正常，是否有报错，一般日志正常、没有报错，那么就算是启动成功了，有时候也会让QA/PM做一个线上验证

那么万一发布失败了呢？此时就得回滚，因为不同的上线是不一样的，有时候你仅仅是对代码做一些微调，大多数时候是针对新需求有上线，加了新的代码/接口，有时候是架构重构，实现机制和技术架构都变了

所以回滚的话，也不太一样，比如你如果是加了一些新的接口，结果上线失败了，此时心接口没人访问，直接代码回滚到旧版本重新部署就行了；如果你是做技术架构升级，此时失败了，可能很多请求已经处理失败，数据丢失，严重的时候会导致公司丢失订单，或者是数据写入了但是都错了

此时可能会采用回滚代码，或者清洗错乱数据的方式来回滚，总之，针对你的发布，你要考虑到失败之后的回滚方案，回滚代码，就得用旧版本的代码，然后重新在各个机器依次部署，就算是一次回滚了，至于丢失了数据没有，要不要清洗数据，这个看情况来定

滚动发布的话，风险还是比较大的，因为一旦你用了自动化的滚动发布，那么发布系统会自动把你的所有机器都部署新版本的代码，这个时候中间很有可能会出现问题，导致大规模的异常和损失

所以现在一般中大型公司，都不会贸然用滚动发布模式了

（2）灰度发布

灰度发布，指的就是说，不要上线就滚动全部发布到所有机器，一般就是会部署在比如1台机器上，采用新版本，然后切比如10%的流量过去，观察那10%的流量在1台机器上运行一段时间，比如运行个几天时间，观察日志、异常、数据，是否一切正常，如果验证发现全部正常，那么此时就可以全量发布了

全量发布的时候，就是采用滚动发布那种模式

这个好处就是说，你先用10%以内的小流量放到灰度新版本的那台机器上验证一段时间，感觉没问题了，才会全量部署，这么操作，即使有问题，也就10%以内的请求出现问题，损失不会太大的，如果你公司体量特别大，灰度也可以是1%，甚至0.1%的流量

因为体量太大的公司，1%的流量就很大了

如果灰度的时候有问题，那么立刻把10%以内的小流量切去请求老版本代码部署的机器，灰度版本的机器立马就没流量请求了，这个回滚速度是极快的

通常灰度验证过后，全量发布，都不会有太大的问题，基本上再出问题概率就很小了，所以现在中大型互联网公司，一般都是灰度发布模式的

（3）蓝绿部署

蓝绿部署的意思是说，你得同时准备两个集群，一个集群放新版本代码，一个集群放老版本代码，然后新版本代码的集群准备好了过后，直接线上流量切到新版本集群上去，跑一段时间来验证，如果发现有问题，回滚就是立马把流量切回老版本集群，回滚是很快速的

如果新版本集群运行一段时间感觉没问题了，此时就可以把老版本集群给下线了

那么为什么有灰度发布了还要用蓝绿部署呢？

是这样的，灰度发布过后，还是要全量部署的，但是有时候，如果涉及到一些新的架构方案，或者是新的接口，10%以内的小流量可能没办法暴露出线上的高并发问题，所以灰度验证没问题，结果全量部署还是有一个小概率会失败

此时全量发布用滚动发布的方式，逐步部署过去，很快会引发大规模的失败，此时回滚，是很慢的，因为要一台一台逐步回滚

所以说，一般针对那种改动不太大的小版本，比如加个接口，修改一些代码，修复几个bug，类似这种整体变动不大的情况，建议用灰度发布，因为这种一般灰度验证没问题，全量部署也不会有问题

但是如果涉及到那种很大规模的架构重构或者架构升级，比如数据存储架构升级，或者是技术架构整体改造，或者是代码大规模重构，类似这种场景，最好是用蓝绿部署，也就是说，完全部署一个新的集群，然后把较大的流量切过去，比如先切10%，再切50%，最后切到100%，让新集群承载100%的流量跑一段时间

过程中一旦有问题，立马流量全部切回老集群，这个回滚速度就比灰度发布的全量部署回滚要快多了，因为仅仅是切流量而已，不需要重新部署