Dubbo与SpringCloud的区别

1、dubbo由于是二进制的传输，占用带宽会更少

2、SpringCloud是http协议传输，带宽会比较多，同时使用http协议一般会使用json保温，消耗会更大

3、dubbo的开发难度较大，原因是dubbo的jar包依赖问题很多大型工程无法解决

4、SpringCloud的接口协议约定比较自由且松散，需要有强有力的行政措施来限制接口无序升级

5、dubbo的注册中心可以选择zookeeper，redis等多种，springcloud的注册中心职能用eureka或者自研

但如果选择，我会用springcloud。

从公司整体规划：我不会选择很久没人维护的dubbo，重启之后也未必是原班人马

从程序员招聘难度：招springcloud的程序员会更好招，因为更新更炫

从系统结构简易程序：springcloud的系统结构更简单、“注册+springmvc=springcloud”，而dubbo各种复杂的url，protocol，register，invocation，dubbofilter，dubboSPI，dubbo序列化......炫技的成分更多一些

从性能：dubbo的网络消耗小于springcloud，但是在国内95%的公司内，网络消耗不是什么太大问题，如果真的成了问题，通过压缩、二进制、高速缓存、分段降级等方法，很容易解决

从开发难易度：dubbo的神坑是jar包依赖，开发阶段难度极大。springcloud比较自由，但带来的问题是无法“强力约束接口规范”，建议用行政方式解决，且我们团队的强力行政约束做的还是比较好的，在接口管控层面比较强效，一个没有行政组织能力的IT团队真的是废渣，用什么框架都不好使。

从后续改进：dubbo的改进是通过dubbofilter，很多东西没有，需要自己继承，如监控，如日志，如限流，如追踪。springcloud自己带了很多监控、限流措施，但是功能可能和欧美习惯相同，国内需要进行适当改造，但更简单，就是ServletFilter而已，但是总归比dubbo多一些东西是好的

从配套措施：springcloud一直宣称自己是“一套全方面的解决方案”。我起初信了，后来发现上当了，实话说，有，但是不是很健全，很多需要自行改造。而dubbo相反，一直宣称自己是“一套全方面的服务化方案”，纯服务化顶个鸟用，任何系统都是相辅相成配套的，一个完整的系统，要有前台、中台、后台、前台包括前端和交互，中台包括交易、任务、数据，后台包括财务、账户、管理......单纯的服务化解决不了“任何问题”，唯有体系才能解决。在这个层面，springcloud是个往“体系”方向发展的方案，而dubbo仅是个工具而已，两者相比，就好比始祖鸟与草履虫的区别。

从技术实力层面：对比双方的源码，不得不说dubbo作者的技术能力要高于springcloud，而springboot的作者技术能力要高于dubbo。即，springboot>dubbo>springcloud。我喜欢springboot这种大道至简的风格，keep it simple and stupid。而dubbo，先看看dubboSPI，再看看Protocol$Adpative里面那一群绕来绕去的瞎几把绕的玩意儿，你会迅速判断出：这群屌丝在炫技。尽管dubbo从上之下分为十层四五十个组件，第一感官上是哇塞好全面好伟大的样子，但深入之后你会觉得，这技术是很炫，设计的确实很全面，但是用不到，例如：在zookeeper地址中，使用逗号分隔和分好分隔地址的区别。。。用途不大，但是代码里面为了这个就走向了完全不同的一条分支。用俗语平均，就是“臃肿且无用代码多，在文档里还非得为了这个说出123456来”。说完dubbo再说springcloud......他没有技术含量，完全没有，就是一堆简单组件拼装在一起，如configserver、eurekaserver、robin、feignClient、hystrix等，每个特别简单，没什么技术含量。

最后说springboot，我要用“纯粹”两个字类评价这个框架，真的很纯粹，并且从其代码架构的总体思路一致性，目标的纯粹性，具体模块的干净利落，确实是个好框架，值得大家一度。