链滴

dubbo服务化设计原则

作者：hzshouchen
原文链接：https://ld246.com/article/1467115435771
来源网站：链滴

服务化是一个抽取基础业务的过程，基础服务为上游业务提供支撑，但基础服务是无状态的，有类似于实用服务，但又不是实用服务，基础服务还是与业务存在关联的，不同项目的基础服务一定是同的，但是实用服务却是可以复用的。基础服务可以随着系统的符合灵活伸缩来提供服务。
<h2>服务化子模块设计</h2>
将一个大的系统拆分成多个子模块是服务化的第一步。划分子模块就会涉及到一个粒度的问题。

划分的过细会破坏业务的独立性，许多模块之间会相互调用，而且部署和运维工作量也大，独立程占用的内存较多；
划分的过粗又会导致模块之间不能很好的解耦，开发和维护工作都不好分工，升级维护涉及面太。

这里具体要怎么划分还是需要看具体的业务以及项目所考虑的侧重点。但是一下几点还是需要注一下的：

各个服务之间尽量不要存在涉及到数据库层面的耦合（表查询）
避免服务之间出现循环依赖调用的情况（A——>B——>C‐—>A）
服务之间的依赖关系不要过长（最好不要超过三个）
尽量避免分布式事务（尽量把能做的事情在一个服务中做完）

<h2>服务化接口设计</h2>

controller——facade——service——dao
服务接口尽可能大粒度，每个服务方法都应该代表一个功能，而不只是一个步骤
服务接口建议以业务场景为单位划分，并对相近的业务做抽象，防止出现接口数量暴增（接口爆）
不建议使用过于抽象的接口，如Map query(Map)，不利于后期的维护
每个接口都应该定义版本号，为后续的兼容升级提供可能
接口兼容性:服务接口增加方法，或服务模型增加字段，可向后兼容；删除方法或删除字，将不兼容，枚举类型新增字段也不兼容，需要通过变更版本号升级。
异常处理:建议使用异常汇报错误，而不是返回错误码，异常信息能携更多的信息，以及语义更友好。
在Provider上尽量多配置Consumer端属性:如调用的超时时间，合理重试次数，并发控制数量，负载均衡 ，等等
