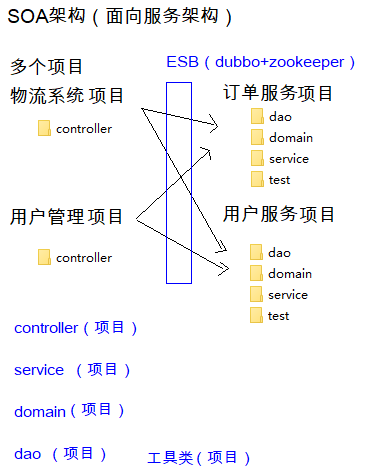
# 11\_09中午演讲

## 请介绍你理解的SOA架构

面向服务的架构（SOA）是一个组件模型，它将应用程序的不同功能单元（称为服务）进行拆分，通过这些服务之间定义良好的接口和契约联系起来。

接口是采用中立的方式进行定义的，它应该独立于实现服务的 硬件平台、操作系统和编程语言。

这使得构建在各种这样的系统中的服务可以以一种统一和通用的方式进行交互



**具体到项目中怎么用:**

## 请介绍你理解的微服务架构

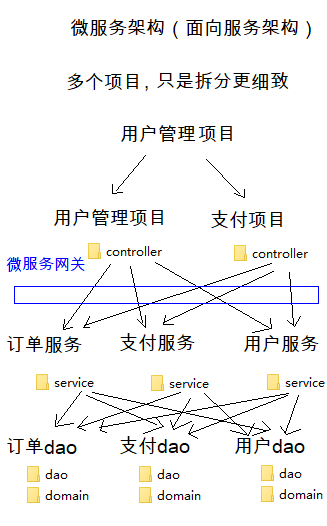
1.微服务把每一个职责单一的功能放在一个独立的服务中 。

2.每个服务有多个实例在运行，每个实例可以运行在容器化平台内，达到平滑伸缩的效 果，单个微服务启动较快。

3.每个服务应该有自己的运营平台，以及独享的运营人员，这包括技术运维和业务运营人 员：每个服务都高度自治，内部的变化对外透明。

4.易于开发和维护：一个微服务只会关注一个特定的业务功能，所以业务清晰、代码量较少。开发和维护单个微服务相对简单。

5.局部修改容易部署：单体应用只要有修改，就得重新部署整个应用。微服务解决了这样的问题。一般来说，对某个微服务进行修改，只需要重新部署这个服务即可。

6.技术栈不受限制：在微服务架构中，可以结合项目业务及团队的特点，合理的选择技术栈

## 请介绍dubbo的架构执行流程

**先说清分4部分,再介绍之间的关系**

0 start: 启动Spring容器时,自动启动Dubbo的Provider

1、register: Dubbo的Provider在启动后自动会去注册中心注册内容.注册的内容包括:

1.1 Provider的 IP

1.2 Provider 的端口.

1.3 Provider 对外提供的接口列表.哪些方法.哪些接口类

1.4 Dubbo 的版本.

1.5 访问Provider的协议.

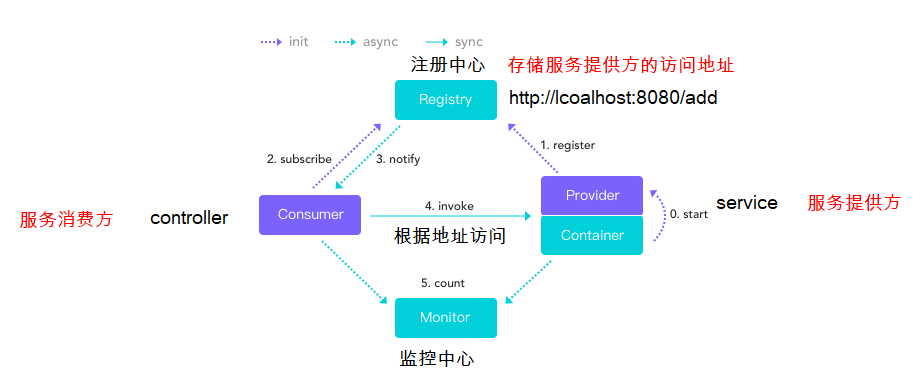
2、subscribe: 订阅.当Consumer启动时,自动去Registry获取到所已注册的服务的信息.

3、notify: 通知.当Provider的信息发生变化时, 自动由Registry向Consumer推送通知.

4、invoke: 调用. Consumer 调用Provider中方法

4.1 同步请求.消耗一定性能.但是必须是同步请求,因为需要接收调用方法后的结果.

5、count:次数. 每隔2分钟,provoider和consumer自动向Monitor发送访问次数.Monitor进行统计.



## 在使用zookeeper作为注册中心的时候，如果zookeeper挂掉了，服务消费方还可以访问服务提供方吗？

Zookeeper本身是有缓存的，缓存的是就是服务提供方数据和消费方的数据。

首次操作：新的服务提供方启动，要往注册中心中注册访问地址，注册中心挂掉了，会注册失败，所以这个时候服务消费方是不能访问服务提供方的

再次操作：服务提供方启动，往注册中心中注册访问地址，注册中没有挂掉，消费方可以从注册中心中取出地方访问服务，但是在再次访问的时候，注册中心挂掉了，这个时候服务消费方会使用zookeeper缓存获取提供方地址，访问，依然可以访问成功