## 系统架构的演变

<https://www.bilibili.com/video/BV1oy4y1b7Yt?p=4&spm_id_from=pageDriver>

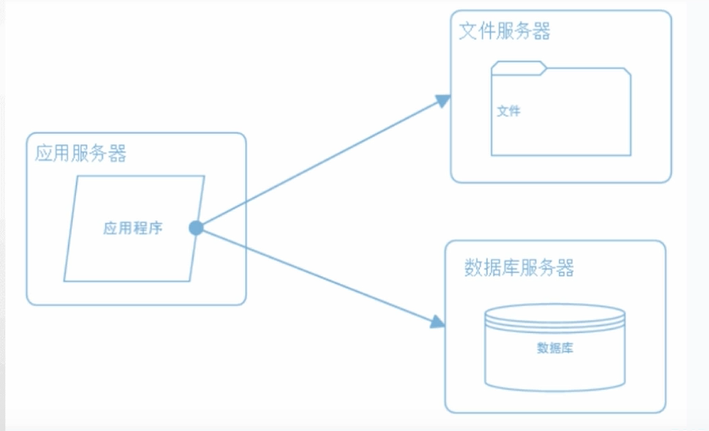
教学视频

## 分布式、集群、微服务区别

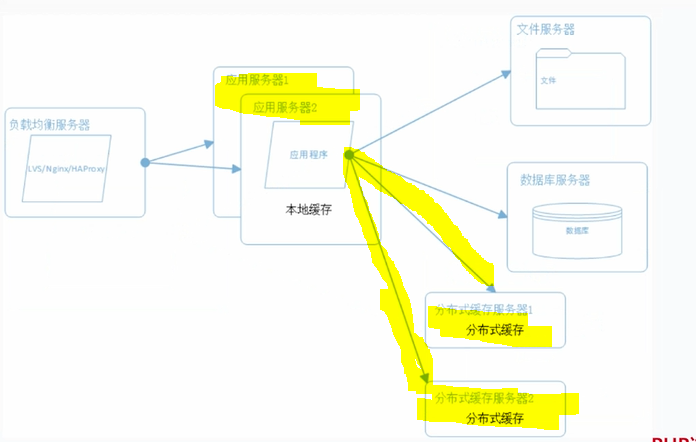
### 分布式

将项目分开部署在多台服务器上叫分布式

#### 基础的分布式



#### 采用负载均衡的分布式



应用程序和分布式缓存属于集群

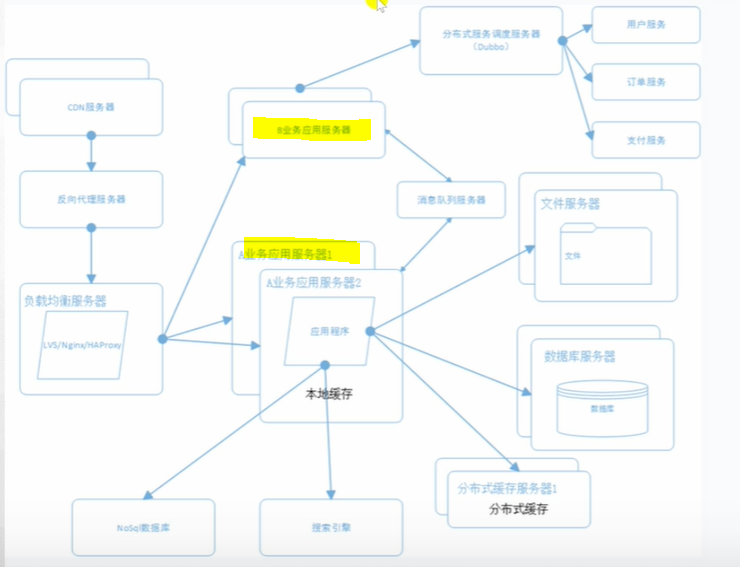
应用服务器1和应用服务器2的功能一样

### 集群

在多台服务器上部署功能一样的项目加集群，比如mysql主从，redis主从

### 微服务

将一个应用程序的业务拆分成小的服务，并且拆分的服务不能脱离这个项目单独运行



## 高并发原则

拆分、服务化与无状态、服务通信、数据异构、缓存银弹、并发化。

### 拆分

服务拆分、业务拆分。

#### 拆分原则

1. 系统维度：根据功能拆分。

如商城系统根据商品、订单、支付、用户拆分

1. 功能维度：

如支付、优惠券=》创建优惠券、领取、使用

1. 读写纬度：

如商品拆分成读服务器、写服务器

1. 模块维度：代码模块 mvc三层架构
2. AOP维度

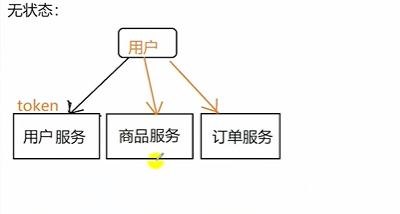
### 服务化

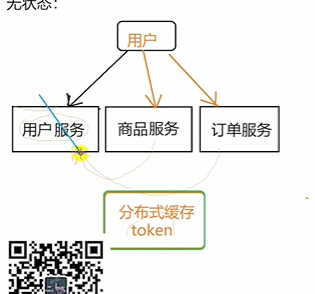
服务化：基于拆分之后的操作

相对独立的功能、内聚的一组业务、数据自我包含。Rpc、api

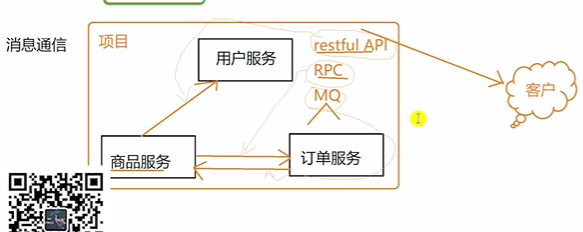
### 无状态

项目拆分为下面三个服务，

将用户token放在用户服务器，商品、订单服务器通过访问用户服务器查看用户登录状态，一旦用户服务器炸掉，其他两个也无法工作

将用户的登录token放在分布式缓存，用户服务器崩溃，商品服务、订单服务不受影响，这个就是无状态，即用户处理一次后其他不影响

### 消息通信



### 数据异构

将数据按需求异地构建存储

### 缓存

### 并发化

主要是不同业务的操作

## 高可用原则

限流、降级、熔断、切流量、可回滚

### 1.限流

Nginx限流、redis限流

#### Nginx限流的方式

参考：https://www.cnblogs.com/niyueling/p/11572003.html

限制访问频率

限制并发连接数

#### Redis限流方式

### 2.降级

Redis🡪nginx->静态

### 3.熔断

目标服务器调用过慢或有大量超时请求

### 4.切流量

### 5.可回滚