前言15972284201

其实从这个课程开始，就正式进入了咱们的真正的架构师课程

思路：实际上就是用一个完全真实的项目背景来驱动，在这个有复杂业务场景的项目中，去讲解各个技术是如何落地使用的

我自己之前讲课，spark，elasticsearch

还有我观察到，行业里其他做培训的机构，包括说一些现在也在做java架构师培训的一些机构，采取的还是老一套的办法

6个月

5个月，就是一个一个的技术，一个知识点，一个知识点的去讲解，可能每个知识点，会给你做一个案例，做一个demo。每个月的知识学习之后，给你一个什么什么项目（demo），复杂的业务需求，纯就是用很简单的业务做一点东西出来，demo

学东西一定能学到，黑竞争对手，用实力说话

一定能学到一些东西，但是问题是在于，真正如何跟项目中的业务场景结合，技术的解决业务问题的理解深度，实践深度，技术本身的理解，不够。很容易遗忘

讲课的方法，脱胎于大学里，先是很枯燥的，书本，一章一章的讲解，全部是理论，小的项目实战，案例实战，练习

1、纯技术，纯理论的学习，会让人感觉很枯燥

2、你学习了很长时间的技术之后，如果没有像公司里做项目一样，立即将这些技术使用到你实际的业务需求里，用技术去做需求。很快会把学习到的一些技术给忘了

引申一点点：为什么说你的技术是由多年的工作经验积累出来的。因为你实际在工作中，比如说半年，或者1年的一个大项目，做了。为了做这个项目，为了实现里面的业务需求，业务场景，解决问题，性能问题，并发问题，可用性问题，稳定性问题，线上运维，可能做了很多的事情

在工作里面会由项目需求驱动，学习了一些技术，快速的将这些技术应用到了你的工作场景中，效果还不错，实际解决了项目里的技术问题，成就感，印象很深刻

如果是经过项目中的业务需求洗礼的话，你对一个技术的理解深度、掌握程度、记忆深刻度，至少是你纯理论学习的10倍以上，5倍~10倍

BAT公司里的人出来，说我是技术大牛，包括我也一样

实际上来说，我们学习跟大家学习，技术，不管是通过什么途径去学习，这个本质差别不大

BAT里，公司，高并发、高可用、高复杂度、线上系统运维坑多、性能问题多，挑战摆那儿，我们学习技术之后，大量的在公司项目里实践，这种情况下，积累出来的技术、架构、深度，绝对是说，会达到普通人，一些小公司的同学，自己看看视频，自己看看书，10倍以上

真正在项目里面踩过坑，遇到了，有绝对复杂的业务场景，技术一定是在一个业务场景中，深刻理解业务的情况下，用技术去解决业务的问题，反过来对你对技术的理解和掌握，那是意义非常的大，比你单纯看本书来说，意义要大很多

3、对技术的解决业务问题的理解深度，实践深度，技术本身的理解，不够

4、出去，感觉到虚，实际上来说，你没有真正用技术去做过项目，10几个项目，6个月做十几个项目，四五个项目，还排除了技术学习的时间

正儿八经，做一个比较成熟，有深度的项目，架构演进，1.0 -> 2.0 -> 3.0，解决了大量的问题，这才是一个项目，周期至少1~2年

redis、mysql分库分表、mq、dubbo、spring cloud，hold住高并发的场景吗？

在过往的6个月里面，普通的培训课程，只是把这些技术一块一块的给学会了，敢说自己能到任何一个公司，那个项目并发量上万/s，你敢说自己就能Hold住？

那我觉得，第一，我觉得你hold住；第二，可能大部分自己都觉得自己虚，hold不住

我要说一下，我对这个架构班课程的，长达1年多，东西非常多，我随时还要加入很多的东西

我的一个授课理念，思路，设想

1、真实项目驱动，10几个项目，做一个项目，大项目，为期1年，1年多的这么一个项目

2、完全真实的业务需求和场景，淘宝，京东，但是至少说，是我梳理过后，我认为是一个完整的电商的系统，可能里面会删减掉很多的细节，但是整体流程是完全可以跑通的。麻雀虽小，五脏俱全，一个项目。至少20个左右的服务。

业务流程，业务需求，业务场景，整体的项目思路，基本就是按照完全真实的去做

但是因为不可能说真的做出来一个京东商城，上千个人在做，所以会保留核心链路，黄金链路，砍掉一些琐碎和细节的需求

3、按照一个大型项目，在1年期间完全真实的一个架构演进的过程去讲解

完全按照真实的去走

（1）一开始，就是几个哥儿们开始创业，做电商网站，卖东西。你就是一个架构师，带着2个小弟。其实就是用最简单的技术把整个这个电商网站的核心业务功能给做出来。技术架构可能就比较low，部署一台服务器。spring boot + mysql + tomcat + 一台阿里云服务器，就可以跑起来了。

可能用户注册数量就10万个人，每天日活，活跃用户，一两万个，访问量就几十万，高峰期每秒并发在**10/s**左右

在这个过程中，我们重点就将spring boot技术穿插在这个业务场景里面去讲解

架构设计的流程，架构设计的几个视图，画哪些架构图，架构怎么去做设计，概要设计，详细设计，设计评审

系统开发流程与规范，BAT公司的编码规范，BAT公司的单元测试规范，BAT公司的多环境测试规范（集成测试、功能测试、性能测试、并发测试、QA测试），BAT公司的上线发布规范，回滚规范，监控规范

设计模式，如果是一个有经验的架构师，在一开始写业务代码的时候，就会很注重设计模式的使用，是落地在类图设计里的，在详细设计的阶段，架构师会去把握这块的东西

代码如果用设计模式，复杂的业务，代码很优雅，在后面在这个代码基础之上，扩展新的功能和代码，都会很轻松，不需要大量的重构

【结果】互联网公司实用型的项目管理方法，20人以内的一个项目，合理的管理，基本上都够了

（2）后面用户量开始增加，增加到了百万级别的用户量，高峰并发可能就**几百**了

而且业务功能不断快速迭代和增加，研发团队开始扩充，可能都扩充到了你一个架构师带了10个小弟

针对这样的业务场景，需要做微服务化，拆分成多个服务，每个服务一台应用服务器，每个服务一个库，应对日益增长的并发量

开始运用微服务+DevOps+敏捷开发+敏捷项目管理，去做

为了解决的问题就是不断迭代的业务需求，不断增加的工程师数量，用上面那套东西如何去抗住百万级的用户访问量，同时将业务需求开发迭代速度支持到最快

（3）用户量都到了千万级了，高并发了，每秒的并发可能都达到**几千**qps（Queries Per Second意思是“每秒查询率”）那样

这套技术架构是撑不住的，整个大电商系统，要针对高并发做整体的架构改造，引入一系列的技术，redis缓存、mysql分库分表、mq、elasticsearch、nginx、集群

（4）高可用，安全性，高稳定性，扩展性，伸缩性，架构问题，需要去解决

（5）**亿级**用户量，亿级流量，整体架构更加复杂，为了解决亿级流量的访问，光是一个电商系统的购物车，就非常复杂

（6）亿级流量了以后，一定会有基础架构组，这个时候会带着大家去研发自己的基础架构，基础中间件

（7）一个公司到了这个程度之后，实际上就会开始做一些大数据的东西，带着大家去学习整个大数据技术栈的一些，java架构师如何将大数据技术栈融入自己的系统架构中

我希望的是每个阶段到下一个阶段的时候，第一个阶段到第二个阶段（微服务阶段）

我希望有一个真实业务场景中，承上启下的这么一个过程，就是说针对第一个阶段的系统架构，给大家真实的去演示他的一些问题，比如说能承载的qps太低了，或者是多个人在里面频繁修改代码，会导致整体的效率非常的底下

从第二个阶段到第三个阶段（高并发阶段）

比如说我要使用redis了，在整个架构中，我会先承上启下，给你去演示一下，现有的系统架构直接基于MySQL去搞，问题在哪里，可能就是撑不住一定的并发量，会跨掉，所以此时在什么什么环节要引入redis去重构这个架构，支撑更高的并发量

（8）总结：完全真实业务，完全真实的购物流程，有一定复杂度的业务场景 -> 1年期间，完全真实的业务发展历程以及技术挑战 -> 完全真实的技术架构演进的过程 -> 所有的技术和架构，穿插在整个项目的过程中去讲解 -> 所有的技术完美的落地实践，与项目结合起来 -> 所有的技术在支持了业务需求之后，还要有足够深度的讲解，比如说spring boot，spring cloud源码的剖析