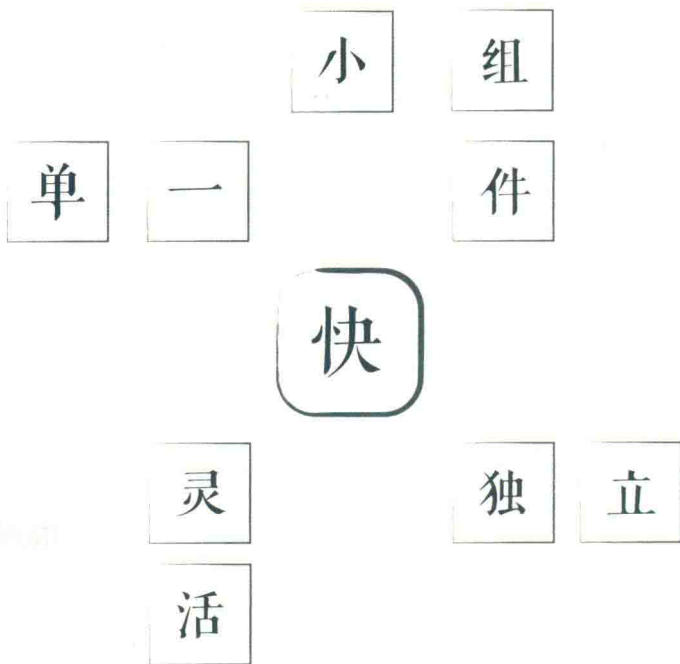


解密微服务 Spring Boot Spring Cloud 技术实践

# 微服务



纪晓峰 著

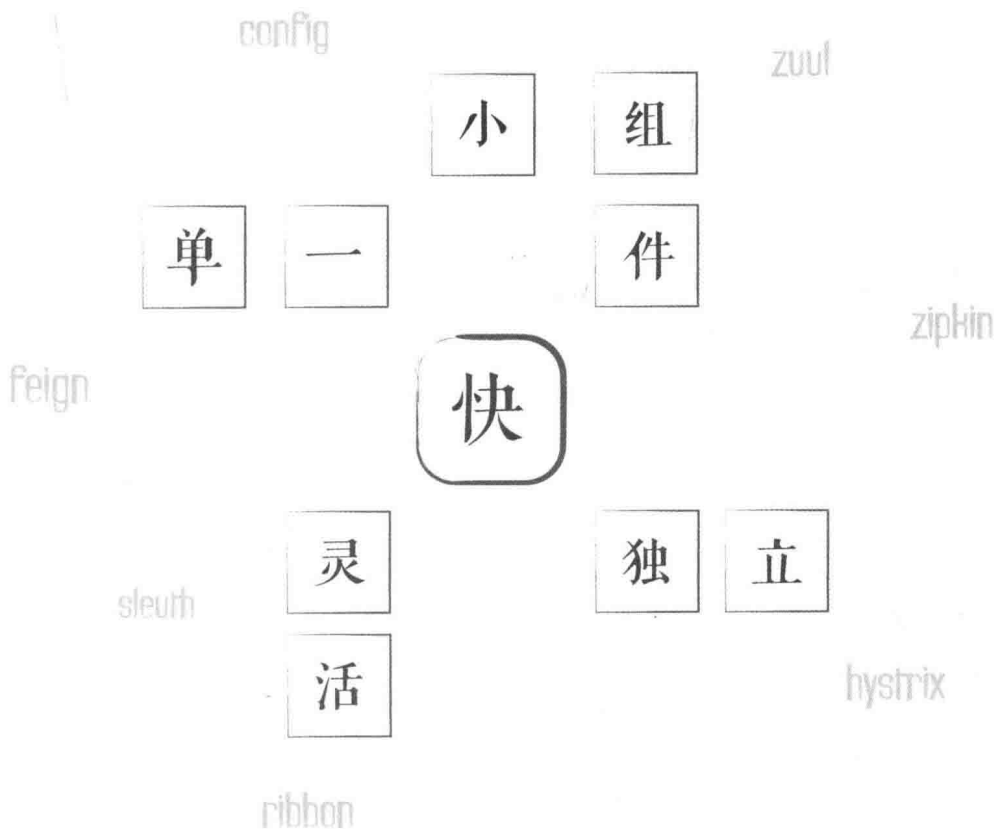


解密微服务 Spring Boot Spring Cloud 技术实践

# 微服务



纪晓峰 著



电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

本书以幽默风趣的语言生动地讲述了微服务，将微服务相关的理论知识、技术实现和项目实践一网打尽。全书共分为三大部分：第一部分为“微服务解惑篇”，让读者对微服务有一个清晰的认识，消除对微服务的种种疑惑，同时对用户十分关注的如何拆分、如何使用等问题做了很全面的讲解；第二部分为“技术实现篇”，通过对当前世界最为流行的微服务框架 Spring Boot+Spring Cloud 的常用组件的详细解读以及其他相关技术、工具的介绍，来帮助读者了解整体架构及其中各个组件的原理，结合代码帮助读者快速构建项目，搭建一个比较完整的微服务框架；第三部分为“项目实战篇”，通过案例来展示不同业务系统在微服务使用过程中如何梳理业务、如何拆分、如何使用微服务重构等。

本书不仅适合架构师、开发人员、测试人员及运维人员阅读，也适合正在尝试使用微服务架构解耦历史遗留系统的团队或个人参考，希望本书能在实际工作中对读者有所帮助。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

### 图书在版编目 (CIP) 数据

微服务那些事儿 / 纪晓峰著. —北京：电子工业出版社，2017.4

ISBN 978-7-121-31142-0

I. ①微… II. ①纪… III. ①互联网络—网络服务器 IV. ①TP368.5

中国版本图书馆 CIP 数据核字 (2017) 第 057517 号

策划编辑：王二华

责任编辑：王二华

印 刷：中国电影出版社印刷厂

装 订：中国电影出版社印刷厂

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱 邮编：100036

开 本：720×1000 1/16 印张：13.5 字数：310 千字

版 次：2017 年 4 月第 1 版

印 次：2017 年 4 月第 1 次印刷

定 价：65.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010)88254888，88258888。

质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn)，盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

本书咨询联系方式：(010)88254532。

# 前 言

当莱斯特城获得 15/16 赛季英超冠军的时刻，我毫不犹豫地亮出了自己隐藏了 30 年的狐狸城球迷身份。

随着微服务在国内的炙手可热，很多人亮出了隐藏多年的微服务专家身份。

然而有一种专家就是那种告诉你 1000 种谈恋爱的办法，但自己从来没有谈过恋爱的人。

他们讲讲亚马逊的订单，谈谈 Uber 的服务，然而就是没有自己做的。

不同于这些专家，笔者曾是 AC 微服务方面的咨询，曾长期供职于 HP，具有丰富的架构设计和实战经验，先后参与过多个微服务项目，对微服务有着较为深刻的理解。

微服务的概念提出已经有几年了，相对于传统技术架构来说微服务不仅仅是新一代架构，更是具有划时代意义的架构。目前国外一些大型互联网公司如亚马逊、Netflix、Spotify 和一些传统公司如沃尔玛都已经采用微服务架构并且实际效果非常好。

与此同时，微服务架构在国内一直不温不火，大多呈观望态势，直到容器技术的出现，为微服务架构的落地完成了最后一块拼图，国内 IT 公司对微服务架构的热情才大增，从各种微服务培训大会一票难求可见一斑。

## 为什么写这本书？

国内很多公司已经开始或正在着手准备使用微服务架构，与市场需求暴增对应的是巨大的人才缺口，市面上相关材料、书籍的信息量很小，且大多偏技术实现，既缺少理论的指导又缺少项目实践，无法满足最终用户的需求。最重要的是还没有介绍 Spring Boot+Spring Cloud 的书问世，它可是当前世界上最主流的微服务技术实现，这多少影响了微服务在国内的发展。

技术类书籍大多有一个普遍的问题，就是枯燥乏味、晦涩难懂。不要说非技

术人员看不懂，连技术出身的人看着也吃力，更别提那些翻译过来的技术书籍。

究其原因是因为描述技术时用的语言是技术语言，而不是生活语言，自然难以理解。

技术不是玄学，技术应该是面向生活的。

所以本书会通过大量生活中的例子对微服务架构及原理进行解释，有趣易懂，可以说是接地气的一本技术书籍。

本书内容都是从实际出发，均来自本人常年在前方发回的“报道”，这里有客户最关心的话题，如何拆分服务、如何使用微服务等。本书有针对性地对这些问题进行了解答，同时提供了一套完整的理论体系，让读者知其然知其所以然。

一本书是一个作者智慧的结晶、知识的沉淀、思维体系的形成。

做人呢，开心最重要。

写书呢，态度最重要。

本人能力一般、水平有限，如有不足之处，欢迎“拍砖”。如果通过这本书能让更多人了解和更好地使用微服务，那么我的目的就达到了。

## 您适合看这本书么？

本书的最大特点是贴近生活，简单易懂，同时又与实践紧密联系。这本书的读者不仅限于技术人员，同时也适用于非技术人员。技术人员看完可以轻松搭建一套完整的微服务环境，开始微服务之旅；架构师看完可以规划设计出更好的架构，业务人员看完也可以从中了解微服务。

只要是对微服务有兴趣的同道中人，都可以从这本书中有所收获。就像 Spring Cloud 的工具包，内容丰富，如果要使用微服务总有一款适合你。

## 本书是如何组织的？

本书分为三大部分。

第一部分微服务解惑篇包含了前 5 章

微服务会给很多人带来疑惑，它是什么，可能一说就懂，但是该不该使用它，以及如何使用等问题就不是三言两语能讲得清楚、说得明白的。这部分内容可以

让读者对微服务有个清晰的认识，消除对微服务的种种疑惑，同时对用户十分关注的如何拆分、如何使用等问题做了很全面的讲解。可以让之前的迷惑与误解消失不见。

### 第 1 章：微服务架构

本章介绍了微服务架构的产生背景、特点、优势及不足，还有在使用过程中会遇到的一些挑战，让读者对微服务有一个整体的了解。

### 第 2 章：为何选择微服务

本章主要讲选择微服务的理由，介绍传统架构的缺陷，而微服务就是它的“解药”，随后介绍了微服务带来的价值及通过它最终要达成的目标，最后介绍了成功人士的选择，让你用着放心。

### 第 3 章：我拆我拆我拆拆拆

本章介绍了微服务的拆分原则，以及拆分过程中需要注意的地方。它是微服务使用过程中非常重要的一环。

### 第 4 章：如何使用微服务

本章介绍了微服务在使用时应该如何规划，并以秦国都城举例说明规划的重要性及规划的方法。随后介绍微服务的使用方法，让读者在使用过程中有一个理论上的指导。

### 第 5 章：微服务的朋友圈

本章介绍了微服务架构的圈子，它不是一个独立的存在，有一系列热门词围绕着它，如容器、DevOps 等。

## 第二部分技术实现篇包含了第 6~9 章

通过对当前世界最为流行的微服务框架 Spring Boot+Spring Cloud 的常用组件的详细解读以及其他相关技术、工具的介绍，来帮助读者了解整体架构及其中各个组件的原理，结合代码帮助读者快速构建项目，搭建一个比较完整的微服务框架。另外，对一些误读进行了澄清，如 Spring Boot 究竟是不是微服务框架。

### 第 6 章：Spring Boot

本章介绍了基础框架 Spring Boot 的一些情况，包括特点、如何使用等。

### 第 7 章：Spring Cloud

本章介绍了 Spring Cloud 中比较常用和重要的一些组件的原理和使用，如

Eureka、Config、Zuul、Hystrix、Ribbon 等。

#### 第 8 章：其他相关技术和工具

本章介绍了微服务的使用过程中需要用到的一些工具和技术，如 Liquibase、Swagger、Kafka、Spring Security 等，这些工具、技术可以辅助微服务被更好地使用。

#### 第 9 章：测试相关

本章介绍了微服务在使用过程中测试的重要性，以及需要用到的一些测试方法和工具，如 Mockito 等。

### 第三部分项目实战篇包含了第 10~11 章

#### 第 10 章：三个典型系统案例

本章通过三个案例来展示不同业务系统在微服务使用过程中如何梳理业务、如何拆分、如何使用微服务重构，等等。

#### 第 11 章：开发管理

讲了开发管理中的管理原则、日常管理、代码质量管理、工作方式及相关人员的工作职责等内容。

## 我要致谢的人

首先感谢我的妻子，妻贤夫安，谢谢妻子对我的理解和包容，给我创造了愉悦的写作环境。谢谢妻子对家的付出，让我无须操心家中事，将更多的精力投入到写作中。谢谢妻子在我写作遇到瓶颈时给予的支持和鼓励，感恩有你的陪伴。

我还要感谢我曾经效力多年的中国惠普，一家非常伟大的公司，惠普之道让我有家一般的归属感，我在这里受益良多，结识了许多正能量的朋友，这些对我来说都是非常宝贵的财富。每当我回想起曾经在惠普的那段岁月，心中总是充满了感恩和美好的记忆。

鸟随鸾凤飞腾远，人伴贤良品自高。要感谢的人太多太多，是你们这些良师益友让我一直在荆棘中不断前行，采撷鲜花。

最后还要感谢电子工业出版社王二华编辑，在我写作期间给予了很大的帮助和指导。

## 专家推荐

近几年，微服务的兴起对从事资讯科技的各级别、各功能、各种技术的人员来说是一重大挑战，也是一个突破的机会。本书深入浅出，让读者很容易就了解到什么是微服务。另外，本书也让已经对微服务有一定认知的读者了解到微服务如何可以应用，在什么环境、什么情况下，会面对什么困难和问题，如何通过不同方法加以解决，让微服务可以顺利地应用，让微服务所带来的好处可以大大地体现出来。对微服务感兴趣的读者和对微服务的应用感到困惑的读者，以及对微服务的场景应用需要更深入了解的读者，可以阅读这本书，不可错过。

电讯盈科解决方案中国区总经理 潘家驰 (William Poon)

软件架构模式的衍变往往有其深刻的历史和业务背景，而在互联网+、云计算、大数据、区块链等新的商业模式及相关技术体系大行其道的今天，微服务模式很好地适应了时代的发展，其辅以相应的管理流程、体系，实践了 DevOps 理念，为 IT 产业的蓬勃发展奠定了技术基础。

市面上关于微服务的书浩若烟海、不胜枚举，本书的作者从实际工作的丰富积累出发，清晰地描述了微服务的“微”、“独”、“轻”、“松”等特点的具体实践。针对企业应用微服务模式中遇到的选型、服务拆分、开发技术及相关工具等重点、难点做了阐述，配合实际生产案例，便于读者快速理解。

作者是我在惠普时的老友、多年的同事，一直在一线从事软件研发和项目实施相关工作，具有丰富的软件架构设计经验。本书将给有志于规划、设计微服务模式的架构师、软件设计和开发者们提供一条可行的路。

云星数据（深圳）有限公司 副总裁兼研发中心总监 楼炜

随着技术的不断发展，企业使用的软件功能变得越来越强大，开发维护的难



度也呈指数级上升。如何高效率地开发一个庞大系统，同时做到后期维护成本低，是每一个架构人员考虑的问题。Dubbo 的分布式服务架构，让大家更多地关注了微服务。在大型的互联网电商平台，需要应对极高的并发场景，分布式服务贯穿其中，有成百上千的应用实例在对外提供着微服务。

在传统的企业，由于没有这样的高性能要求，对于微服务的意识并不强。公司前段时间招聘一批技术人员，应聘者很多，但懂微服务的人却很少。只有一些有电商背景和喜欢钻研技术的人，才对 Dubbo+Zookeeper 有比较深入的了解。我甚是感慨，公司很重视微服务技术，但外面用的又不多，希望借此书也能将微服务做一些推广。这是一种潮流，需要我们去引领。不过还好，技术的领导者已经嗅到了微服务未来的蓬勃发展，在 Spring MVC 的基础上又推出了 Spring Boot，简化了 REST 服务开发。后期又推出了微服务架构 Spring Cloud，打通了微服务的全流程。这对于想了解微服务架构，或者想步入互联网公司的人员来讲，无疑是一个福音。

有了技术领导者的带路，还必须有大量的推广者，才能使一项新技术发扬光大。本书作者纪晓峰就是微服务技术的发烧友，一个坚定的推广者。为了普及微服务知识，特意撰写了这本书。他是我 HP 的同事，工作认真，性格开朗，项目经验丰富，对 Spring Cloud 的微服务框架有很深的研究，能够用通俗易懂的语言让读者在短时间内熟悉微服务。

相信读者看了此书之后，一定能被作者的行文风格所吸引，在潜移默化中加深对微服务的认识。读此书，就如同读小说，又如同读人生。你会同时收获到微服务以外的很多其他知识。

京东商城架构师 孙海波

# 目 录

## 第一部分 微服务解惑篇

第 1 章 微服务架构	2	2.1.2 宝宝心里苦	19
1.1 不微你就 OUT 了	2	2.2 微服务有药	21
1.2 产生背景	3	2.2.1 讳疾忌医	21
1.2.1 时代的召唤	3	2.2.2 朋友听说过安利么	22
1.2.2 容器神助攻	4	2.2.3 沙盘演练看效果	22
1.2.3 微服务的进击	4	2.3 微服务的价值	23
1.3 微服务的标签	5	2.3.1 资源价值	24
1.3.1 单一职责	6	2.3.2 业务价值	24
1.3.2 微	6	2.3.3 技术价值	24
1.3.3 面向服务	7	2.3.4 用户价值	25
1.3.4 自治	8	2.3.5 未来价值	25
1.3.5 易扩展	9	2.3.6 小结	26
1.3.6 流程化	10	2.4 定个小目标	26
1.4 实力碾压	11	2.4.1 持续交付	26
1.4.1 组件化	11	2.4.2 业务敏捷	26
1.4.2 快速	13	2.4.3 独立演进	27
1.4.3 可复用	13	2.4.4 高可用	27
1.4.4 机动灵活	13	2.4.5 高性能	27
1.5 不要奶我	14	2.4.6 站在云端	28
1.5.1 不足	14	2.5 别人家的公司	28
1.5.2 挑战	15	2.6 来不及解释快上车	29
1.5.3 小结	16	第 3 章 我拆我拆我拆拆拆	30
1.6 本章小结	17	3.1 拆还是不拆	30
第 2 章 为何选择微服务	18	3.2 如何拆	32
2.1 传统架构的病	18	3.2.1 庖丁解牛	32
2.1.1 中年危机	18	3.2.2 数据模型和业务模型	33

3.2.3	金字塔结构图	33	4.3	本章小结	45
3.2.4	关键指标	33	<b>第 5 章</b>	<b>微服务的朋友圈</b>	46
3.3	粒度	35	5.1	好朋友容器	46
3.3.1	五笔字根	36	5.1.1	容器的来历	46
3.3.2	被遗忘的字典师傅	37	5.1.2	容器的好处	47
3.3.3	小结	37	5.1.3	小结	49
3.4	边界	37	5.2	天生一对 DevOps	49
3.5	本章小结	38	5.2.1	DevOps 是什么	49
<b>第 4 章</b>	<b>如何使用微服务</b>	39	5.2.2	为什么采用 DevOps	50
4.1	如何规划	39	5.2.3	小结	51
4.2	微服务重构	42	5.3	SOA 是谁	51
4.2.1	影响因素	42	5.3.1	接着忽悠	52
4.2.2	重构原则	43	5.3.2	SOA 是个什么鬼	52
4.2.3	重构方法	44	5.3.3	小结	53

## 第二部分 技术实现篇

<b>第 6 章</b>	<b>Spring Boot</b>	56	7.4.4	Eureka 机制	71
6.1	介绍	56	7.4.5	如何使用	72
6.2	特点	57	7.5	网关 Zuul	77
6.3	Spring Boot 和微服务之间的关系	57	7.5.1	Zuul	78
6.4	如何使用	58	7.5.2	反向代理	78
6.5	可定制化	61	7.5.3	如何使用	79
6.6	本章小结	64	7.6	客户端负载均衡 Ribbon	84
<b>第 7 章</b>	<b>Spring Cloud</b>	65	7.6.1	负载均衡	85
7.1	Spring Cloud 优势	66	7.6.2	如何使用	85
7.2	Spring Cloud 特点	67	7.6.3	补充说明	91
7.3	Spring Cloud 之服务管理	68	7.7	断路器 Hystrix	92
7.4	注册中心 Eureka	69	7.7.1	如何隔离	93
7.4.1	什么是注册中心	69	7.7.2	电路熔断器模式	93
7.4.2	为什么要有注册中心	69	7.7.3	回滚降级	94
7.4.3	Eureka 来历	70	7.7.4	如何使用	95
			7.7.5	交通疏导	101

7.7.6	小结	101	第 8 章	其他相关技术和工具	131
7.8	分布式配置中心 Spring		8.1	数据库开发持续集成工具	
	Cloud Config	101		Liquibase	131
7.8.1	为什么要有配置	102	8.1.1	特性	132
7.8.2	集中配置的好处	103	8.1.2	使用	136
7.8.3	配置方式的演进	103	8.2	Swagger	144
7.8.4	如何使用	104	8.2.1	为什么使用 Swagger	144
7.9	服务之间调用 Feign	114	8.2.2	如何使用	145
7.9.1	Feign 如何使用	114	8.2.3	小结	149
7.9.2	补充说明	118	8.3	权限 spring security	150
7.10	服务追踪	119	8.4	微服务架构的通信方式	152
7.10.1	Spring Cloud Sleuth		8.4.1	同步	153
	介绍	119	8.4.2	异步	156
7.10.2	Sleuth 如何使用	120	8.5	服务编排	160
7.10.3	日志聚合 Zipkin	122	8.6	管理工具 JIRA	162
7.10.4	小结	127	第 9 章	测试相关	166
7.11	Spring Cloud 和 Dubbo 的		9.1	单元测试	166
	比较	127	9.1.1	Mockito 如何使用	168
7.11.1	生的伟大	127	9.1.2	Mock 与 InjectMocks 的	
7.11.2	死的憋屈	127		区别	171
7.11.3	没比较没伤害	128	9.2	接口测试	173
7.11.4	现状	128	9.3	代码质量管理工具	
7.12	Spring Cloud 与 Spring			Sonar	174
	Boot 的关系	129	9.3.1	特点	174
7.13	本章小结	130	9.3.2	使用	175

### 第三部分 项目实战篇

第 10 章	三个典型系统案例	188	10.1.4	解决方案	190
10.1	企业运营支撑系统	188	10.2	车队管理系统	193
10.1.1	背景	188	10.2.1	背景	193
10.1.2	现状	188	10.2.2	技术架构	193
10.1.3	原因调查	189	10.2.3	服务如何拆分	194

10.3	财务管理系统	195	11.2	日常管理	198
10.3.1	背景	195	11.3	代码质量管理	199
10.3.2	现状	195	11.4	工作方式	199
10.3.3	原因调查	196	11.5	BA 的职责	199
10.3.4	解决方案	196	11.6	SA 的职责	200
10.4	本章小结	197	11.7	DEV 的工作原则	201
第 11 章	开发管理	198	11.8	本章小结	201
11.1	管理原则	198	后记		202

第一部分

## 微服务解惑篇

一谈起微服务，很多人都是满脑的问号，具体如下：

- ◇ 微服务是什么？
- ◇ 我为什么要选择微服务？
- ◇ 微服务能带给我什么？
- ◇ 我该如何使用微服务？

第一部分通过一套完整的知识体系，让读者对微服务有一个全方位的了解。读者有了这把钥匙就可以解开心中的疑虑。

## 1.1 不微你就 OUT 了

---

现在众多国外大型互联网公司和传统型公司都使用了微服务架构，这里面比较典型的有亚马逊、沃尔玛、eBay、Spotify、Netflix 等。国内的一些公司，包括一些互联网公司和世界五百强企业也已经投入轰轰烈烈的微服务工作之中。

如果你还不知道微服务是什么，你就 OUT 了。

微服务架构就像葫芦娃兄弟，每一个葫芦娃都有自己的能力果实，随便拎出一个来都可以独当一面，大力果实、火火果实、隐身果实等。每个葫芦娃都是能力单一且唯一的，组合在一起就是无所不能的葫芦小金刚，然后就如意如意随心心了。

最能体现微服务架构思想的就是活字印刷，二者有异曲同工之妙。

传统的雕版印刷，经常出现一个字刻错了，整个雕版都作废的情况，耦合度非常高，很像传统应用里一个错误就拖垮整个应用的情况。

后来毕昇为了解决这种问题放了一个大招，发明了活字印刷。“毕姥爷”的活字印刷闻名于世，每个胶泥制字都是一个独立的组件，再通过组合的方式完成一套完整的印刷版面，实现了完全彻底的解耦合，再也不会因为一个字刻坏了，而废弃整个雕版的情况。

活字印刷术的发明，不仅非常环保，减少了浪费，而且降低了时间成本、物

料成本、印刷成本，极大地提高了生产力，缩短了交付周期，更是改变了历史，为人类文明的发展做出了巨大贡献，影响深远。

毕昇是世界上最早的架构师，向“毕姥爷”致敬。

微服务就像活字印刷一样，每个服务都是一个组件，通过编排组合的方式来使用，从而真正做到了独立、解耦、组件化、易维护、可复用、可替换、高可用、最终达到提高交付质量、缩短交付周期的效果。

随着架构新时代的到来，是选择上微服务的船驶向新世界还是偏安一隅？生活不止眼前的苟且，还有诗与远方。

## 1.2 产生背景

---

微服务架构是互联网行业高速发展的必然结果，传统应用架构已经在持续增长的业务需求、日益激烈的竞争、瞬息万变的市场情况下独木难支，纵然使出“洪荒之力”也枉然。

面对更高要求、更快交付、更多挑战的市场，为了达到敏捷、精益的目标，业界迫切需要一种新的架构能够力挽狂澜，接棒传统应用，继续前行。

微服务踏着七彩祥云就出现了。

随着传统行业也加入微服务大军，微服务架构不再只是互联网行业的专属，大部分传统行业也可以采用这种架构。随着应用范围的逐步扩大，微服务将以星星之火可以燎原之势大放异彩。



### 1.2.1 时代的召唤

It was the best of times, it was the worst of times.

狄更斯笔下的时代已经被历史车轮远远地甩在身后，现在的年代是最快的年代。

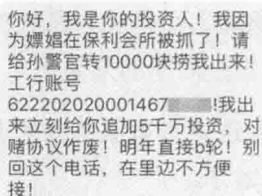
这不仅是大鱼吃小鱼的年代，更是快鱼吃慢鱼的年代。

行业竞争激烈，需求增长快，变化快。面对竞争，快才是硬道理。



业务领域需求变更频繁，如销售、市场、营销都对系统提出了越来越高的要求，需要快速上线，缩短上线周期。

当近期北京打掉三个涉黄场所之后，骗子闻风而动，炮制了下面的诈骗短信，如图 1-1 所示。类似的热点事件经常被蹭。



你好，我是你的投资人！我因为嫖娼在保利会所被抓了！请给孙警官转10000块捞我出来！工行账号 622202020001467 [redacted]！我出来立刻给你追加5千万投资，对赌协议作废！明年直接b轮！别回这个电话，在里边不方便接！

图 1-1 诈骗短信

现在连骗子都知道紧跟时事、蹭热点，更何况企业。

如果不能在瞬息万变的市場情况下做出快速反应，那只能被竞争对手远远地甩在身后。



## 1.2.2 容器神助攻

在容器技术没有出现以前，微服务所面临的“最后一公里”问题很棘手，因为服务多了以后如何自动化部署、如何运维是头等大事，没有合适的解决方案之前，很多人都对微服务心猿意马。

但是随着容器技术的出现，一切都不同了。

容器技术解决了微服务“最后一公里”落地的问题，从而完成了微服务生态的最后一块拼图。



## 1.2.3 微服务的进击

微服务概念的提出已经有几年时间，微服务相对于传统应用可以说是脱胎换骨。

人们常说好的架构不是设计出来的，是进化出来的。

非著名相声演员曾经说过，很多相声演员相声说得好，是因为这些演员就是相声世家，父亲就是说相声的，孩子从小受熏陶，所以说得好，是站在父亲的肩膀之上。