Atitit 微服务之道 attilax著 微服务的艺术

[1. 什么是微服务架构？与一些理论 2](#_Toc678)

[1.1. 、微服务与SOA的关系 :微服务架架构师面向服务架构（SOA）的一种特定实现 2](#_Toc26875)

[1.2. 微服务与康威定律 2](#_Toc19311)

[1.3. 微服务的一些设计 断路器 幂等 版本管理 3](#_Toc25958)

[1.4. 章 康威定律和系统设计　161 4](#_Toc3708)

[1.5. 微服务和分布式的区别 4](#_Toc15738)

[2. 架构的历史 微服务发展历史 Web》soa》msa 4](#_Toc17450)

[3. 微服务的优点 4](#_Toc32203)

[3.1. 微服务最大特点 独立部署 4](#_Toc3526)

[3.2. 性能负担 5](#_Toc30061)

[3.3. 增强稳定性，独立部署 5](#_Toc19747)

[3.4. 协调人力资源，使用不同的擅长的技术来实现不同的模块 5](#_Toc22958)

[3.5. 拆分，有利于开发人员项目规模的轻量化，提升开发速度 5](#_Toc31227)

[4. 核心业务拆分　与　远程接口 5](#_Toc1146)

[4.1. 拆分 5](#_Toc11064)

[4.2. 拆分粒度　　对于大部分项目来说，不到十 5](#_Toc2815)

[5. 微服务的远程接口 cli 与rest接口 5](#_Toc15250)

[5.1. RPC 6](#_Toc5710)

[5.2. Ｒｍｉ 6](#_Toc27696)

[5.3. 6](#_Toc30207)

[5.4. Ｗｓ soa 6](#_Toc15651)

[5.5. dcom/.net 6](#_Toc25907)

[5.6. CORBA 6](#_Toc13051)

[6. 如何实施微服务 6](#_Toc31847)

[6.1. 章 分解单块系统　66 7](#_Toc14793)

[7. 其他 8](#_Toc24293)

[7.1. 第2　章 演化式架构师　11 8](#_Toc6459)

[7.2. 章 集成　32 　与第三方软件集成　6 8](#_Toc26854)

[7.3. 第 11章 微服务与持续交付 ............ 131 8](#_Toc19146)

[7.4. 为什么corba不流行 9](#_Toc2977)

[8. 参考资料 10](#_Toc18948)

[8.1. ATITIT 三种架构 集中，分布式，微服务架构.docx 11](#_Toc11985)

[8.2. 《微服务设计》([英] 纽曼（Sam Newman）) 11](#_Toc30842)

[8.3. 从源头入手，一分钟秒懂为什么要搞微服务架构？ - 推酷.mhtml 11](#_Toc27520)

[8.4. 微服务实战（一）：微服务架构的优势与不足\_知识库\_博客园.html 11](#_Toc14816)

[8.5. 微服务设计－读书笔记 - 简书.html 11](#_Toc21015)

[8.6. 微服务与康威定律 - 推酷.html 11](#_Toc21084)

[8.7. Atitit 提示稳定性 ---------msa微服务部署计算页码服务 v1.1 .docx 11](#_Toc15875)

[8.8. 微服务架构与实践 11](#_Toc8215)

# 什么是微服务架构？与一些理论

Martin Fowler认为，微服务架构是一种独立部署的软件应用设计方式。这种架构方式没有准确的定义，但是在业务能力、自动部署、端对端的整合、对语言及数据的分散控制上有着共性。Martin Fowler曾在文章中详细阐述了微服务的特征，资深架构师顾伟在分享中总结了其中最重要的三点：轻量可复用、安全可伸缩、失败设计。很多企业在发展中遇到了瓶颈，CIO们纠结如何让企业的架构更有弹性、并节约成本的增加弹性、如何开放服务数据、并规避开放之后的安全问题。而微服务架构正能够满足这些需求。但是，微服务架构也为企业带来一些挑战：微服务的粒度更细，导致了更多的进程；微服务架构整合了多种服务形态，却需要提供统一的接口；开放服务之后，业务量不稳定，增加了CPU和内存的负担。

微服务架构就能够很好地解决这个问题。微服务架构自 2010年开始逐渐被大家熟知，通过对传统单块应用进行服务化切分，把一个大而复杂的问题化解为多个小而简单的问题，服务之间通过契约来约定依赖，做到服务独立发布和演进。今天，微服务架构已经被广泛运用在像 Google、 Facebook这样的大型互联网公司，为他们的快速交付和持续创新提供软件架构支撑。本书中有大量微服务架构实战经验的总结，不仅仅有应用架构设计的内容，还涵盖了微服务大背景下应用测试、发布、日志、监控等方面，让读者可以全面应对微服务架构需求。

## 、微服务与SOA的关系 :微服务架架构师面向服务架构（SOA）的一种特定实现

## 微服务与康威定律

Melvin Conway于1968年发表的论文《 ***[How Do Committees Invent](http://www.melconway.com/Home/pdf/committees.pdf" \t "http://www.tuicool.com/articles/_blank)*** 》指出：系统设计的结构必定复制设计该系统的组织的沟通结构。这一论断被称为“康威定律”。在《***[Exploring the Duality Between Product and Organizational Architectures](http://www.hbs.edu/faculty/Publication Files/Research Policy 41 (2012" \t "http://www.tuicool.com/articles/_blank)*** %201309%E2%80%93%201324\_c5c2350e-013c-4065-a2f9-d95eb32177d5.pdf)》一文中，作者发现紧密耦合的组织（例如典型的商业产品公司，所有员工在同一地点工作，具有高度一致的愿景与目标）开发的软件倾向于较少模块化，而松散耦合的组织（例如分布式的开源社区）开发的软件则倾向于更加模块化、耦合较少。

## 微服务的一些设计 断路器 幂等 版本管理

2）断路器  
使用断路器，当请求下游服务发生一定数量的失败后，短路器打开，接下来的请求快速失败。一断时间后，查看下游服务是否已服务，重置断路器。  
3）舱壁  
未每个下游服务建立单独的连接池。超时和断路器资源受限时释放资源，舱壁第一时间确保它不成为限制。还有一个拒绝请求的舱壁，用以避免资源饱和，称之为减载。  
4）隔离  
当下游服务离线，上游服务不受影响。设置成为服务间隔离。

5、幂等

幂等操作，多次执行所产生的影响，均与一次执行影响相同。可以把某些特定业务操作设计成幂等的，比如客户下单送积分。

4、版本管理

1）尽可能推迟破坏性修改  
宽进严出的原则  
 3）不同的接口版本共存  
最好共存两个版本

## 章 康威定律和系统设计　161

## 微服务和分布式的区别

# 架构的历史 微服务发展历史 Web》soa》msa

Web》soa》msa

# 微服务的优点

## 微服务最大特点 独立部署

微服务特点的描述。

大概从以下四个方面来说：

* 根据业务模块划分服务种类。
* 每个服务可以独立部署并且互相隔离。
* 通过轻量的 API 调用服务。
* 服务需要保证良好的高可用性。

## 性能负担

## 增强稳定性，独立部署

## 协调人力资源，使用不同的擅长的技术来实现不同的模块

## 拆分，有利于开发人员项目规模的轻量化，提升开发速度

# 核心业务拆分　与　远程接口

## 拆分

## 拆分粒度　　对于大部分项目来说，不到十

# 微服务的远程接口 cli 与rest接口

关于轻量 API， 微服务本身是推荐使用轻量的通讯协议和简单的数据结构，实际上，实施环节通常采用的都是 http+json 的方式。

## RPC

## Ｒｍｉ

## 

## Ｗｓ soa

我用CORBA做过一点开发，当时是为Lucent和上海贝尔的HLR写接口程序，经验不多。  
我觉得主要是2个原因。1是过于庞大复杂，用起来比较繁琐，对开发人员的要求较高，  
2是缺乏大公司的推动和支持，Microsoft主推自己的，SUN推销Java/RMI  
IBM/BEA等鼓吹Web Service、SOA，其实后来的SOA等技术都从CORBA借鉴吸收了很多内容，  
分布式开发的思想还是不变的，只是具体的实现有所不同，又何必在意CORBA是否流行呢？

## dcom/.net

## CORBA

公共对象请求代理结构(Common Object Request Broker Architecher,CORBA )是对象管理组织(Object Management Group,OMG)针对数量激增的软硬件产品之间互操作的现实需求而提出的中间件解决方案。CORBA定义了接口定义语言和应用编程接口,使对象可以按照特定的对象请求代理(Object Request Broker,ORB) 方式进行交互。 [1]

CORBA从概念上扩展了RPC。用RPC开发的分布式应用是**面向过程**的，而CORBA是完全**面向对象**的。CORBA协议在许多大型企业级应用中使用得很多。因此，CORBA对RPC的扩展就像C++语言 对C语言的扩展一样。 DCOM是微软自己开发的，面向对象的分布式应用模型。它的设计目标和CORBA一样，但它远 没有CORBA那么健壮和丰富，因此，真正在企业级应用系统中使用DCOM协议的并不太多。

# 如何实施微服务

要搞微服务了，有啥建议么?通过我们不断的摸索和总结，要做好微服务，就要做好一定的准备工作。

从五个具体的方面来谈：

业务拆分，体现在设计环节： 在设计的时候，要有足够的判断力来合理的规划服务之间的界限。

服务治理，底层技术的支持： 首先要选一款适合自己实际情况的分布式服务基础框架，对于服务的发现、治理、熔断、降级，都要做好相应的技术准备。

自动测试，一定要自动化。 微服务一个明显的表象就是随着服务的增多，如果继续沿用传统的测试模式就会遇到瓶颈，为了保证高效的迭代，尽量做到更多的环节实现自动化。

自动运维 ： 微服务拆分之后，每个服务都可以独立部署，进而言之应该是随时随地可以升级。尤其当互联网发展到今天，业务要保持对市场变化的一个高效响应，自动化运维就是提升交付速度的一个重要环节。

监控： 包括硬件环境、服务状态、系统健康度、接口调用情况、异常的实时告警以及潜在问题的事先预警等等。监控在实施微服务过程中会重要到什么程度呢?一句话：没准备好监控，就不要搞微服务。

Line 301: 第1　章 微服务　1

## 章 分解单块系统　66

Line 434: 第6　章 部署　86

Line 458: 第7　章 测试　110

Line 487: 第8　章 监控　131

Line 501: 第9　章 安全　143

# 其他

## 第2　章 演化式架构师　11

Line 341: 第3　章 如何建模服务　24

## 章 集成　32 　与第三方软件集成　6

Line 396: 4．15　与第三方软件集成　61

## 第 11章 微服务与持续交付 ............ 131

第 12章 微

Line 555: 第11　章 规模化微服务　171

Line 606: 第12　章 总结　204

第 1部分 基础篇

第 1章 单块架构及其面临的挑战 ............................................. 3

第 2章 微服务架构综述 .................... 13

第 2部分 实践篇

第 3章 构建\*\*个服务 .................... 41

第 4章 Hello World API .................... 45

第 5章 构建 Docker映像 ................. 61

第 6章 部署 Docker映像 ................. 71

第 7章 持续交付流水线 .................... 85

第 8章 日志聚合 .............................. 97

第 9章 监控与告警 ......................... 105

第 10章 功能迭代 .......................... 115

第 3部分 进阶篇

## 为什么corba不流行

我用CORBA做过一点开发，当时是为Lucent和上海贝尔的HLR写接口程序，经验不多。  
我觉得主要是2个原因。1是过于庞大复杂，用起来比较繁琐，对开发人员的要求较高，  
2是缺乏大公司的推动和支持，Microsoft主推自己的com/.net，SUN推销Java/RMI  
IBM/BEA等鼓吹Web Service、SOA，其实后来的SOA等技术都从CORBA借鉴吸收了很多内容，  
分布式开发的思想还是不变的，只是具体的实现有所不同，又何必在意CORBA是否流行呢？

你可以看看《The rise and fall of CORBA》，作者 Michi Henning  
可是corba方面的大大大牛啊，当年是corba的推动者，他写的书《Advanced CORBA programming with C++ 》是用C++开发corba的参考手册。后来他进入反对corba的阵营。他的观点，应该是多年经验的总结。  
我用过micro ORB， ACE的TAO，ominiORB，我个人的感受：   
1.标准是个大问题。标准里面没有定义到每个具体的细节，结果不同厂家都按自己的方式进行了扩展，而且又互不兼容，不同厂家的ORB，调用方法并不完全相同，不同厂家的ORB做server和client端会有兼容性问题。  
2.复杂。可以这么说，如果用C++做corba开发，没有2年多C++的开发经验+1年的corba方面的经验+一定的网络通迅方面的经验，那写出来的程序基本上是经不住压力的。  
3.与流行标准不兼容，没有紧跟时代潮流，但又不便于扩充。corba诞生的年代，网络安全问题还不像现在这么严重，内网基本上没有几个有防火墙的，可是现在不一样，大企业内部基本上都有防火墙，但是corba要穿透防火墙，设置起来非常的麻烦，而且还需要程序内部进行一定的特殊处理，对那些已有的应用，这点就行不通。corba不能跟其它技术进行集成，这在进行快速开发的年代，是致命缺陷。  
4.corba太老了，催生corba的因素，很多都已不存在。新产生的问题，corba解决不了。  
5.最最重要的一点：利益。 各自为了自己的利益，联盟里的厂家都不想让步，结果呢，就只能造成现在的局面。

# 参考资料

## ATITIT 三种架构 集中，分布式，微服务架构.docx

## 《微服务设计》([英] 纽曼（Sam Newman）)

## 从源头入手，一分钟秒懂为什么要搞微服务架构？ - 推酷.mhtml

## 微服务实战（一）：微服务架构的优势与不足\_知识库\_博客园.html

## 微服务设计－读书笔记 - 简书.html

## 微服务与康威定律 - 推酷.html

## Atitit 提示稳定性 ---------msa微服务部署计算页码服务 v1.1 .docx

## 微服务架构与实践

CORBA的兴衰\_百度文库.html