云原生计算研究报告

用友研究院出品

PREFACE

导言

云原生是近几年非常热的一个概念，但云原生到底是什么，云原生起源是什么，云原生的价值是什么，业界众说纷纭。

另外，IT 各界人士在提到云原生的时候都会提到微服务和容器，到底云原生和微服务、容器有什么关联关系？云原生的体系还应该包含哪些？

云原生到底是一时的一个热门概念，还是代表业界未来技术、架构走向？云原生体系未来会如何发展？

为了指明未来技术发展方向，用友研究院特形成专题研究报告。

CONTENTS

目录

01

CHAPTER

第一章

微服务的起源

1953 年 IBM 发布 650 大型主机时并发布了汇编语言，使人类终于告别了直接用最原始的 01010 机器码编写程序的历史。汇编语言是人类计算机史上第一个编程语言，但是当时并没有编程显示器、开发 IDE、编译器，IBM 公司创始时发明的打孔机就相当于是编译器，它会按照在打孔键盘上输入的汇编指令，自动在打孔纸带上按映射规则打出小孔（有孔代表机器码 1）。程序员把打孔纸带放入计算机的读纸带设备里，计算机就会自动按顺序读入计算机进行运算处理，我们可以把这种程序编写方式称作为面向代码流。

1957 年，IBM 发明了 Fortran 语言，Fortran 是世界上第一个高级编程语言，出现模块、函数这样的复杂程序需要的关键字。而且 IBM 还发明了 Fortran 编译器，这也是世界上第一个编译器。函数有明确的输入输出参数数量与类型，并且还具有确定类型的返回值。函数即约定。Fortran 开启了人类面向函数的编程方式。

随着函数数量增多，1972 年，世界上第一个面向对象的编程语言 SmallTalk 产生。物以类聚，关联紧密的函数归集在一个类中，只允许内部调用的函数申明为 Private，允许公开调用的函数申明为 Public。从此，人类从面向函数走向面向对象编程。

面向函数和面向对象，都是为了解决代码工程有组织性结构性的问题。随着人类应用软件越来越复杂、代码规模越来越庞大、函数之间调用越来越交叉，人们发现类的定义申明、函数的定义申明如此重要，一个函数改变了接口的输入输出参数数量或类型，其他关联代码都需要进行连锁修改。所以人们需要把类（函数）的定义申明单独分离出去，交给有丰富经验的高级程序员去精良设计。

SmallTalk 和 C 语言都能在源代码级做到定义与实现分离，但是第一个把 Interface 作为保留字的开发语言是 1995 年出现的 Java 语言。从此，人类从面向对象编程走向面向接口编程。面向接口编程是为了解决不同开发语言的应用代码互相直接调用的问题，这已经属于应用程序互操作性的问题范畴了。

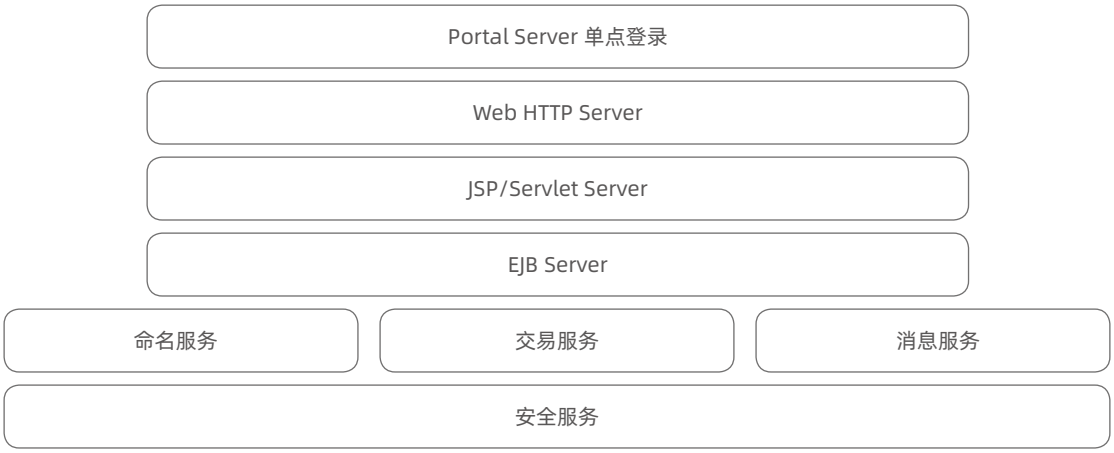
90 年代初，应用程序之间的互相调用，甚至跨机器之间互相调用成为关键需求。1990 年微软为解决 Office 套件之间互相调用（如在 Word 中编辑表格时直接打开 Excel 的编辑功能），发明了 OLE 技术，后来逐步发展为 COM、DCOM、COM+ 技术。

1990 年，IBM、HP、Sun、Novell 等一些厂商发起了 OMG 对象管理组织，1991 年发布 CORBA（Common ObjectRequest Broker Architecture 公共对象请求代理体系结构）技术规范。CORBA 技术更进一步，采用中立的 IDL 接口描述语言，与具体的编程语言实现动态绑定做到编程语言无关性，采用 ORB Stub/Proxy 技术进行数据传输序列化与反序列化，采用 IIOP（互联网 ORB 协议）进行数据传输。CORBA 还定义了公共服务，提供组件命名注册发现服务、事件通知服务、交易事务保证服务、容错保证服务等。但 CORBA 并未定义标准的组件规范。

1998 年，Sun 发布 EJB1.0 技术标准，补齐了 CORBA 体系没有的组件规范，并用 Java 语言在 Java VM

平台上重新实现了一套基于组件技术的 RPC 机制（PRC 为 Sun 公司于 1984 年发明的跨服务器调用的技术）：RMI（远程方法接口）。至此业界三大组件技术规范均已产生。

三大组件技术体系均为解决跨应用程序之间、跨机器之间的互操作问题。三大组件技术体系均有共性特点：拥有接口定义语言，支持多种接口调用协议与网络通讯协议，拥有公共总线服务（命名注册发现服务、生命周期管理服务、安全服务、事务处理服务、事件服务、通知服务等）。组件模型、组件容器、组件总线中间件，成为了面向组件编程的三大核心。IBM WebSphere、BEA WebLogic、Red Hat JBOSS 应用中间件成为了当时支持 Java EJB 组件模型、组件容器、服务总线的代表性产品。



图例为常规应用服务器组成模块。应用经由 Eclipse Studio 进行开发、编译、自动化测试、调试后打包成 JAR 或 WAR 包，实施人员将应用包部署到 Web HTTP Server、JSP Server、EJB Server 容器中，形成了以下技术栈关系：

1995 年，全球第一次互联网热潮爆发，HTML/XML 数据格式、HTTP 网络传输协议，比二进制数据格式、TCP/IP 网络传输协议更具有互联性。1998 年，W3C 开始制定 Web Service 技术规范，以便满足互联网级的应用逻辑调用。目前又逐步演进到 JSON 数据格式、RESTful 接口风格。人类从面向组件编程进化到面向服务编程（SOA），而这里的服务，特指的就是 Web Service。面向服务编程 SOA 主要解决的就是应用程序之间跨互联网互操作的问题。

但是经过多年完善，组件技术也逐步复杂沉重，业界开始兴起反组件、反企业总线中间件热潮。实用的工程界开始尝试使用单纯的 Java 类来代替沉重的 EJB 技术，原有的 EJB 容器被 Docker 为代表的容器技术所代替，企业服务总线承担的命名注册发现服务被独立的 API 网关代替，企业服务总线承担的事件与通知服务被独立的消息队列中间件代替，组件、组件容器、企业服务总线正在逐步分解、消融。人类从面向服务编程（SOA）走向微服务编程，主要是为了解决组件技术体系太沉重复杂的问题。这个阶段技术体系尤其以开源的 Struts 框架（代替 JSP 中间件职责）、Spring 框架（代替 EJB 中间件职责）、Hibernete 框架（代替 JPA 中间件职责）为代表。中间件产品的职责被上移到应用开发层。

目前，AWS 更在提倡无服务器编程：Function as Service，颗粒度从 Class 微服务缩小到 Function 级。把原来由各个应用中间件承担的职能，如命名注册发现、容错服务、日志服务、监控服务、安全服务、调度服务，也都从应用编程框架、应用中间层的级别，下沉到系统层，应用软件开发商不仅不用关注复杂的组件技术，也不用关注复杂的大规模分布式中间件技术，只需要关注自己的核心业务逻辑即可。

可以这么说，面向函数和面向对象是解决代码组织性问题，面向接口、面向组件、面向服务都是解决应用程序之间互操作性问题，面向微服务以及面向无服务器编程都是为了解决分布式应用技术体系过于沉重复杂的问题。至于代码模块可灵活替换、灵活组合、灵活修改这些需求，都不是上述技术的解决焦点，只能靠架构师和程序员进行精心的接口设计以及精心的程序代码编程实现。

02

CHAPTER

第二章

云原生概念的起源

云原生概念，是 Pivotal 公司的 Matt Stine 于 2013 年首次提出。Matt Stine 曾经作为咨询解决方案架构师，为大型企业客户的 IT 部门提供如何敏捷地应用最新技术。Matt Stine 目前担任 Pivotal 全球架构首席技术官，本质为 Pivotal 公司的技术布道师，大部分时间都在向企业 IT 领导层建议如何有效采用云技术架构。

Pivotal 公司最初来源于 1989 年成立的 Pivotal 实验室，，2012 年被 EMC 收购。Pivotal 公司是由 EMC 和 VMWare 在 2013 年 4 月共同重组，将两家公司相关业务剥离出来成立一家单独公司，EMC 剥离出来的最核心软件资产是大数据产品 Greenplum，VMWare 剥离出来的最核心软件资产是应用中间件平台 CloudeFoundry 和 SpringCloud。Pivotal 公司创始人为前微软高管保罗马里茨（1986 年加入微软，曾负责微软操作系统产品线研发管理），2000 年加盟 VMWare，2008 年担任 VMWare CEO，2013 年组建 Pivotal 担任董事长。Pivotal 创始 CEO 为前 Greenplum 产品公司创始人，该公司于 2010 年被 EMC 收购。 Pivotal 于 2018 年 4 月 20 日在纽交所上市（最高市值 60 亿美元），于 2019 年 8 月被 VMWare 并购（收购价 27 亿美元），目前 Pivotal 全线产品已经被整合进 VMWare Tanzu 产品家族。而 VMWare 也是 2004 年被 EMC 收购。EMC 在 2016 年又被 DELL 收购。

Pivotal 的主要产品线有：

- 服务层产品线：微服务中间件 SpringCloud（2009 年 SpringSource 被 VMWare 并购）
- 中间件层产品线：消息队列中间件 RabbitMQ（2010 年被 SpringSource 收购）
- 数据层产品线：分布式关系数据库 Greenplum（基于伯克利大学开源的 PostgreSQL）、分布式 KV 数据库 Redis（2010 年起由 VMWare 支持，2013 年起由 Pivotal 赞助）、分布式内存缓存 GemFire、Pivotal HD（Pivotal 的 Hadoop 发行版）、Pivotal HDB（Pivotal 的 SQL on Hadoop 发行版）
- 平台层产品线：Cloud Foundry（2008 年被 SpringSource 并购）

现在 SpringCloud 在开源的推动下，也形成了完整的技术栈：

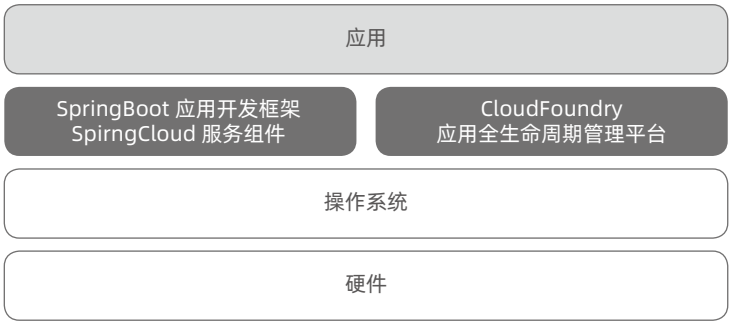
- 服务调用产品线：客户端负载均衡 SpringCloud Ribbon、API 网关服务 SpringCloud Zuul、声明式服务调用 SpringCloud Feign
- 服务总线：消息总线 SpringCloud Bus、消息驱动的微服务 SpringCloud Stream、分布式配置中心 SpringCloud Config
- 服务管理：服务发现 SpringCloud Eureka、服务容错保护 SpringCloud Hystrix、分布式服务跟踪 SpringCloud Sleuth

微服务时代，中间件产品的职责被从应用中间件层上

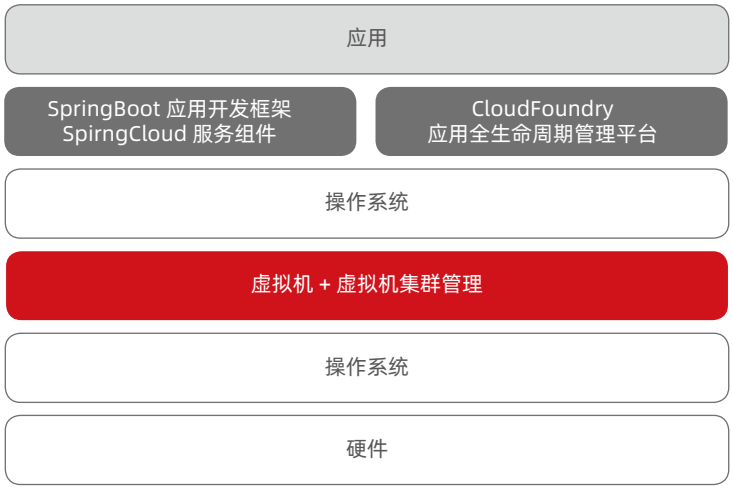
移到了应用开发层，而且商用应用中间件套件被分解为更为细分的专业开源组件，这不仅加重了开发人员、测试人员的技能要求，也加重了实施部署、技术运维人员的工作。

各应用框架和专业细分的组件，如何进行相互关联性地打包，并且还能解决不同框架组件之间的版本兼容性问题？Cloud Foundry 应用而生。随着 Cloud Foundry 的不断进化，Cloud Foundry 已经形成应用全生命周期管理全套职能栈：

- 集成阶段
  - ◇ 解决各应用框架与组件依赖性完整打包问题
  - ◇ 解决各应用框架与组件版本兼容性问题
- 部署阶段
  - ◇ 统一部署，屏蔽多云技术之间的差异，如虚拟机、容器
  - ◇ 统一部署，屏蔽多云部署之间的差异性，如 AWS、Azure
- 运行阶段
  - ◇ 应用运行资源隔离
- 运维阶段
  - ◇ 应用部署资源管理与调度、自动扩展
  - ◇ 应用运行状态监控、日志采集



微服务时代（基于裸金属服务器部署）



微服务时代（基于虚拟机部署）

基于 Pivotal 公司这套全栈技术生态体系，应用开发人员可以做到快速开发、简化部署、快速部署、多云部署、故障转移、自动扩展。

虽然“云原生”一词由 Matt Stine 首先提出，但 Matt Stine 并未给云原生下明确定义。他只是指出：云原生是他根据自己多年的架构与咨询经验总结出来的一套思想集合，其中囊括了：微服务技术、敏捷基础设施、程序设计方法、持续集成和 DevOps，甚至涉及了组织文化（康威定律）。这套思想集合与十多年前畅行的敏捷研发有很多相似之处，而且微服务技术也是 Martin Fowler 于 2004 年提出的概念。至于云原生为什么要包含这些思想、方法、技术、工具，以及云原生和云计算到底存在什么关系，Matt Stine 并未说明。

另外，Matt Stine 还非常推崇 HeroKu 公司在 2012 年针对微服务设计、开发、部署、运行、管理制定的 12 条原则，也被纳入到他的思想集合中。Matt Stine 认为没有良好的微服务设计，即使有很好的微服务平台技术，照样发挥不了微服务的价值作用。这十二条微服务原则是：

1、设计：显式声明 API、显示声明依赖关系

2、设计：要把应用配置放到部署环境中存储，代码和配置严格分离

3、设计：应用的进程必须是无状态且无共享的

4、设计：微服务应该可以快速启动和优雅终止

5、设计：要通过 URL 绑定的方式来提供微服务

6、开发：一份代码库，对应多套部署（如开发环境、测试环境、生产环境）

7、开发：日志，一定要日志，而且要把日志当做事件流的汇总

8、部署：要严格区分构建、发布、运行三个步骤三套环境，不要直接修改运行环境中的代码和配置

9、部署：开发环境与线上环境要等价

10、运行：通过启动新的多个进程来进行扩容

11、管理：把后端应用服务（如邮件发送任务）和数据服务（如数据计算任务）都让系统管理员统一管理起来

12、管理：一个批处理命令一次性完成管理任务

03

CHAPTER

第三章

云原生计算基金会

CNCF，The Cloud Native Computing Foundation 的缩写，云原生计算基金会。这是业界首个以云原生为主题的组织。

CNCF 从属于 Linux 基金会。Linux 基金会是一个非盈利性联盟，其目的在于协调、规范、保护、宣传 Linux 发展，以便最大限度地保持各个商业公司基于开源 Linux 构造的商业发行版之间的兼容性。Linux 基金会是 2007 年，由开源码发展实验室（Open Source Development Labs，OSDL) 与自由标准组织（Free Standards Group ,FSG) 联合起来成立的。Linux 基金会扶持与孵化了多个成功开源项目，如：开源虚拟机 XEN、开源软件定义网络 OpenNFV 和 OpenSwitch、开源人工智能可迁移模型 ONNX 等等。

2014 年，Cloud Foundry 成立基金会，并隶属于 Linux 基金会。2015 年 7 月，云原生计算基金会（CNCF）成立，也隶属于 Linux 基金会。云原始生计算基金会（CNCF）宣称：我们致力于加强和维护一个厂商中立的开源生态体系，来推广云原始生技术。

CNCF 基金会给云原生下了明确定义（https://github.com/cncf/toc/blob/master/DEFINITION.md）：

云原生的技术有利于各组织在公有云，混合云、私有云环境中，构建和运行可弹性扩展的应用。云原生的代表技术包括容器、服务网格、微服务、不可变基础设施、声明式 API。这些可靠的自动化手段、云原生技术，使工程师能够轻松地对系统做出切实可行的改进和可预测的重大变更。

从定义中我们可以理解，云原生的目的是构建和运行可弹性扩展的应用，手段是代表云原生的核心技术，如容器、服务网格、微服务、不可变基础设施、声明式 API 等。

自从人类走入局域联网、互联联网、分布式计算、多层技术架构时代以后，我们的技术栈就日益复杂：

- 前端层：
  - ◇ 开发框架：JQuery、BootStrap、AngularJS、React、Vue、Flutter
  - ◇ 开发技术：移动原生开发技术、小程序开发技术、CDN
  - ◇ 开发语言：Dart、Swift、Kotlin
- 前端服务层：
  - ◇ 开发框架：Struts、Ruby on Rails
  - ◇ 开发技术：JSP、ASP.NET、HAProxy/Ngnix
  - ◇ 开发语言：PHP、Python、Ruby、Perl、Node.JS
- 应用逻辑层：
  - ◇ 开发框架：SpringCloud
  - ◇ 中间件技术：Kafka、Zookeeper
  - ◇ 开发语言：Golang、Java、C#
- 数据层：
  - ◇ 存储：Ceph、GlusterDFS、HDFS

◇ 数据库：MySQL、Hbase、Redis、MongoDB、NEO4J

◇ 数据工具：ELK、Flume、Presto、ElasticSearch

◇ 数据计算平台：Hadoop MR、Spark、Flink

- 系统层：

◇ 中间件技术：Docker、Kubernetes

◇ 开发语言：Rust

开发人员、测试人员、实施人员、运维人员，不仅要关注客户业务需求、应用软件功能，还得关注如此复杂的中间件技术和开发框架，还得应对分布式计算所面临的服务发现、服务负载均衡、容错限流、自动伸缩与自愈、调用链监控等等巨大挑战。尤其在全球化海量用户、需求快速变化浪涌浪退、24x7 不间断运行、海量数据的应用场景下，这个挑战愈显艰难。如何简化开发、部署实施与运维，还能满足应用需求，成了互联网厂商、云计算厂商、开源技术社区共同面临的问题。

CNCF 的领导厂商为 Google，目前有 500 多个企业成员加入了 CNCF，铂金级的会员包括：

- 应用厂商：SAP、Oracle
- IaaS 厂商：AWS、Azure、阿里云、IBM Cloud、Google Cloud、JD Cloud
- 系统软件厂商：Red Hat、VMWare
- 网络厂商：Cisco、华为
- IT 设备厂商：Dell、NetApp、富士通
- 芯片厂商：Intel、ARM

CNCF 最早起源于 2007 年 Google 研发出 CGroups，这种技术机制，可以使 Linux 进程能够做到对 IT 资源（如 CPU、磁盘、网络）的限制与隔离。2007 年，CGroups 被集成到 Linux 的内核当中（2007 年，KVM 虚拟机技术也被纳入到 Linux 内核中）。2008 年，Linux 发布新版本，Linux 容器功能包含在内。

2013 年，Docker 创业公司推出第一个开源容器引擎版本。2014 年，Google 开源容器资源管理与调度平台 Kubernetes，并纳入到 CNCF 基金会的孵化项目中。Google 希望借助 Kubernetes，统一管理多个操作系统之上的所有容器技术，不限于 Docker 公司出品的容器。

经过六年的孵化毕业以及业界其他产品的兼容性认证，CNCF 也形成了一个完整的技术栈图谱：

- 服务管理产品线：服务网格 Linkerd 和 Istio
- 中间件层产品线：远程调用中间件 gRPC、消息队列中间件 Nats、消息分发中间件 CloudEvents
- 数据层产品线：对象文件存储 Rook、关系数据库 Vitess（基于开源 MySQL）、KV 数据库 TiKV

- 容器层: Containerd（是 Docker 的底层运行时管理器，2017 年被 CNCF 接受）、容器网络接口 CNI、容器网络发现 CoreDNS

- 容器管理层：容器资源管理 Kubernetes、容器镜像仓库管理工具 Harbor、容器镜像分发工具 DragonFly、容器运行日志管理工具 Prometheus、容器运行安全监控工具 Falco

2019 年 10 月，CNCF 发起第一次用户调查（CNCF Survey 2019），共收到 1337 份有效问卷，欧洲占 37%、美洲占 38%、亚洲占 17%，三分之二的人来自软件 / 技术 / 专业服务厂商，其中 41% 是软件架构师、24% 是后端开发工程师、39% 是 DevOps 运维工程师，71% 的人工作在 100 人以上的公司，30% 的人工作在 5000 人以上的公司。

调查得知：

- 服务器使用：15% 的公司拥有 5000 台以上的服务器，70% 的公司拥有 20 台以上的服务器。62% 的服务器是公有云服务器，30% 的服务器是私有云服务器

- 容器使用：84% 的公司已经使用容器技术，容器数量大于 250 个的公司超过 50%

- Kubernetes 使用：82% 的人在使用 Kubernetes 技术

- 服务网格使用：18% 的人使用服务网格

- 无服务器技术使用：41% 的人在使用无服务器技术

- Open API 技术使用：14% 的人对外输出 API

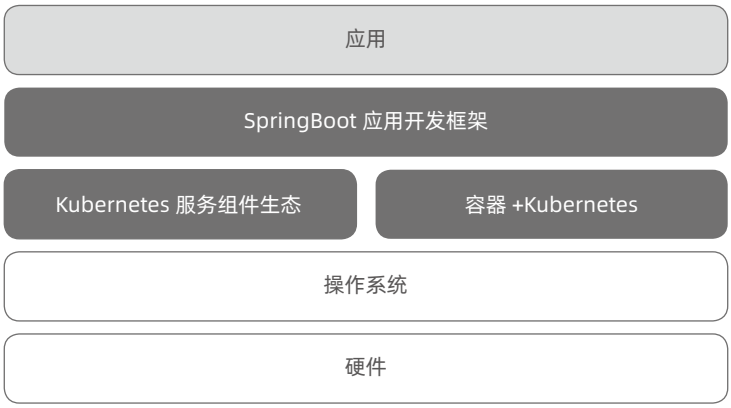
- CNCF 数据存储技术使用：14% 的人在使用 CNCF 的数据存储技术

- 持续集成 / 持续发布技术使用：40% 的人使用无状态测试

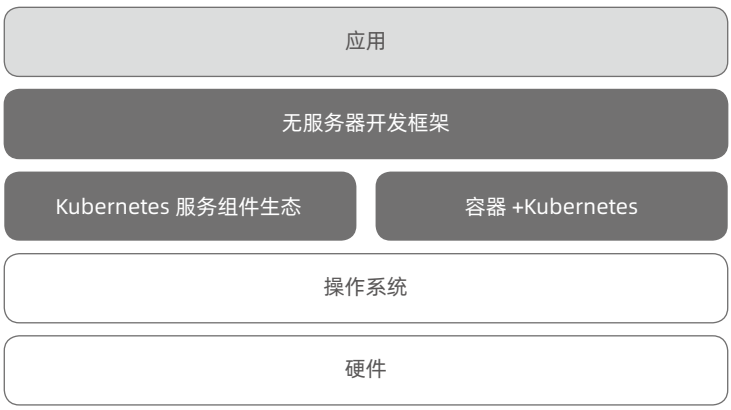
52% 的人认为云原生计算可以让部署更有效率，45% 的人认为云原生计算可以让应用更具有弹性伸缩，39% 的人认为云原生计算可以提高可迁移性和应用高可用性。

我们也做了 SpringCloud+CloudFoundry，和容器 +Kubernetes 两个生态体系的技术对比：

	SpringCloud 生态体系	容器 +Kubernetes 生态体系
服务发现	Eureka	Ingress
服务负载均衡	Ribbon	Service
API 网关	Zuul	Ingress
配置管理	SpringCloud Config	ConfigMap
容错限流	Hystrix	Health Check&resource isolation
日志监控	ELK	EFK(Fluentd)
度量指标监控	Actuator	Prometheus
调用链监控	Sleuth	OpenTracing，Zipkin
应用打包	JAR	Pod
服务框架	Spring（Boot）	无
发布和调度	CloudFoundry	Scheduler&Deployment
作业管理	Spring Batch	Jobs
自动伸缩与自愈	HealthIndicator	Autoscaler
进程隔离	CloudFoundry	Pod
环境管理	CloudFoundry	Namesapces
资源配额	CloudFoundry	Namespace resource quotas
集群	SpringCloud Cluster	Pods



微服务时代（基于容器部署）



无服务器时代（基于容器部署）

从对比中我们可以洞察到，CNCF 形成的容器 +Kubernetes 生态体系已经和 SpringCloud+CloudFoundry 生态体系旗鼓相当，CNCF 把过去暴露在应用框架和应用中间件层的技术下沉到系统层，应用开发人员、测试员、部署实施和配置人员、运维人员，已经不用在应用层级关心这些复杂技术。而且，CNCF 并不主打微服务，而是主打无服务器编程 Serverless。

04

CHAPTER

第四章

从云计算到云原生

1995 年全球互联网热潮爆发以来，除了创业的纯互联网站以外，还有来自企业 IT 市场人士也在思考如何利用互联网。

1998 年，IDC 运营厂商 Rackspace 抛出 ASP 概念：应用服务托管提供商（Application Service Provider）。Web 技术大多应用于企业外部业务（如企业和消费者之间、企业和上下游合作伙伴之间），因而 Web 应用具有天然的外部公开性以及高并发性，所以 Web 技术体系的要求比 C/S 技术体系更高。有了了应用服务托管提供商，企业无需自建 IDC、无须自己招募更多更熟悉 Web 专业互联网安全专业的 IT 专家来运维，完全可以交给 Rackspace 来进行服务器托管、软件迁移、数据备份、性能监控、安全防护。

1999 年的 Salesforce 更是提出：No Software。企业不用自建或租用 IDC，不用购买、托管、租用服务器，不用外包给其他 IT 运维厂商来做安装部署，不用担心数据备份和丢失，不用担心全球黑客攻击，只需要在线充值在线注册在线开通使用就好。

连续创业家 Diane Greene 看到了这个机会，于是在 1998 年创建 VMWare 公司，专门研发一种虚拟机技术：

- 隔离环境：让不同应用的部署，不受基础软件版本、配置参数不同的影响
- 节省成本：一台服务器上可以运行多个虚拟机，合理组合利用资源
- 批量部署：一次配置成虚拟机镜像文件后，以后在多点部署的时候可以做到批量快速部署
- 快速迁移：把应用软件和依赖基础软件环境打包成一个虚拟机镜像文件后，可以走到应用快速迁移到其他服务器上

Rackspace 除了开展企业服务器托管到 IDC 的业务以外，也利用开源虚拟机技术 XEN 和 KVM 做虚拟服务器对外订阅租用。为了管理众多托管的裸金属服务器和虚拟服务器，Rackspace 也研发了海量 IT 资源管理平台，这就是 2010 年开源的 OpenStack。本质上来讲，OpenStack 就是云时代的资源集群管理操作系统，便于管理海量的分布在多处的计算资源、存储资源、网络资源。

经过十年开源发展，OpenStack（2019 Train 版本）也形成了一套完整的 IT 资源管理框架：

服务类别	服务组件
服务器、虚拟机 容器、Serverless 资源管理工具	Bare Metal service (ironic) 裸金属服务器资源管理服务
	Compute service (nova) 虚拟机服务
	Image service (glance) 虚拟机镜像资源管理服务
	Containers service (zun) 容器服务
	Container Infrastructure Management service (magnum) 容器集群部署服务
	Function as a Service for OpenStack (qinling) Severless 服务
存储资源 管理工具	Block Storage service (cinder) 块存储服务
	Object Storage service (swift) 对象存储服务
	Shared File Systems service (manila) 共享文件系统服务
网络资源 管理工具	Networking service (neutron) 网络虚拟化服务
	Load-balancer service (octavia) 负载均衡服务
	DNS service (designate) DNS 服务
数据库资源 管理工具	Database service (trove) 数据库管理工具
	Big Data Processing Framework Provisioning (sahara) 大数据框架配置工具
安全管理	Identity service (keystone)。用户身份认证服务
	Key Manager service (barbican) 密钥管理服务
	Application Data Protection as a Service(karbor) 应用程序数据保护服务
资源编排工具	Orchestration service (heat) 资源编排服务
	Clustering service (senlin) 集群管理工具
	Resource reservation service (blazar) 资源预定服务

服务类别	服务组件
运维管理工具	Telemetry Data Collection service (ceilometer) 监控数据收集服务
	Telemetry Alarming services (aodh) 告警服务
	Application Catalog service (murano) 应用目录服务
	Indexing and Search（searchlight）资源索引与搜索服务
	Dashboard (horizon) 可视化管理操作台

OpenStack 生态体系虽然也在不断融合容器技术，但整体优势仍然在虚拟化多租户切割的服务器资源、存储资源、网络资源的管理方面。RackSpace 从切割虚拟服务器进行租用出发、VMWare 从简便运维出发，都采取了虚拟机 + 虚拟 IT 资源管理平台技术。但 VMWare 和 OpenStack 技术都不是为了解决分布式并行计算和分布式海量存储问题而来的。从技术价值来看，OpenStack 更倾向于资源的集群管理，而 Kubernetes 生态体系更倾向于分布式计算。

容器 +Kubernetes 生态体系比 OpenStack 生态体系更靠近裸金属服务器和操作系统，而且容器 +Kubernetes 生态体系更把微服务时代的服务组件从应用层下沉到系统层。从这个视角来理解，云时代的分布式计算编程范式，越来越趋向单机编程范式。这可能是云原生的价值本质。

2001 年，全球电子商务巨头亚马逊推出第三方商家平台。2002 年为了方便第三方商家平台能和它们自己的内部应用软件打通，亚马逊开放了 Open API。但是由于商家的服务器环境、网络质量参差不齐，导致亚马逊第三方商家平台 -Open API 平台 - 商家自己应用系统之间总是存在不稳定性，给商业正常运转带来很多麻烦。于是，亚马逊决定自己做 IT 资源订阅租用服务提供给商家使用，2006 年，AWS 发布。

AWS 经过十多年的发展，已经形成完善的产品体系，从企业应用中间件、多媒体处理中间件、人工智能中间件，到 SQL、NOSQL 数据库，到大数据湖、数据仓库、大数据实时计算平台、大数据可视化、大数据查询与搜索、大数据挖掘，甚至协同处理组件（IM、互联网音视频会议、云呼叫中心、文档、日历）、亚马逊开放平台等。不管是

企业 IT 部门，还是中小软件公司，都可以基于他们提供的丰富的产品以及 API，来快速开发出自己所需的商业应用。

AWS 不重点宣扬容器、虚拟机、微服务，虽然这些服务他们都提供。他们也宣称云原生应用，但是他们宣称的云原生应用，却是利用他们的云开发工具集，直接调用他们的云产品 API，直接编写应用，直接发布到他们的应用市场中，直接购买直接运行，开发者和使用者均不用关心底层技术细节和运维。

2014 年以来，AWS 引领业界走向无服务器架构：Serverless，落地的服务产品就是 AWS Lambda。AWS Lambda 把微服务更微缩成一个匿名函数：Function as Service，可以使用各种开发语言进行编写（如 Node.js、Python、Java、C#）。AWS 会自动把此函数代码打包到容器中，自行根据 IT 资源情况进行自动部署，以便做到毫秒级启动、执行完毕即刻销毁。所以 Serverless 既屏蔽了底层技术的复杂性，又提供了极具性价比的 IT 资源利用。目前，AWS Lambda 按照调用次数、执行时间长度、占用内存大小来进行计费，每 100 万次调用价格 0.2 美元、每 GB 每秒 0.00002 美元。AWS 预测在未来：IaaS、PaaS、SaaS、互联网电子商务服务会极大丰富，新的应用只需要调用他们持续在线提供的 Open API 进行粘合，即可组合出新的应用。

所以云原生目前在业界有三股力量，均宣称云原生概念，但对云原生的定义与目标均不一样：

- 来自技术产品提供商，如 Google 领导的 CNCF Kubernetes 生态体系、如 VMWare/Pivotal 领导的 SpringSource+CloudFoundry 生态体系，他们要么重点宣扬基于容器之上开发 Severless 微应用，要么重点宣扬基于微服务中间件之上开发微服务。
- 来自公有云计算提供商，如 AWS，他们重点宣扬基于他们通用的 IaaS、PaaS、云开发工具链、无服务器编程技术、云应用商店，直接开发 SaaS 应用或互联网应用。
- 来自公有 SaaS 提供商，如 Salesforce，他们重点宣扬基于他们的应用开发平台、应用开发语言（如 APEX）、应用开放 API 平台、应用商店，直接开发他们的 SaaS 应用和生态应用。

这三者从价值作用来讲，都可以称作云原生应用，适合不同技术实力、不同目标的应用开发组织。

05

CHAPTER

第五章

云原生计算对于中国企业客户的价值

中国市场企业客户呈现多样性：

- 有 GDP 超过一个欧洲国家的地方政府，有营业额超过一个欧洲国家的巨型央企
- 有极具爆炸式成长、极速伸缩变化的互联网电子商务企业
- 有几千万家小微、中小企业
- 有几百万家中型、中大型企业
- 有几万家大型企业

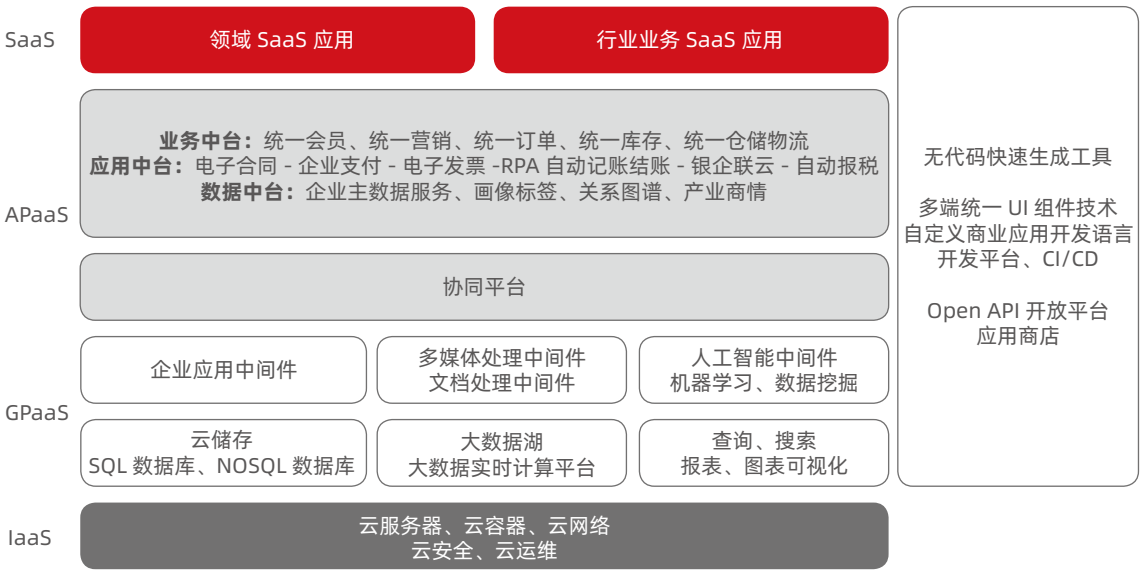
虽然群体多样，但企业应用诉求却趋同，呈现两个方向：

- 业务开展方式：趋向于在线化，趋向于直接连接用户、直接连接上下游合作伙伴、直接连接社会。
- 业务开展范围：趋向于全国化甚至全球化。

在技术诉求方面，也呈现三个共性：

- 多端统一技术：一套代码适配 PC Web、移动 Web、移动 App、移动小程序
- 微服务技术：便于性能提升、敏捷修改
- 大数据技术：便于性能提升、数据深度分析

面对三类云原生技术提供商，不同组织如何满足应用诉求与技术诉求？



对于中国一二线城市、地级市政府应用，一般会选择类似 CNCF、Pivotal 这样的技术栈，但会通过系统集成解决方案专业服务提供商（如软通动力、文思海辉等）来开发上层应用。

对于拥有上万人研发团队独立科技公司的巨型央企，一般也会选择类似 CNCF、Pivotal 这样的技术栈来自我研发上层商业应用。

对于极具爆炸式成长、极速伸缩变化的中国互联网电子商务企业，他们本身就是纯互联网应用，所以他们会积极拥抱公用云厂商，在公有云的产品之上开发他们的互联网应用。在中国云计算不成熟的过去时期，有一些快速长成的巨无霸确实会有重新自建私有云的反复，但相信在未来云计算成熟时期，新一批快速成长的互联网电子商务企业，他们会永久生长在公有云上。这无关 IT 成本事宜，主要和 IT 弹性伸缩、组织弹性伸缩、社会化资源整合文化相关。

对于中国几千万家小微企业来讲，他们可能会多直接采用中国互联网电子商务平台商提供的在线经营管理工具。对于中国上千万家小、中小企业，他们会直接采用中国 SaaS 提供商的服务。

中国还有几十万甚至百万级的中型、中大型、大型企业，他们不仅会采用 SaaS 商的 SaaS 服务，而且他们也会采用 SaaS 商的应用开发平台和 Open API 开放平台，不管是自组研发团队还是外包给专业开发服务商，他们会基于应用开发平台开发他们的商业创新应用。

06

CHAPTER

第六章

中国企业对云原生计算的关注点

中国地方政府、中国巨型企业，对云原生计算的关注重心是：安全性高、国产化、自主可控。

对于极具爆炸式成长、极速伸缩变化的中国互联网电子商务企业，他们对云原生计算的关注重心是：快速开发、弹性伸缩、高稳定高可用、高安全。

对于中国几千万家小微企业，他们一般会直接使用中国互联网电子商务平台商提供的在线经营管理工具，并不接触到云原生技术。对于上千万家小、中小企业，他们只直接使用中国 SaaS 商提供的 SaaS 服务，对于云原生应用，他们希望这些应用能够像苹果 App Store 一样，有丰富的应用插件、模板，易于寻找、购买、支付，一键下载安装、统一计费精确计费、统一运维。

对于还有几十万甚至百万级的中型、中大型企业，他们不仅希望 SaaS 提供商有可视化的配置平台、快捷的无代码快速生成工具，也希望有丰富的应用商店应用插件、模板、表单、报表、模块级生态应用。另外他们还希望有免费、API 丰富开放、高性能高可用、高数据安全的 Open API 开放平台，便于和他们的其他应用进行集成。

对于几万家中国大型企业，他们希望 SaaS 提供商提供专属云部署的低代码开发平台、Open API 开放平台和应用组件、协同平台与应用商店，便于快速开发他们自己的商业创新应用。

商业创新 如此便捷

用友网络科技股份有限公司

地址：中国北京市海淀区北清路68号用友产业园
网址：www.yonyou.com
邮编：100094

yonyou Network Technology Co.,Ltd.
Address: yonyou industrial park, no. 68, Beiqing road, Haidian district, Beijing, China
Website: www.yonyou.com
Postal code: 100094