2021年
云原生行业研究报告(一)：微服务

2021 Cloud Native Industry Research Report (1): Microservice Architecture

クラウドネイティブ産業調査レポート(1): アプリケーションアーキテクチャ

报告标签：应用架构、Serverless、PaaS
主笔人：胡竣杰

概览摘要

应用架构的演进和生物物种的进化有着共同的内核：物竞天择，适者生存。随着信息技术的日新月异，传统架构已经不能适配不断迭代的底层硬件和不断升级的顶层业务发展诉求。

微服务架构在2016年后随着容器技术一起成为技术热点，以Spring Cloud为代表的侵入式开发框架占据着微服务市场的主流地位，Spring微服务框架+Docker容器+K8s集群管理被誉为在春天的货船上的盛世。

逐渐，非侵入式的服务网格Service Mesh走向成熟，到Linkerd、Envoy、Istio、Conduit等框架选型的出现，微服务市场从框架、组件、到容器等产品系列与生态的不断发展与完善，大大降低了微服务从技术理念到产品落地的门槛，使微服务企业用户最为重视的架构方式之一。

头豹 LeadLeo
www.leadleo.com
400-072-5588

01 微服务的技术概述

微服务通过细粒度切分、单独进程、轻量级通信、独立部署四个特征解决了单体应用架构衍生的集中式项目迭代流程。

正如微服务是 SOA 实践中正确的部分一样，微服务架构在经过了四次迭代后，Mecha也在Mesh的实践中吸取经验和教训， 有望成为微服务落地实践的下一站，而Mecha也不会是微服务或应用架构演进的终点。

02 微服务的应用概况

微服务具备敏捷开发、弹性伸缩和高可用的优势，但由于其复杂且高昂的的前期实现搭建，微服务并非适合所有场景。

微服务通过领域驱动设计完成业务拆分，协同IT团队的架构调整与平衡系统架构的设计，逐步完善微服务架构改造。

03 微服务的市场概况

互联网厂商、传统IT厂商以及开源微服务项目为企业端进行微服务架构改造提供了多种成熟稳定的方案，推动了微服务行业向各行业场景进一步积累技术储备并开拓企业用户资源。

微服务市场吸引了互联网厂商、云服务厂商、传统IT实施服务商、应用软件开发厂商及初创企业参与布局，从提供的产品或服务的视角区分，微服务市场参与者主要分三类。

目录

CONTENTS

图表目录

名词解释

- **云原生：**区别于从本地环境移植到云上的大部分程序，云原生强调最初的开发就是为了最终部署到云环境上。在公有云、私有云和混合云等新型动态环境中，赋能组织或企业去构建和部署可弹性扩展的应用。
- **持续交付：**(Continuous delivery, CD)，是一种软件工程手法，让软件产品的产出过程在一个短周期内完成，以保证软件可以稳定、持续的保持在随时可以发布的状况
- **微服务架构：**微服务架构 = 80% 的 SOA 服务架构思想 + 100% 的组件化架构思想 + 80% 的领域建模思想，系统中的各个微服务可被独立部署，各个微服务之间是松耦合的。每个微服务仅关注于完成一件任务并很好地完成该任务。
- **DevOps：**Development和Operations的组合词，是一组过程、方法与系统的统称，用于促进开发（应用程序/软件工程）、技术运营和质量保障（QA）部门之间的沟通、协作与整合。
- **容器技术：**有效的将单个操作系统的资源划分到孤立的组中，以便更好的在孤立的组之间平衡有冲突的资源使用需求，这种技术就是容器技术。
- **正交分解：**将一个力分解为Fx和Fy两个相互垂直的分力的方法，叫作力的正交分解。
- **无状态服务：**stateless service， 对单次请求的处理，不依赖其他请求，也就是说，处理一次请求所需的全部信息，要么都包含在这个请求里，要么可以从外部获取到（比如说数据库），服务器本身不存储任何信息。
- **SideCar 模式：**将应用程序的组件部署到单独的进程或容器中，以提供隔离和封装。 使用此模式还可以使用异构组件和技术来构建应用程序。称作“挎斗”(Sidecar)，是因为它类似于三轮摩托车上的挎斗。在此模式中，挎斗附加到父应用程序，为应用程序提供支持性功能。 此外，挎斗与父应用程序具有相同的生命周期：与父应用程序一起创建，一起停用。 挎斗模式有时也称为搭档模式，这是一种分解模式。
- **Localhost：**在计算机网络中，localhost（意为“本地主机”，指“这台计算机”）是给回路网络接口（loopback）的一个标准主机名。
- **运行时rumtime：**本身是一个相当混淆的概念，不需要分那么清。对于不同层级的应用，其 runtime 的含义不同。一个编程语言的 runtime 概念范围比用它写的某个应用程序的要小；另一个基于此应用的应用程序的 runtime 包含了前者。
- **PHP语言：**PHP（PHP: Hypertext Preprocessor）即“超文本预处理器”，是在服务器端执行的脚本语言，尤其适用于Web开发并可嵌入HTML中。PHP语法学习了C语言，吸纳Java和Perl多个语言的特色发展出自己的特色语法，并根据它们的长项持续改进提升自己

01

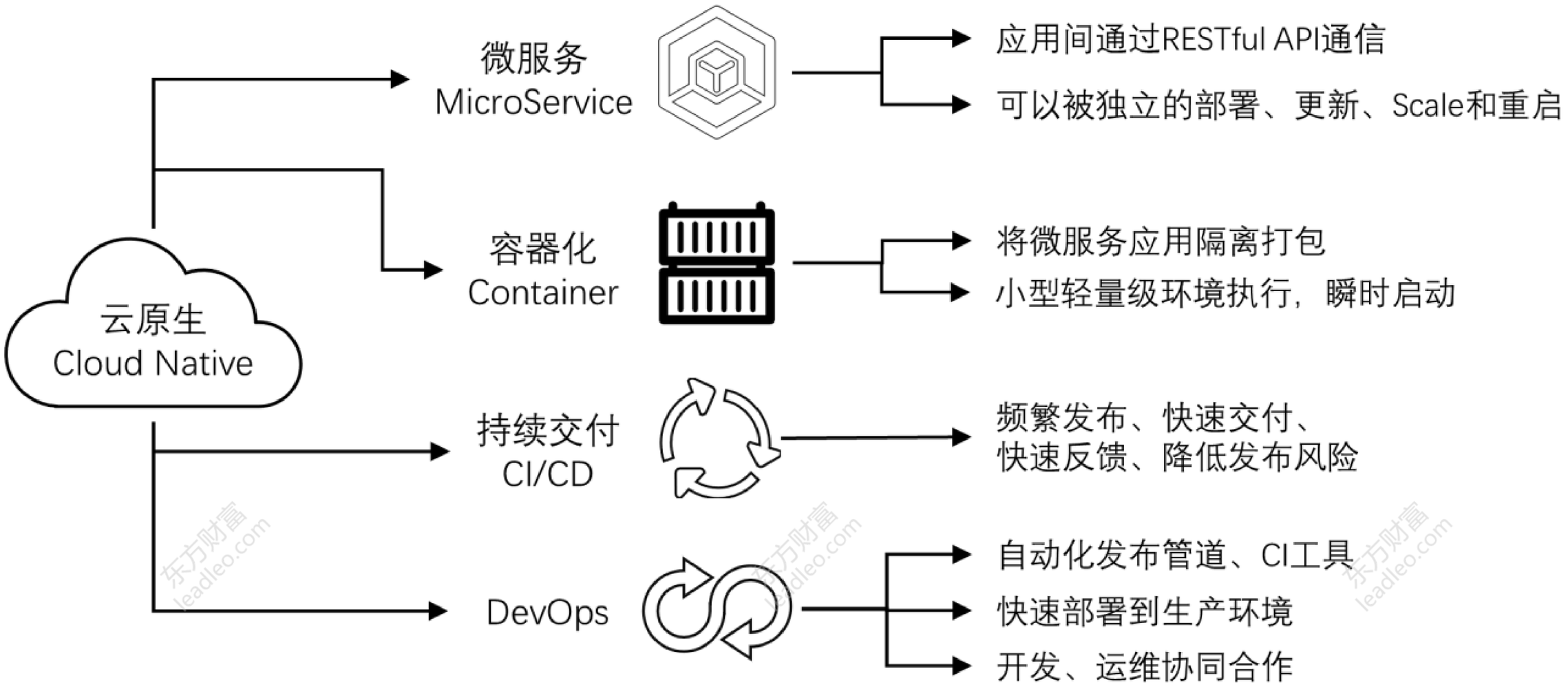
应用架构综述

- 云原生概述
- 应用架构的演进路径

云原生概述

- 云原生通过微服务、容器化、持续交付、DevOps等技术，赋能企业在云基础设施上，实现速度、可扩展性和利润的提升

云原生构成



来源：CNCF, 头豹研究院

云原生的定义

区别于从本地环境移植到云上的大部分程序，云原生强调最初的开发就是为了最终部署到云环境上。在公有云、私有云和混合云等新型动态环境中，赋能组织或企业去构建和部署可弹性扩展的应用。

云原生的目的

- **速度：**将想法快速实现推向市场在竞争中具有极大的战略意义。云原生本质上是关于如何解决技术风险的策略，配合企业内部文化向渐进式改进的开发模式转变，依靠采取小型、可逆和低风险的步骤来实现迅速前进，极大缩短开发周期。
- **可扩展性：**云原生应用将其状态存储在数据库或其他外部实体中，不依赖于底层基础设施，只需向集群添加节点就可以扩展应用程序。
- **利润：**企业支出从前期的 CAPEX（购买新机器以预期成功）变成了 OPEX（按需支付额外的服务器）。实现只在新客户上线时购买所需的额外资源的战略目标，满足灵敏响应和可控管理成本的需求。

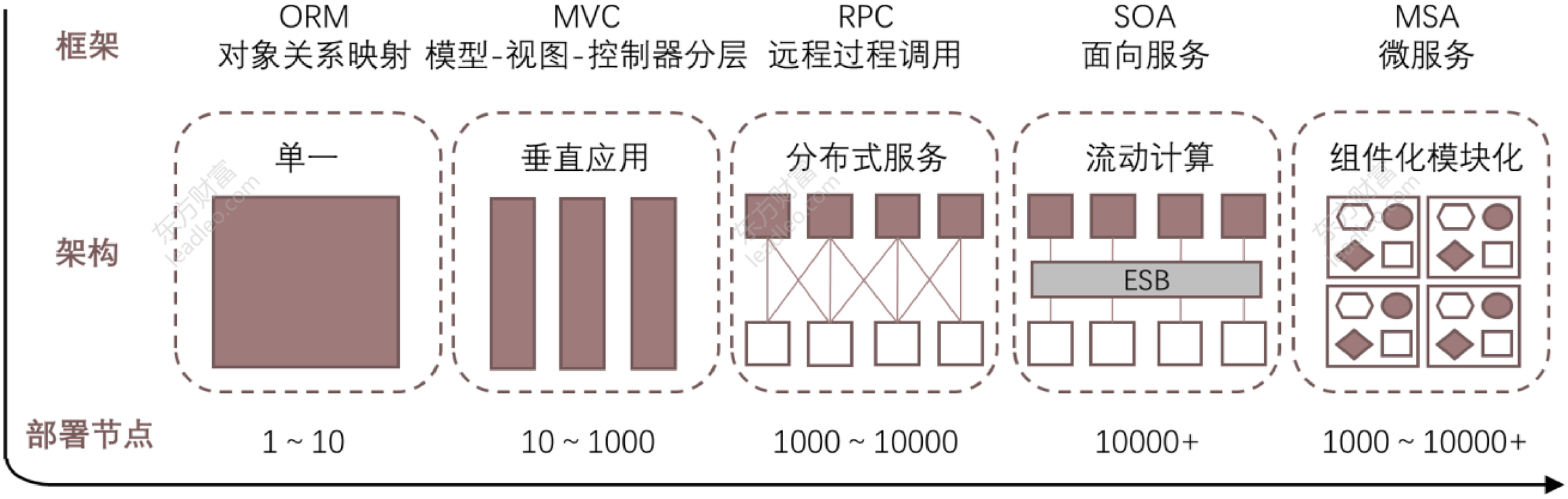
应用架构的演进路径

- 应用架构为了满足扩展弹性与适配大型项目开发运维的需求，逐渐向松散耦合，细粒度的微服务应用架构发展

❑ 应用架构的定义

根据业务和功能要求，应用架构描述了业务中使用的应用程序的行为，着重于它们之间以及与用户的交互方式，确保组织用来创建复合体系结构的应用程序套件是可伸缩，可靠，可用和可管理的。

应用架构的演进



来源：Dubbo，头豹研究院

❑ 单一应用架构

当网站流量很小时，只需一个应用，将所有功能都部署在一起，以减少部署节点和成本。数据访问框架(ORM)可用于简化增删改查工作量。

优点：架构简单；前期开发成本低；开发周期短

缺点：技术栈受限；扩展成本高、有瓶颈；大型项目不易开发维护

❑ 垂直应用架构

当访问量逐渐增大，单一应用增加机器带来的加速度越来越小，将应用拆成互不相干的几个应用，以提升效率。此时，用于加速前端页面开发的Web框架(MVC)是关键。

优点：架构简单；开发成本低；不同的项目可以采用不同的技术

缺点：扩展成本高、有瓶颈；大型项目不易开发维护

分布式服务架构

当垂直应用越来越多，应用之间交互不可避免，将核心业务抽取出来，作为独立的服务，逐渐形成稳定的服务中心，使前端应用能更快速的响应多变的市场需求。此时，分布式服务框架(RPC) 可用于提高业务复用及整合。

优点：将重复的功能抽取为服务；针对不同服务制定集群及优化方案

缺点：系统与服务界限模糊，不利于开发维护；服务粒度大，系统与服务之间耦合性高

流动计算架构

当服务越来越多，容量的评估，小服务资源的浪费等问题逐渐显现，此时需要增加一个调度中心（企业服务总线ESB）基于访问压力实时管理集群容量，提高集群利用率。此时，用于提高机器利用率的资源调度和治理中心(SOA)是关键。

优点：体现解耦思想；采用ESB减少系统中的接口耦合

缺点：服务接口协议不固定；服务粒度大，系统与服务之间耦合性高

组件化模块化架构

虚拟化技术与DevOps文化的快速发展以及传统单块架构无法适应快速变化，微服务不再强调传统SOA架构里面比较重的ESB企业服务总线。微服务把所有的“思考”逻辑包括路由、消息解析等放在服务内部，去掉一个大一统的ESB，服务间轻通信，是比SOA更彻底的拆分。

优点：拆分粒度更细，利于资源重复利用；RESTful轻量协议通信，比ESB更轻量；更精准制定每个服务的优化方案，提高可运维性；迭代周期更短

缺点：微服务过多时服务治理成本高；开发技术成本高；使用团队挑战大

演进路径总结

伴随着企业业务发展所需的软件代码库的扩张，由于集中式研发、测试、发布、沟通模式，应用迭代效率显著下降。应用架构的技术方案演进由紧耦合向松耦合，粗粒度向细粒度的规律。

02

微服务的技术概述

- 微服务的定义与原理
- 微服务技术的演进与发展

微服务的定义与原理

- 微服务通过细粒度切分、单独进程、轻量级通信、独立部署四个特征解决了单体应用架构衍生的集中式项目迭代流程

❑ 微服务的定义

微服务架构（MSA，Microservices Architecture）是一种架构风格和设计模式，提倡将应用分割成一系列细粒度的服务，每个服务专注于单一业务功能，运行于独立部署的进程中，服务之间边界清晰，采用如HTTP/REST等轻量级通信机制。提炼出四点微服务的特征：

1. 细粒度切分 2. 单独的进程 3. 轻量级通信 4.松耦合，可独立部署

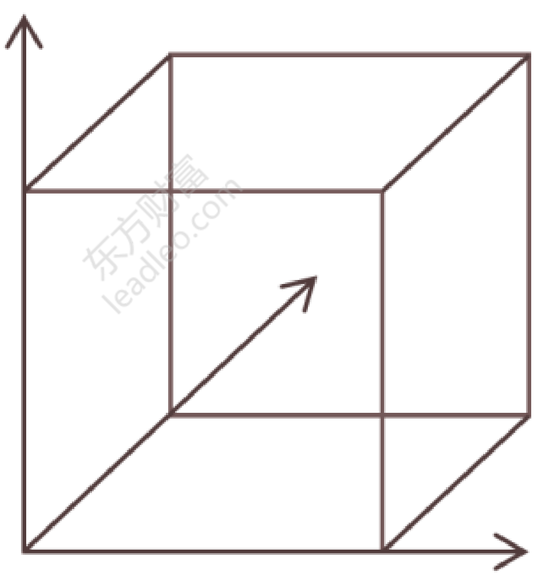
❑ 细粒度切分

微服务应用所完成的功能在业务域划分上相互独立。相比单体应用强行绑定语言和技术栈，微服务的好处是不同业务域有着不同的技术选择权，比如推荐系统采用Python要比Java的实现效率要更高。

- 于**组织层面**上，微服务对应的团队更小，“一个微服务团队一顿能吃掉两张披萨饼”是业内对正确划分微服务在业务域边界的隐喻，通过最大化“适度职责”实现相对自治，增益开发效率。
- 于**开发效率**上，微服务团队虽小却要求着更高的开发迭代速度，业内评价标准是至少两周完成一次迭代，所以也反向对微服务的业务域边界划分提出了要求。

综上，微服务的细粒度划分不是为了微而无限拆分，而是按照问题对单体应用做合理的切分。

微服务AKF可扩展模型



X轴-服务实例水平扩展
- 复制应用程序的副本扩展，让每个副本均衡负载，
- 保证可靠性与性能。

Y轴-功能的扩展
- 功能的扩展，将应用程序拆分为多个不同的服务，
- 服务单一职责，功能独立。

Z轴-数据分区
- 基于每个实体的数据行，通过一组数据库对数据进行分区/分片，
- 每个数据库服务器仅处理数据的子集，故障隔离，
- 数据独立，可靠性保证。

来源：AKF，头豹研究院

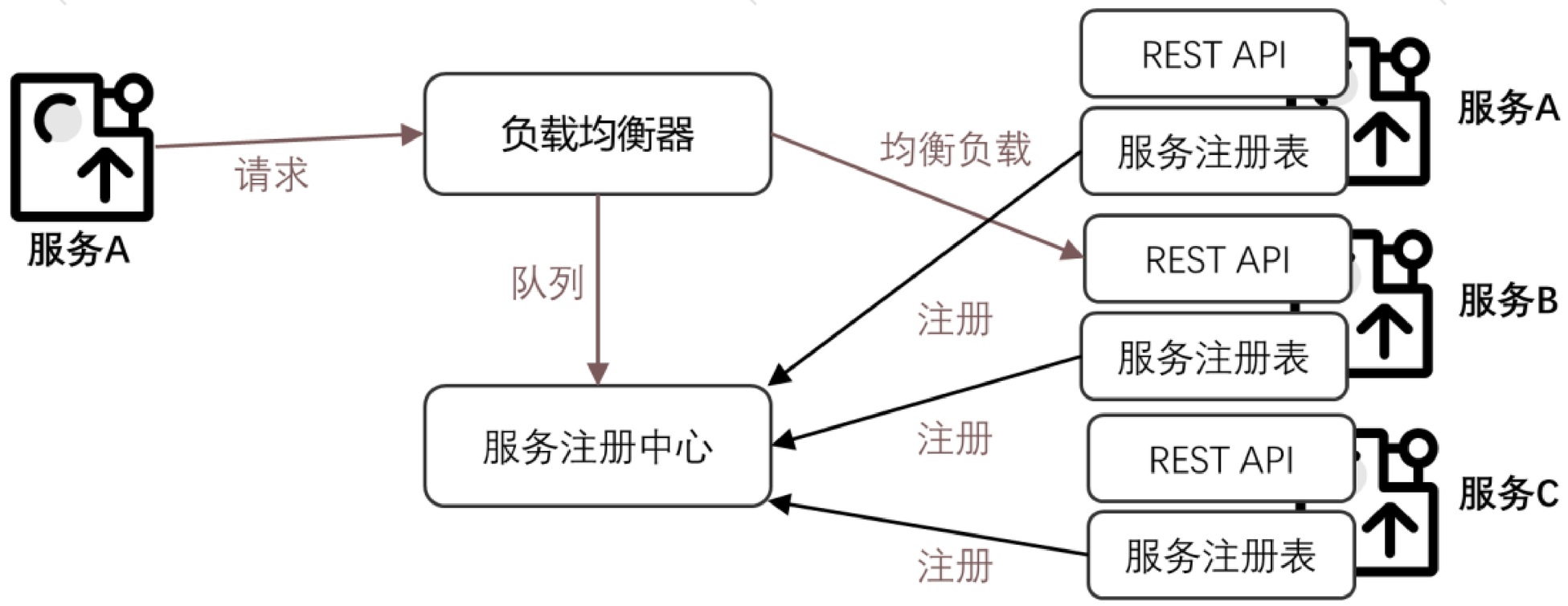
微服务具备正交分解特性，在横向与纵向两个维度对架构进行职责划分。

❑ **单独的进程与轻量级通信**

在合理划分后，主要从可发现性和可交互性处理微服务间的横向关系：

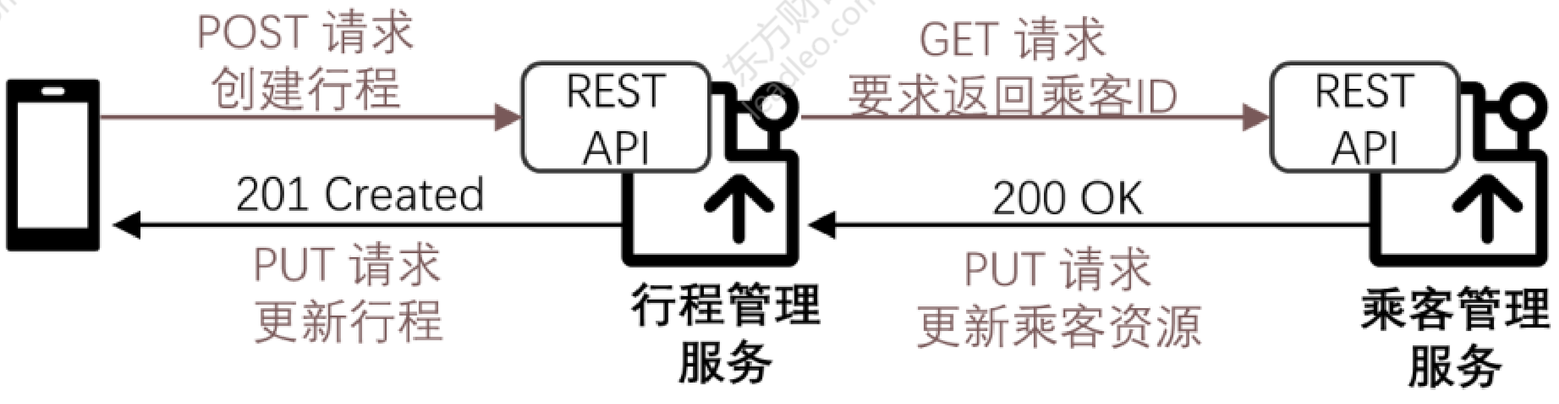
- **可发现性**：当服务A发布和扩缩容时，依赖服务A的服务B如何自动感知服务A的变化？于是引入了第三方服务注册中心来满足服务B对服务A的可发现性。服务注册中心的推送在大规模微服务集群中尤为重要。
- **可交互性**：由于服务自治与技术栈不对等的约束，服务之间的调动需要采用与语言无关的远程调用协议，比如REST协议。而在高性能场景下，基于IDL的二进制协议更优。目前业界的微服务实践中，还是需要提前约定接口来完成服务与服务之间的调用。

服务端发现模式



来源：CloudFoundry，头豹研究院

打车软件中的REST协议示例



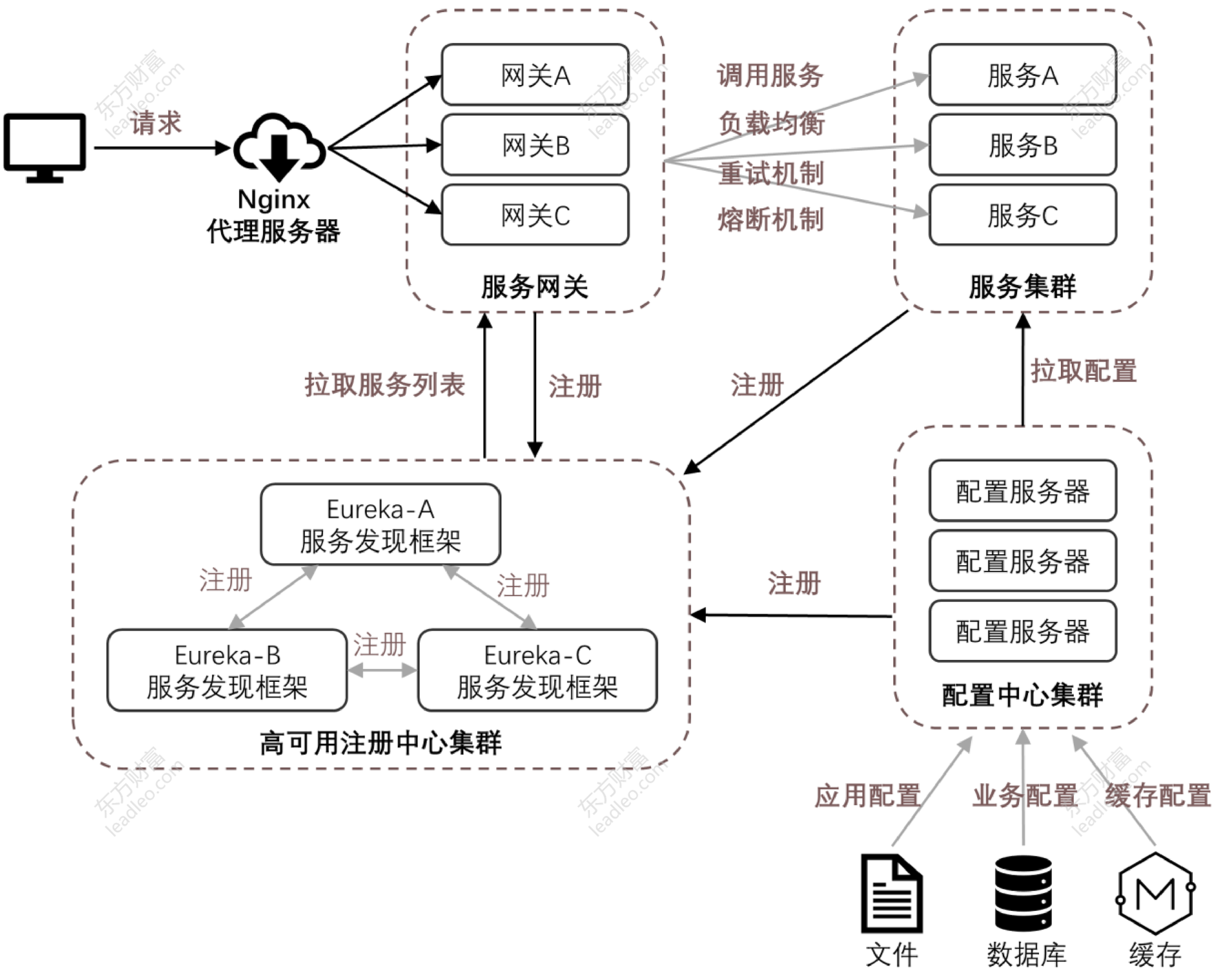
来源：CloudFoundry，头豹研究院

❑ 松耦合，独立部署

微服务采用数据存储隔离原则，即数据是微服务的私有财产，对于该数据的访问都必须通过当前微服务提供的API进行访问，从而避免了数据层产生耦合。同时，通常采用读写分离手段来提高性能和存算资源的可扩展性。

微服务的设计遵循无状态设计原则，解耦上层应用与底层基础设施，使得微服务可以在不同容器间自由地被调度。而对于有数据存取（有状态）的微服务，采用存储分离的方式，将数据下沉到分布式存储，以此达到一定程度的无状态化。

微服务运行流程



来源：kubernetes，头豹研究院

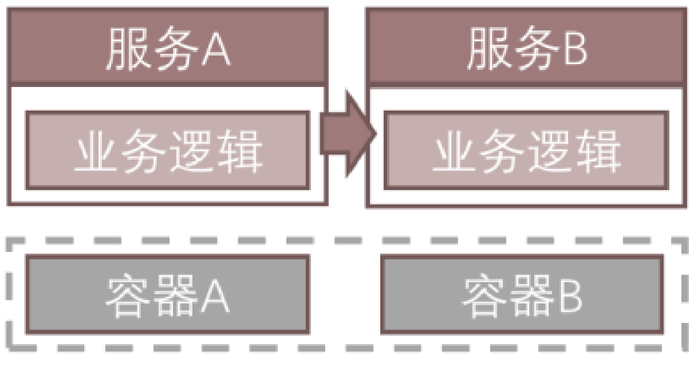
微服务技术的演进与发展

- 正如微服务是 SOA 实践中正确的部分一样，Mecha也在Mesh的实践中吸取经验和教训， 有望成为微服务落地实践的下一站，而Mecha也不会是微服务或应用架构演进的终点

❑ 第一代微服务架构

应用服务除了需要实现业务逻辑，还需要额外自行解决寻址、通讯、容错等微服务的基础能力。哪怕是同一编译语言下的另一个应用，也需要重新实现上述维服务能力。随着维服务规模扩大，通讯寻址变得复杂，第一代的微服务架构将不堪重负。

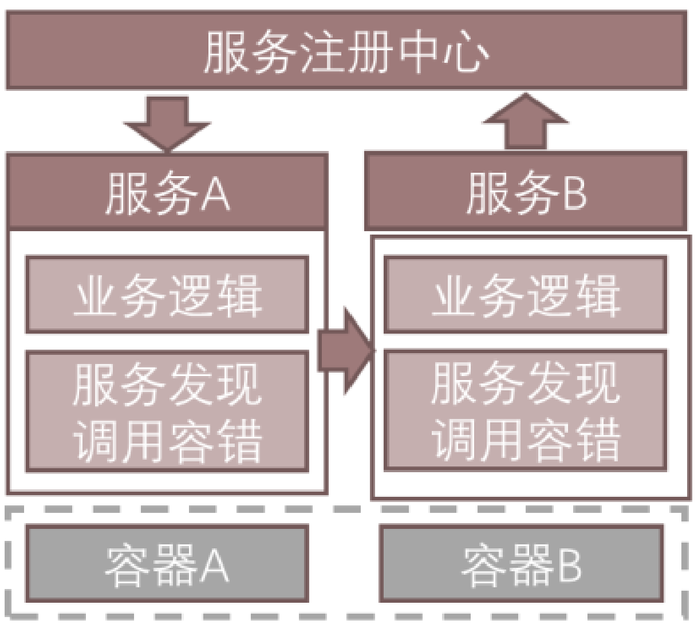
第一代微服务架构



❑ 第二代微服务架构

引入了旁路服务注册中心作为协调者来完成服务的自动注册和发现。服务之间的通讯及容错机制开始模块化，形成独立的服务框架。但随着服务框架内的功能日益增多，不同的编译语言构成的不同基础功能难以复用，违背了微服务的敏捷迭代的目的。

第二代微服务架构



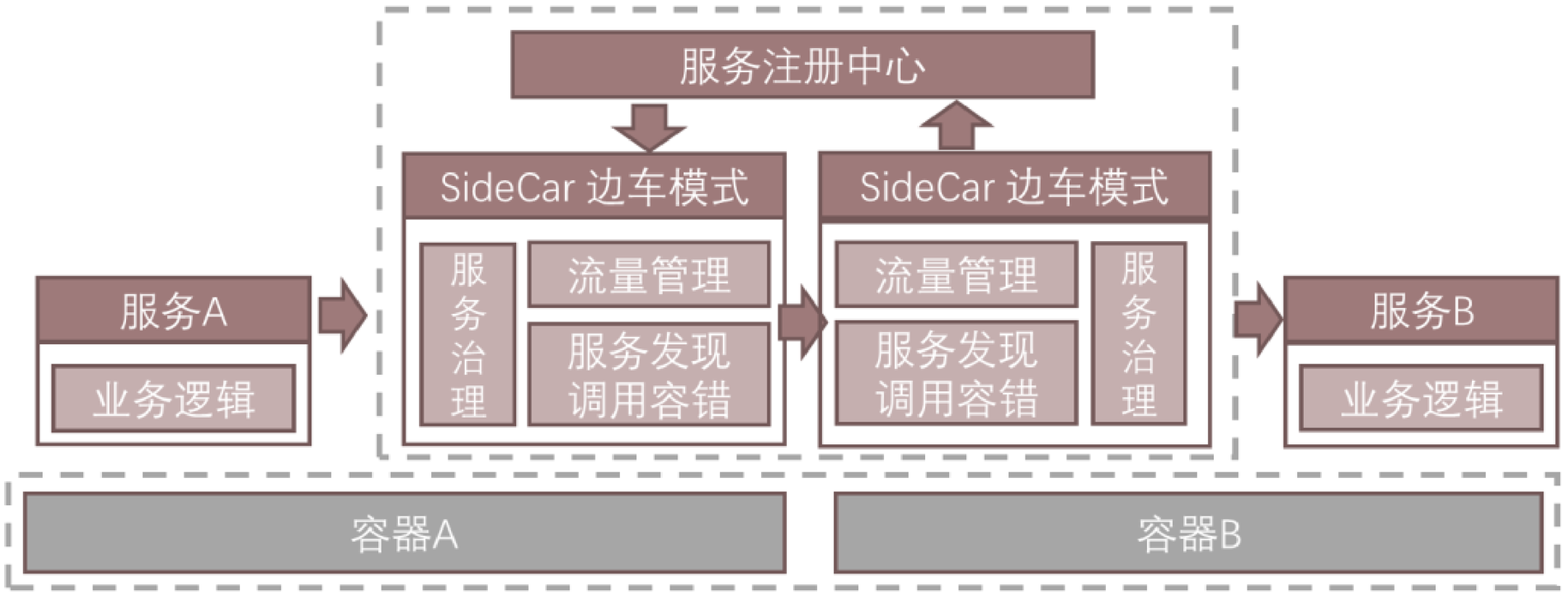
❑ 第三代微服务架构-服务网格 Service-Mesh

将微服务基础能力从服务框架模块进一步演进成一个独立进程-Sidecar，解决了第二代架构中多语言支持问题，微服务基础能力演进和业务逻辑迭代彻底解耦，真正实现了云原生微服务架构。

Sidecar进程接管了微服务应用之间的流量，承载了包括服务发现、调用容错这些服务间通讯功能，同时还支持权重路由、灰度路由、流量重放、服务伪装等服务治理功能。

来源：阿里云，头豹研究院

第三代微服务架构



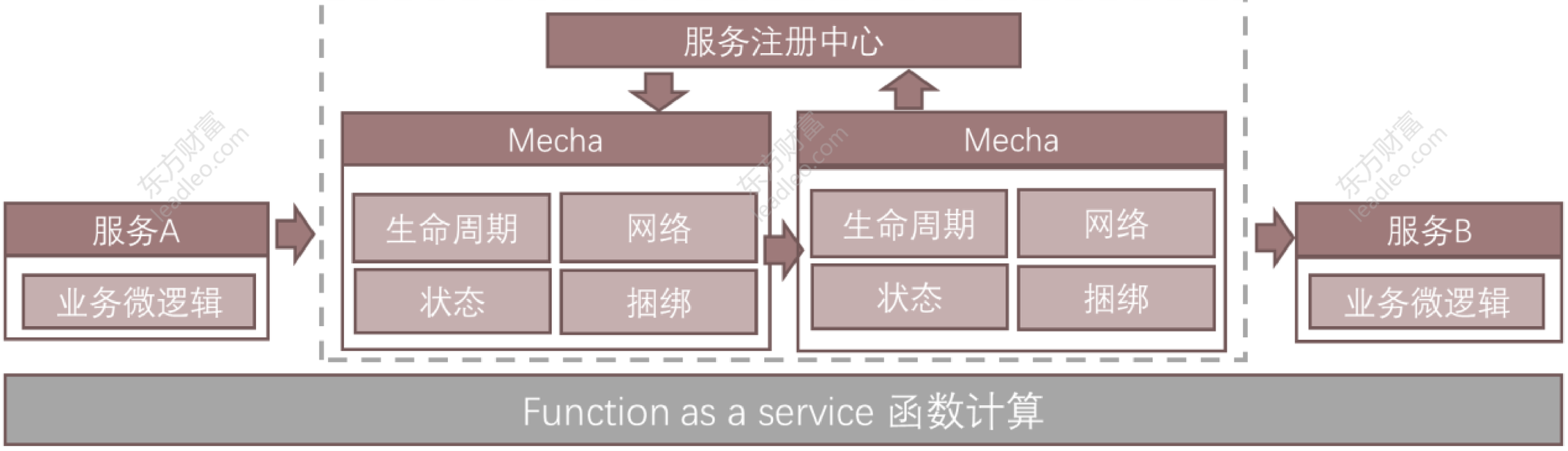
❑ **第四代微服务架构- 多运行时微服务架构 Multi-Runtime Microservices**

自AWS Lambda出现后，应用开始尝试利用Serverless无服务器来补充微服务架构，促进了没有虚拟机或者容器的微服务。当服务应用的一个组件请求更多的计算资源，低资源的事件就可以触发Serverless的函数计算，从而提供另外的组件实例，而无需创建一个独立的组件来处理这个事件。于是在此架构中，微服务进一步由一个应用简化为Micrologic微逻辑，从而对SideCar提出了更高的诉求，将更多可复用的分布式能力从应用中剥离下沉到SideCar中，例如状态管理、资源绑定、链路追踪、事务管理、安全等等。

随着Serverless资源的实现和云原生思想的影响，未来的架构趋势是将Mesh深化，应用需要的分布式能力需要继续下沉，越来越多的能力会以 Sidecar 的形式出现，但运维压力也催生了新的形态Mecha来解决SideCar过多的问题。此时，应用核心的业务逻辑（称为微逻辑）和强大的开箱即用的分布式原生功能Mecha组件共同组成多运行时微服务。

Mecha的目的是成为面向应用的分布式能力抽象层。目前还处在一个起步探索的过程，在Service Mesh 现有的固定模式之上，尝试Node模式取代SideCar，尝试用Runtime解耦底层实现取代通讯协议转发，尝试保留轻量SDK提供统一的API接口，开始提倡localhost编程的开发侧理念。

第四代微服务架构



来源：阿里云、亚马逊云、Apache, Red Hat， 头豹研究院

03

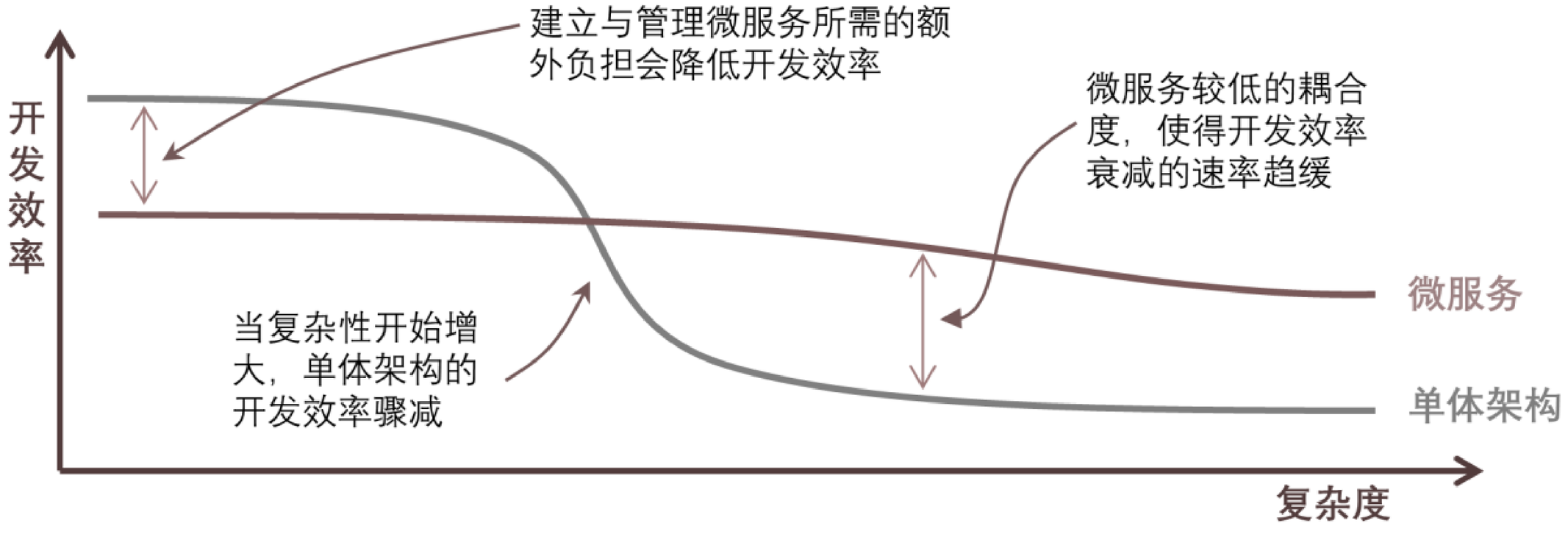
微服务的应用概况

- 微服务与企业团队架构
- 微服务的适用场景
- 2020年微服务技术应用现况图谱

微服务的适用场景

- 微服务具备敏捷开发、弹性伸缩和高可用的优势，但由于其复杂且高昂的的前期实现搭建，微服务并非适合所有场景

微服务对比单体架构的效率对比



来源：Thoughtworks，头豹研究院

❑ **适用场景一：企业规模较大，系统复杂**

微服务需要对前期基础设施投资，架构本身的额复杂性高，涉及的技术栈多。如果小企业采用微服务，首先对问题领域不了解导致很难划分服务的边界，开发的成本和风险较高，更推荐单体架构。当公司发展至业务已经到达了一定的复杂性，微服务可以提升研发和交付的效率，满足增长需求。

❑ **适用场景二：软件即服务（Software as a Service, SaaS）**

SaaS具有高度场景化、关注用户体验和短流程的特点。用微服务架构提供SaaS服务，可以同时高度满足定制开发、可配置、多租户、高性能、伸缩性的最高级别成熟度模型。

虽然微服务架构下的SaaS需要面对较高的运维、前期领域驱动设计和DEVOPS要求所带来的成本，但微服务能够提供容器化独立部署，可以直接被外部或其他服务调用，将每一个碎片化的SaaS服务场景由微服务平台统一管理和编排。

❑ **适用场景三：混合云/多云架构应用管理**

混合云和微服务架构各自的性质契合，凸显了微服务在混合IT架构管理方面的适用性：混合云的各项服务相互响应的能力必须满足松耦合，而其也同样遵循API控制、负载均衡等管理基础技术。

❑ 适用场景四：IT架构云端迁移

迁移效率和可用性是大型企业IT架构云端迁移关注的重点。由于涉及基础设施架构的跨度较大，IT系统在从金属机向虚拟机迁移过程中，如何快速高效得实现应用与基础设施的解耦是关键的问题。

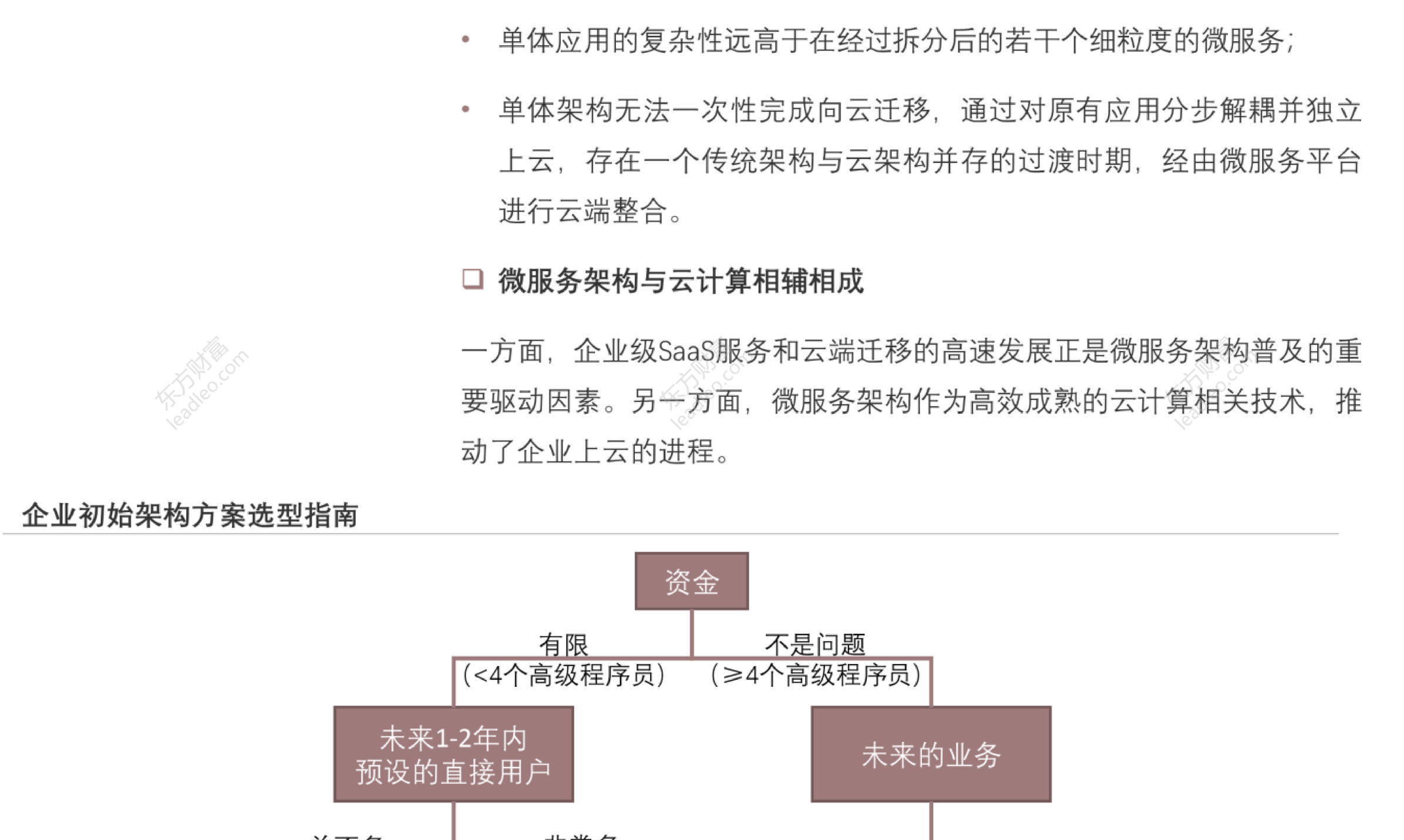
微服务架构相较于单体架构能够极大地提升云端迁移的效率，原因如下：

- 单体应用的复杂性远高于在经过拆分后的若干个细粒度的微服务；
- 单体架构无法一次性完成向云迁移，通过对原有应用分步解耦并独立上云，存在一个传统架构与云架构并存的过渡时期，经由微服务平台进行云端整合。

❑ 微服务架构与云计算相辅相成

一方面，企业级SaaS服务和云端迁移的高速发展正是微服务架构普及的重要驱动因素。另一方面，微服务架构作为高效成熟的云计算相关技术，推动了企业上云的进程。

企业初始架构方案选型指南



来源：头豹研究院

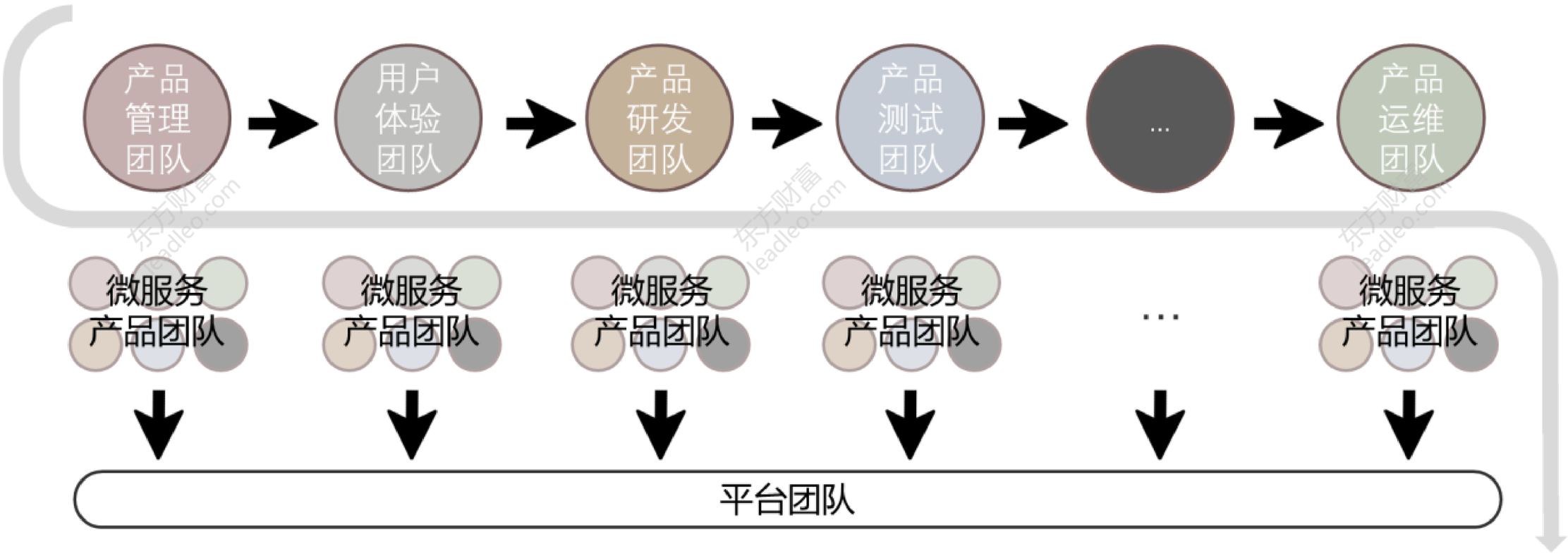
微服务与IT团队架构

- 微服务通过领域驱动设计完成业务拆分，协同IT团队的架构调整与平衡系统架构的设计，逐步完善微服务架构改造

❑ 应用架构与IT团队架构同步演进

在传统单体架构下，企业中不同的IT团队以不同的职能划分，通过不同团队之间的协作共同开发项目。而微服务架构改造后，原有的IT系统被拆分成大量的微服务组件，需要围绕每一个独立的微服务能力组建一个从开发到运维的完整团队，从而满足“康威定律”保持技术架构与企业架构一致。

微服务下IT团队架构的变化



来源：头豹研究院

❑ 领域驱动设计

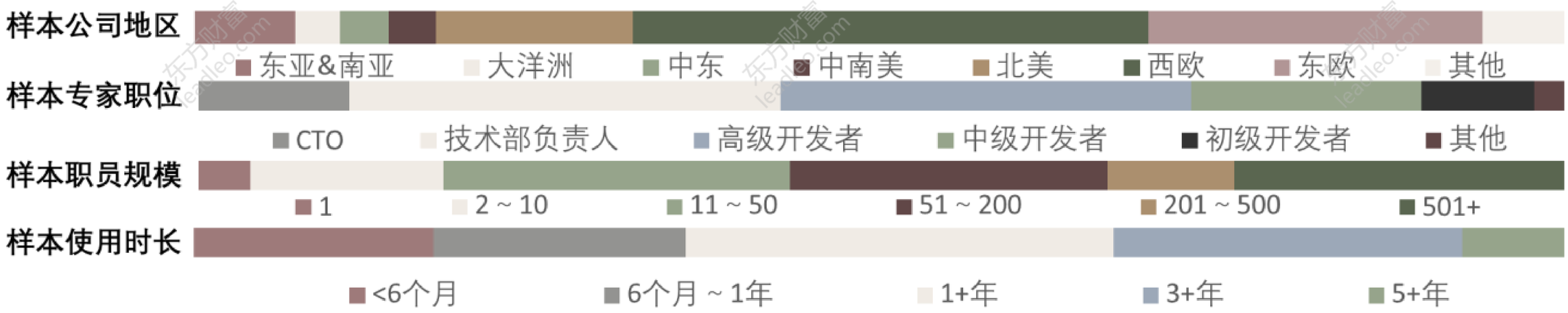
对原有系统的有效拆分是微服务实现的关键，在微服务设计过程中，要先对企业原有的业务和架构进行梳理和规划，据此进行领域驱动设计（Domain Driven Design，DDD）。DDD追求精确反映领域中某一知识元素的载体，通过与领域专家进行频繁的沟通将专业知识转化为领域模型。

❑ 落地方式

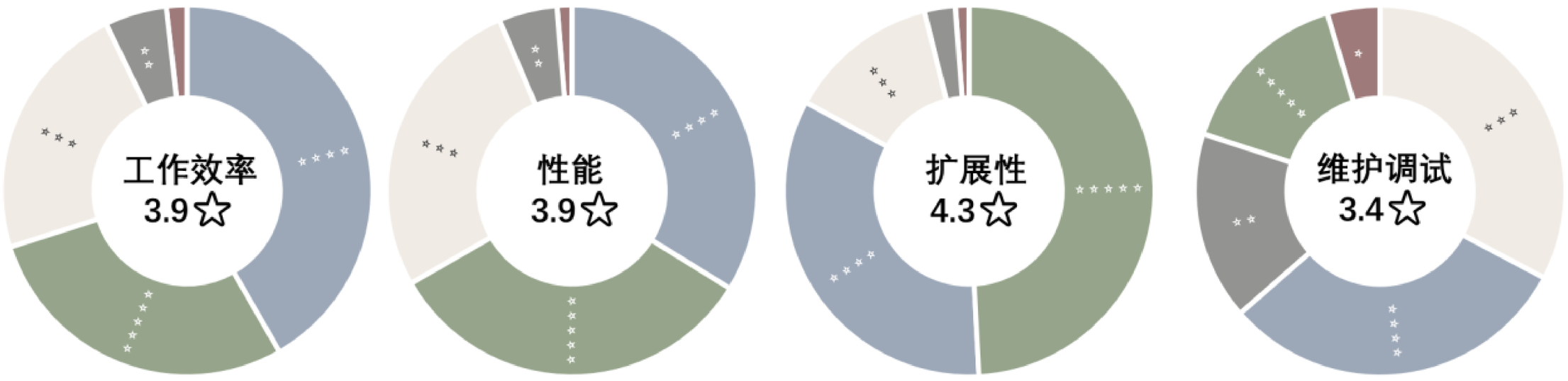
不同行业及不同企业中的业务拆分方案不同，微服务设计要求不同的对包括团队架构和技术架构的实现过程，体现了微服务落地存在极高的个性化特色，兼具行业壁垒和技术壁垒。落地方案有两种：

- 借力外部架构咨询公司提供架构DEMO和培训服务助推内部技术团队。
- 招聘相关经验丰富的人员进来，自行研究和搭建架构并做内部培训。

2020年微服务技术应用现况图谱



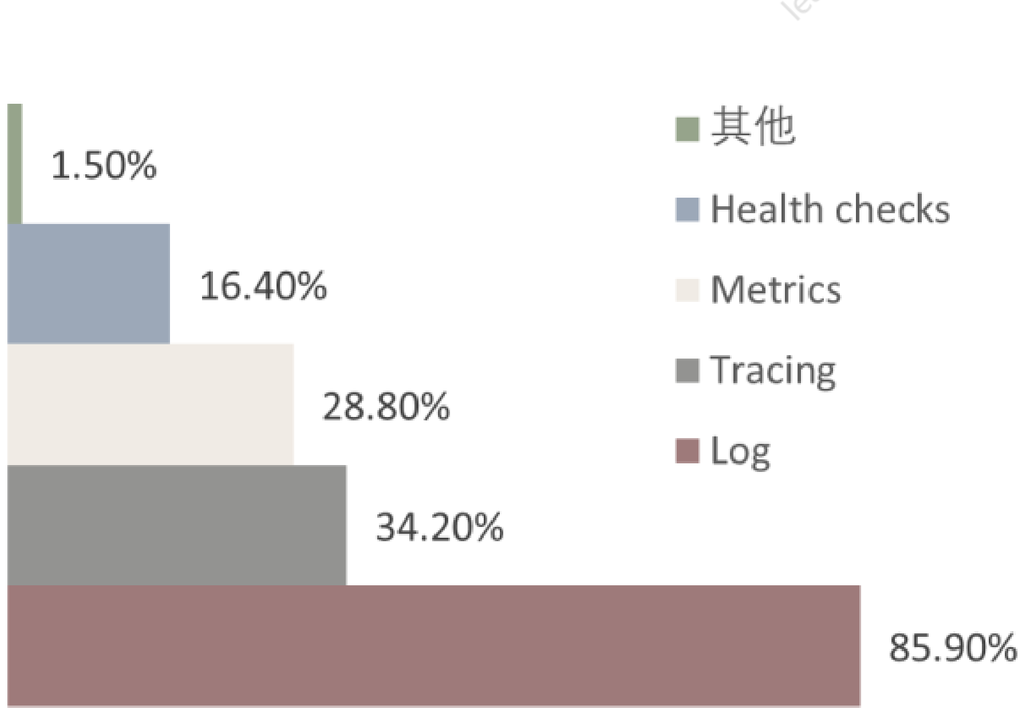
微服务技术成熟度评分



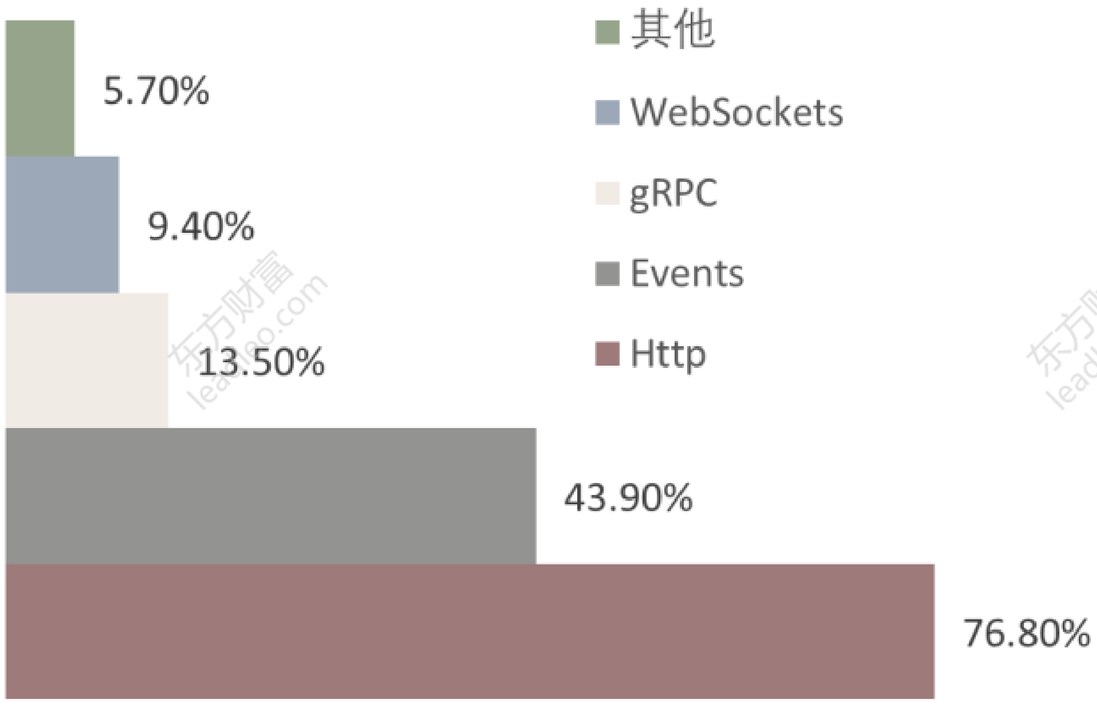
微服务编译语言选择

https://www.leadleo.com/sizepro/details?id=6109f0ddfa08a02df8f0fcbd&core=611c3faa0dd177911b516f39

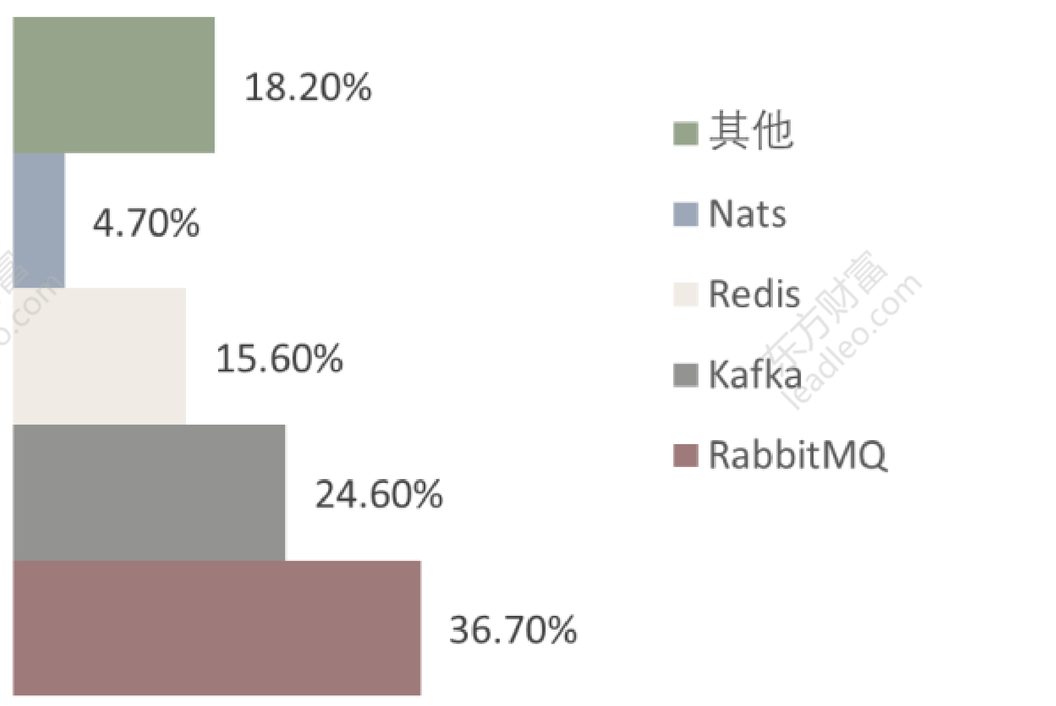
微服务调试方法选择



微服务通信协议选择



微服务消息代理选择



来源：The Software House，O'Reilly，Neal Ford，头豹研究院

04

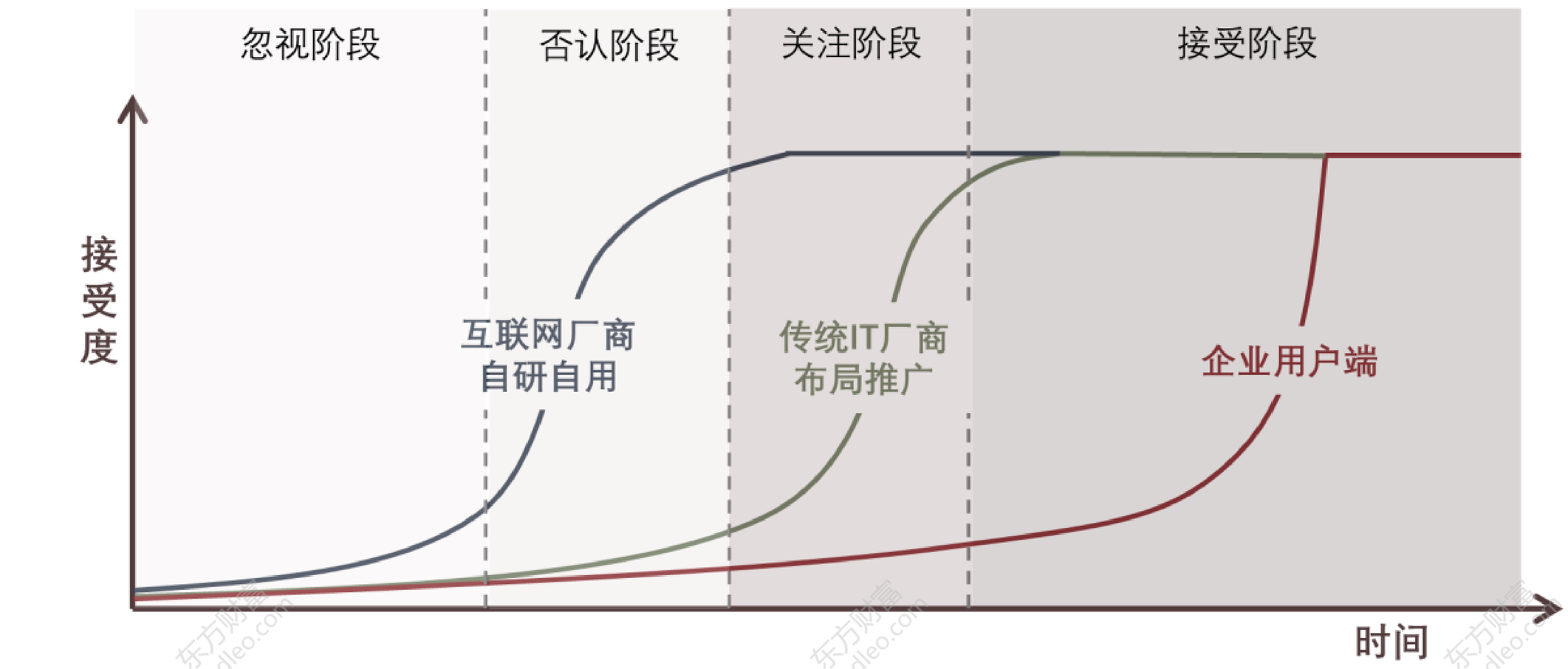
微服务的市场概况

- 微服务市场发展驱动因素分析
- 微服务市场参与者梳理
- 微服务市场参与者图谱

微服务市场发展驱动因素分析

- 互联网厂商、传统IT厂商以及开源微服务项目为企业端进行微服务架构改造提供了多种成熟稳定的方案，推动了微服务行业向各行业场景进一步积累技术储备并开拓企业用户资源

企业用户对微服务技术的接受度曲线及阶段



来源：Gardeviance, 头豹研究院

❑ 大量成熟的微服务供应商可供企业用户选择

互联网厂商和传统IT厂商已完成前期的技术积累和应用试错经验，在企业IT开始考虑进行微服务架构改造时，微服务供应商已可提供大量成熟的微服务应用案例并为企业用户提供相关的咨询与实施服务。

❑ 微服务领域的开源数字化

Quarkus 作为云原生微服务框架，全球 GitHub 开源项目活跃度中排名 40；Spring 作为 Java 微服务框架事实标准，在微服务框架中活跃度位列二三；Apache Dubbo 作为中国本土开源的项目，微服务框架活跃度排名第五。微服务的活跃开放，促进了标准与接口的统一，推动了微服务行业的发展。

2020上半年微服务项目在GitHub所有开源项目的活跃度排名

项目	全球排名	加权活跃度	项目	全球排名	加权活跃度
Quarkusio/Quarkus	40	3684.58	spring-cloud-alibaba	1674	461.16
Spring-cloud	/	3222.89	spring-cloud-gcp	1953	416.34
spring-projects/spring-boot	203	1704.33	spring-cloud-netflix	4969	221.44
micronaut-projects	/	1393.23	spring-cloud-aws	4976	221.34
apache/dubbo	693	801.63	spring-cloud-huawei	20845	76.47

来源：GitHub, X-Lab, 头豹研究院

微服务架构转型需求激增

微服务技术在互联网公司快速发展的过程中已经发挥了重要的作用。Netflix在2008年因为全站瘫痪被迫停业3天后，开始进行服务化架构改造，经过十余年的开发实现了从单体架构到微服务的变迁，支撑了业务的千倍增长，是微服务架构的先驱，带动了国内外互联网领域的微服务化。

在传统行业中，包括工业制造、交通物流、金融、零售、地产、医疗等领域正面临大量新兴业务场景的出现，服务变得复杂，单体架构已不能满足业务发展对迭代速度、可用性、流量峰值和系统互通的要求，微服务化需求明显。

云计算技术红利吸引

底层硬件的迭代和顶层业务发展诉求不断升级共同驱动了计算机软件技术架构的进化。云计算进入工业化应用后，用虚拟机替代了物理机，如今的容器技术与Serverless的融合跨代际得实现了计算密度的升级，提供了最佳的性能功耗比和性能价格比，大幅降低了用户使用门槛的同时吸引着各行各业的IT团队对其上层应用微服务化以享受云计算带来的技术红利，其灵活性与成本优势加速了数据服务的云原生化。

微服务架构贴合未来软件技术发展趋势

微服务架构具有两大特点：

1. **独立通讯 Transport Independence**：意味着消息如何传输是无关紧要的，消息传递模型的级别 是同步或异步无关紧要。重要的是，消息成为了唯一的界面，通过组合服务，大大减少了组件的集成面。

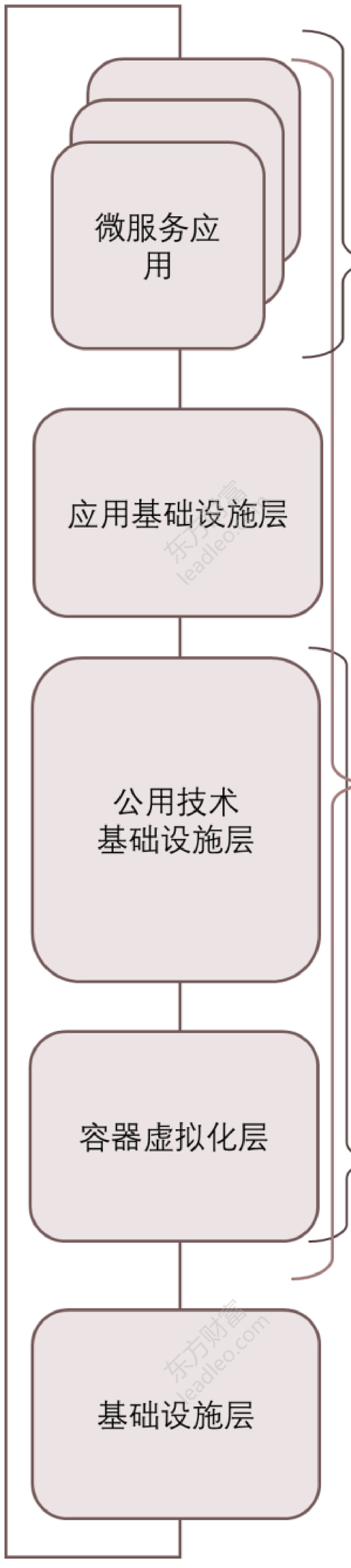
2. **无身份 Zero Identity**：微服务和组件不得互相了解。消息简单地发送和接收，无需考虑目的地。这种方法使动态的实时修改系统成为了可能。在一些实例中，仍能看到依赖于嵌入在服务中的身份，而这是一个最大的错误，导致了大多数微服务的崩溃。

有了这两个特点作为微服务的底层原则，或许在未来将看到“微服务”这个词会消失，意味着它们将真正融入成为软件开发的主流架构。兴许在未来还会出现Macro-service或Nano-service命名的新一代应用架构，但微服务架构带来的理念和技术积累将继续沉淀。

微服务市场参与者梳理

- 微服务市场吸引了互联网厂商、云服务厂商、传统IT实施服务商、应用软件开发厂商及初创企业参与布局，从提供的产品或服务的视角区分，微服务市场参与者主要分三类

微服务市场参与者分类



参与者描述	产品/服务
微服务应用软件提供商 以应用软件开发厂商为主， 为企业用户提供基于微服务架构的软件应用产品。基于企业现存IT系统，以全局视角交付微服务应用。	满足工业智造转型、金融科技创新等不同领域的场景需求，将传统业务抽象为一个个业务服务，通过微服务架构加快迭代速度、提升可用性、灵活应对峰值、促进系统互通。 1.企业采用微服务架构，微服务应用按照统一的API接口集成。 2.企业采用SOA架构，微服务应用以整体软件形式通过ESB统一管理。
综合型咨询实施服务提供商 以微服务实施初创企业与传统IT实施服务商为主， 提供定制化设计及实施方案包括开发落地及后续运维工作的全栈式服务。	**前期咨询服务**：引入外部技术专家，协助业务架构有效拆分，以项目制收费。 **定制化平台开发**：进行系统对接迁移、DevOps架构设计，前期实施以项目制收费。 **应用部署实施**：通过开发测试运维平台实现自动化发布和传统应用迁移，项目制收费。 **平台及应用运维**：对定制化平台、云原生应用进行升级、运维托管，以订阅方式收费。
通用型容器平台供应商 以互联网厂商、云服务厂商为主， 提供实现容器运行及管理的能力，其中不包含实现业务功能的中间件或微服务应用组件，涉及较少下游需求理解。	**容器服务**：提供高性能可伸缩的容器应用管理，提供面向IaaS、网络、微服务的完整监控报警能力。 **应用PaaS**：提供应用托管、支持多语言互通、流量和稳定性治理、微服务治理。 **消息代理**：构建应用异步化、实现低延迟消息、开箱即用兼容主流开源版本。 **性能测试工具**：提供性能测试、API调试、系统监控等功能

来源：阿里云、网易数帆，头豹研究院

微服务市场参与者图谱

	公司名	相关产品	相关介绍
微服务应用软件提供商	Kingdee金蝶 云管理，触手可及	金蝶云苍穹平台 金蝶SaaS平台 各场景云应用	创立于1993年，包含服务、财务、人力、采购、制造、销售、物流、渠道，为企业级应用提供场景化平台服务。 **客户案例：** 建发集团、温氏集团、光峰科技
	Neusoft东软®	神州新桥 微服务支撑平台MSP	创立于1991年，业务领域覆盖智慧城市、医疗健康、智能汽车互联及软件产品与服务。 **客户案例：** 华夏银行、广州电视台、广东法院
	信雅达 SUNYARD 金融IT创新先锋	信审系统 资产负债系统5.0 智能风控决策引擎	创立于1996年，以金融作业和管理信息化建设作为主营业务发展方向提供系统软件与服务。 **客户案例：** 宁波银行、晋中银行、中国外贸信托
	SAP Hybris　金证股份　中百信 ZBX　YSSTECH 赢时胜软件 YSSTECH SOFTWARE　RedXun		
综合型咨询实施服务提供商	灵雀云 alauda.cn	云原生平台 ACP 3.0 Alauda Kubernetes ACE企业级容器	成立于2014年，创始团队出自Azure。 容器PaaS领域有着多年经验;咨询与服务贯穿企业IT应用的全生命周期。 **客户案例：** 中石油、中国移动、民生银行
	accenture	埃森哲中国信息技术交付中心（CDC） SaaS、PaaS业务整合	与容器平台提供商合作，从应用架构、敏捷改造、DevOps、运维，涵盖了全生命周期的咨询。 **客户案例：** 雷诺日产、蛇口集装箱码头
	YSSTECH 赢时胜软件 YSSTECH SOFTWARE	智能架构服务AIAS 资产管理系统AMS 资产托管系统ACS	创立于1998年，金融IT解决方案综合服务商。 2016年开发基于微服务架构的PaaS云平台体系，综合应用云计算、大数据、AI等技术实现生态环 **客户案例：** 中国银行、中国农业银行、交通银行
	用友 yonyou　DaoCloud　博云 BoCloud　时速云　THUNISOFT 华宇　天玑科技 Pioneer in IT Service		
通用型容器平台供应商	阿里云	容器服务ACK 分布式应用服务EDAS MSE微服务引擎 ACM应用配置管理	EDAS 是一个面向微服务应用的应用全生命周期PaaS平台，支持HSF、Dubbo、Spring Cloud技术体系，提供ECS集群和K8s集群的应用开发、部署、监控、运维等全栈式解决方案。
	华为云	云容器引擎 CCE 微服务引擎 CSE 应用编排服务 AOS 管理运维ServiceStage	ServiceStage针对微服务解决方案提供了一站式管理平台：使用微服务引擎，令开发者专注于业务开发；使用DTM分布式事务来解决微服务复杂的本地事务，保证全局的数据一致性。
	亚马逊云科技	Amazon ECS AWS Lambda Amazon EventBridge Amazon AppFlow	Amazon ECS 是一个完全托管的容器编排服务。可帮助您轻松部署、管理和扩展容器化的应用程序或构建微服务。它与 AWS 平台的其余部分深度集成，可提供安全、易于使用的解决方案。
	腾讯云　金山云　网易数帆　IBM Cloud　Microsoft Azure		

来源：各公司官网，头豹研究院

方法论

- 头豹研究院布局中国市场，深入研究10大行业，54个垂直行业的市场变化，已经积累了近50万行业研究样本，完成近10,000多个独立的研究咨询项目。
- 研究院依托中国活跃的经济环境，从社会保险、人工智能、大数据等领域着手，研究内容覆盖整个行业的发展周期，伴随着行业中企业的创立，发展，扩张，到企业走向上市及上市后的成熟期，研究院的各行业研究员探索和评估行业中多变的产业模式，企业的商业模式和运营模式，以专业的视野解读行业的沿革。
- 研究院融合传统与新型的研究方法，采用自主研发的算法，结合行业交叉的大数据，以多元化的调研方法，挖掘定量数据背后的逻辑，分析定性内容背后的观点，客观和真实地阐述行业的现状，前瞻性地预测行业未来的发展趋势，在研究院的每一份研究报告中，完整地呈现行业的过去，现在和未来。
- 研究院密切关注行业发展最新动向，报告内容及数据会随着行业发展、技术革新、竞争格局变化、政策法规颁布、市场调研深入，保持不断更新与优化。
- 研究院秉承匠心研究，砥砺前行的宗旨，从战略的角度分析行业，从执行的层面阅读行业，为每一个行业的报告阅读者提供值得品鉴的研究报告。

法律声明

头豹研究院简介

- 头豹是中国领先的原创行企研究内容平台和新型企业服务提供商。围绕“协助企业加速资本价值的挖掘、提升、传播”这一核心目标，头豹打造了一系列产品及解决方案，包括：**报告/数据库服务**、**行企研报服务**、**微估值及微尽调自动化产品**、**财务顾问服务**、**PR及IR服务**，以及其他企业为基础，利用大数据、区块链和人工智能等技术，围绕产业焦点、热点问题，基于丰富案例和海量数据，通过开放合作的增长咨询服务等
- 头豹致力于以优质商业资源共享研究平台，汇集各界智慧，推动产业健康、有序、可持续发展

四大核心服务

研究咨询服务

为企业提供**定制化报告**服务、**管理咨询**、**战略**调整等服务

企业价值增长服务

为处于不同发展阶段的企业，提供与之推广需求相对应的“**内容+渠道投放**”一站式服务

行业排名、展会宣传

行业峰会策划、**奖项**评选、行业**白皮书**等服务

园区规划、产业规划

地方**产业规划**，**园区**企业孵化服务

报告阅读渠道

头豹官网 —— www.leadleo.com 阅读更多报告

头豹小程序 —— 微信小程序搜索“头豹”、手机扫上方二维码阅读研报

添加右侧头豹分析师微信，身份认证后邀您进入行研报告分享交流微信群

详情咨询

客服电话

400-072-5588

上海

王先生： 13611634866

李女士： 13061967127

深圳

李先生： 18916233114

李女士： 18049912451

南京

杨先生： 13120628075

唐先生： 18014813521

头豹 Project Navigator 领航者计划介绍

每个季度，头豹将于网站、公众号、各自媒体公开发布**季度招募令**，每季公开

125个

招募名额

头豹诚邀**政府及园区、金融及投资机构、顶流财经媒体及大V**推荐共建企业

头豹共建报告

2021年度特别策划

Project Navigator

领航者计划

头豹诚邀各行业**创造者、颠覆者**

领航者

知识共享、内容共建

头豹邀请**沙利文**担任计划**首席增长咨询官**、**江苏中科院智能院**担任计划**首席科创辅导官**、**财联社**担任计划**首席媒体助力官**、**无锋科技**担任计划**首席新媒体造势官**、**iDeals**担任计划**首席VDR技术支持官**、**友品荟**担任计划**首席生态合作官**

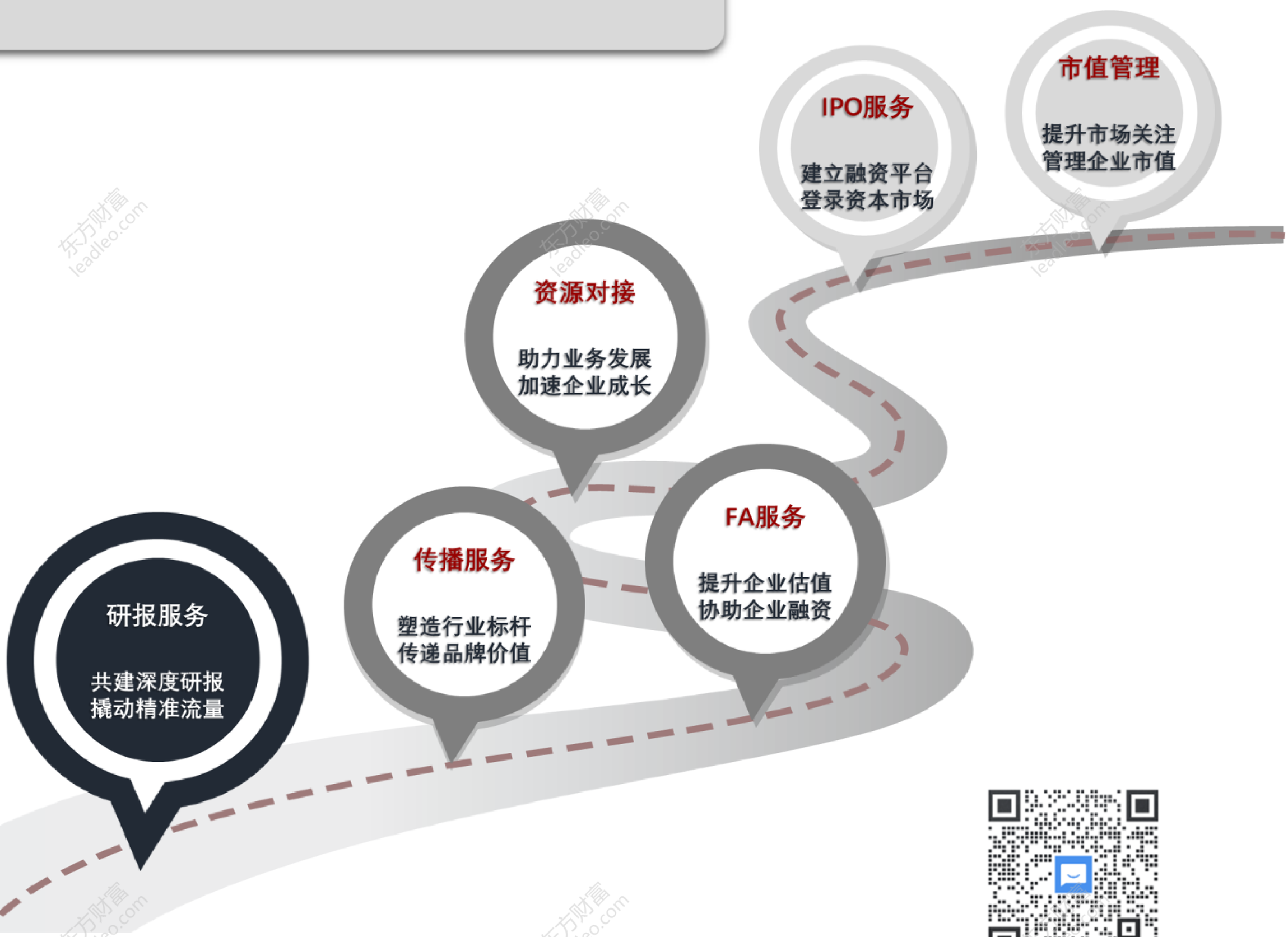
共建报告流程

企业申请共建 → 头豹审核资质 → 确定合作细项 → 报告发布投放 → 信息共享、内容共建

备注：活动解释权均归头豹所有，活动细则将根据实际情况作出调整。

头豹 Project Navigator 领航者计划与商业服务

- 头豹以**研报服务**为切入点，根据企业不同发展阶段的资本价值需求，以**传播服务**、**FA服务**、**资源对接**、**IPO服务**、**市值管理**为基础，提供适合的**商业管家服务解决方案**



扫描上方二维码

联系客服报名加入

备注：活动解释权均归头豹所有，活动细则将根据实际情况作出调整。

读完报告有问题？

快，问头豹！你的智能随身专家

扫码二维码
即刻联系你的智能随身专家

千元预算的

高效率轻咨询服务

STEP01 智能拆解提问

人工智能NLP技术
精准拆解用户提问

STEP02 云研究院后援

云研究院7×24待命
随时评估解答方案

STEP03 解答方案生成

大数据×定制调研
迅速生成解答方案

STEP04 专业高效解答

书面反馈、分析师专访、
专家专访等多元化反馈方式

头豹 LeadLeo
www.leadleo.com
400-072-5588