云原生、DevOps和平台工程都是十分繁杂的概念，其内涵不断延伸，有许多重合的部分。尽管如此，三者的出发点却并不相同，但却可以形成有机的整体。

云原生

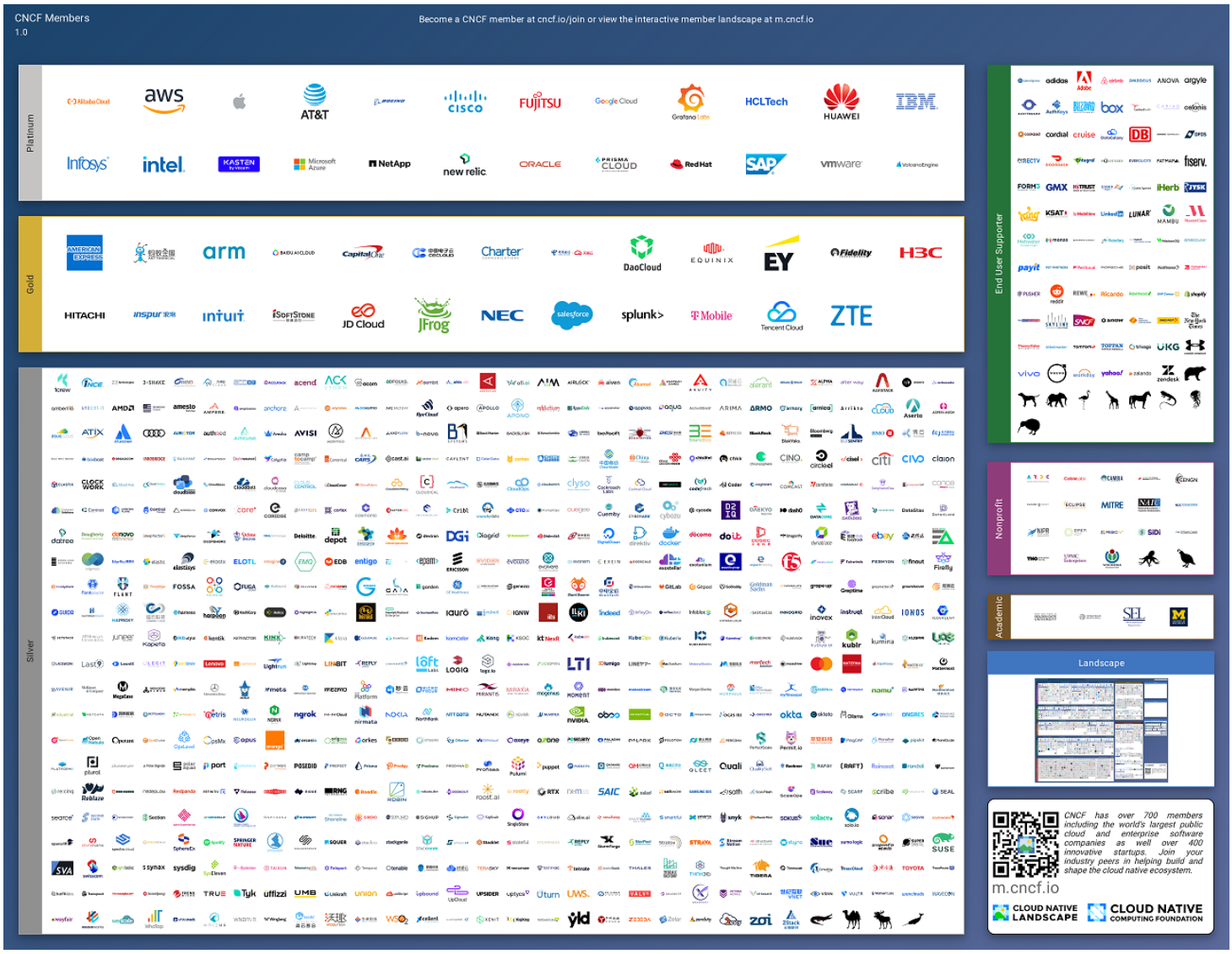
云原生基金会关于云原生的定义如下：

> 云原生技术有利于各组织在公有云、私有云和混合云等新型动态环境中，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、不可变基础设施和声明式API。
>
> 这些技术能够构建容错性好、易于管理和便于观察的松耦合系统。结合可靠的自动化手段，云原生技术使工程师能够轻松地对系统作出频繁和可预测的重大变更。

可以看到云原生的目的是“**够构建容错性好、易于管理和便于观察的松耦合系统**”，而采用的方法包括“**容器、服务网格、微服务、不可变基础设施和声明式API**”，而这一些还有个前提，需要是在“**在公有云、私有云和混合云等新型动态环境中**”。

这不像一个定义，倒是像一个“万能”盒子，似乎可以装任何“正确的”方法。不过这个定义与实际情况匹配，开发与运维面对的问题很多，并没有单一方法来解决。人们在实践中可能发现某个工具很好用，然后渐渐发现其背后的思想是成功的关键，于是这个思想被挖掘出来成为一种方法论，然后衍生出更多的工具。一个很棒的例子就是 Kubernetes 的声明式 API，以及其实现方式 Operator，许许多多的工具，都借用了这个思想，实现了自己的 Operator，共同组成了了一个松耦合的复杂的系统。这些方法最后成为了云原生的一个方法。

可以看出来，“云原生”只是告诉你这有一套方法很好。工具提供者可以用这个方法开发工具，最终用户应该用云原生的工具。但是基于这个思想的工具太多了，该如何选择组合成了一个大问题，所以云原生基金会给大家展示了这样一个图景：



可怕，所以做好云原生的一个关键问题是如何选好工具，这就要求 CTO 和架构师需要持续关注云原生解决方案，需要更多的同行沟通，这也是为什么我们的公众号“云云众生s”为什么要一直给大家介绍一些工具。

不过，任何事情想做好，都不仅仅是技术的问题，许多云原生践行者，尤其是在传统企业实践的同僚一定对此深有体会。云原生聚焦于技术，并没有太多的意识形态，但不知不觉中技术改变了人的协作关系，由此引发的问题需要有别的方法来解决。

DevOps

与云原生相比，DevOps 有着众多存在差异的定义，先看看 Atlassian 的定义：

> DevOps 是一套实践、工具和文化理念，可以实现软件开发团队和 IT 团队之间的流程自动化和集成。它强调团队赋能、跨团队沟通和协作以及技术自动化。

> DevOps 运动始于 2007 年左右，当时软件开发和 IT 运营社区开始担忧传统的软件开发模式。在此模式下，编写代码的开发人员与部署和支持代码的运营人员会独立工作。DevOps 这一术语由“开发”和“运营”两个词构成，它反映了将这些领域整合为一个持续流程的过程。

可以看到，相对于云原生比较关注方法和工具，DevOps 包含了“**实践、工具和文化理念**”三个方面，但是其突出的价值观是什么，可以从其起源追溯。Atlassian 在 DevOps 历史中写到：

> 尽管敏捷开发方法已兴起，但多年来，开发团队和运营团队仍然处于孤立状态。DevOps 是协作工具和实践的进一步发展，以更快速地发布更优质的软件。

关于 DevOps 的起源，马丁·福勒的 Bliki 中有以下描述：

> Agile software development has broken down some of the silos between requirements analysis, testing and development. Deployment, operations and maintenance are other activities which have suffered a similar separation from the rest of the software development process. The DevOps movement is aimed at removing these silos and encouraging collaboration between development and operations.

按照此文的说法，DevOps 解决的问题是“敏捷开发”的不同角色的竖井问题，而不仅仅是运维和开发两个组织的沟通协调问题。此外，”敏捷开发“这个词也隐含的说明 DevOps 的前提是敏捷开发，即使不是，也可能如 ThoughtWorks 网站上所提到的需要考量的因素：

> 而且，DevOps 并不是打从一开始就适合每个组织——组织可能必须要做出一些改变。应时刻将康威定律铭记于心——这是一句格言，指出组织所设计的系统反映了组织自身的沟通结构。 这意味着，如果您使用一个大型单体应用程序来运行绝大多数业务关键应用程序，那么 DevOps 可能不适合您的组织。DevOps 最适合那些能够将工作分解成一个团队可以拥有的若干离散小块的组织。

确实如此，并不是每个组织都能开展 DevOps。一个组织的部门墙，越难以实施 DevOps。

尽管不同的组织对于 DevOps 的定义有许多差异，但从前面的描述中，我们也能发现 DevOps 是敏捷开发的延伸。较早采用 DevOps 的组织本身的部门墙不高，开发人员的眼界和能力都比较好，所以能够快速的吸收运维方面先进的最佳实践，DevOps 实践也能实现比较好的效果。

但是对于大多数组织来说，未必有着这样高能的开发人员，由”Dev“人员推动的”Ops“可能并不专业，再加上高高的部门墙，推动 DevOps 阻力非常大。

面对 DevOps 在传统企业的这种不适，许多厂商也推出了改良版的 DevOps 理念。或者打着 DevOps 的旗号推出了许多 DevOps 产品，这些产品回避了 DevOps 中”麻烦“的部分，聚焦于工具，或多或少发挥了一些作用，尤其是在开发团队中，但是往往没有把能力延伸到运维阶段。

还有许多组织，成立了一个专门的 ”DevOps 团队“，负责DevOps工具和实践的开展。这样可以，但是这与开发团队把控自己运维的原则有一些背离。

面对企业级研发的现状，需要有一个新的指导方法来解决 DevOps 的问题。

平台工程

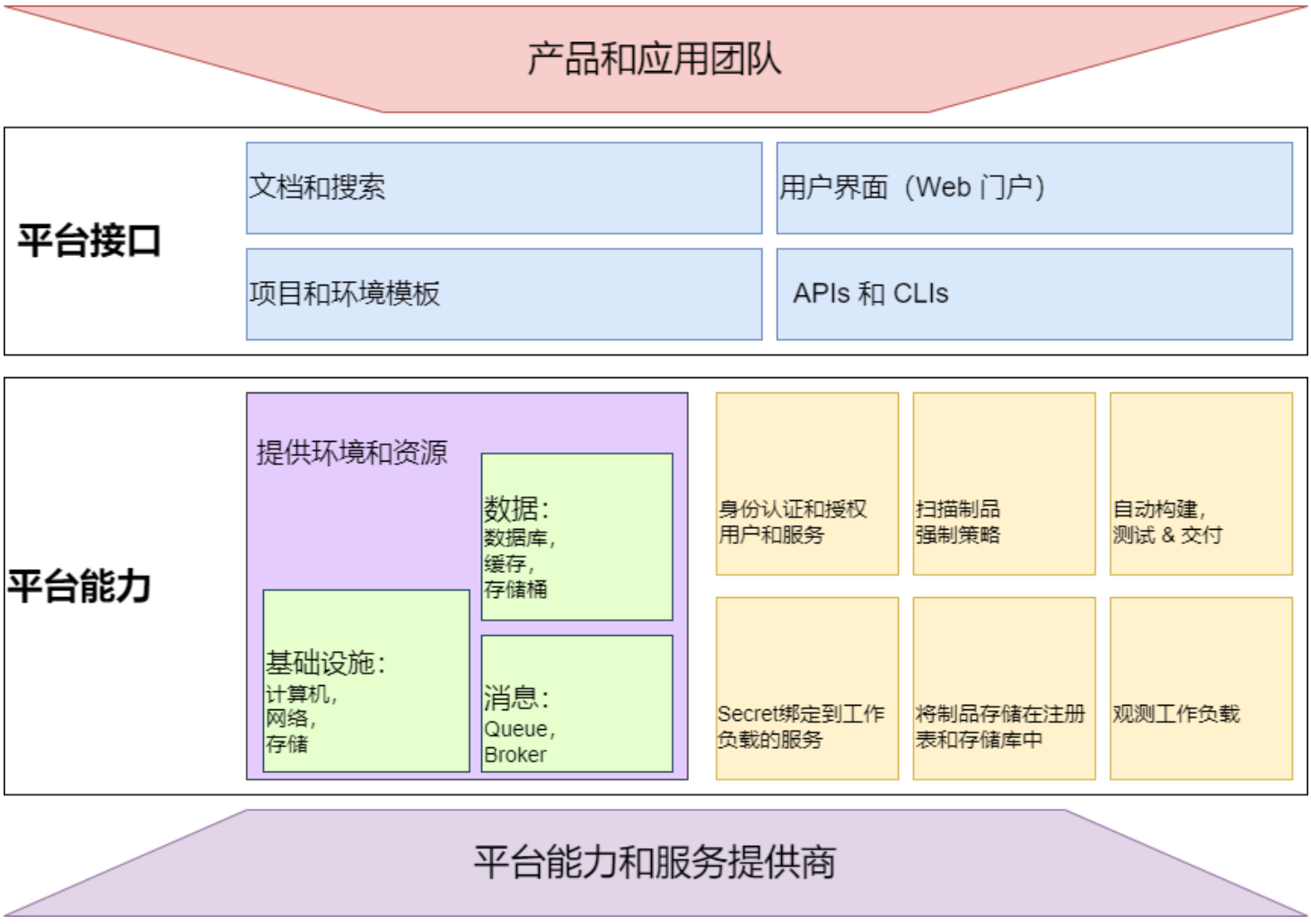
DevOps 运动已经有了 15 年的历史了，但对于大多数企业来说，并没有实现全面的 DevOps，也没有享受到其好处。这种自治的 DevOps 团队不仅没有改善组织的交付水平，反而带来了许多安全问题。

另一方面，随着云原生技术的发展，创建”平台“不再是一件难事以一个围绕平台的社会技术管理实践应运而生，这就是平台工程。

根据 CNCF 平台工程白皮书(翻译)的说法：

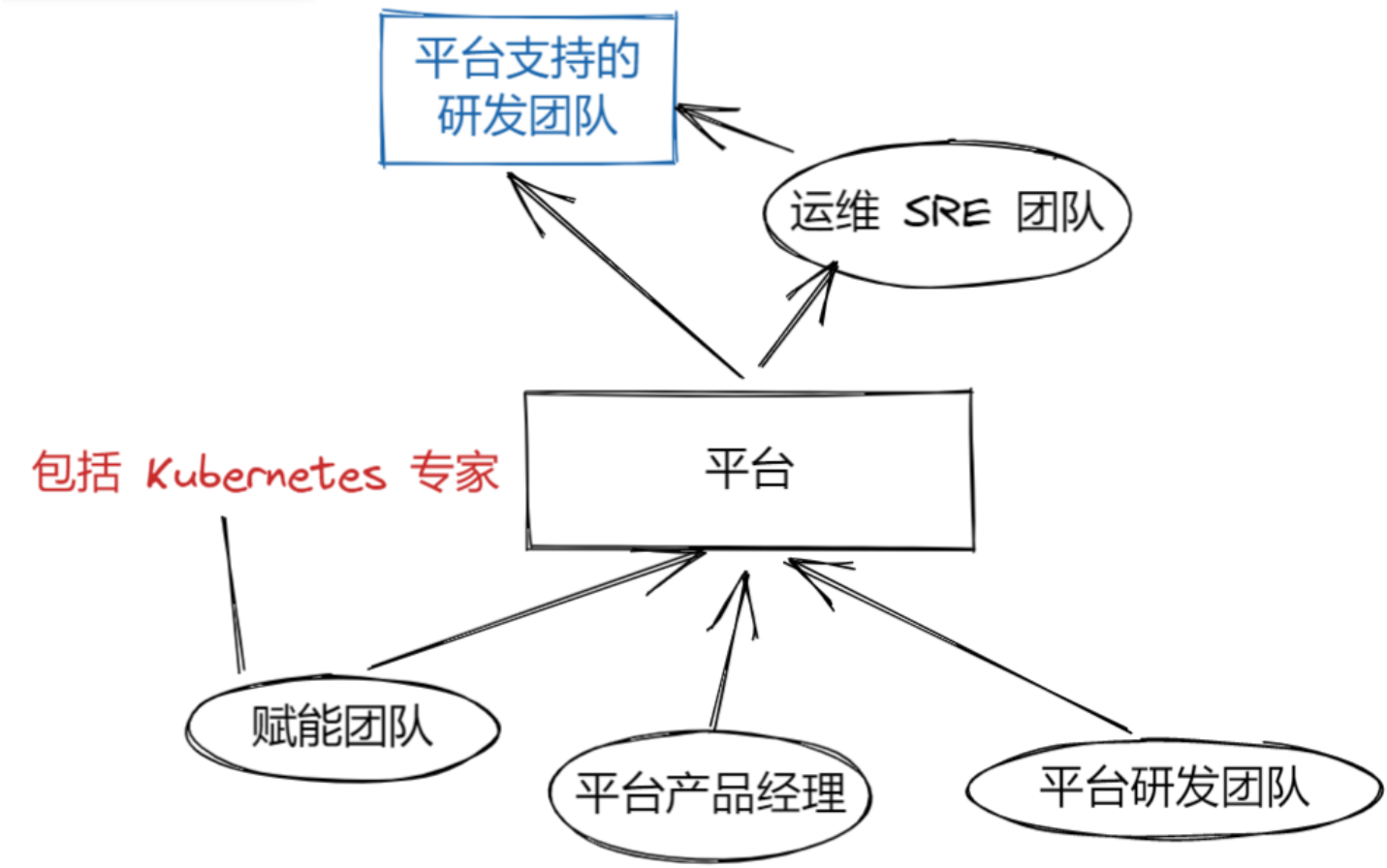
> 分布式计算中的平台是为多种用途提供通用支持能力和服务的层。平台为获取、使用和管理这些功能和服务提供一致的用户体验，包括 Web 门户和页面、特定于场景的代码模板、可自动化的 API 和命令行工具。

可能用一张图更清楚：



这里说的平台，大家公认的是内部开发者平台（Internal Developer Platform, IDP）。

这个平台成为平台团队与应用团队新的分工界面之所在。团队之间的协作会调整为以下结构：



运维团队的专家会变成赋能团队，主要职责是将能力集成到平台。原来许多手工的运维工作，已经在平台上实现，而相关的人则可能要进入平台研发团队，负责将运维工作抽想到平台。当然，可能也需要一个平台产品经理把控平台的方向，这可能是原来运维经理的角色。

平台隐藏了包括 Kubernetes 的基础设施的复杂性，也将最佳实践封装，因而研发团队可以更轻松的按照业务需求构建自己的基础设施环境，又不必担心违反了安全规定，或者是没有遵守最佳实践。研发团队可以通过平台提供自助服务完成自己的大部分任务，可以为自己的应用”负责“，这与 DevOps 的原则相符。

那平台工程中的”平台“与之前的平台的区别是什么？我觉得有几个方面。

- **平台的用户不同**。传统的平台的用户可能是运维团队，而内部开发者平台的用户是开发者，也就是应用团队。
- **两个平台的建设者不同**。传统的平台往往是单独的研发团队研发，这个团队最强的能力的是开发。这个模式的一个原因是之前的工具比较原始，需要较高的抽象才能运行，对研发的要求更高。这个模式的一个问题是运维知识与平台的分离，作为平台的使用者，他们的知识并不能很好的集成到平台上。而内部开发者平台的则是偏向运维的平台团队建设，各类运维专家更加清楚日常有哪些任务，可以更好的结合运维专业知识。

- **平台的建设思路不同**。传统的平台往往是专注于能力封装，也就是将以前各种运维任务简化。而内部开发者平台则更注重实际的研发流程，关注如何帮助开发者快速的实现自己的目标。

如果说 DevOps 是从敏捷开发向运维的延伸，那么平台工程则是来自管理层对于 DevOps 的回应。就像以前的 ITIL 一样，现在可以从组织架构的角度开始设计，而不必像 DevOps 那样倒推改革。

三位一体的未来

现在我们再来看一下为什么是三位一体：

- **云原生技术为构建平台提供了能力**。对于大多数组织来说，在没有云原生相关的技术之前，构建自己平台的难度是非常大的，但现在云原生给许多组织同样的能力。
- **DevOps 为协作提供了文化基础**。越来越多的案例证明，传统的开发运维分工不利于高效的交付产品。鼓励交流协作的 DevOps 文化已经成为了现代应用交付的基础，即使有了平台，也需要不同的团队保持相同的文化。
- **平台工程则在组织层面给出了指导方针**。平台工程概念的推出，让许多有相似想法的人有了共同的术语，可以为这个目标一起协作，从而实现整个行业的改变。

如前面所述，这三个概念有着许许多多不同的演绎，但是在本文我希望大家抓住各自的重点，能够体会到三者对我们这个行业的意义。

平台工程是我们期望的一种状态，云原生是我们达到这个状态的工具，而要实现这个状态，需要我们每个参与者永远保持 DevOps 的协作精神，这就是三位一体。

后话

关于“云云众生s”公众号，我写下的简介是“云原生践行者”，而在相关博客（ https://yylives.cc/ ）中，写的是“献给云时代的小人物们”，当时的我感觉要为云原生时代的 IT 人做点什么，但是并没有很明确说出来。通过梳理这个文章，我发现可以做的事情包括：

- 介绍各种解决方案
- 给大家一个共同讨论的平台
- 汇聚大家的声音改变这个行业

第一个正在做，但是目前看来，最需要平台工程的组织还没有接触到足够多的信息，还需要努力。后两个要怎么做还没有那么清楚，可以先从加我的微信（rocksun21）开始。

业内也有很多人和组织也在做类似的事情，例如信通院搞的”平台工程标准“，还有蚂蚁牵头的”平台工程“组织，大家可以共同推动。