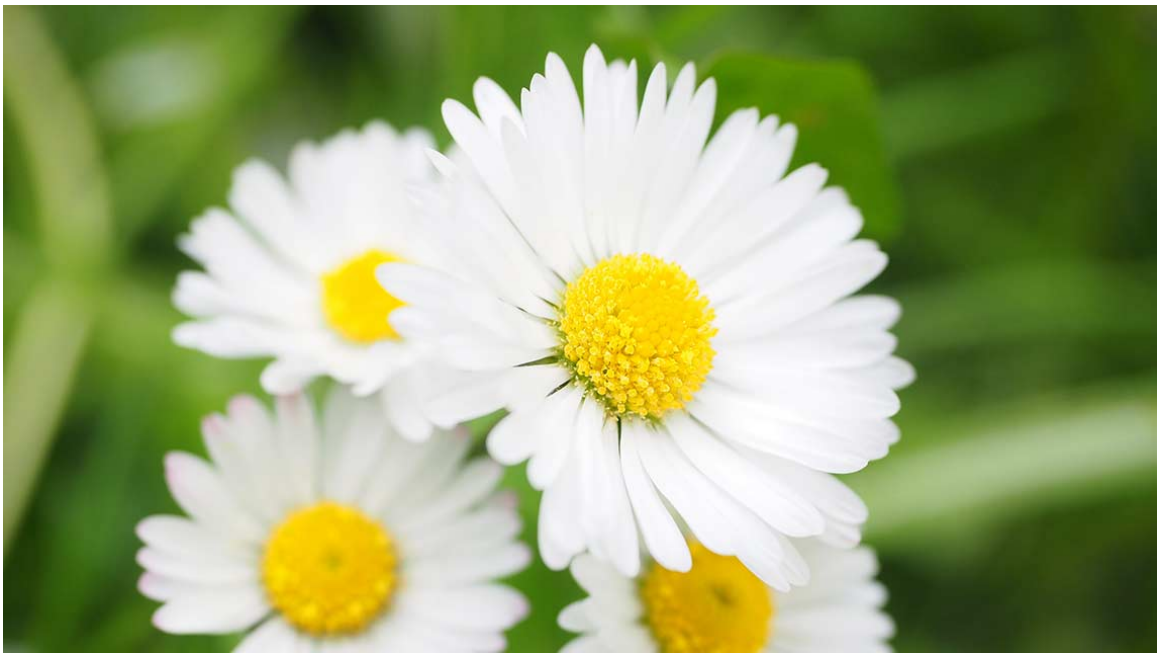
04 | 预习篇 • 小鲸鱼大事记（四）： 尘埃落定

04 | 预习篇 · 小鲸鱼大事记（四）：尘埃落定

张磊 2018-08-31



15:02

讲述：张磊 大小：6.89M

你好，我是张磊。我今天分享的主题是：小鲸鱼大事记之尘埃落定。

在上一次的分享中我提到，伴随着 Docker 公司一手打造出来的容器技术生态在云计算市场中站稳了脚跟，围绕着 Docker 项目进行的各个层次的集成与创新产品，也如雨后春笋般出现在这个新兴市场当中。而 Docker 公司，不失时机地发布了 Docker Compose、Swarm 和 Machine“三件套”，在重新定义 PaaS 的方向上走出了最关键的一步。

这段时间，也正是 Docker 生态创业公司们的春天，大量围绕着 Docker 项目的网络、存储、监控、CI/CD，甚至 UI 项目纷纷出台，也涌现出了很多 Rancher、Tutum 这样在开源与商业上均取得了巨大成功的创业公司。

在 2014~2015 年间，整个容器社区可谓热闹非凡。

这令人兴奋的繁荣背后，却浮现出了更多的担忧。这其中最主要的负面情绪，是对 Docker 公司商业化战略的种种顾虑。

事实上，很多从业者也都看得明白，Docker 项目此时已经成为 Docker 公司一个商业产品。而开源，只是 Docker 公司吸引开发者群体的一个重要手段。不过这么多年来，开源社区的商业化其实都是类似的思路，无非是高不高调、心不心急的问题罢了。

而真正令大多数人不满意的是，Docker 公司在 Docker 开源项目的发展上，始终保持着绝对的权威和发言权，并在多个场合用实际行动挑战到了其他玩家（比如，CoreOS、RedHat，甚至谷歌和微软）的切身利益。

那么，这个时候，大家的不满也就不再是在 GitHub 上发发牢骚这么简单了。

相信很多容器领域的老玩家们都听说过，Docker 项目刚刚兴起时，Google 也开源了一个在内部使用多年、经历过生产环境验证的 Linux 容器：lmctfy（Let Me Container That For You）。

然而，面对 Docker 项目的强势崛起，这个对用户没那么友好的 Google 容器项目根本没有招架之力。所以，知难而退的 Google 公司，向 Docker 公司表示了合作的愿望：关停这个项目，和 Docker 公司共同推进一个中立的容器运行时（container runtime）库作为 Docker 项目的核心依赖。

不过，Docker 公司并没有认同这个明显会削弱自己地位的提议，还在不久后，自己发布了一个容器运行时库 Libcontainer。这次匆忙的、由一家主导的、并带有战略性考量的重构，成了 Libcontainer 被社区长期诟病代码可读性差、可维护性不强的一个重要原因。

至此，Docker 公司在容器运行时层面上的强硬态度，以及 Docker 项目在高速迭代中表现出来的不稳定和频繁变更的问题，开始让社区叫苦不迭。

这种情绪在 2015 年达到了一个小高潮，容器领域的其他几位玩家开始商议“切割”Docker 项目的话语权。而“切割”的手段也非常经典，那就是成立一个中立的基金会。

于是，2015 年 6 月 22 日，由 Docker 公司牵头，CoreOS、Google、RedHat 等公司共同宣布，Docker 公司将 Libcontainer 捐出，并改名为 RunC 项目，交由一个完全中立的基金会管理，然后以 RunC 为依据，大家共同制定一套容器和镜像的标准和规范。

这套标准和规范，就是 OCI（Open Container Initiative）。**OCI 的提出，意在将容器运行时和镜像的实现从 Docker 项目中完全剥离出来。**这样做，一方面可以改善 Docker 公司在容器技术上一家独大的现状，另一方面也为其他玩家不依赖于 Docker 项目构建各自的平台层能力提供了可能。

不过，不难看出，OCI 的成立更多的是这些容器玩家出于自身利益进行干涉的一个妥协结果。所以，尽管 Docker 是 OCI 的发起者和创始成员，它却很少在 OCI 的技术推进和标准制定等事务上扮演关键角色，也没有动力去积极地推进这些所谓的标准。

这，也正是迄今为止 OCI 组织效率持续低下的根本原因。

眼看着 OCI 并没能改变 Docker 公司在容器领域一家独大的现状，Google 和 RedHat 等公司于是把与第二把武器摆上了台面。

Docker 之所以不担心 OCI 的威胁，原因就在于它的 Docker 项目是容器生态的事实标准，而它所维护的 Docker 社区也足够庞大。可是，一旦这场斗争被转移到容器之上的平台层，或者说 PaaS 层，Docker 公司的竞争优势便立刻捉襟见肘了。

在这个领域里，像 Google 和 RedHat 这样的成熟公司，都拥有着深厚的技术积累；而像 CoreOS 这样的创业公司，也拥有像 Etcd 这样被广泛使用的开源基础设施项目。

可是 Docker 公司呢？它却只有一个 Swarm。

所以这次，Google、RedHat 等开源基础设施领域玩家们，共同牵头发起了一个名为 CNCF（Cloud Native Computing Foundation）的基金会。这个基金会的目的其实很容易理解：它希望，以 Kubernetes 项目为基础，建立一个由开源基础设施领域厂商主导的、按照独立基金会方式运营的平台级社区，来对抗以 Docker 公司为核心的容器商业生态。

而为了打造出这样一个围绕 Kubernetes 项目的“护城河”，CNCF 社区就需要至少确保两件事情：

1. Kubernetes 项目必须能够在容器编排领域取得足够大的竞争优势；
2. CNCF 社区必须以 Kubernetes 项目为核心，覆盖足够多的场景。

我们先来看看 CNCF 社区如何解决 Kubernetes 项目在编排领域的竞争力的问题。

在容器编排领域，Kubernetes 项目需要面对来自 Docker 公司和 Mesos 社区两个方向的压力。不难看出，Swarm 和 Mesos 实际上分别从两个不同的方向讲出

了自己最擅长的故事：Swarm 擅长的是跟 Docker 生态的无缝集成，而 Mesos 擅长的则是大规模集群的调度与管理。

这两个方向，也是大多数人做容器集群管理项目时最容易想到的两个出发点。也正因为如此，Kubernetes 项目如果继续在这两个方向上做文章恐怕就不太明智了。

所以这一次，Kubernetes 选择的应对方式是：Borg。

如果你看过 Kubernetes 项目早期的 GitHub Issue 和 Feature 的话，就会发现它们大多来自于 Borg 和 Omega 系统的内部特性，这些特性落到 Kubernetes 项目上，就是 Pod、Sidecar 等功能和设计模式。

这就解释了，为什么 Kubernetes 发布后，很多人“抱怨”其设计思想过于“超前”的原因：Kubernetes 项目的基础特性，并不是几个工程师突然“拍脑袋”想出来的东西，而是 Google 公司在容器化基础设施领域多年来实践经验的沉淀与升华。这，正是 Kubernetes 项目能够从一开始就避免同 Swarm 和 Mesos 社区同质化的重要手段。

于是，CNCF 接下来的任务就是，如何把这些先进的思想通过技术手段在开源社区落地，并培育出一个认同这些理念的生态？这时，RedHat 就发挥了重要作用。

当时，Kubernetes 团队规模很小，能够投入的工程能力也十分紧张，而这恰恰是 RedHat 的长处。更难得的是，RedHat 是世界上为数不多的、能真正理解开源社区运作和项目研发真谛的合作伙伴。

所以，RedHat 与 Google 联盟的成立，不仅保证了 RedHat 在 Kubernetes 项目上的影响力，也正式开启了容器编排领域“三国鼎立”的局面。

这时，我们再重新审视容器生态的格局，就不难发现 Kubernetes 项目、Docker 公司和 Mesos 社区这三大玩家的关系已经发生了微妙的变化。

其中，Mesos 社区与容器技术的关系，更像是“借势”，而不是这个领域真正的参与者和领导者。这个事实，加上它所属的 Apache 社区固有的封闭性，导致了 Mesos 社区虽然技术最为成熟，却在容器编排领域鲜有创新。

这也是为何，Google 公司很快就把注意力转向了动作更加激进的 Docker 公司。

有意思的是，Docker 公司对 Mesos 社区也是类似的看法。所以从一开始，Docker 公司就把应对 Kubernetes 项目的竞争摆在了首要位置：一方面，不断强调“Docker Native”的“重要性”，另一方面，与 Kubernetes 项目在多个场合进行了直接的碰撞。

不过，这次竞争的发展态势，很快就超过了 Docker 公司的预期。

Kubernetes 项目并没有跟 Swarm 项目展开同质化的竞争，所以“Docker Native”的说辞并没有太大的杀伤力。相反地，Kubernetes 项目让人耳目一新的设计理念和号召力，很快就构建出了一个与众不同的容器编排与管理的生态。

就这样，Kubernetes 项目在 GitHub 上的各项指标开始一骑绝尘，将 Swarm 项目远远地甩在了身后。

有了这个基础，CNCF 社区就可以放心地解决第二个问题了。

在已经囊括了容器监控事实标准的 Prometheus 项目之后，CNCF 社区迅速在成员项目中添加了 Fluentd、OpenTracing、CNI 等一系列容器生态的知名工具和项目。

而在看到了 CNCF 社区对用户表现出来的巨大吸引力之后，大量的公司和创业团队也开始专门针对 CNCF 社区而非 Docker 公司制定推广策略。

面对这样的竞争态势，Docker 公司决定更进一步。在 2016 年，Docker 公司宣布了一个震惊所有人的计划：放弃现有的 Swarm 项目，将容器编排和集群管理功能全部内置到 Docker 项目当中。

显然，Docker 公司意识到了 Swarm 项目目前唯一的竞争优势，就是跟 Docker 项目的无缝集成。那么，如何让这种优势最大化呢？那就是把 Swarm 内置到 Docker 项目当中。

实际上，从工程角度来看，这种做法的风险很大。内置容器编排、集群管理和负载均衡能力，固然可以使得 Docker 项目的边界直接扩大到一个完整的 PaaS 项目的范畴，但这种变更带来的技术复杂度和维护难度，长远来看对 Docker 项目是不利的。

不过，在当时的大环境下，Docker 公司的选择恐怕也带有一丝孤注一掷的意味。

而**Kubernetes 的应对策略则是反其道而行之，开始在整个社区推进“民主化”架构**，即：从 API 到容器运行时的每一层，Kubernetes 项目都为开发者暴露出了可以扩展的插件机制，鼓励用户通过代码的方式介入到 Kubernetes 项目的每一个阶段。

Kubernetes 项目的这个变革的效果立竿见影，很快在整个容器社区中催生出了大量的、基于 Kubernetes API 和扩展接口的二次创新工作，比如：

目前热度极高的微服务治理项目 Istio；
被广泛采用的有状态应用部署框架 Operator；
还有像 Rook 这样的开源创业项目，它通过 Kubernetes 的可扩展接口，把 Ceph 这样的重量级产品封装成了简单易用的容器存储插件。

就这样，在这种鼓励二次创新的整体氛围当中，Kubernetes 社区在 2016 年之后得到了空前的发展。更重要的是，不同于之前局限于“打包、发布”这样的 PaaS 化路线，**这一次容器社区的繁荣，是一次完全以 Kubernetes 项目为核心的“百家争鸣”**。

面对 Kubernetes 社区的崛起和壮大，Docker 公司也不得不面对自己豪赌失败的现实。但在早前拒绝了微软的天价收购之后，Docker 公司实际上已经没有什么回旋余地，只能选择逐步放弃开源社区而专注于自己的商业化转型。

所以，从 2017 年开始，Docker 公司先是将 Docker 项目的容器运行时部分 Containerd 捐赠给 CNCF 社区，标志着 Docker 项目已经全面升级成为一个 PaaS 平台；紧接着，Docker 公司宣布将 Docker 项目改名为 Moby，然后交给社区自行维护，而 Docker 公司的商业产品将占有 Docker 这个注册商标。

Docker 公司这些举措背后的含义非常明确：它将全面放弃在开源社区同 Kubernetes 生态的竞争，转而专注于自己的商业业务，并且通过将 Docker 项目改名为 Moby 的举动，将原本属于 Docker 社区的用户转化成了自己的客户。

2017 年 10 月，Docker 公司出人意料地宣布，将在自己的主打产品 Docker 企业版中内置 Kubernetes 项目，这标志着持续了近两年之久的“编排之争”至此落下帷幕。

2018 年 1 月 30 日，RedHat 宣布斥资 2.5 亿美元收购 CoreOS。

2018 年 3 月 28 日，这一切纷争的始作俑者，Docker 公司的 CTO Solomon Hykes 宣布辞职，曾经纷纷扰扰的容器技术圈子，到此尘埃落定。

总结

容器技术圈子在短短几年里发生了很多变数，但很多事情其实也都在情理之中。就像 Docker 这样一家创业公司，在通过开源社区的运作取得了巨大的成功之后，就不得不面对来自整个云计算产业的竞争和围剿。而这个产业的垄断特性，对于 Docker 这样的技术型创业公司其实天生就不友好。

在这种局势下，接受微软的天价收购，在大多数人看来都是一个非常明智和实际的选择。可是 Solomon Hykes 却多少带有一些理想主义的影子，既然不甘于“寄人篱下”，那他就必须带领 Docker 公司去对抗来自整个云计算产业的压力。

只不过，Docker 公司最后选择的对抗方式，是将开源项目与商业产品紧密绑定，打造了一个极端封闭的技术生态。而这，其实违背了 Docker 项目与开发者保持亲密关系的初衷。相比之下，Kubernetes 社区，正是以一种更加温和的方式，承接了 Docker 项目的未尽事业，即：以开发者为核心，构建一个相对民主和开放的容器生态。

这也是为何，Kubernetes 项目的成功其实是必然的。

现在，我们很难想象如果 Docker 公司最初选择了跟 Kubernetes 社区合作，如今的容器生态又将会是怎样的一番景象。不过我们可以肯定的是，Docker 公司在过去五年里的风云变幻，以及 Solomon Hykes 本人的传奇经历，都已经在云计算的长河中留下了浓墨重彩的一笔。

思考题

你如何评价 Solomon Hykes 在 Docker 公司发展历程中的所作所为？你又是否看好 Docker 公司在今后的发展呢？