

开源观止

OPEN SOURCE GUAN ZHI

P05 聚焦

开源办公室大揭秘

P25 创业小辑

“洗钱”·底线·PR·3亿

P41 众说

工程师如何选择开源软件?

P27 溯源

老同事拉我创业，
做一家开源存储公司

P10

OSPO专题

字节跳动

新晋 OSPO 设超十人全职

+

蚂蚁集团

OSPO 组织首次曝光

+

微众银行

OSPO 老兵的深度解析



本期主编 肖滢



www.oschina.net

征集合作

(OPEN SOURCE GUAN ZHI)

《开源观止》是 OSCHINA 于 2022 年 6 月推出的一本关于开源的精选集，旨在集中化地呈现一些关于 FLOSS 的信息与观点。我们有一个美好的愿景：开源，观止矣。而这需要更多开源人、开发者参与进来共建。

因此，OSCHINA 编辑部特向大家征稿，征稿类型如下：

- (1) 开源实务经验：合规、安全、运营等与开源有关的实践过程、方法经验，参考总第1期文章《企业开源运营之道：目标、用户、指标》；
- (2) 开源创业历程：体现创业者的奋斗历程、成长经过，阐述创业项目的运营情况、未来前景，分享创业感悟、创业经验，参考总第1期文章《SphereEx 张亮：创业是为了让开源更好》；
- (3) 开源观点论述，要求观点明确、新颖，论述严谨，参考总第1期文章《开源纯粹主义：每一颗螺丝钉都必须是自由的！》；
- (4) 热点事件解读：针对当前开源领域的热点事件进行解读，挖掘事件背后原因，引发读者思考；
- (5) 开源历史故事：与开源有关的历史故事，以史鉴今，参考总第1期《专题 | 古典主义 Debian》栏目；
- (6) 开源报告、调查数据；
- (7) 开源趣味小故事、冷知识，参考散落各处的《开源识多D》；
- (8)

如欲投稿、合作，请联系：

肖滢 183 7099 8278（微信同号）



《开源观止》总第 1 期

内容合作



卷首语

开源需要专业化

软件吞噬世界，开源吞噬软件。如今，任何商业公司的经营都不可避免地直接或间接依赖于开源软件，甚至公司本身就参与到开源协同的过程中，或者发起新的开源项目。

如何用好开源软件？如何参与开源协同？如何经营公司在开源社群的影响力？如何治理公司发起的开源项目？如何利用好开源浪潮带来的新杠杆？这是面对开源吞噬软件新形势下，每一个商业公司都有可能需要回答的问题。

过去，选择开源软件和开发开源软件的责任由研发部门兼职承担，宣传和品牌建设的责任由市场运营承担，合规与安全问题由法务部门和安全部门应对。

然而，开源社群是一个有机整体。上面提到的问题并不能够由单一部门独立解决，甚至具备解决问题能力的人可能在公司之外。如何围绕开源这个主题，协同企业内不同部门，以及企业内外不同参与者的诉求，利用开源社群的杠杆完成多赢正和游戏，需要专业化的人才与机构来回答。

这就是近几年蓬勃发展的开源办公室（OSPO）机构。谷歌、苹果、微软、华为、阿里巴巴、腾讯、字节跳动……一大批国内外知名公司纷纷设立开源办公室或相同性质的机构，招贤纳士，以期从战略层面统一规划，协同不同部门与公司内外的力量，应对企业在开源的使用、参与和发起方面的挑战。

“夜天之书”公众号作者、Apache Member & 孵化器导师 tison

2022年6月22日



微信公众号 “夜天之书”

关注开源共同体的发展，致力于回答“如何建设一个开源社群”的问题。

CONTENTS 目录

03 卷首语

开源需要专业化

05 聚焦

开源办公室大揭秘

10 专题 ospo

方法论：蚂蚁集团关于 OSPO 的思考

字节跳动加入 OSPO 群聊，将投入超十人全职团队

金融科技 OSPO 先行者，微众银行如何干？

25 创业小辑

“洗钱” | 底线 | PR | 3 亿

27 溯源

老同事拉我创业，做一家开源存储公司

姜宁，带程序员前往开源 “乌托邦”

38 星推荐

Go+ | Skyline | Mastodon | Bitwarden |

PDManer | PaoPao CE

41 众说

工程师如何选择开源软件？

搞懂社区运营：留存与转化的几个常见逻辑范式

53 新鲜事

57 高手问答

从 0 到 1 掌握大数据技术

01 封面

02 征稿

60 Gitee



网址: www.oschina.net

扫码关注，下期推送到手机



微信服务号：OSC 社区

投稿、合作，联系我们

电话：183 7099 8278

邮箱：xiaoying@oschina.cn



新浪微博：开源中国



抖音号：oschina



头条号：开源中国 OSC

版权声明

版权所有，如欲转载，请联系《开源观止》编辑部

开源办公室大揭秘

近年来，国内互联网大厂相继设立开源办公室（Open Source Program Office，简称 OSPO），其中有部分企业在 OSPO 建设方面更是颇有成效。虽说 OSPO 并不是什么新鲜事物，但却少有人揭开其神秘面纱，因此它常常为外界所误解，甚至有人说它是形同虚设，其存在是浪费资源。因此，开源中国邀请了四位业内专家，来谈一谈真实的 OSPO 到底是怎么样的。



王晔惊（头哥）



tison



Keith Chan



堵俊平



边思康

主持人

王晔惊（头哥）：支流科技技术 VP、Apache APISIX Committer

观察员

tison：“夜天之书”公众号作者、Apache Member & 孵化器导师

分享嘉宾

Keith Chan: CNCF 中国区总监、LF 亚太区策略总监

堵俊平：现为华为计算开源业务（OSDT）总经理、LF AI & DATA 基金会主席

边思康：蚂蚁集团技术战略发展部资深专家、开源办公室执行负责人

话题 1：OSPO 是什么？

王晔惊（头哥）：什么是 OSPO？它有哪些形式？先请观察员 tison 来谈一谈。

tison：OSPO，全称叫 Open Source Program Office，翻译成中文的时候，把 Program 去掉了，所以叫开源办公室。从一些企业用例来看，还挺贴切的。对于“OSPO 是什么”，没办法用一句话来回答。

如今，几乎所有的商业活动都建立在软件之上，而软件总会依赖某些开源组件。因此，企业或多或少都会参与到开源之中，总结起来无非是使用开源、参

与开源、发起开源项目。

在此过程中，企业肯定会遇到很多问题，因此就需要“OSPO”这样的部门来解决这些问题。

OSPO 职能涉及很多方面。其最基础的职能是法务和安全，比如要处理开源合规、软件缺陷修复、安全漏洞等问题；还涉及到研发，毕竟开源的核心还是软件；此外，与市场、运营等部门也有一定的关系，因为开源可以被当做一种免费增值的手段（比如 MongoDB），且其本身还自带品牌效应、传播优势。

关于 OSPO 在企业的架构形态，总结如下：

在我国，OSPO 最普遍的形态是虚拟组织。它跟实体部门很不一样，没有可以汇报到 CEO 的汇报线。当前国内大部分企业对 OSPO 的态度都处于尝试阶段，因此倾向于组建一个虚拟部门，典型的如 PingCAP。PingCAP 的 OSPO 在法务、合规以及安全方面，会有全职雇员，或至少有一部分工作与之相关，研发、市场营销专员也会参与进来。

OSPO 作为整体市场策略的一部分，常见于开源技术创业公司。对他们而言，开源软件或者说开源生态是公司的核心竞争力之一，因此往往会借鉴一些成功案例，将开源作为营销方式之一，把 OSPO 跟市场部门挂钩，或者并入到市场部门，作为整体市场策略的一部分。有的创业公司把研发看得比较重，会单独成立一个技术架构委员会，这跟 OSPO 还是有些区别的。技术架构委员会是一个纯粹的技术部门，大部分情况下关注技术领域，而 OSPO 是涉及法务、安全、研发、市场、运营等多维度的部门。开源社群主要分为用户社群、开发者社群，这类 OSPO 更关注的是用户社群。

OSPO 作为实体部门单独存在，可以直接汇报给 CEO；或挂靠在研发部门，汇报给 CTO 或者技术 VP。至于 OSPO 是要处理什么事情，很多时候取决于部门招到什么样的人。如果招聘的是研发出身，那么侧重于技术开发，如果是法务或者安全方面的人才，那么自然会偏向自己擅长的方向。

这三类 OSPO 的共性在于，都属于横向赋能部门，就算是虚拟部门也不例外，虽然与其他部门有业务合作关系，但不会或很少直接承担业务。

也有少部分 OSPO 是纯粹的横向部门，提供完整的开源咨询能力，但很少参与具体业务的经营或执行，最多会安排联络员作为咨询师参与到其他部门。

总结一下，从务实的角度来讲，OSPO 要解决的，就是在使用开源、参与开源、发起开源项目遇到的方方面面的问题。从务虚的角度来讲，OSPO 是一种文化。比如红帽的开放文化在某种程度上，也可以认为是 OSPO 的一部分，当然，这也是企业文化。

Keith Chan: 我用比较简单的比喻来解释吧。我

觉得，OSPO 就是一个外交官，既是外部的外交官，也是内部的外交官。

每一家公司的 OSPO 有自己的定位，职能也都不一样，但是总的来说，必须要建立起沟通的能力。即使是一件很简单的事情，比如开源合规，也要与研发部门沟通。

为什么要做这件事？许可证有没有问题？版权有没有问题？这些问题都是要讲清楚的，尤其是涉及出口更要搞清楚。不是每一个人都了解这些，因此沟通能力非常重要，这也是我称 OSPO 为外交官的原因。

很多公司的领导层，不太知道开源的重要性，尽管他们使用的软件或多或少都跟开源有关系，都是睁只眼闭只眼，至于为什么要管理开源，用了多少开源，这些开源项目有多重要，都不知道。有人还会把这些问题扔给研发去管。我看到很多公司都面临这样的问题，既没有在战略上把开源管理好，也没有足够的精力去付诸行动。

事实上，不管是对内还是对外，都应该由 OSPO 来把这些事讲清楚。

话题 2：OSPO 要解决什么问题？

王晔惊：从企业的角度来看，OSPO 主要想解决什么样的问题，具体怎么去落地的？

堵俊平：为什么要有 OSPO 这么一个组织？抛出这个问题的前提，一定是在开源的使用或者贡献过程中遇到了一些障碍。

开源已经席卷了 IT 史三十多年，诞生了 Linux、MySQL、K8S 等一批熟悉的开源软件。在早期，开源软件数量不多，可能并没有太多的问题出现。但现在，很多企业的 IT 基础设施底座基本上都是以开源为主体。之前 Linux 基金会发布的报告显示，全世界 80% 以上的软件都是基于开源软件开发而来。

不仅仅是 IT 企业，传统企业也无法脱离开源，华为也经历了好几个过程。大概在 20 多年前，华为还在做硬件卖盒子的时候，开始逐渐使用开源软件，比如 Linux，这时面临的问题是如何用好管理开源软件。在很长一段时间内，OSPO 都不是一个成型的

部门，但有一个类似的职能部门负责开源的管理，包括引入、选取。我们知道，随着开源软件供应越来越丰富，社区也越来越多样，如何选用一个可以发展或适合承载业务使命的开源软件越来越重要。选用不合理的开源软件对整体业务来说，相当于一笔巨大的技术债务。

因此，在早期，华为的 OSPO 主要解决的问题是如何选好开源软件，如何支撑业务发展，如何合规使用。

差不多从十年前开始，华为的业务从传统产业转向 IT 产业，计算业务、云业务都涉及大量开源软件。到了这一阶段，除了使用开源，华为还贡献开源，甚至有一些自有的优势项目要开源出去，去构建一个产业生态，做大自己的伙伴圈，从整个共享生态当中获益。这就是 OSPO 另外一个职能，也就是构建所谓的开源战略。

开源战略一定是围绕业务的。业务有多种形态，是面向硬件，还是面向软件，又或者是云，亦或者消费者？不同的业务形态，开源战略的打法也不一样，但基本的原则和目的是一样的，就是服务好我们的业务战略，构建起整体的产业生态。OSPO 的作用，就是要识别出重要的、关键的开源领域，在里面持续发力，包括构建开源文化，提升技术影响力，把整个生态做大做强。

话题 3：OSPO 面临的难题是什么？

王晔惊（头哥）：开源管理的复杂性，到底在哪里？复杂到需要建设一个专门的机构来解决这个问题吗？

堵俊平：正如 tison 介绍的那样，OSPO 有不同的组织形式，有的是虚拟组织，有的是实体组织，这两种我都经历过。华为的 OSPO 是实体组织，有专职的团队在做开源。但我之前所在的公司，其 OSPO 是一个虚拟组织，各业务团队会抽调一些人加入这个虚拟组织，来解决开源相关的问题。

虚拟组织和实体组织，面临的挑战是不一样的，

因为它承载着不同业务的使命。

一种观点认为，OSPO 是一个能力中心。什么叫能力中心？就是由专家组成一个 OSPO 团队，在各业务部门有法务、技术、运营或营销需求的时候，为其提供帮助。

这种模式的问题在于，当其他部门有需求时，不能很快地响应，无法支持高频次地调用。对于采用虚拟组织形式的 OSPO 来说更是如此。实际上，很多业务决策，是有时效性的，最佳窗口期错过就没了。当然，在一般情况下，能力中心是足够支撑业务的，是没有问题的。比如怎么用好开源，怎么正确地避坑，怎么制定开源版本规划等等，这些问题的时效性并不那么强。

另一类观点是，OSPO 是业务中心，要承载一些业务使命，作为整个生态团队的一部分配合公司的业务。这种情况下，如果是虚拟组织，可能就不太适合。因为虚拟组织无法及时有效地响应，可能会导致更多问题。

现在已经不是二三十年前开源稀缺的年代了，不是开源一个项目，别人马上就会拿来用。在开源项目之外，还要做运营、做推广。有很多事情要考虑的，比如推动开源项目，你要和哪些人建立合作关系；做宣传推广的时候，要如何持续定点地去突破。如何做好外交家，是业务中心要解决的一个难点。

开源社区是由各种不同背景的人组成的，不管是用户、贡献者，甚至是旁观者，都有各自的想法。推广开源项目的时候，出现了争议声音，要如何解决？那么，如何进行有效沟通，化干戈为玉帛，持续把产品有效地推广出去，与大家建立连接；如何把社区的声音带回内部，以推动工程团队来改进技术产品，这些都是 OSPO 团队要解决的问题。

边思康：有一点不能忽略的是，OSPO 既要解决内部问题，也要解决外部的问题。OSPO 的成员要同时扮演 COO、CMO 的角色，甚至是 CTO 和 CEO 的角色。当然，这些事情可能不是一个人来做，而是 OSPO 的所有成员群策群力。这种情况下，一般业务的复杂性就翻了一倍。

做技术的时候，大家会考虑得很具体：业务用

户是谁，想要什么样的效果，可以提供什么样的技术解决方案，做成一个什么样的设计……这个链路非常清楚。然而，在开源的环境下，不再是“一对一”或者“一对 N”的模式，更像是“N 对 N”的交流。

另一方面，做开源肯定会与规范、安全产生联系，因为开源本身就是一个相应的风险点。与此同时，开源会有大量的对外交流，会与市场部门产生联系。这些关系错综复杂，导致 OSPO 要解决的问题很多。即使我们对所有的问题进行二级抽象，它仍然是一个复杂度很高的事，因为我们并不能解决所有人的所有问题。

然而，在这过程中，OSPO 如果去教别人做事，也是不合适的。开源本来就是一个自适应、社区型的生长环境。我们常常拿雨林来做比喻开源，就是因为开源没有一个固定形态。我们也会从其他组织那里学习很多先进做法，但直接拿来用又不现实。

最后，我认为做开源，核心的还是那句话——“授人以鱼不如授人以渔”。其实，我们可以花多一点时间，让大家把这个心智培养起来。但这里的核心难点是，我们如何能够系统化地把开源讲清楚？因为开源不是一个非常具体的事情。

话题 4：OSPO 的运营困境是什么？

王晔惊（头哥）：tison，作为局外人，你看到的 OSPO 现在面临的一些客观问题是什么？

tison：华为、蚂蚁这样的公司，更多是关注战略上的东西。其 OSPO 做开源治理、开源使用，更多的工作还是局限于公司内部，还是比较封闭的。从研发角度来看，开源软件的开发不是一个企业独自地开发，并不是企业开发者开发了一个软件拿出来就叫“Open Source”，这与真正的开源大相径庭。

现实中，企业会把一些开源软件捐赠给开源基金会，基金会有一些基础规则来帮助开源项目，比如要怎么做决策。这对企业传统的开发者来说，遵循这些规则是一个挑战。不过，像 Apache 软件基金会这样的社群，有很明显的企参与痕迹，如此一来，企

业开发人员从社群里成长出来之后，自然就会明白如何做好协同。

如果缺少了这样的能力，不知道如何外界协同、不知道怎样基于行业力量去发布软件，那么就还是停留在封闭的软件开发模式之中。开源协同的开发效率要远远高于封闭的开发，Linux 有数百万贡献者，这是企业无法相比的。

运营也是一个很典型的问题，很多企业尤其是技术型创业公司在这方面表现得最明显。大企业的项目也可认为是一个小型的、虚拟的技术创业公司，会遇到这些的挑战。

有时候运营对外去输出内容（比如一篇文章），但他自己并不知道具体需求是什么？反正文章就这么出去了。国内的大部分运营的经验还是面向 C 端，而不知道面向开发群体应该是什么样的。在很多企业，研发和运营之间的沟通是断裂的，运营永远不会跟研发说，你的文章哪里写得不好，顶多会反馈为阅读量不好。

更多的时候，企业应该依靠开源社群获得资源，而不是企业花钱雇佣一大堆员工，整天去回答社群问题，这是很吃亏的。让用户能够自发地互相去解答问题，才能让社群更加健壮。

一些大厂本身不依赖开源软件赚钱，他们的目标是吸引开源人才。运营发挥得好，对企业的品牌建设和行业影响力的提升有很大帮助。如果你在开源社群建立起了声望，就会吸引到很多人才。比如谷歌就因为发表了很多影响力巨大的论文，吸引了很多工程师。

在这些方面，企业的 OSPO 有很明显的价值。而且，越是到深层次、长期的价值上，实体 OSPO 的作用越大。

话题 5：什么情况下要组建 OSPO？

王晔惊（头哥）：多数业务驱动型公司对 Open Source 认知不够，认为开源就是免费。我之前也走过弯路，我曾经在公司大张旗鼓地推行开源委员会，主张把中间件贡献给基金会，但事实上却很难执行下去。那么，在什么情况下，企业才需要建立或者需要

思考去开始建立 OSPO 呢？

Keith Chan: 无论公司大小，只要用到开源，就会发现开源可以省下大量成本。但是，如果企业只是使用开源，而没有回馈社区，还需要请很多人来维护代码。有一家规模很大的公司因为没有参与 OBS 上游，投入了 100 多人去维护项目。

不止是 IT 业，金融业或者传统行业都在用开源，现在企业使用的代码 80%~90% 都是开源的，但很少有人去想怎么去管理这些代码，慢慢就变成了“代码负债”。

而且，当下很多人才、技术大神都是从开源挖掘出来的。企业如果在开源方面没有知名度，很难吸引这些人才。因此，如果不做开源，企业的竞争力会越来越低，成本也会越来越高。

堵俊平：我非常认同 Keith 的观点。不是只有想通过开源挣钱的公司才需要建立 OSPO。只要你想用好开源，不管是节省成本、提升效率，或者是加强组织文化，都可以尝试去建立 OSPO。其中，最核心的点在于你是否遇到了痛点，不管是在使用开源或者推行开源业务遇到的。

前段时间爆出了不少开源漏洞，很多相关业务和软件都受到了影响。开源软件千千万万，上面有几千万个代码仓库，项目怎么选？选完之后怎么维护？所有公司都会面临这样的问题。

除此之外，当你分叉了一个开源项目，随着时间推移，运行过程中多多少少都要修复一些 Bug。如果你不从上游直接拿到修复代码，而是依靠自己的团队缝缝补补，最后运维团队只会越来越大。

尤其是我们互联网公司，从小公司成长起来，回头一看，发现投入的开发人员比社区现存的 PMC 还要多。为什么会出现这种情况？我认为，首先是没有长线的开源规划，没有清晰的理念；其次，没有 Upstream First（上游优先）。有长期开源经历的人就会发现，上游优先是一个非常经济的管理开源的方式，它同时也契合了开源文化，开源即回馈社区。

如果没有一个有经验的开源团队，企业很可能在物质和精神层面都有损失。因此，当团队遇到搞不定的痛点，不妨去引入外部资源和人才，去组建

OSPO。

边思康：如果把 OSPO 定位为外交官，那什么时候需要去建立 OSPO？就是需要去对外建交的时候。谁需要对外建交？任何一个企业都可能需要。

不要为了做开源而做开源，否则大家都不舒服。但话说回来，如果开源能够切实地帮到你，那建立这样一个外交官的角色，对沟通是有极大帮助的。

我觉得建立 OSPO 和其他的决策链路都是类似的。公司一开始只有十个人的时候，你不需要效能部，因为每个人都要对效能负责。当公司到 30 个人的时候，你不需要技术风险部，因为核心技术风险全部都体现在业务代码里。

但是，当解决开源问题的潜在收益出现规模效应的时候，当开源项目开始出现在公司的多个 BU (Business Unit, 业务单元) 里的时候，当开源出现“烟囱”的时候，那 OSPO 肯定会帮到你。

实际上，很多公司的架构部在最开始解决的其实就是 Upstream 问题。当公司内不同团队的架构重复出现，就会形成一个个的“烟囱”。很多东西都是可以被复用的，为什么还要重复造轮子？我能不能去做内源？一堆衍生问题就出来了。

我们并非一定要建立一个 OSPO，但企业一定需要一个类似的负责人，能够提供开源方面的帮助，能把开源的事情讲清楚，并付诸实践。

除此之外，我认为不管你是不是做开源，不管你的业务是 to B 还是 to C，其实都可以了解一下开源精神是什么，开源的发展史是什么。开源精神带来的帮助，不只局限于 OSPO。因此，我认为应该更早把开源精神用起来。

王晔惊（头哥）：很多公司建设到一定程度一定会去搭建工程效能、搭建 PMO。我反问一下，为什么要搭建呢？为什么是在这个时候而不是半年前去搭建呢？是因为工程太多了、太慢了，企业才会想去做这件事。开源也一样。当你的开源使用率够大、开源基建数量够多到一定程度的时候，就是需要去治理它。开源像空气，如果空气已经污染，人们都开始咳嗽了，不治理能行吗？

方法论：蚂蚁集团关于 OSPO 的思考

开源办公室（Open Source Program Offices，OSPO），这两年在国内大厂之间慢慢流行开来。2021年初，蚂蚁集团也成立了 OSPO，时间虽不算长，却也渐渐摸索出了一条自己的路。开源中国编辑部邀请了蚂蚁集团 OSPO 执行负责人边思康，来谈一谈蚂蚁集团的 OSPO 究竟形成了怎样的方法论来指导开源实践。

01 既是运营官也是外交官

蚂蚁集团的 OSPO 成立于 2021 年初。这个时间节点，好像是蚂蚁特意等来的时机成熟，又好像是顺其自然迎来的恰到好处。当开源的过程中积累了一些问题，边思康的一句“可以试试”，OSPO 就这么开始了。

边思康：一个公司成立 OSPO，这件事最好是自然发生的，而不是刻意的。一定是先有开源，再有 OSPO 。

很多 BU（Business Unit，业务单元）都会涉及到优秀的开源项目，而项目在发展过程中，会逐渐积累一些问题，比如风险、成本和成效等问题。之后，大家慢慢会对这些“痛点”有所感知，继而有所认知，然后形成判断——“需要有个组织来系统性地承载类似需求”。

成立 OSPO 的逻辑，可能和成立效能部、安全部等部门的思路是一致的。业务发展到一定规模之后，能有一个组织来进行业务统筹、治理和运营，会给整体效率的提升带来更大帮助。

我来公司不久后，就接到了这个命题，一时有点懵，因为跟原本预想的方向不太一样，但抱着“非我莫属”的精神，还是觉得可以试试看。后来在大家支持下做出了些心得，慢慢地，也就更加得心应手了。

蚂蚁集团的 OSPO 是虚拟组织，有核心成员，参照了 TODO Group、基金会（比如 CNCF）等组织的设计。



我们考虑的维度可能没有那么复杂。从“我们想解决谁的什么问题”开始，明确用户，明确短期目标，然后明确所需的专家经验和资源，取得共识，明确机制后，就开搞。如果还缺少某些方面的专家支持，组织可以随时进化调整。这一设计在内部也是开放的，可以根据收集的建议向前迭代。

我很喜欢一本书《硅谷生态圈：创新的雨林法则》。这本书讲的是，世界上很多地区试图模仿硅谷，以建造能够孵化伟大公司的科技园和创意园，但是鲜有成功的，它们欠缺的不是资金和人才，而是一个利益各方共赢的生态圈。这对开源同样适用。

书中提到一个词，叫做“deal maker”，指的是在硅谷的那些社会网络节点，能够链接资本和创业者，让创新能自然发生的那些人。

OSPO 扮演的就是这种角色。OSPO 对内要像运营官一样思考，如何让开源项目更好地协同和发展；对外则要像外交官一样思考，如何寻求与社区的合作，



共同追求可持续的互惠互利健康发展，以及如何通过开源更好地抓住创新的机会。

02 承载蚂蚁开源的核心宗旨：把关键技术开放

OSPO 是什么？要解决的问题是什么？它在蚂蚁集团的定位是什么？是如何对开源项目产生影响的？除了推动项目开源之外，OSPO 还有哪些作用？对此，蚂蚁集团已经有了一些答案。

边思康：蚂蚁开源的核心宗旨，是把关键技术开放给大家用。而 OSPO 是该核心宗旨的一个承载。

开源项目的生命周期，类似 CNCF 的 sandbox（沙箱） - incubation（孵化） - graduation（毕业）的孵化思路。

一个项目的成长，也是从 sandbox 阶段开始。该阶段的项目有开源的意愿和业务需求，但有点类似于人类幼崽，并不能很系统地自主操作一些事情。这时，OSPO 就能提供成长支持。

其中非常重要的一点就是 OSPO 会协助开源团队理清楚“开源的目标是什么”。目标明确之后，项目会进入到孵化阶段，OSPO 则能够帮助团队做一些落地开源的筹备工作。

与此同时，OSPO 还会协调内外部资源来配合孵化，确保项目在开源之后，以更专业的方式来打造优秀的社区。我们遇到过部分很优秀、有潜力的开源项目因团队对项目发展方向考虑不到位而暂停开源的情

况。当然，在与团队业务 leader 理清思路之后，最终我们成功地将其开源。

OSPO 的工作，还是要按照需求来。有些团队的开源经验丰富，可以自运营社区；有些团队需要利用开源来发展业务，但实际执行的人员没有接触过开源，也缺少社区经验，我们会在这样的项目和人员身上投入更多的精力。

OSPO 是一个新兴的、待明确的业务方向，要深入认识它，其实需要我们系统性地、自上而下地、整体地思考。这就要回到老子的“道法术器”模型。

“道”的层面，我们认为 OSPO 是一个“catalyst of enablement”，也就是催化剂，而不是化学反应本身。催化剂有两个作用，让不可能发生的化学反应发生，或者让发生的化学反应更高效。

往下拆解一层，“法”是回答“我们需要解决的问题是什么”。对内，OSPO 解决的是控制风险、降低成本、提升产出等问题；对外，OSPO 关注的是如何促成与开源圈的合作，如何寻找业务机会，如何助力业务结果。

到“术”的层面，OSPO 提供具体的机制流程、最佳实践、优质案例以及人才培养等。当方法论成型之后，再辅以相应的“器”来进行工具升级。

OSPO 整体思路的核心还是从业务诉求出发，先解决缺少机制以及因为缺少机制所带来的风险敞口的问题，然后服务好项目，提升整体开源认知水位。

总体来说，开源不能急功近利，它是一件长期的事情，这点必须要跟所有人达成共识。

在企业内部达成开源共识不容易。在一个概念尚未被系统性普及，且大家接触比较少的时候，都会面临“难以达成共识”的挑战。比如，大家很容易对“数据库”形成共识，但对“元宇宙”包含什么、怎么做，则不太容易形成共识。

国内开源的整体状态，其实还处于以用为主的阶段，至于“贡献开源”，还有蛮大的进步空间。因此，很多人常常意识不到开源蕴含的巨大价值。

“回馈上游，异步沟通，重视文档”等开源方法论，可以用于企业内部赋能技术开发和技术文化。只



不过，大家经常会忽略“开源”和技术文化的联系，认为我不做开源，它跟我也没有什么相关性。这是一个常见误区。

另一个问题是，对开源的认识容易走向极端。有人会过度美化开源的价值，认为项目开源后就会有发展，会有人来用，也会有人来贡献，这个想法其实还需多加斟酌；有的人担心开源的潜在风险，认为代码外部可见会导致更多合规和安全问题被发现，这都是对开源理解不够全面的表现。

OSPO 的作用就是在一定程度上应对这些挑战并找到解决方法。如果没有 OSPO，相应的挑战可能就会变成“无人区”，项目团队容易基于当时的认知来做判断，从而留下后遗症。有了 OSPO，就可以把开源社区专家的经验以及对内部业务的系统了解结合起来，得到一个融合的、有效的解决方案。

03 OSPO 提供开源专家支持，自主决策权仍在团队

一个项目要长期、可持续发展，少不了多个部门通力协作，在涉及开源时更是如此，那么此时明确主次就很重要。在蚂蚁集团，不论是使用开源，参与开

源，还是发起开源项目，OSPO 的角色定位始终都很明确。

边思康： 使用开源的过程比较复杂，同时也不是孤立的。当然，其中涉及到的业务决策、安全决策、合规决策等，其实在有 OSPO 之前早就存在了，这点需要明确。

在开源使用层面，由团队自主决策比较适合，尤其是业务选型和安全判断。因为这并不是一个开源场景，而是比较具体的技术选型和软件供应链问题。

在个别场景下，我们会对开源组件的使用有一定限制，但这并不会因为是开源有很大不同，毕竟闭源组件引入和技术选型，也是有相应风险的。OSPO 在这个维度上，一般是作为业务和安全侧的“开源专家支持”出现。

开源参与层面，首先还是要明确，团队会有自己的自主决策权，OSPO 提供的是专家经验。举个例子，如果业务团队认可现有的某个开源项目能更好地解决技术问题，那么该团队很可能就会变成项目社区的参与者，`fork` 这个项目，按照社区的要求结合内部的业务需求来推进。如果团队与社区的想法不可调和，则可能会形成一个内部分支，但大多数情况下都是以回馈上游和共建为主要思路。

另一方面，参与开源本身并不需要相应的激励机制，因为“使用项目支持业务”的外在动因足够强。我们技术同学关注的领域也都是相对前沿的，对于一些出名的、好用的开源项目，会有足够的内在动因去参与和贡献。举个例子，我们是 Ray 社区的重要参与方，因为 Ray 很好地解决了我们面临的业务问题，所以我们并不需要依靠激励来动员大家贡献，业务的存在已经足够驱动大家做出贡献。

蚂蚁集团相信微小而美好的改变，所以最开始打造 OSPO 的时候，我们核心的对外开源宗旨是希望大家“有信心，有目标”地开源。一方面，要求项目能带来实在的技术业务价值，另一方面也要求项目能够“可持续”地健康发展，吸引外部参与，从根本上推进技术和行业的长期进步。

OSPO 成立之初最主要的工作，就是制定一套能够实现这些愿景的机制，然后明确开源必须要走流程。同时，我们对流程进行了相应文化配套，有专门的自服务文档来辅助团队同学进行开源设计，明确开源目标、社区治理和运营方式。之后，项目会参与开源评审。

走过这一流程的同学，会提升对开源的整体认知。这种认知的提升对于长期的开源成功是必不可少的。

04 结合行业需要，在借鉴的基础上创新

一年多的时间，蚂蚁集团 OSPO 一步步建立起了自己的开源管理流程。事实上，在这个过程中，蚂蚁并不是单打独斗，已经有不少先行者留下了丰富的经验。当然，在将这些理论和经验用于内部实践时，蚂蚁会根据自己的情况有所侧重，找出自己的创新点。

边思康：“一步步”背后的核心逻辑，其实是优先级问题。我们的核心痛点是上面提到过的“对外开源的整体设计和收口”，做好这件事情其实会牵扯蛮大的精力。这之后，我们再慢慢把工作拓展到像开源使用、开源孵化这样的重要但没有那么紧急的部分。

Linux 基金会和 TODO Group 其实提供了蛮多很不错的教育文档，我们有所参考。有趣的是，虽然



我知道 TODO，但是在最早开展业务的时候，其实我没有仔细看 TODO 的所有设计，而是从内部的业务需求开始，以产品设计思维结合用户体验设计，与主管一起找出并决定了我们要做的前几件事情和整体的工作目标切分。而这些事情做完之后，有一次我仔细看 TODO 的内容，才发现很多内容与我们的情况一致，殊途同归了。

这种殊途同归的发现，其实也直接展现了我们的创新点。创新机会一在“把事情做实”，讲求“为世界带来一些微小而美好的改变”，所以我们更重视“未开源还不知道怎么做”的项目和人，帮助他们把一些相应的准备工作做实，让大家更懂开源。这也是一个“风险左移”的过程，可以在更上游管控潜在风险。

另一潜在创新点在开源 OSPO 的服务边界。现在业界对于“如何开展和衡量开源办公室的工作范围”这一问题，仍然处于早期探索阶段，且没有达成共识。我们的一些探索和实践，比如开源相关的工具和方法论，则可能成为一种创新。

蚂蚁的 OSPO 会去看行业需要什么，在这个维度上寻求双赢甚至多赢的机会。如果目前国内开源圈需要的是标准和工具支持，那么我们会去思考，要如何结合内部的业务诉求来对外提供帮助。

企业推广开源与社区推广开源，核心上并没有很大不同，毕竟要触达到的人群是类似的，需要找到的同路人也是类似的。

社区开源强调的是工程师文化、合作共建等，而企业作为盈利机构，在这之上，还会有更明确的商业目标，但这并不是一个社区冷启动时要考虑的事。毕

竟一个项目要解决实际问题，并且社区足够吸引人才能带来相应的成长，只靠运营是没有意义的。

企业推广开源需要注意的方面还蛮多，如果挑重点点说的话，应该是对于“开源社区”要有足够的好奇心和敬畏心。开源社区是个很有意思的存在，一方面是“理科得不能再理科”的代码，另一方面是“人文得不能再人文”的“共同体”，认知范围非常广。

弄懂开源需要大量的技术知识储备、历史储备和社会科学储备，如果不敬畏这种沉淀，则可能会导致一些不受社区认可的风险。

05 绩效量化确实很难，但有迹可循

就当前普遍情况来看，OSPO 的工作无法立即显效，而且难以量化和评估，但并不意味着无从下手，仍然有一些可用的指标可以考虑。关键是如何用互联网的方式，小步快跑，快速迭代，总结出一些规律来。

边思康：评估 OSPO 工作成效的确很难，且没有那么快能解。开源是需要长期主义精神的一个领域，这是我们必须要接受的一件事。

如果为了做成绩而做开源，则很容易跑偏，比如过度注重发声的频率和影响力。作为开源生态中一部分，OSPO 也要接受这一事实，并在这一事实的制约下去做事。情况是否会变好？我认为是肯定的。当然，这也要靠大家群策群力来形成共识，共同推进。

量化评估一个组织的效果，有三种方式，一种是 testing 选定测试维度，另一种是 benchmark 明确平均水位，第三种是专家认可。就开源领域而言，第三种衡量方法可能并不适用，所以需要从前两种入手。

测试维度其实很难有个定论，因为连我们提到的“OSPO 工作所涵盖的范围”可能都无法达成共识。

所以我们要对其中的一些维度加以甄别。比如治理水平、项目数量和质量、社区活跃指标等维度，肯定是需要的；但开源统筹水准、开源商业化产出、开源共建值这些，并不一定是每个 OSPO 都需要。可

能我们要做的，就是用互联网公司的方式，小步快跑，快速迭代，看是否能总结出一些规律来。

如果用平均水位来评估，首先要参考我们上面提到的测试维度，其次在每一个维度上，还要比较明确地定义出水位。这就更难了。以项目数量为例，多少算多？一个组件项目和一个数据库项目是否能够等同而论？另外，如果一个项目的社区做得好，如何证明是因为 OSPO 的存在才做到相应水位？

BVP 的研究显示，不同开源类型的项目，活跃度可能会有比较大的差异，泛前端的项目活跃度会两倍于数据库和 AI 工程类型的项目活跃度。这种情况下，benchmark 还是需要考虑到各个社区的自身状态。

06 OSPO 应具备三大能力

加入蚂蚁 OSPO，是边思康从事的第一份开源相关的工作，一切都是新鲜的，同时也是混沌的。经过一年多的摸索，蚂蚁集团在建设 OSPO 方面，已然总结出了一些经验。

边思康：OSPO 是外交官，最重要的是建立起沟通的能力。这个观点来自 Linux 基金会的 Keith，他是我很尊敬的一位专家，在开源领域有多年积累。很多公司会把 OSPO 看成是一个运营的岗位，对内是工具的角色，承担合规和优化的工作。但我觉得 OSPO 还有一层角色，就是外交官，而且它作为外交官的能力被低估了，这块儿会持续探索吧。

二是战略定力要与各业务团队的支持结合。这种战略定力不能只来自于表面，还是要小步快跑，把事情做实，深入浅出地服务好各个项目的业务属性。对于成熟的项目，OSPO 要能够提供外部合作渠道以及牵引更多共建机会；而对于新的开源项目，OSPO 要能够沉下来理解业务需求，并帮忙做统筹规划和运营设计，考虑你的客户需要什么。

三是 OSPO 要懂开源。你并不需要成为最懂每个社区的那个人，但一定要对开源是什么，治理是什么，运营是什么，开源如何服务于公司的各条业务线等这些问题形成自己的认知和判断，并不停地去迭代。

不要怕犯错误，也不要怕被别人认为不懂，开源圈其实是个很包容的圈子。

虽然有时候会有些自己的小骄傲和坚持，但以现在的宏观环境来说，不妨放下些姿态，因为我们需要更多对开源有兴趣、想做事的人进来，一起把这件事做起来。

最后，我以一年多的感受来收个尾吧。做开源，是需要有一种信念的。用蚂蚁人的话说，叫做“相信相信的力量”。你可以不相信开源然后来做开源相关的工作，但总会很拧巴。我觉得开源是一股很强大的力量，是一种先进的生产关系，是一种商业手段，也是一种寻求全局最优解的、有效的探索路径，我个人很相信这种开放生态的力量。

虽然开源并不能解决“内卷”的问题，但“内卷”的反义词可以是开源！如果我们能持续推动一些开源侧微小而美好的改变，能够让我们的开源技术圈和社区专业度在全球更上一个台阶，这是很令人兴奋和开心的一件事情。



边思康 蚂蚁集团 OSPO 执行负责人

边思康，蚂蚁花名“边澄”，于 2020 年底加入蚂蚁集团技术战略发展部。2021 年年初，边思康参与了蚂蚁 OSPO 的创建，之后该方面业务成为其主要工作方向。与此同时，他还兼顾技术洞察和技术战略发展相关工作。加入蚂蚁之前，边思康一直在美国从事软件研发相关工作，先后入职微软、旧金山 Square Inc（即现在的 Block Inc，一家金融科技公司）。



开源知多口

乔布斯曾在 2000 年为 Linus Torvalds 提供一份工作，工作内容是负责苹果基于 Unix 打造的内核，条件是他停止在 Linux 上的开发。但 Linus 拒绝了，还表示不喜欢苹果的 Mach 内核。

不过 Linus 却对 MacBook Air 情有独钟，还曾称赞其是“拥有过的最好的笔记本电脑”。

字节跳动加入 OSPO 群聊， 将投入超十人全职团队

2022 年 5 月，字节跳动官宣成立开源委员会，这一举动在业界的意料之中。

一方面，开源来势汹汹，正在席卷整个软件行业；另一方面，2021 年字节团队接连出现与开源相关的合规性事件。作为近年来在世界范围内都风头正盛的互联网大厂，字节跳动迎来了必须规范和重视开源的重要节点。由此，开源中国编辑部请到了字节跳动开源委员会治理运营负责人张鑫，让他来谈谈字节跳动成立 OSPO 的前因后果。

01 筹备半年，字节跳动成立 OSPO 是硬需求

开源中国：字节跳动成立 OSPO 的初心是什么？为什么选择在这个时间点做这个动作？之前的一些合规性事件在其中有何推动作用？

张鑫：2015 年至今，字节主动开源了超过五十个项目，但我们发现此前绝大多数的开源项目都是由我们工程师的个人兴趣驱动的，因此在过程中就遇到了一些规范性和管理上面的问题。这使我们意识到，开源仅仅靠工程师的个人兴趣驱动是不够的，还需要引入公司级的策略、规范与流程机制，这也是字节跳动开源委员会首先要做的工作。

同时随着业务规模的逐渐扩大，也产生了很多开源安全合规和社区运营等方面的需求，需要有一个部门能够及时响应并提供有力支撑。公司也希望能够成立一个专门的 OSPO 团队可以拉通产品、研发、市场等相关资源，对重要的战略项目进行管理，同时也可作为公司在开源领域的“外交官”，推进与各个组织、社区和行业合作伙伴在开源领域建立更好的合作关系。

开源中国：在此之前，字节跳动对于开源的态度



更倾向于？有何变化历程？对此，公司高层秉承怎样的理念？

张鑫：字节跳动对于开源的态度一直是开放、欢迎的，并且希望能够将自身实践和技术创新在社区内经验共享。和很多业内同行一样，字节接触开源也经历了从使用开源，到参与开源，到主动开源这样一个由浅入深的阶段。在业务发展过程中，字节也广泛采用了开源技术，并对原有开源项目进行技术优化，反哺社区，形成了一种良好的开源文化。

开源是软件世界的根基，字节跳动看重参与开源的长期价值，在创新生态日益复杂的今天，开源软件极大地提高了我们创新的速度和质量。拥抱开放式创新，字节跳动将始终抱着学习和合作的态度，通过更好地参与到开源生态当中，不断拓展新的技术和创新

空间，携手产业上下游的合作伙伴共同摸索前行，促进生态发展。

开源中国：成立开源管理办公室。在做此动作之前，字节酝酿了多久？前期又做了怎样的准备？调研的重点是什么？

张鑫：我们是从去年年底开始在内部筹备 OSPO 的成立工作，到正式成立用了大概半年的时间。

在前期准备的过程中，我们系统地盘点了积累的开源项目，和此前在参与开源过程中的相关流程规范，以及相关人力情况，讨论哪些是可以复用的，哪些还存在不足。

此外，我们还对业界在开源管理方面的做法进行了调研，主要分析了国内外云厂商的开源管理和商业化流程，以及如何处理开源和云业务的关系，并且重点研究了一些在开源管理方面走在前面的公司的管理经验。

02 战略评审与治理运营并举，将投入超十人全职团队

开源中国：请您简略介绍一下字节跳动 OSPO 的组织架构设置。在设置上，除了虚职，会投入多少全职同学？

张鑫：当前字节开源委员会已经完成了整体架构的搭建和角色划分。

由于大的开源项目往往需要长期投入，同时又需要协调公司内外很多资源，因此由字节跳动 CEO 梁汝波和两位技术最高负责人杨震原、洪定坤担任开源委员会的 Sponsor。在此基础上，由洪定坤担任开源委员会的主席，负责协调和支持开源相关事务，下设两个小组，分别是王剑负责的战略评审组和我负责的治理运营组。

其中，战略评审组统筹开源战略制定与开源项目评审，比如对于开源项目，按照技术影响力分成不同层级，对应地制定合理的运营目标和价值体系。小组成员包括各个技术领域（如基础架构、数据、AI 等）



字节跳动开源委员会组织架构图

的代表和 ToB 业务（主要是火山引擎和飞书）的代表。

而治理运营组统筹开源合规治理与开源运营，包括搭建整个开源运营的基础设施、做好开源文化和最佳流程的宣贯和落地，以及为重要开源项目提供一线运营和治理方面的帮助等等。小组成员由项目布道师、市场运营团队、开源法务，以及搭建平台和工具相关的同学等组成。

合作和复用是开源委员会展开工作的重要准则之一，目前团队的运作方式是虚实结合，既有实线，会投入超过十位全职同学来牵头市场运营、法务、安全合规等工作；同时也会有虚线的合作和复用，比如相关平台工具的搭建就会由分散在不同研发团队的同学来贡献，最大程度地复用各个团队已经做过的工作。

开源中国：资料显示，字节此次的开源委员会定位为“资源中台”，具体来说，是怎样的一个“中台”？

张鑫：所谓的“资源中台”的核心，就是希望开源委员会可以对字节的开源发展提供更好的支持，主要体现在以下两个方面：

首先，开源委员会战略评审组会和各项目维护团队一起，逐一对项目做评审，看是否应该加大投入力度或者有更高的目标，为项目团队提供开源战略支持；

其次，在治理运营方面，对于重量级的项目，在运营、用户增长等方面投入的资源相应会更多一些。项目维护团队需要持续维护项目技术层面的迭代，而治理运营组负责提供更加体系化的支持，比如宣发渠道统筹、内容策划、社区开发者关系维护等。

03 对内简化流程、强调有用性， 不为开源设立 KPI

开源中国：在目前的 OSPO 设置下，字节跳动给予项目的自由度有多大？资料显示，字节认为应该在规则设置上尽量减少“打扰度”，为什么？要如何简化流程呢？

张鑫：很多情况下，字节一直鼓励大家对上游社区提交贡献，免去了内部流程的麻烦。从我们 OSPO 的架构设置也可以看出，OSPO 的成员，尤其是战略评审组的成员都是来自于各个技术领域和一些业务线的代表以确保技术中立，一个基本的审核标准就是这个开源项目要对其他人、组织、公司有用，而且我们能保持开放的心态去维护。而治理运营组的主要职责是帮助项目守好底线，与各团队充分协同，提供资源对重点项目做好支撑。

在规则设置上，我们默认开放信任，有效合作，避免简单事情复杂化，这也就是在规则设置上减少“打扰度”。目前开源委员会的相关流程都做得很轻，还是以提供能力和资源支持为主，希望在保持个人自发性的同时，通过消除障碍鼓励大家更多地参与开源。但我们也组织很多合规相关的培训，保证大家在使用开源、贡献开源的同时不踩红线。

开源中国：“不会为了开源设立 KPI”，那么 OSPO 的工作要如何考核和衡量呢？

张鑫：字节一直秉持的原则就是不为开源项目设 KPI，我们看重开源项目是否真的有用、有长期价值，不希望因为设定了硬性的 KPI 而导致动作变形，同时我们也不会将开源项目的商业变现作为唯一目标。

对于 OSPO 的工作以及一些开源项目的成果，我们会设定一些“北极星指标”，比如看整个开源项目在行业的市场占有率，这是跟这个开源项目到底有没有人用强相关的一个指标；比如看开源项目的贡献度，包括项目被 fork 的情况、第三方开发者的数量和应用的数量，如果是一个足够好的项目，自然会有很多贡献者愿意参与进来。除了使用的广度，也会关注使用的深度，比如在行业里有没有企业用户在深度使用这个项目，是否沉淀了一些标杆案例出来。

归根结底，“北极星指标”更多是围绕这个开源项目是不是能实际落地、有没有人愿意用、能不能在生产环境或者实际业务中产生价值来展开的。这些都比 GitHub 上简单的 Star 数更有意义，同时，即使是这些“北极星指标”也不会变成强 KPI，而是作为辅助评估的工具和引导工作方向的手段。

04 对外项目要“少而精”，注重 合规和价值

开源中国：自主对外开源方面，字节有什么样的标准和原则？这是基于什么之上的考量？

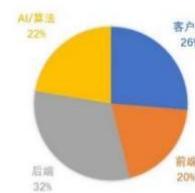
张鑫：在对外自主开源方面，首要标准就是保证安全和合规，希望通过 OSPO 引入公司级的战略、规范和流程机制。

同时，我们希望来自于字节的项目是“少而精”的，能够更加充分地利用公司的各种资源，开源出一批真正对产业界有价值的项目，期待字节自主对外开源的项目能够成为源头活水带出奔涌江河，参与甚至引领全球软件创新生态。

开源中国：目前，字节对外开源的情况是怎样的？有没有在 GitHub 或 Gitee 上专门整理成库？现在，有什么开源项目是字节主推的？

张鑫：目前，字节已经开源了超过五十个项目了，按照技术领域分类，其中项目数量排在前三的分别是基础架构、AI 平台和音视频领域。比较有代表性的项目包括现代 Web 开发框架 Modern.js、微服务中间件 CloudWeGo、高性能分布式训练框架 BytePS、联邦学习平台 Fedleaner 等等。

字节跳动的开源现状与项目概览



ByteDance 字节跳动

这些项目我们会按照技术领域进行整理，主要的考虑是：

- 通过独立的组织来汇总字节云原生态的项目，输出统一的开源品牌认知和推广，同时也便于版本管理；
- 期望通过相对中立的组织来推动三方合作，包括开源基金会（CNCF）、开发者用户和企业用户；
- 在云原生方向，业界主流的做法都是采用独立的组织来运作（主要目的也是前 2 点），我们也期望保持一致。

今年我们还将推出一个数据平台方面的精品项目，敬请期待。

05 首先侧重运营治理的体系搭建，未来将与行业共同成长

开源中国：字节的 OSPO 相较于其他公司，差异化大吗？有什么创新点？之后还会有演变吗？

张鑫：字节的 OSPO 才刚起步，还有很大的发展空间，我们也一直抱着谦虚开放的心态希望向业界学习更多开源治理方面的经验。

接下来，OSPO 在短期内的重点工作将侧重在运营治理工作的体系化建设，中长期来看希望能够打造出一批精品开源项目。字节希望通过开源委员会一段时间扎实的工作，能够更好地摸索出一条企业级开源治理流程和方法，同时也在摸索中不断调整战略方向，从而能够真正地提升字节内部每一位使用开

源、贡献开源的同学的开源能力和认知水平。

开源中国：实际情况中，很多程序员开源知识储备不足，OSPO 还需要肩负布道者这样一个角色，字节这方面有何投入或计划？

张鑫：字节开源委员会的成立一方面是希望帮助我们的开发者在使用、贡献和创建开源项目时能更加顺畅，另一方面是希望通过积极参与并回馈社区来促进创新。

要达到这样的目的，一定是要 OSPO 的同学有能力、有热情、有方法地承担起开源布道者的角色。这不仅需要将开源理念和方式在内部进行传播，另外也需要将字节在开源领域的一些积累和最佳实践传播到业界。

未来，我们希望字节的 OSPO 能够更加主动和具有战略性，将 OSPO 打造成为一个卓越的咨询中心，在指导公司了解最新的技术趋势、开源发展方向以及最佳实践方面发挥重要作用。

开源中国：整体上，公司未来对于开源的战略是怎么样的？有没有一个规划蓝图？

张鑫：这次 OSPO 的成立意味着字节跳动的开源战略更加清晰，但我们还有很长的路要走，希望通过 OSPO 的成立帮助字节吸引更多顶尖人才，站在全球的视角上，通过开源软件激发创新力量。

开源已经成为了技术创新的关键路径，字节跳动希望自己的开源战略能够更加落到实处，主导一些精品开源项目，培育社区增长，借助开源力量推动更多的行业内和跨行业的合作，与行业共同成长。

张鑫 火山引擎副总裁，字节跳动开源委员会治理运营负责人

原才云科技 CEO（连续入选杭州准独角兽企业），曾是美国谷歌资深软件工程师，6 次获得谷歌副总裁和总监颁发的即时奖励。曾作为技术带头人从事谷歌容器化集群管理系统的研发，自动化管理 95% 以上的谷歌数据中心服务器，并最早参与了谷歌公有云的产品设计与研发。

张鑫在美国卡内基梅隆大学（CMU）获得计算机博士学位，期间在分布式系统和网络安全领域的顶级国际会议发表学术论文数十篇，被引用上千次；研究成果曾被美国 Economist、英国 BBC、瑞士 RTS 电视台等国际媒体报道；并获得国际顶级安全学术会议 IEEE Security and Privacy 的“Test of Time”论文奖。张鑫曾获“海归科技创业者 100 人”、“清华大学优秀毕业生”等称号，并入选清华五道口金融学院学员。



金融科技 OSPO 先行者， 微众银行如何干？

2019 年年中，微众银行开始组建开源管理办公室，这个时间点对于一家国内的金融科技企业来说，是很超前的。微众银行是一家互联网模式银行，他们认为一个开放银行，除了建设自身的开放平台之外，还需要在技术上开放创新，最终实现与各类合作伙伴在业务场景上的开放协作。

而开源是开放创新的最有效手段，只有通过开源，所有参与者才能一方面享受到开源带来的红利，同时也能够通过自身努力让生态更加繁荣，加快技术创新的步伐，实现共赢。于是，开源中国编辑部特意请到了微众银行 OSPO 负责人钟燕清，来讲述微众银行的 OSPO 故事。

01 因高层重视而起，微众 OSPO 是自上而下的

开源中国：作为一个 OSPO 老兵，您这些年最大的感悟是什么？

钟燕清：大家好，我是钟燕清，目前是微众银行开源管理办公室的负责人，同时也负责微众银行内部科技 PMO 及开发过程管理等方面的工作。

在 OSPO 工作的这段时间，我感触最深的就是开源在我国的发展越来越快了，企业对开源的认知也越来越深刻。三年前，基本上只有少数头部企业在参与开源，只有大企业才有类似于 OSPO 的组织来系统地推进开源战略。现在，一大批各种类型的科技公司都纷纷成立了开源办公室，大家携手共进，一起推动企业拥抱开源的速度。

开源中国：您在之前的演讲提过，CIO 马智涛对开源的重视是微众银行成立 OSPO 的一大原因。具体来说，他是如何推动的？高层重视对 OSPO 的最后落地有多重要？

钟燕清：首先，我们的 CIO 马行是一个非常有技术情怀以及崇尚工程师文化的领导。我现在还清楚地记得，2019 年上半年，马行邀请我加入微众银行并且负责开源管理办公室时候对我讲的话，他认为开



源在过去 20 多年的互联网技术发展中已经发挥了决定性的作用，继续往前看，开源对于技术发展、技术创新、技术生态形成的作用只会越来越大。而且，未来的商业模式一定是分布式的，一定是多方参与、共享资源、智能协同、价值整合、模式透明、跨越国界的。

而构建这样一个分布式商业模式所使用的技术也一定是分布式而且应该是透明的，开源无疑是构建这个生态的技术底座的最佳选择甚至是唯一选择。

所以，在 2019 年年中，我们一方面在银行内部加快开源治理体系的搭建，成立 OSPO 团队；同时，我们在 7 月底举办了首届 FinTech Day（金融科技



开放日），在会上马行首次系统阐述了微众银行在金融科技领域的创新进展，并且对外宣布“金融科技全面开源”。

现在再回头看，仍然可以感受到当时我们的很多观念和行为都是非常有前瞻性的，特别是在金融行业，我们是较早能够将开源与公司的技术战略紧密挂钩的企业之一。马行对开源的深刻理解以及对技术趋势的洞察，是我们这么多年能够在拥抱开源方面取得一定成绩的非常重要的原因。

所以，对于很多准备加强拥抱开源的企业而言，我认为相关执行团队与 CIO 等高层领导一起达成共识、明确目标，是至关重要的一个环节。

02 参照开源基金会模式，微众 OSPO 在不断成长

开源中国：2019 年 7 月，微众银行对外宣布金融科技全面开源，并成立开源管理办公室。在此动作之前，微众银行酝酿了多久？前期又做了怎样的准备？

钟燕清：微众银行的管理团队和技术团队，对于开源的认识应该是非常深刻的。从 2014 年立行伊始，我们就坚定地走了一条基于开源产品构建自主可控和分布式架构的银行核心系统的技术路线，各类先进的开源技术应用在微众银行众多关键业务场景中；2017 年，依托金链盟（微众银行牵头发起的金融行业的区块链联盟组织）开源工作组协作打造了金融级联盟链底层开源平台 FISCO BCOS；2019 年，成立开源管理办公室，对外宣布金融科技全面开源，进一步加速推动开源生态的建设。

所以，我们可以看到，微众银行拥抱开源的路径也是一步步向前推进的。从广泛应用开源技术，到以重点项目为核心发展开源项目社区，以及从公司整体

战略角度出发，全方位地推动开源，每个阶段也积累了丰富的经验以及打下了坚实的基础。

开源中国：请您简略介绍一下微众 OSPO 的组织架构设置。在设置上，除了虚职，有多少全职的同学？据知，微众银行的整个开源管理工作组属于银行信息科技委直接领导，这意味着什么？能否为 OSPO 的横向打通上加分？

钟燕清：微众银行的 OSPO 组织定位就是银行内部的“开源基金会”，所以在组织架构还有职能定位等方面我们都会深入的去研究和参考开源基金会的管理体系与运作模式。

开源管理工作组隶属于银行信息科技委直接领导，在这个组织里面包括我们的 CIO 以及各大主要研发部门、职能部门的负责人等。

这里简单说一下，为什么我们强调开源管理工作组隶属银行信息科技委直接领导呢？因为信息科技委是银行最高的与信息科技相关的决策机构。这相当于我们在这个体系里面将开源与研发管理、运维管理等放到了同一个层面，可见微众银行对于开源的重视程度。

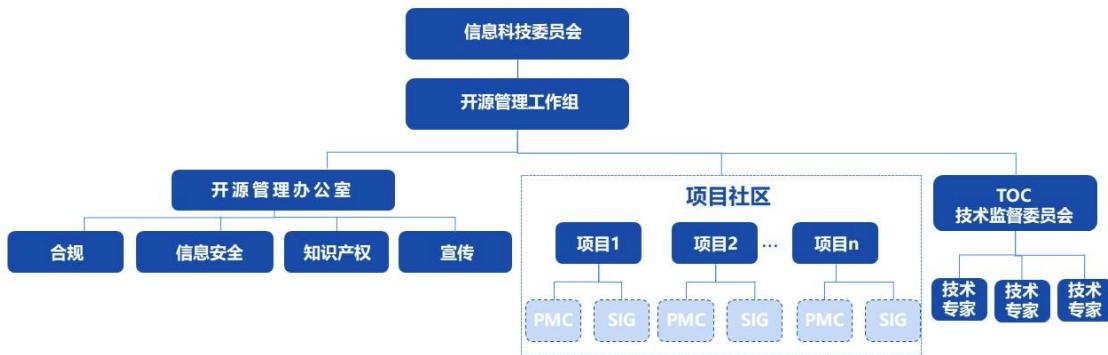
除了 CIO 之外，其他中高层管理团队成员也能充分理解开源战略并且在具体实施推进过程中给予指导和支持，这也是组织设计上的考量点之一。在开源管理工作组领导之下，其中包含了开源管理办公室，我们相当于整个公司具体负责开源战略落地以及日常运营的组织，这里全职参与的有 4 名同事。当然，还包含合规、信息安全、知识产权、品牌宣传等专业岗位。

另外，也包含 TOC 团队（Technical Oversight Committee，技术监督委员会），里面由不同部门的技术专家组成；还有就是每个具体的开源项目或者项目群的内部社区运营团队。

开源中国：微众银行在内部成立 TOC 技术委员会来技术驱动决策。这与开源社区所存在的 TOC 有何不同？其中的成员是如何选举上来的？他们的权限如何？对项目有决策权还是建议？

钟燕清：微众银行内部的 TOC 组织并不是在成立 OSPO 的同时就成立的，而是经过近一年的准备在去年 2021 年才正式成立的。成立 TOC 组织主要

参照开源基金会模式的开源治理组织架构



是与我们在大力推动内部开源有关，也跟微众银行内部的科技团队组织架构有关。

我们的科技条线组织架构核心是产品驱动的敏捷模式，按照不同业务板块划分了几个大的技术部门，每个部门都包括产品经理、研发、测试及运营等职责。所以，我们总体上在技术上是比较开放、包容的，不同部门也逐渐形成了自身的技术风格。

成立 TOC 是为了加强跨部门的技术交流、共享与协作。针对 TOC，我们也设计了任期的机制，首批 TOC 成员是由每个部门分别推选的，任期周期到期可以连任也可以推举其他同学接任。总体上，这批人都是在公司内部公认的有影响力的技术专家，并且对各自部门的技术栈、架构等都非常熟悉。

他们主要做以下一些事情：

第一、制定技术标准和规范，我们希望银行内很多技术相关的标准和规范不是由某一个集中的部门制定，其他组织或团队仅仅是执行而已，而是通过这种分布式、共同参与的方式去推动技术标准和规范的完善；

第二、解决技术难题，因为 TOC 成员都是每个部门顶尖的技术大拿。大家可以在一块去讨论很多技术难点，共同去攻克技术难题。同时，能够经验共享，避免持续的轮流踩坑等现象；

第三，推动内部开源，作为每个部门的先锋代表，连接部门内部和其他部门之间的内部开源的项目推广、

协作等一系列事情；

第四，推动一些公共平台的建设，与第一点一样，我们希望公司内部的公共平台通过利用内部开源、共同协作的方式不断完善。

总体来讲，微众银行内部的 TOC 并不是一个具备很强的管理职能的组织，而是纯粹从技术角度大家通过分享、协作等方式达成共识，共同推动公司技术氛围越来越浓。

03 安全是底线，旗帜鲜明支持开源和对上游反馈

开源中国：在开源使用层面，微众银行有建立起怎样的合规机制和流程？

钟燕清：对于金融机构而言，合规、安全是非常重要的，是底线。所以，在开源治理的各个方面，我们是有比较完善和严谨的流程制度保障。

在开源使用层面，我们有应用开源软件的管理规范，定义了企业应用开源软件的全生命周期，比如引入阶段、使用阶段、退出阶段，同时明确了整个流程参与的各类角色。在流程的每个环节也明确了具体的操作标准与责任人等。

其实，除了制定流程规范之外，我们更看重的是通过系统集成的方式实现自动流程的控制。所以，我们也在现有的 DevOps 体系里面加入了应用开源

软件的各个环节，能够真正落实管理要求、快速识别和防范各类风险。

开源中国：对金融来说，开源安全是关注的重中之重。微众银行是如何通过 OSPO 来提升开源安全性的？有什么专门的针对措施吗？可以是技术上的，也可以是管理机制上的。

钟燕清：前面我们也反复提到，应用开源最重要的就是合规、安全，我们有专业的信息安全部队利用各种技术手段保障公司各个环节的信息安全。

OSPO 也会从开源治理的角度设计各种流程来降低我们应用开源以及对外开源的各种风险。我们的重心还是建立系统化的 DevSecOps 体系，并且建立与开源技术相关的知识库，植入各类管控规则（比如对待不同等级漏洞的策略、黑名单策略、许可证兼容策略）等，真正做到及时、准确和动态的管理。

开源中国：在开源参与层面，微众银行又有怎样的策略和激励机制？对上游的反馈机制如何？

钟燕清：在参与开源的激励方面，我们正式发布了开源激励管理办法，全方位对参与开源的各类行为进行奖励。

比如，我们设置了杰出贡献个人、杰出贡献团队等奖项。当一个开源项目取得一定成绩、社区发展良好、取得了一些阶段性成果时，就会被开源管理办公室提名。相关团队和个人都可以申请杰出贡献团队及个人等奖项。

另外，我们也是旗帜鲜明地支持对上游开源社区的深度参与。如果某个同学获得知名上游社区的 committer 身份或者认可，也会成为杰出贡献个人奖项的获得者。当然，我们还有很多很丰富的积分方式，鼓励所有同学都能参与到开源活动中去。

04 核心技术全面开源，企业主导 开源必须严把质量关

开源中国：除了使用和参与，微众银行也是开源项目的发起者。对此，微众银行对外开源的宗旨是什么？微众银行制定了怎样的流程来规范扶持？以企业名义来推广开源，与社区开源有哪些异同？有什么特



2021 年 8 月，微众银行为优秀开源个人颁奖

别要注意的？

钟燕清：微众银行对外开源的主要宗旨就是核心技术全面开源。另外，就是一些前沿技术，我们也会花大力气去深入研究，并且通过开源的方式加快产业发展。比如我们在区块链领域，包括从底层平台到中间件再到上层应用组件等 10 多个项目都进行了开源。

同时，为了保障对外开源项目的质量，并且符合银行在安全、合规等方面的要求，我们也是有一整套软件对外开源的管理制度和流程，包括开源项目立项、实施、对外发布以及正式运营等各个阶段。每个阶段也都有明确的要求，需要经过各方充分的讨论以及相关领导的审批，确保此项目是可持续发展的。

以企业名义来推广开源，最终的目标还是希望能够推动社区繁荣，有更多的参与者和合作伙伴，最终实现社区共治。但是在此过程中，发起开源项目的企作为最重要的组织，需要在社区建设、合规安全等方面高度重视，因为我觉得开源社区本身是一个高度透明的环境，项目本身质量不过关或者社区氛围不好，对于主导企业也是有所损害的。

开源中国：这一整套开源管理流程，是如何一步步建立起来的？借鉴过谁？创新点又在哪里？

钟燕清：Linux 基金会与 TODO Group 一直是致力于推动企业拥抱开源及建立 OSPO 的组织，发表了一系列企业参与开源的指南文章以及很多优秀企业的经验分享，非常有参考价值。

我们也是通过学习相关理论体系以及研究其他在开源领域有卓越影响的头部公司的开源路径，来制定适合微众银行的开源相关战略、阶段性目标和发展路径等。如果讲我们的创新点，我个人觉得目前我们试图将内部开源、外部开源以及内部的技术文化等融合贯通，通过开源的形式推动开放式组织的形成，应该是一个非常有意义的尝试。

05 国内金融科技开源建设还在起步，需要长期主义

开源中国：资料显示，微众银行的开源战略愿景是“希望成为金融科技领域开源生态建设的领导者”。目前，金融科技领域的开源现状是怎样的？存在怎样的问题？

钟燕清：我个人认为在国内金融科技领域的开源生态建设还是处于起步阶段。在过去 30 多年的开源文化发展中，更多的是以使用者的身份在参与，对上游生态的贡献以及由金融科技领域企业发起的顶级开源项目都比较少，还没有形成广泛的真正符合金融行业需求和特点的开源社区。

当然，随着各类新技术的快速发展以及金融行业的数字化转型，很多企业在快速向参与者、贡献者及发起者这些方向前进。特别是，去年 10 月底五部委联和发布了《关于规范金融业开源技术应用与发展的意见》，应该讲这个《意见》对于金融机构如何利用开源、参与开源提供了明确的方向和各类原则标准，也进一步加强了金融机构使用开源及参与社区的积极性。我相信，金融领域除进一步提高应用开源技术的能力之外，很快也会在社区贡献及社区建设等方面发挥更大的作用。

开源中国：“对企业而言，拥抱开源是长期主义。”的确，OSPO 的工作无法立即显效，而且难以量化和评估。对此，您有什么看法？微众银行在这几年的 OSPO 实践，有什么可以总结出来的经验和成就吗？

钟燕清：首先，开源本身就一定是个长期主义，任何一个开源社区、一个技术生态都不可能是短期能够形成的。

另外，除了一些基于开源建立商业模式的开源公司之外，大部分企业参与开源的收益可能都不是非常显性的，比如技术的影响力、对人才的吸引力、参与社区对企业内部的技术应用效率提升等，并不能很直观地衡量。

所以，我觉得一个公司决心要通过建立 OSPO 来推动拥抱开源的进程，首先一定是要在内部得到高度认同，并且得到高层领导的大力支持。在具体执行阶段，也应该客观评估企业自身的技术积累和发展阶段，制定比较务实的发展路径，明确在参与开源这件事情上的角色定位。

我个人认为，并不是一定要主导发布一个开源项目才叫拥抱开源，能够在使用开源的过程中积极的参与社区，反馈问题或建议，或者在上游社区里面做更多的代码贡献，这些也都是非常有价值和有影响力的。

我的观点一直都是，企业参与开源一定不是一个追热点、赶潮流的事情，一定要踏踏实实地参与到社区中去。正所谓“念念不忘，必有回响”，长期地努力和坚持必然也会得到社区的回馈与尊重。

钟燕清 微众银行 OSPO 负责人

在加入微众银行之前，钟燕清一直从事与研发以及技术团队管理等方面有关的工作，先后在平安、腾讯、顺丰等企业任职研发经理、研发总监等岗位。目前为止，其整个职业生涯都没有离开技术。

前期的从业生涯中，钟燕清主要作为开源软件的使用者的身份接触开源。他身为研发总监，需要带领团队对所使用的语言、开发框架以及一些应用组件等进行判断选型。因此，在这一阶段，他更关注不同开源项目的技术特性、生态成熟度、发展前景以及与团队的能力适配等。

2019 年年中，钟燕清加入微众银行开始组建开源管理办公室。这是一个全新的视角，他开始站在一个企业的角度，去看待开源。同时，钟燕清也成为了一名开源的布道者。无论是在企业内部，还是面向社区，他希望能够更广泛地推广开源文化，让更多组织和个人理解开源，认同开源，参与开源。

“洗钱”

在 Deepin（即深度公司，统信软件的前身）刚刚创立时，基本上全靠创始人刘闻欢个人的资金在维持。

引用 Deepin 副总张磊的话来说：“他原本就已经财富自由了，每年都花几百万来做这件事，花着花着财富就不自由了。”

这段时期还发生了一件有趣的乌龙事件：当地税务局发现这家公司每年都有几百万的开支，却一点收入都没有，怀疑他们从事的是违法洗钱活动，为此还特地派稽查人员上门调查，最后得出的结论是：他们确实不挣钱。



公司简介：统信软件技术有限公司由国内领先的操作系统厂家于 2019 年联合成立，专注于操作系统的研发与服务，总部设立在北京，同时在武汉、上海、广州、南京、成都、重庆、西安、太原、深圳等地设立了地方技术支持机构、研发中心和通用软硬件适配中心。

底线



在创业初期，KodeRover 创始人 Landy 和 Grant 有一年多不仅没拿一分钱工资，还用自己的积蓄、早期客户的付款、再加上银行贷的 30 万给员工发工资，最穷的时候公司账上只剩 700 块，工资都发不出来。

Landy 坦言，自己当初设定的心理底线就是个人负债 100 万，如果超过这个数，可能就会考虑放弃创业回去打工。

而现在，KodeRover 已然是一个高速发展中的开源创企新星。

公司简介：KodeRover 是开源分布式持续交付产品 Zadig 背后的团队，专注于云原生软件交付产品的研发。KodeRover 由来自七牛云、斯坦福大学和 Google/Facebook 的资深专家和连续创业者联合创办。目前，KodeRover 已经服务于互联网、云计算、新零售等多个行业，但在云原生分布式交付的赛道上，才刚刚启程。为迈向更丰富的企业级技术和业务场景，KodeRover 核心产品 Zadig 于 2021 年 5 月全面开源。未来可期，KodeRover 已经准备好全新启程。

PR

“今天你提 PR 了么”，已经成为飞轮科技内部流传的梗。

除了 HR、财务等职能同学，几乎人人都是开源社区 Contributor，就连市场和运营同学都会给开源社区做些力所能及的贡献工作，例如文档纠错、用户手册编撰、技术内容翻译等，开源氛围可想而知。

飞轮科技唯一的行为规范是自由，不打卡不考勤，没有任何繁文缛节，极度扁平化，鼓励自由思考，鼓励平等沟通。



3 亿

飞轮科技于去年 12 月成立，迄今还不到一年，目前正在基于 Apache Doris 研发新一代云原生实时数据仓库 SelectDB。

今年 4 月，飞轮科技完成了超 3 亿元的天使轮和天使+轮融资，投资方为 IDG 资本、红杉中国等顶级 VC。飞轮科技表态称，开源与云的结合是其在这一赛道赢得胜利的关键。在多云场景的选择上，独立数据仓库厂商的中立地位难以被取代。



公司简介：飞轮科技是 Apache Doris 社区的贡献者力量之一，创始人团队由原百度智能云的初创成员和 Apache Doris 项目的核心成员组成。目前正在基于 Apache Doris 研发新一代云原生实时数据仓库 SelectDB。当前，飞轮科技已经同部分业内头部云厂商开展合作，同时与更多公有云、行业云厂商的合作也将逐步落地。

《创业小辑》栏目征稿

随着开源在我国不断发展，越来越多依托开源项目创业的企业相继涌现。目前这支队伍并不算壮大，但相信在不久的未来，将会发展成一股不容小觑的力量。《创业小辑》栏目旨在记录这些勇立潮头的开源创企，展现它们的奋斗历程、文化氛围。如欲投稿，请联系我们：肖滢 18370998278（微信同号）



老同事拉我创业，做一家开源存储公司

从豆瓣出来后的一段时间，科班出生的苏锐做了一些不那么“技术”的事儿。

2014~2015年，苏锐在O2O领域里打拼，瞄准的是上门汽车服务这一细分领域。“在O2O创业的这个过程里，我不再是公司里的一个职员，而是站在了更高处，可以更全面地去看待市场是什么样的，用户是什么样的，公司又是什么样的。”苏锐表示。

在这样一个位置上，苏锐开始拥有更完整的公司运营视角。

另一头，苏锐在豆瓣工作时的mentor刘洪清(Davies)，跑去美国成为了Databricks的早期员工，当他提议为Databricks自研一个新的存储方案时，却并没有得到公司的支持。后来，他利用业余时间单枪匹马地写了个原型出来。

2017年，刘洪清叫上了苏锐，说要创业。

终于，苏锐又回到离技术更近的地方——云原生文件存储，俩人随即成立Juicedata公司，并将其产品命名为JuiceFS。

作为1号员工，苏锐不仅是Juicedata的合伙人，还充当了“技术与市场的桥梁”这样一个角色。“Juicedata最初瞄准的就是全球开发者，Davies抓住技术痛点以及可行路径，而我要把技术创造出来的产品，带到市场、用户那边去，建立两者之间的连接。”苏锐说。

01 被AWS S3绊住后，Juicedata诞生

“互联网技术一直在更迭。仅从我自己的角度来说，就已经经历了三个（技术）周期了。从中国接入宽带互联网，到移动互联，再到现在，其实C端网民都已经被普及得差不多了。这几年，To C的领域里，大家都在讲用户红利消失。那么，还有什么红利是存在的呢？还有什么东西是不断增长的呢？”

“现在，整个世界都在数字化，数据领域变得越



苏锐（右）与刘洪清（左）的合照

来越有挑战性。企业端正在快速普及云计算，围绕数据的各种需求都在催生新事物的出现。”

云原生文件存储技术，在苏锐的眼中是个不那么大众的细分技术领域。

简单来说，云原生文件存储就是运用在云原生环境的存储技术，一般具有高可用性、强大的可扩展性、可靠性、动态部署等特性；并且针对公有云、私有云和混合云等不同形态的平台，也有不同的解决方案。

在苏锐的描述中，存储技术演化到今天，是跟随技术变迁进程的。

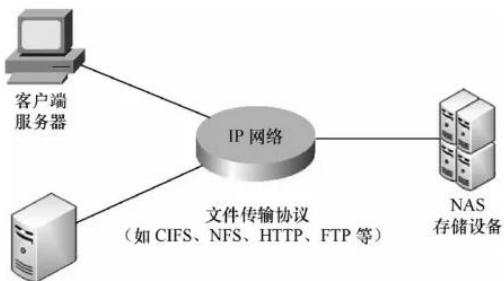
在计算机发展之后，文件存储其实是一个很古老的传统领域。它的第一代产品诞生于90年代——传统旧式的单体式存储，它与专有硬件和专有软件进行捆绑销售。

90年代，上个网还需要拨号。当时的网络存储

一般使用的是专有硬件设备方案（通常指 NAS，Network-Attached Storage，网络接入储存），通过特殊的高性能通讯硬件给其他应用提供访问接入。

1992 年，NetApp 诞生。它是这一代企业存储技术的代表，其产品形态就是软硬件一体的，发展至今 NetApp 成为这一行业的龙头。（PS. NetApp 是闭源的。）

“那个时候，每家公司想要买存储容量，就必须买一个硬件柜子，里面插满了硬盘。在门户网站的时代，网站只能发布信息，这样的存储技术是够用的。”苏锐表示。

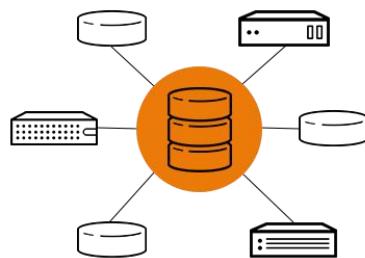


2003 年，Google 的 GFS 开创了先河，第一次用普通的 x86 机器和普通硬盘搭建了大规模存储，分布式存储出现了。2005 年，这一成果与 MapReduce、BigTable 一起成为谷歌引爆大数据时代的“三驾马车”。

同样在 2005 年左右，Web2.0 产品形态涌现，其最大的特点是 UGC（User Generated Content，用户生产内容）。其中，以 Facebook、Twitter、人人网等社交网络为代表。这意味着，每天每个网民都在创造大量数据，互联网的数据量比上一个时代有了数量级上的提升，继续用“买柜子”的方式，根本满足不了这么大的数据需求。

于是，企业存储的“二代目”——SDS（Software Defined Storage，软件定义存储）出现了。不同于 NAS，SDS 一般都在行业标准的 x86 系统上执行，消除了软件对于专有硬件的依赖性，大幅提高了灵活性。

在 2005~2009 年这段时间窗口，全球范围内诞生了很多的分布式文件系统的产品，比如 HDFS



SDS 是一个技术堆栈层，可以使用行业标准服务器提供服务

（Hadoop）、Ceph、GlusterFS、MooseFS 等等，这些软件都得到了广泛应用。而且值得注意的是，其中大多数都为开源软件。

又一个十年后，移动互联网让互联网上的数据再次以数量级形态上升，让上一代的文件系统有点捉襟见肘了。

此前，亚马逊推出 AWS 公有云，云计算搅动一潭春水。AWS S3（Simple Storage Service）赶上了好时候，它有弹性、可伸缩、想存多少就存多少，其拓展能力和成本优势立马让它脱颖而出，整个互联网的数据都在往 S3 上搬。

但是，S3 在满足拓展能力的同时，相比之前的存储产品又舍弃了很多其他功能，比如丰富的访问接口、数据的强一致性保证、高性能的元数据访问、原生目录树结构、有原子性保证的 rename 等基础操作等。很多公司不得不做大量的工程改造才能把 S3 用起来，Juicedata 创始人刘洪清当时所在的 Databricks 就在其中。

2016 年，刘洪清给公司提案，要自研新的存储方案，以系统性地解决问题。当时的 Databricks 也还在创业初期，公司研发团队大多擅长分布式计算领域，认为还不适合投入分布式存储的研发。这个决定也促成了 JuiceFS 的诞生。换个角度说，是 S3 的短板促成了 JuiceFS 的诞生。

在设计中，JuiceFS 是一款面向云原生环境设计的高性能分布式文件系统，为云环境设计，提供完备的 POSIX、HDFS 和 S3 API 兼容性。使用 JuiceFS 存储数据，数据本身会被持久化在对象存储

(例如, Amazon S3), 相对应的元数据可以按需持久化在 Redis、MySQL、TiKV、SQLite 等多种数据库中。

如果说 2005 年左右那次的技术更迭是面向机房的。那么, 这次 (2015~2016 年) 存储技术的创新冲动则是由云时代开启的。

“我们希望产品能在云上使用, 但这些产品在架构上并不适合云, 也发挥不出云的优势。而 S3 又存在功能上的不足, 满足不了复杂的数据需求。如果能有一个全功能的文件存储, 又同时能兼具 S3 的产品优势, 就非常理想了。” 苏锐表示。

2017 年 4 月, 刘洪清和苏锐着手组建了 Juicedata。在他们的构想中, Juicedata 要解决过去文件系统在规模和能力上的一些限制, 在云上提供一种新的体验, 并且要继承以前几十年积累下来的文件系统上层构建的生态。

02 转向开源后, 我们从 AGPLv3 换成了 Apache 2.0

2018 年, JuiceFS 很快推出了云服务版本, 走上了商业化运营的正轨。

最初 Juicedata 只能请一些很早期的初创企业来尝试; 到后来, 有更成熟的 B 轮、C 轮融资阶段的企业敢用了; 又随着他们的背书, IPO 阶段的企业也加入了进来。这是一个不断摸索的过程:

“一开始, 我们非常坚定地拥抱公有云, 认为最好的用户体验就是公有云这样的全托管服务——开发者不用再自己维护。毕竟, 维护是有一定门槛的, 分布式系统很复杂。

“我们就那样推向市场了, 但实际上, 市场接受一家新的存储企业、建立信任是需要一个周期的。毕竟, 在大家眼里我们就是一个新品牌, 一个“黑盒”, 要把数据存在里面, 是要承担一定风险的。所以, 我们需要一步步的市场策略去建立与客户的沟通和信任。”

那么, 有没有机会加速这个过程呢? Juicedata

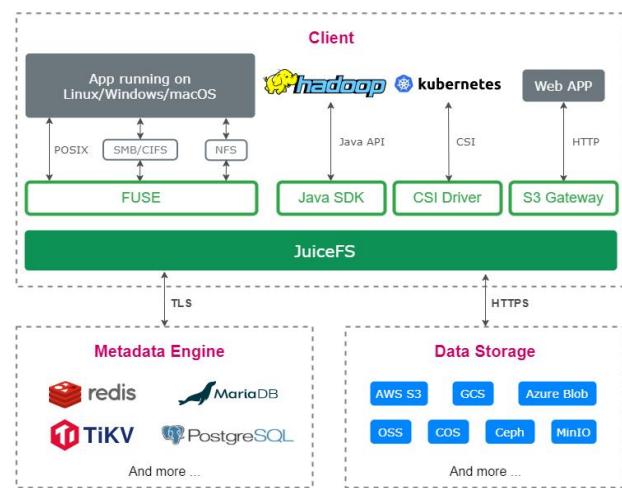
找到了开源这个办法。

首先, 开源意味着代码向大家敞开, 通过代码来建立信任, 让第一批掌握代码的开发者先熟悉和理解起来;

其次, Juicedata 在公有云上也创造了新的架构设计, 而对于众多开发者来说, 他必须要有自己的公有云账号, 才能感受到这个新的体验, 那么开源版本则可以降低这一门槛;

最后, Juicedata 自身关注的业务场景和领域是有限的, 文件系统却有非常多用户场景和应用行业, 开放的方式能将这些可能性交给社区去创造, 让社区用户大开脑洞。

2021 年 1 月 11 日, JuiceFS 开源了, 推出的开源版是一个建立在 Redis 和 S3 等对象存储 (比如 AWS S3、阿里云 OSS) 之上的开源 POSIX 文件系统。



在 Juicedata 的设计中, Metadata Engine 和 Data Storage 里面很多组件可以相互替换

它是为云原生环境设计的。通过把元数据和数据分别持久化到 Redis 和对象存储中, 它就相当于一个无状态的中间件, 帮助各种应用通过标准的文件系统接口来共享数据。

值得注意的是, Juicedata 将元数据引擎的相关代码进行了“可插拔”改造, 引入了对关系型数据库和事务型 KV 存储的支持, 解决了可靠性和可扩

扩展性问题，用开放式的架构去支持社区里各种各样开源的存储引擎，让他们可以结合到 JuiceFS 上来用。

这样一来，社区版本的用户可以结合自己的场景去选择合适的引擎；同时，用户在其他存储引擎（比如 Redis、MySQL 等）上积累的经验可以使他很快上手，大大降低了用户的使用门槛。

目前，JuiceFS 的开源引擎更多主打海量文件规模以及极致性能这样的场景。这个场景不是所有企业都有需求，多数企业从社区版就能满足自己的大部分业务需求，而一些特定业务则是需要企业版本支持。

至此，JuiceFS 的社区版本和企业版本形成了一个较好的相互补充关系。

在开源的第一年里，JuiceFS 收获颇丰，有超过 40 位贡献者完成了超过 800 次 Pull Requests，有过 4400 多位来自全球的开发者给 JuiceFS 点了赞。这样的发展速度，是苏锐没有想到的。

“（JuiceFS）在开源方面的发展速度，超出了我们的预期。文件存储领域离应用层有一定距离，又没有数据库那么被大众开发者所熟知。因此，我们假设这么底层的项目，并不会得到大家的广泛关注。

“另外，我们也评估了市场上其他的存储开源项目。比如零几年就已经开源的 GlusterFS 或者是

Ceph，他们已经做得很好了，却也没有像别的社区那样活跃。我想，JuiceFS 开源的超预期发展也印证了，大家对云上文件系统有强烈需求，只是缺乏一个简单上手的产品而已。”

目前，JuiceFS 累计在 GitHub 上获得了 5K Star 和 441 Fork，并且拥有 53 个 contributors。据苏锐介绍，其中只有 20% 是 Juicedata 的全职员工，其余则均为社区贡献者。而 JuiceFS 所对标的 Ceph，在 GitHub 上的表现则为 10.3K Star、4.8K Fork 和 174 个 contributors。

2022 年 1 月，Juicedata 团队决定自 JuiceFS v1.0 起将许可更改为 Apache 2.0。而在此之前，他们一直采用 GPLv3 许可证。

“刚开源的时候，我们认为需要选择一个能够保护我们的许可证。因为我们看到这几年，一些知名的开源项目和云厂之间有一些纠纷。如果我们选择 GPLv3，那么厂商就不能拿你的项目去做托管服务。”苏锐表示。此外，文件存储界使用最广泛也是 GPLv3。

但随着 JuiceFS 的不断迭代，有一些开源社区和开发者希望将 JuiceFS 作为存储底座，整合到自己的项目中，但 GPLv3 与其他开源协议（比如 Apache 协议）的兼容性却不太好。为了建立更加丰

The screenshot shows the GitHub repository page for 'juicedata/juicefs'. The main navigation bar includes 'Code' (selected), 'Issues 55', 'Pull requests 9', 'Discussions', 'Actions', 'Wiki', 'Security', and 'Insights'. Below the navigation, there's a summary of branches ('main', '58 branches', '29 tags') and a 'Go to file' button. A list of recent commits is displayed, starting with a commit from 'davies' upgrading tikv to v2.0.0 (#1643). The commit message is 'Workflow: canonicalize Hadoop Java SDK name (#1640)'. Other commits include 'upgrade tikv to v2.0.0 (#1643)', 'update image in gateway deploy yaml (#1424)', 'Docs: minor update about how to get access log in K8s (#1639)', 'speedup listing on file (#622)', 'upgrade golang-cross version to v1.17.8 (#1539)', 'refactor intergration test and add webdav protocol test (#1513)', and 'upgrade tikv to v2.0.0 (#1643)'. On the right side, there's an 'About' section with a brief description: 'JuiceFS is a distributed POSIX file system built on top of Redis and S3.' It also lists tags like 'redis', 'golang', 'distributed-systems', 'storage', 'filesystem', 's3', 'posix', 'hdfs', 'cloud-native', and 'object-storage'. Below the description are links to 'Readme', 'Apache-2.0 License', 'Code of conduct', '5k stars', '84 watching', and '441 forks'.

(以上数据截至 2022 年 3 月 24 日)

富的生态，Juicedata 最终转向了 Apache 2.0 许可证。

“其实，这样我们就承担了更多商业上的风险，但我们也由此获得了更丰富的生态，这是一个取舍的过程。”当被问及如何看待云与开源之间的关系时，苏锐则表示：

“从 2018 年开始，我们上线了全球所有的公有云，帮助云厂商去服务好他们的客户，去提供更好的体验。另外，我们也是公有云的客户，我们构建这些云服务也是通过购买公有云的资源而构建的。

“新事物产生过程中，调整和博弈是在所难免的。相比于整个公有云上的生意来说，两者之间的利益冲突其实是很小的。两者之间是个共荣共生的关系，互相合作能带来新的价值。

“其实，云为开源产品提供了更好的商业化方式。”

03 开源创企绝不是夹缝求生

的确，就在这个云时代中，不断有开源企业创业成功、越做越大。国外的 Elastic、Confluent、MongoDB、HashiCorp，包括 Juicedata 创始人刘洪清的老东家 Databricks 都是活生生的例子。这些公司在十年前左右涌现出来，从开源到盈利，最后演变成为估值数十亿甚至数百亿美元的独角兽。

反观国内，这一进程可能要更加滞后一些。当国外开源企业走向 IPO 时，我国开源创企更多集中在 A、B 轮融资阶段。《2021 中国开源开发者报告》显示，2021 年，国内开源融投资金额创历史新高，达到了 52 亿元人民币。

尽管资本火热，但似乎有个更重要的问题摆在眼前：因为 SaaS 环境不成熟，且处于后发劣势等原因，国内开源创企要如何实现可持续性地商业化变现，并且让公司持续盈利呢？以开源 + 云的战略切入云存储赛道的 Juicedata 要怎么走呢？

在苏锐看来，Juicedata 一开始就是立足国际市场的，并且，国内 SaaS 市场其实也正在快速成长，与国际相比，差别并不大，只不过不是同步成熟而已。

因此，国际上开源创企已经走通的路，国内创企完全可以借鉴。

而且，国内正处在全面数字化的阶段。公有云、私有云、混合云这些云服务的市场容量正在越扩越大，加上 SaaS 这种降本增效的服务形式，给了开源创企很大的生存空间。Juicedata 只需做出最好用的存储产品，为全球市场提供同一套标准的服务，绝不是夹缝求生。

此外，在技术赛道方面，尽管云存储领域是竞争对手林立，尤其是一些同时提供云服务的大厂。但 Juicedata 对此并不担忧：

“首先，Juicedata 在最初场景和产品关键能力的选择上，就倾向于选择那些云厂商还没有覆盖的部分，避开那些大众常规的场景，打差异化。

“而且，做产品最核心的是要贴近客户需求。开源创企在产品研发和资源投入上，比大厂要更聚焦。我们做这件事情，是用这个领域最资深的工程师、架构师，集合大家十几年的经验，用一种 All in 的创业心态来做这么一个产品。

“而在大厂，某个产品未必是公司里最重要的产品线，大家的投入心态是完全不一样的。开源创企反而更具优势。”

Juice FS Gitee 地址：<https://gitee.com/juicedata/JuiceFS>

Juice FS GitHub 地址：<https://github.com/juicedata/juicefs>

Juicedata 官网地址：<https://www.juicefs.com/>

开源知多D

自由软件之父 RMS 曾使用龙芯笔记本作为主力设备，但后面不幸被偷，现在换成了 ThinkPad。

姜宁，带程序员前往开源“乌托邦”

在各种会议或者视频中做自我介绍时，姜宁常常用这张照片——穿着蓝白黑的格子衫，背着双肩包，脸上带着笑，两手直直地放下，任谁看一眼都能猜到，他是程序员。

他确实做了十几年的程序员，但有一点特别的是，他为 Apache 软件基金会（ASF）的开源项目写代码。一开始，开源于他而言，不过是一份挣钱的职业。被推着走了很久之后，有一天终于意识到，在工作之外，自己要担负起开源布道的责任。

一开始只是组织贡献者线下见面，后来发起了 Apache 北京本地社区，协助成立 Apache 深圳本地社区，将 ASF 项目开发者，与社区开源爱好者聚在一起。这两年，他还开起了播客，以嘉宾访谈的形式，宣传开源相关的知识和文化。

姜宁将“Open Source Community”翻译成“开源共同体”，称之为程序员的乌托邦，是除了 996，程序员可以拥有的另一个选择。“在这里，程序员们通过邮件，代码评审一起交流编程心得，一同协作开发足以改变世界的开源项目，在这里，菜鸟可以得到高手的帮助，快速成长成为业界大牛。”

今年 3 月初，姜宁当选 Apache 软件基金会（ASF）2022 年度董事。他在董事竞选宣言中说，“希望能够帮助 ASF 打破地域、文化、语言的障碍”。

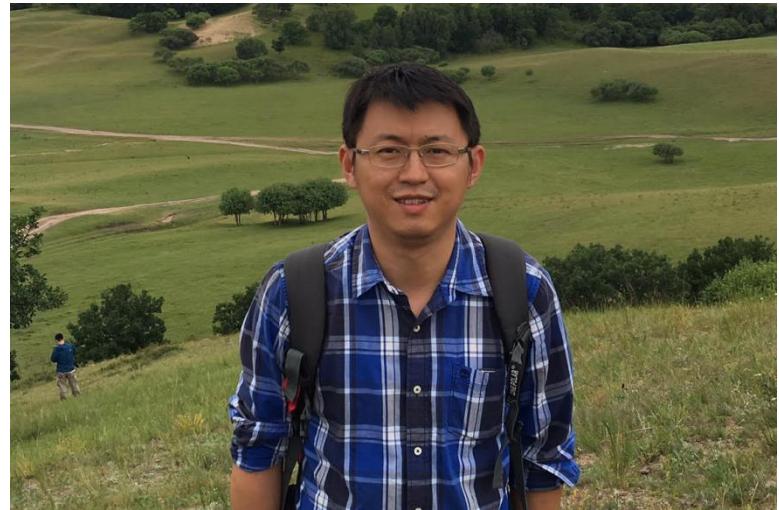
在开源布道的路上，他已然看到了横亘在面前的难以逾越的大山。纵使如此，姜宁也要踏过去。

01 成为开源布道师

对姜宁来说，四年前的“抄袭”风波仍然在敲响着警钟。

2017 年 6 月，华为开源了微服务框架 ServiceComb，托管在 GitHub 上。及至年底，该框架的 GO 语言版本 go-chassis 也开源了。

不料十几天后，go-micro 的作者 Asim 前来讨说法，因为 go-chassis 使用了其部分代码，但没有



按照开源许可证的要求作出声明。经过沟通之后，ServiceComb 团队立即修改了代码，并得到了原作者的认可。事情到这里似乎已经解决。

但传到国内后，一切都变了味，舆论把它演变成了华为抄袭。

姜宁赶紧在知乎上解释，但并没有多少人关心事情的详细经过，以及他们做了什么，大部分人都盯着他们没做什么，以此发泄情绪，“故意”、“抄袭”、“狡辩”，即便有人发表中立观点，也被指是在为华为洗地。百口莫辩。之后，他没有再追上去解释。

现在回过头来看，姜宁正处在职业生涯转型的关键时期。2017年1月，做了十几年开源程序员的他，来到了华为担任开放能力中心的技术专家，成为了一名开源布道师。“我来华为，就是要把我对开源世界的理解，把我做开源的经验，转化成一些基本实践原则，来帮助项目成长。”没成想，却在一年之后发生了 go-chassis 事件。

从事开源开发十多年，他也是第一次遇到这样的事。教训很深刻，让他看到了很多，也学到了很多，深感开源布道重要性。“以前做开源，是跟着大牛学习，只需要遵守社区的规则，按部就班参与进去就可以。但现在，要带着没有开源协作经验的小朋友们一起开源，需要考虑的事情更多了。”

姜宁曾向项目团队成员交代过，要规范使用代码，但很多开发者不重视开源代码的版权，缺乏合规意识，只想快速实现功能。“以为开源的代码，想怎么用就怎么用，想怎么改就怎么改。这种想法是有问题的，最终会付出沉重代价。”姜宁评判说。

事实上，在项目正式开源之前，所有自研代码都会用工具进行查重审查，以确认代码是否符合规范，但工具不能识别所有的问题。

他认为，唯有提升程序员的版权意识，避免片段化引用第三方代码才能彻底解决这一问题。“尽量以库的方式来引用第三方开源代码。否则，很难说清楚自研代码和第三方开源代码的关系，操作稍有不慎，就很容易被扣上抄袭的帽子。如果要修改第三方开源代码，要把需求或者补丁提交到上游代码库。这样即减轻了我们维护第三方代码的工作量，也进一步完善了第三方开源代码的使用生态。”

他也意识到，开源之后，所有问题都是敞开的，如果涉及大厂，问题会被放大，影响更恶劣。“版权意识，是每个开源人都要具备的。”

这场风波直接让姜宁的职业生涯陷入了危机，但无暇顾及太多，他还有很多事要忙。就在两个月前，ServiceComb 项目捐赠给了 ASF，进入了孵化阶段。

姜宁意识到，要想让项目从 ASF 顺利毕业，除了合规发版，还需要构建起健康发展的共同体，发展



ALC 北京成员聚会（2020 年 8 月，北京）

外部贡献者。而要发展外部贡献者，就要扩大项目影响力。影响力又是靠用户不断积累起来的，所以要发展更多的用户。

他采取的方式很原始，主动跟对项目感兴趣的人聊天，聊业务背景，聊实现细节，聊如何解决大家的痛点问题。起初，姜宁的内心有些抗拒，因为他以前是一个很少抛头露面的技术宅。但随着时间的推移，转做社区工作让他结识了很多志同道合的小伙伴。

这个方法颇有效果，Apache ServiceComb 逐步吸引了一些外部贡献者，不到一年时间就从 ASF 孵化器顺利毕业，成为顶级项目。

也正是在完整运作 Apache ServiceComb 项目之后，姜宁才对孵化流程中的 License 合规有了系统化的、全面的理解。之后，姜宁对处理项目发展问题更加得心应手。

2021 年，Weex 项目的社区成员不再活跃。没有人提交社区报告，甚至连项目发版时都没有足够的 PMC 成员来进行投票。最终，姜宁结束了 Weex 在 Apache 的孵化，这是第一个在他手上退休的项目。他说感到很遗憾。

姜宁也希望 Weex 能够善始善终。在此之前，已经有多个导师退出了该项目，他是坚持到最后的人。姜宁比以前更从容了，也看得开：“孵化项目失败，我比其他导师经历的事情，也就更多一些。”

随着身份的转变，姜宁对开源的认识也不一样

了。现在的他认为，开源不单单是技术领域的问题。在把开源的成功经验复制到企业内部的过程中，要解释清楚开源的运作原理，还涉及了很多社会学、经济学的知识，它有很强的人文属性。“从这个角度来说，我又找到一个新的，可以为之奋斗十年的一个方向。”

02 程序员的乌托邦

在孵化 Apache ServiceComb 的过程中，姜宁意识到，原来通过布道可以影响更多的人。于是在 2018 年及 2019 年，姜宁先后组织了两次国内 ASF 贡献者线下聚会，之后又发起了 Apache 北京本地社区，协助成立 Apache 深圳本地社区，将开源爱好者汇聚在一起，做一些有意思的事。

渐渐地，姜宁认识了越来越多的程序员，特殊的角色让他成为了一个观察者，看到了很多困在程序员身上的枷锁。开源布道的责任，就这样在他心里慢慢生长起来。他想，要用自己十几年的开源经历去告诉大家，还可以有另一种选择。

“开源共同体是程序员的乌托邦。”姜宁如此认为。他将“Open Source Community”翻译成开源共同体，而不是开源社区。

“‘社区’一词，容易和居委会，我们居住的社区混在一起。”姜宁解释说，“Open Source Community 既有生产者（开发人员），也有消费者（使用者），大家为了共同的目标（把开源软件打磨得更好）聚在一起，相互协作共同成长。而且在开源社区，会有很多大神级程序员扮演导师的角色，无条件地把经验传授给年轻的程序员，帮助他们找到技术拓展的空间。”他自己就是从开源世界成长起来的，最直接的收获就是技术能力的提升。

他反复提到一个词，办公室政治。“在开源社区，程序员只要专心研究技术，而不需要考虑办公室政治。”发现问题后，社区成员会主动 debug，不会出现推诿扯皮的情况，而且大家都是通过邮件列表公开交流，目的也都是为了解决同一个问题，所以工作效率很高。

他希望这些企业在开会这件事情上，能向 ASF

的董事会学习一下，会前大家通过异步的方式处理相关的报告，每个月开一次，一次不到一个小时。“把开会的时间省下来，大家会更纯粹一点，把自己手上的事情做好，做精，做尖。长期积累精进之后，效率可能提升十倍，百倍。”

姜宁想，如果大家都具备开源协作的能力，工作效率会提高很多，996 这种事情也会少很多，程序员的生存状态也能变得更好，会有更多的时间去创新，做一些更有意思的事情。

对程序员来说，开源还有一个好处就是，可以长期专注于一个领域，从而沉淀自己的技术能力。姜宁认为这种长期积累是很有必要的。

他提到了具有 15 年历史的 Apache Camel，从诞生之日起，很多基础架构一直都没变，因为其创始人 James Strachan 把架子搭得很好。他还补充道，这种能力很难靠走捷径把它学过来，而是要长时间的编程经验的积累。

但国内程序员没有多少可供长期从事的项目。“我们的项目生命周期可能就是三年，过了三年就会被推倒，换一波人重新写一遍。这么一来，我们只能做一些低水平的重复劳动，不可能会有积累。”这样的情况，姜宁见的不少。而且，大部分代码服务于应用层面，只关注业务实现，没有人关心这些代码是否写得精巧，是否易于扩展，是否易于维护。

当然，开源项目也会有推倒重来的时候，但是他认为二者是不同的，这是基于之前积累的经验、知识进行的重构，而不是低水平的重复劳动。

03 难以逾越的大山

布道，没有开发那么容易。从进入华为那天起他就知道。

以前，姜宁跟机器打交道，只要利用好网上资源，沉下心来就能学到技术获得成长。但现在不一样，这个新角色，要求他具备更多的软技能，比如演讲能力、表达能力、说服能力，还要去思考如何影响更多人。华为需要懂开源社区的人，而不是参与过某个具体项目的人。

他都走过来了。就连谈起身份的转变，姜宁现在都能笑呵呵说：“哎呀，其实是被逼无奈。”

但面前还有一座难以跨过去的大山。

“观念的转变是最难的。”姜宁说，“很多人只是把开源当成免费的、能让他快速交活的工具。”

“开源最大的魅力是可以跟开发者交流，他们会给你带来很多惊喜。你会发现，开源项目居然还能这样用。”姜宁认为，参与开源要有交朋友的心态。做开源项目维护者期间，他最享受的事情就是回邮件。在帮助他人的同时，也加深了他对问题的理解，就像在玩打怪升级的游戏一样。人与人之间互动产生的思想火花的碰撞，会让人成长更快。

在开发 Apache Camel 项目时，只要与他互动的人在北京，姜宁就会想办法跟对方见面。不过最终见面的只有两三个人。他觉得有点奇怪，明明有很多人下载软件，却看不到他们。

他鼓励更多人跟上游交流互动，这是一个深度参与开源的机会。“要勇敢迈出第一步，只要参与进来，会有很多热心的人帮助你。”

对姜宁而言，在进行 ASF 项目孵化的时候会比较轻松，因为大部分开发者对开源运作流程有所了解，只要稍微引导就可以。

但要在公司内部推广开源，姜宁说，太难了。

技术上实施起来并不难，难的是如何让大家主动地参与开源。姜宁看到，虽然代码已经放在内部公共仓，公司也鼓励大家把技术用起来，但还是有很多人认为，自己知道代码的执行细节就行了，不需要告诉别人开发相关的背景和知识。“他没有体会到，公开与代码相关的上下文信息，沟通成本会更低，协作会更容易。”

姜宁说，大量的开发者都没有开源经历，难以理解开源协作是一种怎样的方式。没有实际的体验，大部分人都是机械地按照公司要求来做开源，领导关注事情就能落地，领导不关注的，往往做不到位。如果决策者或者推动者也没有经历过开源，就连沟通开源这件事本身，都可能存在很大阻力。

在很多大公司，跨部门协作也比较难。虽然每个



Apache Meetup (2018 年 10 月 上海)

人能力都很强，但各部门都有自己的地盘，人与人之间，就形成了一堵无形的墙。“我不跨过去，你也不要踏进来。得用各种各样的手段才能让各部门协同。”这让姜宁颇为苦恼，因为在开源社区，加入进来的人都有共同目标，所以在做一件事的时候，大家都是尽量相互配合，给予鼓励，而不是设置障碍。

不管是社区开源，还是公司内部开源，遇到的这些难题，归根结底还是观念问题，大家缺乏开放式协作的意识。

“只能一步一步来。”姜宁说。他能想到的解决之道，就是让更多人参与到开源中来。

“这个过程可能要花很长时间。”他补充说，最好的方式就是，大家直接参与开源项目，边看边学，成长会很快。

04 开源滋养了他

在经历过开源的暗面之后，姜宁仍然能看到开源美好的一面，坚称“开源共同体是程序员的乌托邦”。作为程序员，他真真切切获得过开源的滋养，经历的开源是美好的、轻松的、自由的。

那是 2005 年，姜宁离开国企，进入海纳 (IONA) 亚太研发中心，参与开发 CXF 项目。一年之后，公司把 CXF 捐赠给了 ASF。作为 Apache CXF 的初始 committer，姜宁很轻松地加入了 ASF。

毫无波澜地，他的开源之旅就这样开始了。“没有人专门给我做培训，就把我扔到社区里面。别人怎么做，我们照着那个样子做，很多时候就是边看边学。我们从公司的 nobody 变成了公司的 somebody。”

在大部分人的眼中，开源都是与“志愿”联系在一起，意味着要占用大量业余时间，且没有收入。而姜宁早期经历的开源却不大一样，他是拿着工资做开源，跟正常上班什么区别。

姜宁说，自己是幸运的，没有经历过做开源还要考虑生计这样的事。他曾在播客中提到：“过去十到十五年，我国开源充满了悲壮的个人英雄主义，大部分开源项目都不赚钱，主要靠情怀。然而光有情怀，很难持续下去。”

真正让他有成就感的，却是另一个项目——Apache Camel。当时，Camel 想集成 CXF，作为 CXF 核心开发者的姜宁，正好也想多参与一些 ASF 项目的开发，于是就开始接触 Camel。

为了成为 Camel 的 Committer，他老老实实地按照 CXF 中的“Getting Involved”提示，一边熟悉 ASF 的工作方式，一边不断用提交补丁的方式“骚扰”Camel 项目中的其他 Committer，花了半年时间，最终为自己挣足了信用，在 2008 年初成为了 Committer。姜宁说，这让他“赢得了大家的认可，再次体验了一下成为 Apache committer 的快乐”。

在此期间，姜宁结识了 Apache Camel 的创始人 James Strachan。几年后回过头来看，那时国内开源环境并未形成，没有什么可借鉴的开源经验，一切都要靠自己摸索，而 James Strachan 称得上是姜宁的开源引路人。

“他真正地扩展了我的视野，带领我去追求新的前沿技术。”多年后，姜宁还在各种公开场合多次提及这位引路人。

也正是在 2008 年，IONA 被 Progress Software 公司收购，北京办公室解散，20 多个员工都找工作去了，姜宁和另一个开发同事作为“善后”的留守人员，继续维护这两个项目。姜宁记得很清楚：“我们当时有 12 个开发人员，分成了两拨。我们两人留



下，另外十个人，大部分都去了红帽公司。”

没了办公室，姜宁就在家上班，做的还是那些事儿，只不过，给他发工资的，变成了 Progress Software。两年后，他所在部门从 Process Software 独立，成立了开源创业公司 FuseSource，几年后，FuseSource 被红帽收购，姜宁也就顺势成为了红帽员工，和 IONA 的前同事汇合了。

Apache Camel 这个项目，见证了姜宁完整地成长：从 committer 到项目 maintainer，再到 ASF member。在一则媒体采访的视频中，他回想起收到 ASF 的 member 邀请邮件，填完信息注册表的那一刻，不禁笑了起来：“我好高兴。这相当于 Apache 软件基金会对我的认同。成为 member 之后意味着一件事，就是我可以 mentor 项目。”那是 2011 年 1 月，距离他加入 Camel 开发已近 4 年时间。

2015 年，姜宁从红帽离职，结束了七年专职开源，在家办公的生活。他现在经常怀念那时候——既能自主选择工作时间和内容，又可以照顾家庭。

“我不能保证八小时全情投入，但有五六个小时，肯定是专注的。”上午八点开始工作，中午休息两个小时，之后接着工作，一直到下午四点半，这时候姜宁就要去幼儿园接孩子，然后做晚饭。

占用了一个半小时的工作时间，姜宁就在晚上补回来，“晚上八点到十点，在线 2 小时”。尽管白天工作五六个小时，已经基本把重要的事情都处理完了。

唯一不足的是，工作和生活分不开。邮件驱动他加班。只要用户邮件一来，姜宁就赶紧回复，已经形成了习惯。然而邮件一直来，一直来，妻子说他，“永远有干不完的活”。

十多年前，姜宁的手机还不能收发电子邮件，直到诺基亚 720 手机出来。姜宁记忆深刻：“全键盘的手机，当时就觉得，哇塞，这个手机好好，终于可以随时随地回邮件了，爽很多。”手机到手后，陪妻子逛街时，他就找个角落，隔几秒就刷一遍收件箱，盯着邮件列表上那些问题，抓自己知道的，随时回复。

有时也会有些小小的孤独。他专注工作，对外的连接并不多，跟别人的语言交流很少，更多的是写邮件，而且很多时候是跟老外交流。“在家时间长了，懂我的人比较少。当时在微博上认识了一票人，周末活动的时候见面，特别开心，可以说好多话。”

05 履新之后

今年 3 月初，Apache 软件基金会（ASF）公布了新的董事会成员名单，姜宁名列其中。他说自己曾经是一个佛系的技术宅，然而 ASF 董事这个新身份，却是他主动争取的。20 个候选人竞争上岗，最终只有 9 个人脱颖而出。

这次竞选董事的宣言，姜宁花了很长时间去思考，到底要讲哪些东西。最后出来的稿件，没有什么夸大的言辞，只是把他做过的几件事，以及为什么要做这些事讲了出来。“让大家去相信，我就有这样的能力参与基金会治理。”

唯一一句口号式的句子，是他说，“希望能够帮助 ASF 打破地域、文化、语言的障碍。”多年 ASF 开源项目孵化的经历，给了他说出这句话的底气。这些年，他先后帮助孵化了近 10 个来自中国的 Apache 项目。他深知语言和文化隔阂带来的沟通问题。

成为 ASF 董事之后，姜宁肩上的担子又更重了一些。工作和开源并不能严格分开，偶尔，姜宁也会在工作时间处理一些和基金会相关的事务，但华为没有干涉，反而大力支持。



ALC Beijing 的播客节目，姜宁与 tison 对话

他也清楚地知道自己的短板：“站在董事的角度，要推动 ASF 发展壮大，不仅需要国际化的视野，同时还要立足本土，把国内程序员带动起来。我以前的项目孵化经验，只能起到 40% 的作用，剩下 60% 的那部分，要自己去提升。”

问其为何要这么做，他说：“开源是一种很好的协作方式，能让世界各地的开发者协同起来，跨越公司，跨越国界，去做一些对整个人类都有意义的事。”

在开源这条路上，姜宁要做的事情还有很多，可以预见的是，要面对的难题也有很多。但没关系，当他带着程序员前往乌托邦时，就会有很多和他一样手持炬火的人，照亮前进的路。

开源知多口

Linux 的吉祥物 Tux 之所以是一只企鹅，据 Linus 回忆是因为他曾经被一只愤怒的企鹅咬伤。



Go+



Go+ 是目前第一个顺应“三位一体”发展潮流的编程语言，同时兼顾工程、STEM 教育、数据科学三大领域。

推荐官：许式伟，七牛云创始人兼 CEO、ECUG 社区发起人、Go+ 项目作者。

推荐理由：

Go+ = Go（代表了工程能力最强的语言） + Python（代表了当前低门槛化最成功的语言）。而三位一体，本质上代表的就是工程与低门槛化的融合。

来看看 Go+ 历史上的一些关键节点。Go+v1.0.x 奠定了以命令行风格为基础，极尽低门槛化的代码风格；以及实现类文件 Beta 版，面向对象、领域知识表达的低门槛化。兼容 Go 语法也取得突破性进展，大部分 Go 语法都实现了兼容。Go+ 选择命令行风格的原因在于，想将理解难度降至最低，达到小学生都可以理解的层面。

Go+ v1.1.x 是 Go+ 的第一个工程化版本，可用于生产环境。实现了对 Module 的完备支持；实现了 Go/Go+ 混合工程；提供了 c2go 预览版。

Go+ v1.0.x 是明目标，定风格；Go+ v1.1.x 则是为了进生产环境。无论是 Module 支持，还是 Go/Go+ 混合编程，都为进入生产环境打下扎实的基础。

按照规划，Go+ v1.2.0 将于今年底发布，从而形成 Go+ 特色化。包括类文件转正，结束 Beta 测试；c2go 工程化，进入实用阶段，至少完成 sqlite3 的迁移；以及 Go/Go+ 混合编程增强，支持 Go+ 调用 Go 泛型。未来，Go+ 将在 v1.7.x 实现 Go+ 数据科学技术栈形成；在 v2.0.x 完成支持 Python 语法，可 import Python 包。

总的来说，Go+ 今年内会在工程化与低门槛融合上告一段落；明年起，将会对数据科学发起最后的攻坚战。

<https://www.oschina.net/p/goplus>



Skyline

Skyline 是一款实验性模拟器，可将 Android 设备模拟为 Switch，并运行各种 Switch 自制软件和游戏，支持在 ARMv8 Android 设备上运行。

推荐官：OSC 编辑部

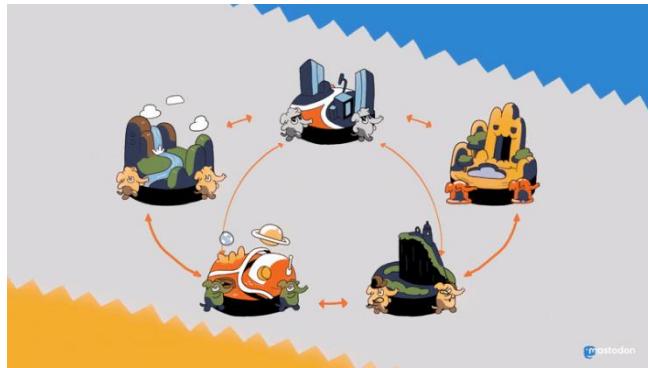
推荐理由：

Skyline 拥有速度快，高兼容性等优点，它支持运行绝大多数 Switch 游戏、支持模拟陀螺仪、倾斜、太阳能传感器，同时还支持作弊码，输入 GameShark/ActionReply/CodeBreaker，即可在游戏运行时即时启用 / 禁用作弊器。

Skyline 对设备和运行环境的要求并不高，它本身是高级 BIOS 模拟，因此无需额外的 BIOS 文件。Skyline 提供 IPS/UPS ROM 补丁，且使用 OpenGL 渲染后端，可以在没有 GPU 的设备上正常渲染。

此外需要注意的是：Skyline 模拟器需要 Android 8.0 及更高版本、四核 2.0 Ghz+ 的 CPU、6 GB 起的内存（推荐 8 GB），以及至少 30 MB 的磁盘空间。如果要玩大型游戏，建议 CPU 的性能不低于高通骁龙 865（玩家实测性能低于 865 的 CPU 可能出现图像无法显示的问题）。

<https://www.oschina.net/p/skyline-emu>



Mastodon



Mastodon 是开源的社交网络服务器，任何人都可以基于它创建类似推特 / 微博的产品。

推荐官：OSC 编辑部

推荐理由：

4月25日，马斯克斥巨资收购 Twitter，并宣称要开源其算法——这份开源承诺无疑值得期待，不过在包罗万象的开源世界里，早已诞生 Twitter 的开源替代方案，Mastodon 便是其中之一。

但 Mastodon 与常见的社交网络不一样，其定位是去中心化的社交平台——可作为商业平台的替代方案，避免了单一公司垄断沟通的风险。

基于 Mastodon 搭建的站点被称为“实例(instance)”，这些“实例”既相互独立，又彼此关联（因为不同站点的用户可以毫无阻碍地交流）。用户可以选择自己信任的实例，无论选择的是哪个，都可以与其他人进行互动。任何人都可以运行自己的 Mastodon 实例，并无缝地参与到社交网络中。

作为一款开源软件，Mastodon 与生俱来的自由气质吸引了众多开发者之外的群体，也让更多人认识了拥有不同兴趣爱好、来自于世界各地的新朋友。

<https://www.oschina.net/p/mastodon>



Bitwarden

Bitwarden 是一个免费开源的密码管理服务，它将敏感信息，如网站凭证存储在一个加密的保险库中。Bitwarden 提供各种客户端应用程序，包括桌面应用程序、浏览器扩展、移动应用程序、Web 界面和命令行界面。

推荐官：OSC 编辑部

推荐理由：

数字时代，人人都需要一个密码管理器。互联网用户每天都需要登录许多网站，数字和字母组合、长度、大写字母或特殊字符等严格的密码要求使人们很难记住所有密码，同一套账号密码到处使用也容易因为撞库导致所有信息泄漏。

如果你对日常生活中创建、使用和记住这些错综复杂的密码感到不知所措，那么密码管理器就是在能够强化用户密码的同时还能方便用户使用的工具。

Bitwarden 支持全平台，这意味着无论你使用/更换什么设备，都能找到对应的客户端软件，确保宝贵的数字资产能在各个设备中保持同步且不会遗失。

Bitwarden 采用了端到端的 AES-256 加密，为用户提供了难以被外人所破解的加密；并且 Bitwarden 具有自托管部署选项，用户可以完全掌握自己手中的数据。反观 Bitwarden 的竞争对手，LastPass 和 1Password 都采用了各自的账号体系，用户需要将数据传到他们的服务器并进行同步，提升了用户资料被黑的风险等级。此外，使用 Bitwarden 的用户除了存储常用的账号密码，还可以存储其他类型的敏感信息，如软件激活码、护照、驾照和银行卡信息等，以备不时之需。

Bitwarden 在近日的更新中还支持了 SimpleLogin、AnonAddy 和 Firefox Relay 来生成用户名，利用电子邮件转发服务可以更好地掩盖真实电子邮件地址，以防止它在数据泄露事件中暴露。

<https://www.oschina.net/p/bitwarden>



PDManer

PDManer 元数建模是一款开源数据库建模软件，基于 Electron+React 技术体系构建，支持多操作系统以及多种数据库，开箱即用。

推荐官：杨松，PDManer 元数建模开源软件项目组织者及发起人，开源生态贡献者，理想主义者，热衷于把产品设计得简单好用好看。同时也是一位“伪”全栈开发者，中度强迫症，对代码观赏性有着极高要求，产品设计精细到 1px。

推荐理由：

PDManer 构建了自己的 UI 组件体系以及前端交互体系，其核心功能是满足数据库建模的关键要求，如表设计、数据表关系 ER 图设计、自由抽象的 ER 图以及逻辑图设计，还能将设计成果导出为数据库

DDL 以及 WORD 文档，实现成果交付及转化。

PDManer 以核心功能为基础做到了生态兼容，例如从数据库中逆向解析数据表，通过 DDL 文件导入，PowerDesigner 文件导入等方式，实现对存量元数据的转换导入。此外，PDManer 还打通了对多种数据库的扩展支撑。加入了动态可扩展的模板引擎技术，除了数据库 DDL 之外，也支持程序代码生成，在完成数据表设计之后，能够将数据表直接生成至各种不同语言的程序代码，例如：Java、C# 等。

PDManer 还支撑数据库版本管理，用户在修改数据表以及字段之后，能够很好地记录历史修改痕迹，并且将这种修改直接生成为数据库变更 DDL 语句。目前 PDManer 元数建模平台还在不断精进发展，希望更多的社区用户能够参与开源，共同促进开源事业持续、稳定、健康向上发展。

<https://www.oschina.net/p/pdmaner>



PaoPao CE

PaoPao 是一款基于 Gin+Zinc+Vue3+Typescript 等构建的高性能垂直社交软件，五脏俱全，产品的交互和 Twitter 相似，小清新文艺风格。

推荐官：ROC，PaoPao 作者，全栈开发，区块链技术爱好者，开源软件 ROCOBOSS 作者，百度大商业搜索架构团队资深研发工程师。

推荐理由：

PaoPao 后端主要由 Gin+Grom 构建，采用轻量级全文搜索引擎 Zinc，相比于 ElasticSearch，在海量数据检索的场景下，其整体内存资源开销更低。前端由 Vue3+Typescript 构建，使用 Naive UI 精致且丰富的组件，保障了高度一致的良好用户体验。PaoPao 还提供了由 Rust + Tauri 构建的桌面端。

功能上，PaoPao 提供明暗两套主题，可以根据系统主题色自动切换。用户可以直接通过“@”去提及目标用户，也可以通过“#”去使用或创建自定义



标签，摒弃传统分类的设计。得益于 Go 的高性能，即使是海量标签，PaoPao 也可以在 20ms 左右完成信息检索。除了对常规图片、视频的支持外，PaoPao 还支持附件上传、外链等，且可以设置附件价格，可以轻松帮助用户完成知识变现。

此外，PaoPao 提供了 Docker Compose 容器编排，方便一键部署，告别复杂的环境依赖，同时提供 embed 编译选项，可以将所有资源嵌入到一个可执行程序中，十分便于迁移、扩容。PaoPao 基于 MIT 协议开源，使用了当下较流行的技术组合，项目结构简单清晰，对于想学习全栈技术的朋友，相信 PaoPao 可以起到抛砖引玉的作用。

<https://www.oschina.net/p/paopao-ce>

工程师如何选择开源软件？

【编者按】本文节选自《[工程师如何对待开源](#)》，作者谭中意是星策开源社区发起人、中国开源推进联盟副秘书长，同时也是一名在知名科技企业从事开源相关工作超过 20 年的工程师，亲身经历或者亲眼目睹很多工程师在开源领域的优秀实践，同时也看到了很多 Bad Cases。《工程师如何对待开源》一文凝聚作者诸多心得和体会，除了本文所呈现的“工程师如何选择开源软件”的内容之外，还着重阐述了“工程师如何定制和维护开源软件”“工程师个人成长如何利用开源”两个话题。谭中意说：“希望能帮助工程师更好地成长。”

文/星策开源社区发起人、中国开源推进联盟副秘书长 谭中意

作为技术工程师，工作任务就是用技术手段支持和实现公司所关注的商业目标。在实际工作中，会主动或者被动地使用和维护大量开源软件。据统计，每个工程师在企业内部进行研发和运维等工作的时侯，每年会接触到上千款开源软件，如果是以 Java 或 JavaScript 为主要程序开发语言的工程师，则接触到的开源软件数量更多，在万级别甚至十万级别。（数据来源：《2020 State of the Software Supply Chain》由 Sonatype 发布）

那么如何选择开源软件？这么多开源软件中，如何根据个人需求和业务需要来选择合适的开源项目进行投入？

首先要明确对开源软件的态度，即在现阶段离不开对开源软件的使用。不过，使用开源软件会面临各种各样的风险，包括开源合规、安全、效率等问题。简化为一句：在企业内部高效、安全、合规地使用开源软件，需要遵守该企业对开源软件的内部规定，包括如何引入和如何维护。

回到具体如何选择特定的开源软件的问题上，有如下几个纬度可以进行参考。

- 根据需求
- 根据技术发展趋势
- 根据软件采纳周期的不同阶段
- 根据开源软件的成熟度情况
- 根据项目的质量指标

- 根据项目的治理模式

一、根据需求来选择开源软件

选择开源软件，首先要明确需求，即选择这个开源软件的目的究竟是什么，是用来进行个人学习的；还是用来满足 to B 客户的需求的；或是用来满足内部服务开发的需求的。因为不同的目的，选择开源软件的导向完全不一样。（注意：后两个场景需要先考虑企业开源合规的需求，参考各个企业内部开源合规要求。）

进行个人学习

先说说选择开源软件来进行个人学习，那么要看学习的具体目的究竟是什么。是想学习一种比较流行的技术来完善个人的技术知识结构扩大个人技术视野；还是想看看相应的开源技术项目的具体实现，来作为内部项目技术开发的参考；还是想为了下一份工作有针对性地进行技术准备。

针对前者，显然是什么技术最流行选什么，自己缺什么选什么；针对第二种目的，一般是对该技术领域的知名开源软件或创新性软件有针对性地进行选择，即某个特性是我当前需要的，或者是我当前项目实现不好的，我要看看别人是如何实现的。最后一种，显然是按照下一份工作的职位需要和技术栈要求进行准备，并根据技术栈要求的门槛高低进行选择。

但是注意，从个人需求出发选择开源软件，一般都需要写个小项目练练手，比如一个 Demo 程序或者一个测试服务，因为不用考虑后续的长期维护，所以

尽可以按照个人的想法和个人研发习惯进行各种练习，不用遵循企业内部的开发流程和质量要求，也不用考虑该开源软件的稳定性和社区成熟度等情况，只需要尽情地学习和参考代码就好了。

满足 to B 客户的需求

选择开源软件进行研发的软件是需要提供给客户的，往往可能是以私有云的方式进行交付。基于此类需求来选择开源软件，注意做好平衡，即要平衡好客户的需求和企业自身技术规划或产品的长期规划需要。

以私有云方式进入客户的 IDC 环境，是需要跟客户开发和运行环境的上下游项目进行集成的。这时候要看客户的需求，可能某些客户对开源软件有特定的要求，例如要求使用 HDFS 而且是某个特定版本。客户之所以会指定软件名字和指定版本，可能是因为其当前比较熟悉这个版本，也有可能是因为之前其他软硬件供应商提供的软件和版本，指定的目的是方便集成和后续的使用与维护。

如果这种需求是符合企业项目或者产品的长期发展需求的，则是可以完全满足的。如果甲方非常强势，除了满足他的要求之外没有别的办法，那就选择客户所指定的软件和版本好了。

但如果跟自身项目或产品的长期发展需求不一致，而且具体项目或者版本是可以跟甲方进行协商的，那么需要跟客户协商出一个双方都能接受的结果，即选择特定的开源软件和版本既要做到客户满意并买单，又要做到自身的交付成本可控，还要做到符合自身项目或者产品的长期发展需要。

例如客户使用 Java 的某个老版本，但是企业的 to B 交付的软件要求使用 Java 的较高版本。那么需要跟客户协商，要么切换到企业希望的版本上，还需要帮助客户完成已有系统的升级工作；要么只能降低自身软件的 Java 版本需求，可能还需要对某些自身代码进行修改，还可能对软件中的某些依赖组件进行修改。这个场景下是带有很多客观约束条件下的选择，是需要跟客户，自身的产品经理和架构师一起协商的。

满足内部服务需求

如果场景是为了满足内部服务的需求，即选择开

源软件来搭建的服务是给内部业务或者最终用户来使用的，常见于国内各大互联网公司的互联网服务系统和各种手机上的 App。这时候项目的开发和维护方有较大的自主权，跟 to B 的交付业务完全不一样。此时选择开源软件，就一定要综合考虑开发和维护成本，还要考虑使用该服务的业务所处的阶段。

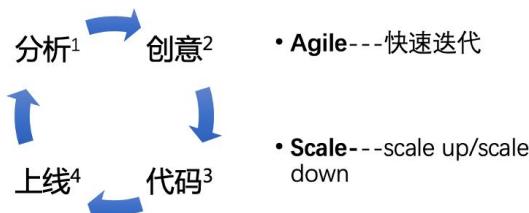
(1) 如果提供的服务是给创新业务使用的，创新业务一般都是试错业务，随时需要根据市场情况的变化和当前执行的状态进行调整，很可能三个月后这个项目没了，即被取消了。这种情况下“糙快猛”的开发方式是比较合适的，不用太多考虑系统的可维护性和可扩展性，就用研发团队最熟悉的软件技术栈，然后用底层技术支撑团队（比如基础架构团队）提供的成熟而且经过验证后的底层基础技术平台就可以，最重要是尽快把系统搭建出来，然后随着产品进行快速的迭代。

这个时候需要尽量降低现有研发运维团队的学习成本和开发成本，不用太多考虑可维护成本，因为需要糙快猛的把系统堆出来，验证产品需求和商业模式是最重要的，时间最重要。如果发现有市场机会，就快速跟进，站稳脚跟之后可以采用省时间但是费资源的方式（俗称“堆机器”）来进行扩展，或者采用“边开飞机边换引擎”的模式进行重写都是比较划算的。对于处于创业阶段的企业或者项目来说，速度胜过一切。

(2) 如果选择开源软件搭建出来的计算机软件系统或者服务需要长期维护，比如是给公司内成熟业务使用的，或者是针对公司内成熟平台的缺点进行系统升级并要替代原有产品的，那么在满足业务需求的前提下，考虑系统的可维护性就变成最重要的事情。

选择对应的开源软件，它是否成熟，是否稳定；二次开发是否友好；运维成本是否比较合算即比较省机器和带宽；运维操作是否方便，例如常见的扩容和缩容操作是否可以高效、自动、无损的完成；Upstream 到上游开源社区是否容易等等，这些都成为需要重点考虑的事情。这种情况下，开发一个系统的成本，可能只占整个系统生命周期内的成本的 1/10 不到。所以在满足需求的前提下，重点考虑可维护性。

二、根据技术发展趋势来选择开源软件



如上图所示，现代计算机软件或者服务的研发，是一个不断运行的循环和迭代过程。从市场分析开始，然后进入到创意阶段，再到编码阶段，最后到上线阶段完成应用的部署和生效，上线之后根据得到的数据反馈，继续进行分析。

这个循环迭代的过程，显然对于一个身处行业竞争激烈的企业来说，迭代的速度越快越好，同时也需要具备快速弹性、低成本伸缩的能力，即产品方向对了，那么赶紧进行系统扩容，承接快速增长的流量，做到快速增长；如果产品方向不对，需要赶紧缩容，把相关硬件和人力资源节省出来，投入到新的试错方向上去。身处同一个行业内的企业，如果企业 A 能以更低的成本，更快的速度进行各种产品和策略的迭代，显然它比迭代速度慢，成本高的企业 B 更具有竞争优势。

现在的开源软件数量非常多，几乎每一个分类下面都有很多的开源项目。针对某一个具体的需求，如何进行选择？一个建议是根据技术趋势进行选择。即现在的计算机系统迭代的方式是 Agile（敏捷） + Scale（扩展）。

显然，能够支持计算机系统进行快速迭代，并能够很方便进行低成本弹性伸缩的开源软件值得进行长期投入。而对一个新的开源软件的学习和使用，学习者是希望该软件的学习门槛越低越好。一个流行的开源软件，内部实现可以尽可能地复杂，但是对于用户来说一定是需要用户友好的。不然即使创新度再好，易用性不好，只有极客才能学习和掌握，创新的鸿沟很难跨越。

例如 Docker 的出现之后，以极快的速度风靡全

球。之所以非常多的工程师喜欢 Docker，就是因为 Docker 的特性——在传统的容器系统之上增加了新特性，包括把应用程序和底层依赖库封装为一个容器镜像，容器镜像有版本，而且可以通过集中的镜像仓库进行存储和大批量分发。

Docker 首先解决了长期困扰工程师的开发、测试、上线环境标准化的问题，能够支持开发者进行快速的迭代。同时使用了统一的镜像仓库来进行镜像的分发，而且底层采用了轻量级虚拟机即容器的技术，可以非常快地被拉起，所以采用 Docker 的系统可以很方便进行弹性扩展。同时，因为把应用 App 封装在一个镜像里面，可以在逻辑上根据 Domain Model 的设计原则进行更好地抽象和复用。

显然，这样的技术值得每一个开发计算机系统的工程师学习和掌握。因为他能带来极大的方便。

相反，在 Docker 产生之前，虽然 Control Group（简称 cgroup）+ Namespace 的技术早就已经出现，并早就集成在 Linux 内核中，Google 的 borg 相关的论文早就已经发表，但是一般的技术研发团队不是很容易就能驾驭容器并把容器系统在公司内部大规模进行部署的。

印象中 borg 论文出现后，国内只有 BAT 级别的互联网公司，才有一小撮精英研发团队来研发和使用容器管理系统，例如百度负责 Matrix 系统研发的团队，阿里负责 Pouch 系统研发的团队，腾讯也有一个小团队负责容器系统的研究。

但是除了那一小部分团队，更多的工程师因为学习难度较大而没有把容器大批量地用起来。而 Docker 这种技术，非常好地顺应了敏捷和弹性扩展的技术趋势，而且提供了非常好的用户易用性，一出场就被非常多的工程师迅速使用，而且成为市场的默认标准。

这些顺应潮流的开源软件是值得选择和投入的。

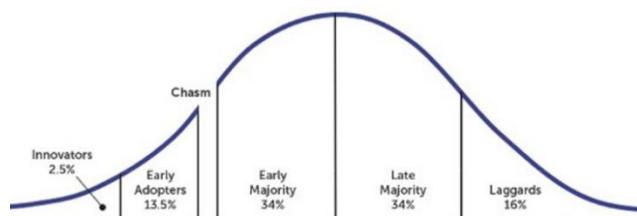
另外一个例子是 Spark，Spark 的出现解决了 MapReduce 在分布式计算过程中因为需要频繁进行 IO 操作导致的性能比较低下的问题，同时在易用性上有较大的提升，所以才取代了 MapReduce 在分布式计算领域内的主流地位。

三、根据开源软件采纳周期的不同阶段进行选择

软件作为智力活动的产物，有它自己的生命周期，一般用软件的技术采纳曲线表示。

开源软件也是软件的一种，也都遵循软件的技术采纳规律。（如下图所示）

Technology Adoption Life Cycle



一个开源软件从创建到衰亡一般会经过 5 个阶段。从创新期（Innovators，占比 2.5%），到早期采纳期（Early Adopters，占比 13.5%），然后跨越鸿沟（chasm），进入到早期大众期（Early Majority，占比 34%），再进入后期大众期（Late Majority，占比 34%），最后进入衰退期（Laggards，占比 16%）。

绝大部分的开源创新项目，没能成功的跨域鸿沟，即从早期采纳阶段进入到早期大众阶段，就消亡了。所以，如果是选择一个需要长期使用并维护的开

源项目，选择处于早期大众或者后期大众状态的项目是比较理智和科学的。

当然如果只是个人想学习一个新的东西，可以看看处于创新者状态的开源项目，或者看看处于“早期采纳者”状态的项目。

注意不管是从长期研发系统的角度，还是从个人学习的角度，都不要再去看处于衰退期（Laggards）的项目了。例如现阶段即 2022 年，是不用再去选择 Mesos，Docker Swarm 之类的项目了。自从 Kubernetes 成为容器调度技术分类的默认标准，这两个项目就已经处于衰退期，他们的母公司都已经放弃了。这个阶段如果还投入较多精力来开发和维护，除非真的是非常强势的甲方要求，把钱砸在工程师面前，逼得他们不得不用。

同学们可能会问，从哪里可以看到这些技术采纳度曲线？InfoQ、gartner、thoughtworks 每年都会更新他们各自的技术采纳度曲线并公布出来，可以根据这些曲线，再结合一些业内的经验，得出自己的判断。

从 2022 年 InfoQ 对 BigData 领域各种流行技术的判断来看，Hudi、Clickhouse、Delta Lake 等开源软件还处于创新者的阶段，即在工业界采纳还比较少，想学习新项目的同学可以重点关注。但是现在这些开源软件还不适合应用在需要长期维护的成熟应用场景里面。

注意这些知名科技媒体的技术采纳曲线是每年都



2022 年 InfoQ 对 BigData 领域各种流行技术的判断

在更新的，在进行参考的时候别忘了注意一下发表的时间。

四、根据开源软件的成熟度情况选择开源软件

还有一点，即根据开源软件本身的成长度来选择开源。即从这个开源软件是否定期发布，是否处于一个多方维护的状态（即使一个公司的战略发生了变化不再继续维护了，还有其他的公司在长期支持），是否文档比较齐全等多个维度来进行成熟度的评估。

对于开源软件的成长度模型，开源社区有很多度量开源项目的成长度模型，其中 Apache 开源软件基金会的项目成长度模型是比较有名的，它把一个开源项目的评估纬度，分为 7 个维度：

- Code (代码)
- License and Copyright (软件许可证和版权)
- Release (发布)
- Quality (质量)
- Community (社区)
- Consensus Building (共识共建)
- Independence (独立性)

每个纬度又有几个考察项。例如针对 Independence (独立性)，又有两个考察项，其一是看这个项目是否独立于任何公司或者组织的影响，其二是看贡献者在社区内活动是代表他们个人，还是作为公司或者组织的代表出现在社区并进行活动的。

Apache 基金会 Top Level 的项目即顶级项目，在毕业阶段都会从这些维度进行综合的判断。只有各方面都达标的项目，才会被允许从 Apache 基金会的孵化状态中毕业而成为成为 Top Level 的项目。这也是逼着个人比较喜欢 Apache 顶级项目的原因。

另外，OpenSSF 项目的 Criticality 评分（参见 https://github.com/ossf/criticality_score）也是一个不错的参考指标，它会度量一个项目的社区贡献者

数量、提交频度、发版频度、被依赖的数量等指标，来判断一个开源软件在开源生态中的重要程度。这里就不详细展开了，有兴趣的同学可以参考它的资料，个人认为是一个值得参考的方向，但是这个评分还处于早期阶段，距离理想状态还比较远。

五、根据项目的质量指标来进行选择

很明显，有些开源软件的代码质量就是比其他开源软件的质量好。有的时候需要从项目的质量情况来选择开源软件。

这个时候，我们需要查看一些被业内广泛证明比较有效的指标。

其中 MTTU 是被知名开源供应链软件供应商 SonaType 所推荐的指标。

MTTU (Mean Time to Update)：即开源软件更新它依赖库的版本的平均时间。举个例子来说，某开源软件 A 依赖于开源库 B，假设 A 的当前版本是 1.0，依赖 B 的版本是 1.1。某天开源库 B 的版本从 1.1 升级到了 1.2，然后一段时间之后，开源软件 A 也发布了新版本 1.1，其中把对 B 的依赖版本从 1.1 升级到了 1.2。这个时间间隔，即从开源版本 B 的版本升级到 1.2 的时间点距离开源软件 A 的新版本 1.1 的发布时间，称之为 Time to Update。

这个时间反映出来的是开源软件 A 的研发团队，根据依赖库的更新周期，同步更新它的依赖版本的能力。Mean Time to Update 是指这个软件的平均升级时间。数值越低表明质量越好，表明该软件的负责人在很快速地升级各种依赖库的版本，在及时修复各种依赖库引起的安全漏洞问题。

据 SonaType 的统计，业内开源软件的更新升级时间 MTTU 越来越短。在 Maven 中心仓库上的 Java 类开源软件，2011 年平均的 MTTU 为 371 天，2014 年平均的 MTTU 为 302 天，2018 年平均的 MTTU 是 158 天，而 2021 年平均的 MTTU 时间是 28 天。能看出来，随着开源软件库更新频率的加快，使用它们的软件也加快了更新版本的速度，

MTTU 相对 10 年前，时间缩短到原来的 10/1 以下。

当然 MTTU 只是项目质量的一个间接纬度。历史上是否爆出重要高危安全漏洞，修复响应是否快速及时等等，也是开源项目质量评价的重要维度。

某些大厂的安全部门，会不断评估开源软件的安全情况，把某些屡屡发生高危安全漏洞，但是修复不及时的开源软件设定为不安全软件，列入到内部的开源软件黑名单中对内公示，并要求各个业务研发团队不再使用这些软件。如果实在因为研发和人力问题不能迁移到新的软件系统，也需要把这些老服务迁移到一个相对封闭的网络环境中，以减少风险可能造成的损失。这个时候，显然应该需要遵守公司的安全规定，不再使用黑名单上的开源软件。

六、从开源软件所属于的开源社区治理模式角度来考虑

还有一个维度，即从开源项目的社区治理模式来考虑，适用于需要长期进行开发和维护的项目。

社区治理模式（Governance Model）主要是指该项目或者社区是如何做决定的以及由谁来做决定。具体表现为：是所有人都可以做贡献吗还是少数几个？决定是通过投票的方式产生的，还是通过权威？计划和讨论是否可见？

常见的开源社区和开源项目的治理模式有如下三种：

单一公司主导：特点是软件的设计、开发和发布都由一个公司来控制，也不接受外部贡献。开发计划和版本计划不对外公开，相关讨论也不对外公开，版本发布时候才对外公开源码。例如 Google 的 Android 系统。

独裁者主导（有个专有名词“Benevolent Dictatorship”，翻译为“仁慈的独裁者”）：特点是由一个人来控制项目的发展，他有强大的影响力和领导力，一般都是该项目的创始人。例如 Linux Kernel 由 Linus Torvalds 来负责，Python 之前由 Guido Van Rossum 来主导。

董事会主导：特点是有一拨人构成项目的董事会来决定项目的重大事项。例如 Apache 软件基金会的项目由该项目的 PMC 决定，CNCF 的基金会的决策是 CNCF 董事会来负责（很多技术决定授权给了 CNCF 董事会下的技术监督委员会）。

个人意见和经验，根据该开源软件背后的开源社区的治理方式来进行选择优先级的排序如下：

优先选择 Apache 毕业项目，因为这些项目的知识产权情况清晰，而且至少有三方在长期维护；

次优选择 Linux 基金会等其他开源基金会的重点项目，因为 Linux 基金会的运营能力很强，每个重点项目后面往往都有一个或者多个大公司在支持；

小心选择一个公司主导的开源项目，因为该企业的开源战略随时可能会调整，很有可能不再持续支持该项目，例如 Facebook 就是一个弃坑很多的公司；

尽量不选择个人开源的项目，个人开源更加随意，风险尤其高，但是不排除某些已经有很高知名度，并且跑出长期维护模式的项目，例如知名开源作者尤雨溪（Evan You）所负责的 Vue.js 开源软件。

这是个人推荐的选择同类开源软件项目的优先级顺序，仅仅代表个人观点，欢迎讨论。

开源知多D

Windows 注册表文件为什么是 "hive"？

Windows NT 的原始开发者之一讨厌蜜蜂。因此，负责注册表的开发者尽可能多地往里面塞入了蜜蜂居住的场所。注册表文件称为 "hive"（蜂房），注册表数据存储在 "cell" 中，这正是蜂窝的组成部分。

搞懂社区运营： 留存与转化的几个常见逻辑范式

近年来，国内的“开源项目”越来越活跃，愈加受到各个厂商的重视。可以看到，很多开发者社区都在组织技术类的线上直播与线下沙龙；在搜索引擎中，与“开源项目”相关技术文档也越来越多。这些运营策略，大部分都是针对开发者群体，以实现“用户增长”和“产品推广”，从而获得更多的用户、贡献者与客户。

那么，获得用户后如何实现留存与转化？这个问题的答案，没有固定的范式，因为每个社区的运营策略并不一样。而开发者又是一个有着“情怀”的群体，对于“留存”与“转化”策略的要求比较高。在这里，我分享一下社区运营中几个比较常用的逻辑范式。

文/ Benker

一、社区运营价值三角

运营社区，首先要了解社区的组织结构与运行原理。在社区运营中，常见的角色有三种，分别是运营方（管理者/组织者）、KOL（意见领袖）、与用户。

运营方

运营方是由发起人或者产品项目方筛选与组织起来的专职或兼职的团队，也是社区中比较活跃的积极用户。运营方在社区运营中承担着主要作用，主要承担社区价值观、社区建设与推广、项目管理与推广、用户管理、内容/项目生态建设与管理等工作。值得一提的是社区发起人，是一个具有情怀与利他的人格（这里不包含商业项目）。

KOL

在社区中有一群积极的活跃份子，对技术或产品有着专业的技术能力，在社区活动中经常分享技术/产品的知识与技术的探索实践。这群人有一个统一的称呼--KOL（意见领袖）。

KOL 是社区中的中坚力量，在运营中起到承上启下的作用。对上能帮助技术/产品探索更多的业务场景，对下帮助用户快速的学习项目中的技术、产品的知识点，带领新人熟悉并快熟使用技术/产品。

KOL 还有一项很重要的工作就是传播技术与产品的知识，业内称为“布道者”，类似宗教的“传道士”，负责传播与教育用户。在社区运营中 KOL 越多，说明用户的覆盖率越大，使用用户也就越多，也能加速产品技术垂直生态的形成。

用户

用户通常是指技术、产品、服务的使用者。广义上讲技术、产品、服务能覆盖到的人，未必参与技术、产品、服务的流转过程中。

用户也可分为“有效用户”与“无效用户”。有效用户通常是指使用产品技术或购买社区中提供的产品技术的人；反推则是无效用户。当然每个公司的定义与口径也不一样，在这里不做过多的描述。

要把社区做到自运营状态，就需要有一个核心的价值观贯穿到社区运营中，也是社区定位中的准则。这里提一下，在资讯开发者社区或者内容类社区中称为“产品意识”。这里可以用一个三角形来比喻，也称之为“价值三角”。

运营方是社区的组织者与建设者、定义与维护社区规范与规则，引导社区用户互动形成社区价值，并将社区价值分配给 KOL 与用户形成“利益链”。组织者获得社区在社会中的精神激励与物质激励。



图1 价值三角

KOL 是社区中产品技术使用的受益者，积极的分享传播产品与技术，探索技术在业务场景中的应用。并在社区中获得个人影响力与间接的物质激励；

用户是社区中产品技术使用者与被影响者，获得社区中的产品与技术，并积极的反馈优化产品技术，不断的提升自己使用产品技术的能力。

在这个“价值三角”中，社区中的任何一个人都是受益者，三者之间相互相助，互相成就，社区才会形成正向运转的“利益链”并达成“自运转”的状态，基于产品技术也会产生垂直的生态，让社区价值利益最大化。

二、决策曲线与社区活跃曲线

在互联网运营中，大家都知道“用户运营”的重要性，互联网用户运营通常按照用户性别、用户年龄、用户所在区域、学历、信用级别、消费行为习惯等属性对用户进行层级划分与用户管理。

而在开发者社区中，同样有用户运营，称为“开发者运营”，开发者运营通常按照行业属性、企业属性、角色属性、个人属性等属性对开发者进行层级划分与管理。值得一提的是，互联网用户运营通常会有产品的生命周期，而开发者运营是一个长尾的效应。

开发者是一个有情怀、自学能力强的群体。从一个职场小白经过几年的职业进化，成为技术专家或行业专家，是一个开发者的成长的路径。

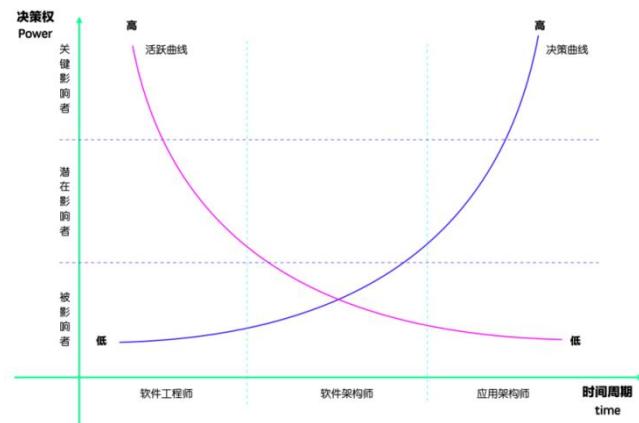


图2 曲线图

通常情况下，我们按照年限对开发者进行划分，0-5年的软件工程师（简称软件工程师），5-8年的软件架构师（简称软件架构师），8年以上的应用架构师（简称应用架构师）。

从软件工程师成长为应用架构师，开发者的决策权与个人影响力也会越来越高，这条蓝色的曲线我们称之为“决策曲线”。（如图2所示）

相对来说，随着职业年龄增加及职位提升，工作事务的压力也会越来越高，个人在社区中的活跃度也会越来越低。这条红色的曲线我们称之为“社区活跃曲线”。“决策曲线”与“社区活跃曲线”成反比。

通过这两条曲线我们可以得知，软件工程师在社区中的活跃度高，应用架构师在社区的活跃度低；而应用架构师在社区中的影响力强，软件工程师与软件架构师在社区中的影响力弱，属于产品技术的潜在影响者或潜在用户。

因此，我们要通过两条曲线的属性设置用户成长体系与激励体系，来将“应用架构师”转化为布道师（关键影响者），通过内容、活动、问答、技术交流等互动，来影响“软件工程师”与“软件架构师”（潜在影响者）转化为产品技术的贡献者与使用者。

软件工程师经过几年的职业进化成长为应用架构师，通过社区的成长体系与激励体系又被转化为“布道师”。周而复始的循环，社区就会达到“自运转”状态，这也标志着社区治理进入了“成熟期”。

三、承诺曲线与用户旅程地图

在社区运营中，还有一个比较常用的框架，就是“承诺曲线”（如图 3）。“承诺曲线”是帮助组织者基于用户的情绪来设置用户使用社区的每一个步骤，从而形成用户旅程地图，帮助用户快速地了解社区、认识社区、参与社区、并最终领导社区。

通过这个工具，可以让用户兴奋，激发他们的好奇心，分享自豪感，找到在社区中的归属感。

介绍下“承诺曲线”的工作原理。通过图 3 我们可以看到，用户从小白到社区领导者，大概分为四个阶段，分别是：Discovering、Onboarding、Engaging、Leading。这四个阶段也是运营方构想社区营销漏斗。用户通过这四个阶段一步一步的加深对社区的投入，参与到社区的建设中从而让自己觉得属于社区，成为社区中的一员，并推进社区前行，找到自己在社区的位置与归属感。

当然，在此过程中，社区一定给到用户积极正面的回应与互动，提升社区对用户的正向认可。

在使用“承诺曲线”之前，我们应该思考几个问题。

- 什么问题会对用户有意义？
- 什么问题会让用户感到不知所措、困惑或烦恼？
- 用户在社区的每个阶段都会存在什么样的问题？



图 3 承诺曲线图

- 怎样让用户更快、更方便的使用技术、产品与服务？

带着这些问题，我们对用户的行为就会有大概的认知，基于用户使用社区的每个步骤，按照使用顺序在承诺曲线的四个阶段中分别标注出来，就形成了一个简单的用户路径图。（如图 4）

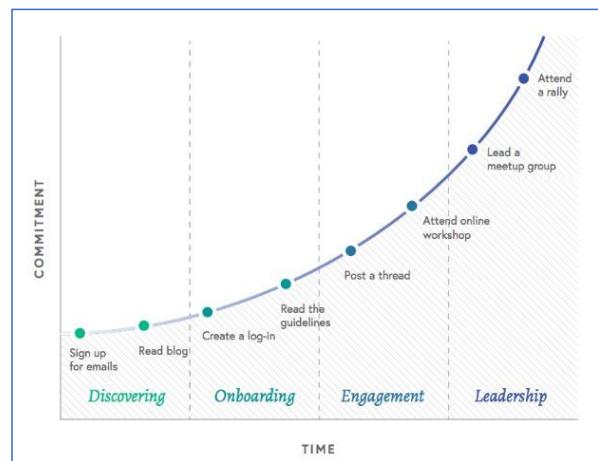


图 4 用户路径图

通常情况下，用户使用产品技术的步骤有：了解探索、加入注册、下载使用、开发测试、服务支持、用户推荐、问题反馈。

基于这 7 个步骤与图 4 中用户路径图的结合，就形成了一个带有触点动作的开发者体验图（如图 5）。

触点是开发者社区通过邮件、推送信息、内容、活动、直播、社群等媒介触达到开发者的动作。

基于开发者体验的每个步骤，并提炼出步骤中的最核心的运营数据就形成了运营指标。

用户旅程图会帮助运营方战略性的调整内容与运营策略，从而提升社区运营中的留存与转化。

可以通过承诺曲线绘制出来一段时间内的用户参与图，从而提升社区运营中阶段内的留存与转化的数据。也可以通过承诺曲线绘制出来某类人群在社区中的使用图，来提升社区运营中垂直人群的留存与转化。

社区承诺曲线是每个社区运营方都可以使用的框架，可以通过这个框架更加直接的关注社区运营中用

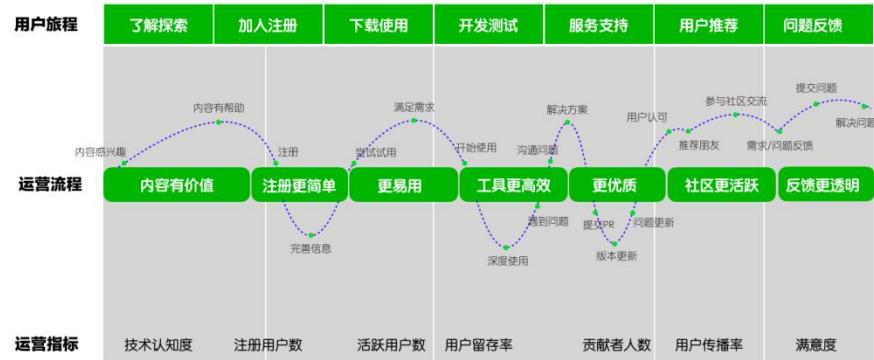


图 5 开发者体验图

户留存与转化的数据，并随时的调整运营策略，提升社区运营的效率。

四、社交因子

大家都知道一个通俗的道理，教师的朋友圈大多是教师，医生的朋友圈大多是医生，也就是说每一类人群朋友圈中，大多数朋友的职业都是一样的。

从定义上面来看，社区是具有相同爱好与共同属性的人群聚集在一起，社区也是具有“社交”因子。而“社交”的本质就是寻找与匹配同频的人，也就是同一类的人。只是社区关注于内容，而社交注重人与人之间的关系。

从以上概念来看，“社交因子”在社区运营中的留存与转化起到了建立信任与信息互相传递的作用。“社交因子”也是社区运营与用户运营的内驱力与核心动能。

常常在思考这样几个问题：

- 用户为什么这么做？
- 怎样才能让用户这样做？
- 用户做这这样动作的核心内驱力是什么呢？

其实，这些问题都归因于社区中用户之间的“社交因子”在驱动，“社交因子”是组成社区运营的基本单元。在社区运营都有哪些“社交因子”驱动着社区前进呢？

下面我们就一起来认识下这四个社交因子：兴趣、需求、个人成长、价值认同。

兴趣

兴趣是个人力求接近、探索某种事物和从事某种活动的态度和倾向，亦称“爱好”，是个性倾向性的一种表现形式。兴趣在人的心理行为中具有重要作用。

当一个人对某个事物产生兴趣时，便对它特别的注意，该事物的曲线变化也就左右这个人的情绪，也就能让人与事物产生情感。兴趣不是靠外界强制力量形成的，而是出于个人的强烈愿望建立和发展起来的。所以，在社区运营中策划的内容与活动一定要基于开发者群体“人性”，引起他们的兴趣，才能抓住了他们的情绪与注意力，达到传播与留存转化的目的。

需求

需求，广义来讲，包含利益、当下遇到的难题、个人技术技能与职业的精进、个人的理想等等。只要是当下有需要的要求，都可以成为需求。在这里我们只谈广义上面的需求。因为在用户旅途地图上，有很多触点都可以与用户的需求发生关联与匹配。故在社区运营中通过用户旅程地图中的关键触点，来设定用户的运营策略，提升社区运营中的留存与转化。

个人成长

个人成长是指开发者在职业生涯精进过程中的自我突破，这里包含自我学习的欲望、技术与职业精进的欲望、自我实现的愿景等等。0-5 年的软件工程师对于技术架构的学习有着强烈的愿望，5-8 年的软件

架构师对于项目的实现与管理有着强大的动力，8年以上的应用架构师对于自我价值的实现有着强烈的驱动力。

这些在个人成长的动机中，都是开发者在每个阶段成长的必经之路。在个人成长路径中需要通过社交与技术交流来满足自身成长的目的。在社区运营中，我们可以通过个人成长路径中的动机，来设定与满足这些动机的策略完成留存与转化。

价值认同

价值认同是指社会成员或组织在社会活动中对某类价值的内在认可或共识，通过这些认可或共识，形成自身在社会实践中的价值定位和定向，由此决定自己的理想、信念和追求。后并形成共同的价值观，它是社会成员对社会价值规范所采取的自觉接受、自愿遵循的态度及甚至服从。

当然价值认同是由个人与社区之间形成正向的沟通反馈形成的，两者缺一不可，在社区中常见策略是激励体系。价值认同也是社区规范与社区价值观最重要的组成部分。通过价值认同，可以筛选与转化用户，一旦被筛选有着超高的黏性。在社区运营中，运营方通过社区行为规范、社区意识、激励体系来维护社区的价值观，有利于社区的长久发展。对于破坏社区价值观的行为，一定及时的纠正与禁止。

这四个社交因子是社交关系中最基本的社交元素，也是在社区运用中最常用的基本单元。在社区运营中往往是多个组合在一起形成一个运营策略，也可以根据运营的目标不同来灵活使用。

五、留存三要素：内容、互动、用户体系

通过上文我们知道了社区中的基本角色，在社区运营中也有着留存的基本元素，这些基本元素也是促进社区自运转不可或缺的必备元素：内容、互动、用户体系。这三个基本元素是相辅相成的，有着基本的逻辑顺序。

内容

内容是社区中最重要的媒介，也是社区中核心的

组成部分。内容不但但是文章，广义上的内容包含资讯、专栏文章、问答与话题讨论、视频课程、开发者文档与手册、用户案例、社区图书刊物、技术知识图谱等形式。内容是触达开发者的重要触点，通过不同的内容形式承接社区运营中的用户拉新、留存、与转化。

不同的内容形式也承接不同的任务。资讯、专栏文章承接引导用户探索社区；问答与话题讨论、视频课程承接用户产生兴趣与参与，并帮助用户解决使用产品技术中遇到的难题与解决方案，让用户更快的使用产品与服务，起到教育与转化用户的目的；开发者文档与手册承接让开发者更顺畅的使用产品技术与教育用户；用户案例承接用户在使用产品技术探索不同业务场景中的实践；社区图书刊物与技术知识图谱起到技术品牌与用户口碑的建立与传播，占领用户的心智，并起到用户的教育、拉新与转化。

内容在社区运营中总体起到的是让用户快速认知产品技术，让用户更快的使用与解决产品技术中的问题，提升开发者使用产品技术的体验与满意度。

互动

互动是用户在社区浏览专栏文章的点赞、关注、收藏、转发、打赏，参与社区问答、话题讨论、社区活动、技术交流分享等互动行为。

互动是可以量化的，通过互动的频率与互动的习惯，可以评估用户的用户的黏性，进一步拉近用户与社区之间的距离，让用户在社区中产生归属感。通过互动，让用户参与到社区的共建中，并认同社区的价值观，形成社区共同体。

互动一定社区与用户之间双方来完成，也是社区与用户之间的沟通，建立社区与用户之间“信息、接收、反馈”这样一个正向的沟通机制，这个简单的机制称为“价值互动”，也是“有效互动”。有效互动是促进社区与用户之间的润滑剂，增加社区与用户之间的信任与黏性。

通过互动也可以发掘产品技术的商机，对用户形成“认知、使用、客户”的角色转化。借助场景化的互动，让用户参与到产品技术的共建当中，加快产品技术在业务场景中的解决方案的建立，并快速形成社区中的垂直生态。

用户体系

是通过数值化、标记的用户行为的类型，对用户分层、分类管理，并通过激励用户行为让用户在平台中形成行为惯性，达到形成共同体的目的。用户行为包含用户成长体系、激励机制、成绩系统、用户管理等系统。用户体系是社区运营中最根本的系统，社区其他的运营策略要基于用户体系之上来建立。

开发者社区中的用户体系与互联网用户体系的原理类似，都是对用户的行为进行激励与管理，但也有区别。因为开发者是一群特殊的群体，有情怀、自学能力强、比较闷、不喜欢套路等特点。在构建用户体系时，要基于开发者的行规范来设定。开发者运营的成长体系与激励机制并没有互联网用户体系那样复杂与精彩。

结语

社区不是一个人的事情，是一个具有相同兴趣与特征群体的共同运营成果，积极发掘每一个成员在社区中的价值，是社区运营要达到的终极目标。社区不仅仅是一个互联网平台，它包含了内容、产品、服务、场景、用户、商业价值等生态组成。在这里只介绍在社区运营中留存与转化的逻辑范式，不具体阐述社区运营中的策略、实践与操作。由于作者水平有限，本文中出现存疑的地方，还请大家多多指教。

特别声明：本文中所展示的图片，转载时请联系作者。

关于作者

Benker，一个产品与运营的从业者，专注于技术与商业应用的实践者。

个人博客：benker.cn



微信公众号：Benker



《众说》栏目征稿

在开源之路上，你经历过什么，你思考着什么，你所处的立场是什么，你秉持的理念是什么，这一切都值得记录下来。不管是对作者而言，还是对读者而言，都将是一笔宝贵的精神财富。

《开源观止》设置《众说》栏目，旨在借众家之口，谈论开源之实务，弘扬开源之精神。这方天地虽小，却也期待百家争鸣。

如欲投稿，请联系我们：18370998278



别了，IE



字节跳动正式开源 Go HTTP 框架 Hertz

Hertz 是 Golang 微服务 HTTP 框架，也是字节跳动内部最大的 HTTP 框架，线上接入的服务数量超过 1 万，峰值 QPS 超过 4 千万，具有高易用性、易扩展、低时延等特点。

Apache Doris 毕业成为 Apache 顶级项目

Apache Doris 成功从 Apache 孵化器毕业，正式成为 Apache 顶级项目 (Top-Level Project, TLP)。Apache Doris 最早是诞生于百度内部广告报表业务的 Palo 项目，2017 年正式对外开源，2018 年 7 月由百度捐赠给 Apache 基金会进行孵化，之后在 Apache 导师的指导下由孵化器项目管理委员会成员进行孵化和运营。

Linux 5.19 已合并龙芯 LoongArch CPU 架构

Linus 已将龙芯 LoongArch CPU 架构的内核移植合并到 Linux 5.19。不过由于一些代码尚未通过审查，虽然 LoongArch CPU 架构代码已经被合并，但缺少一些关键的驱动程序，因此 Linux 5.19 暂未支持在搭载 LoongArch CPU 的设备上启动。

SQLite 推出 Web 版本：SQLite 3 Fiddle

SQLite 推出了实验性的 Web 版本：SQLite 3 Fiddle。开发者无需下载或安装任何东西，直接访问 <https://sqlite.org/fiddle/> 即可在 Web 浏览器运行最新的 SQLite 3.39.0 Beta。官方称，这是在 Web 浏览器的沙箱中运行了一个 WASM 编译的最新 3.39.0 构建的副本。

Archhttp 1.0：简单、灵活、高并发，D 语言 Web 服务端框架

Archhttp 作者说道：“D 语言在国内的环境来说一直热不起来，我和我的开源团队也做过很多开源的框架，全功能的，基本对标 Spring Boot 和 Spring Cloud 那种级别，但是新手用起来不够友好，由于 D 语言在国内的文档匮乏导致入门难，运行这么庞大的框架更是不可想象的难。”

腾讯正式开源 Spring Cloud Tencent

Spring Cloud Tencent 是腾讯开源的一站式微服务解决方案。Spring Cloud Tencent 实现了 Spring Cloud 标准微服务 SPI，开发者可以基于 Spring Cloud Tencent 快速开发 Spring Cloud 微服务架构应用。

KubeSphere 迈入新的里程碑 —— 10000 颗星星！

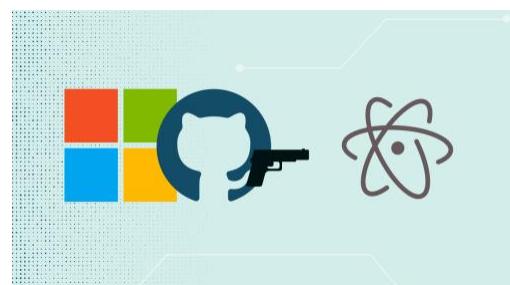
开源项目 KubeSphere 于近期在“云原生分布式操作系统”的征程中迈入新的里程碑 —— GitHub Star 数量突破 10,000！

Linux Lab 项目启动 6 年后，终于发布了 v1.0 正式版

Linux Lab 是一套基于 Docker/QEMU 的极速 Linux 内核学习、开发和测试环境，也可以用于嵌入式 Linux 系统开发。公告写道：“Linux Lab v1.0 是一个很重要的里程碑，不仅意味着其核心功能已经非常稳定和完善，也意味着我们的工作重心将发生调整。在 v1.0 之后，我们将把重点调整到 examples 的开发上，将通过 examples 帮助更多的同学更高效地学习、研究和开发操作系统以及周边的技术。”

GitHub 宣布终止 Atom

GitHub 正式宣布将关闭文本编辑器项目 Atom，所有代码库将在 12 月 15 日归档。

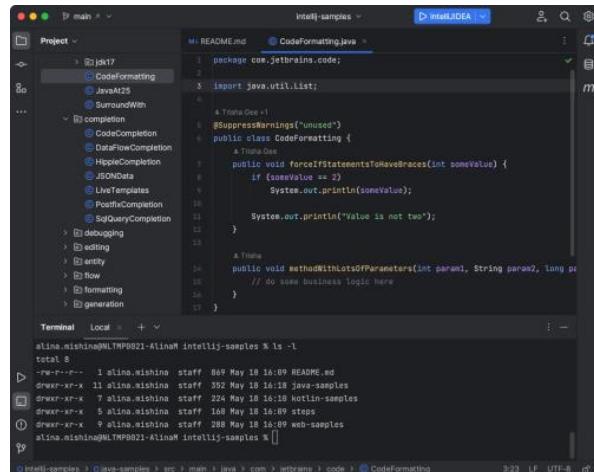


[Apache InLong 毕业成为 Apache 顶级项目](#)

Apache InLong 成功从 Apache 孵化器毕业，成为社区顶级项目。Apache InLong 是由腾讯捐献给 Apache 社区的一站式海量数据集成框架，可以为大数据开发者提供百万亿级数据流高性能处理能力，以及千亿级数据流高可靠服务。这也是 Apache 社区首个一站式大数据集成顶级项目。

[JetBrains IDE 公布全新默认 UI](#)

JetBrains 宣布为其 IDE 系列产品更新了默认 UI。



[Deno 核心开发者开源 Fresh，宣称是下一代 Web 框架](#)

Deno 核心开发者 Luca Casonato 开源了名为「Fresh」的新项目，并称它是下一代 Web 框架。根据官网的介绍，Fresh 是面向 JavaScript 和 TypeScript 开发者的全栈现代 Web 框架，可帮助他们轻松创建高质量、高性能、稳定性好，以及支持定制的 Web 应用。

[Web 框架 mojo.js 1.0 正式发布，从 Perl 到 Node.js](#)

历经一年多的开发，mojo.js 终于发布了首个主要版本 1.0。mojo.js 是 Node.js 实时 Web 框架，可将其视作采用 TypeScript 重写的 Mojolicious（Mojolicious 是 Perl 开发的 Web 框架）。mojo.js 使用了所有最新的 JavaScript 特性，专为聚焦超媒体 (hypermedia) 的后端 Web 服务精心设计。

[WebAssembly 团队成员：谷歌是我工作过的最糟糕的公司](#)

WebAssembly 团队创始成员 Katelyn Gadd 于 2015 年加入谷歌 V8 团队，是编写 WebAssembly 规范的首批成员。她表示，谷歌是她工作过的最糟糕的公司——给她带来了脑损伤，让她难以入睡，每天都感到紧张。Katelyn Gadd 希望通过自己就职于谷歌的这段经历，帮助更多人认识到工作中的“毒文化 (toxic cultures)”，以及帮助新员工在谷歌拥有更好的职业生涯。

[微软推出 Arm PC 和原生 Arm 开发者工具链](#)

Project Volterra 是专门面向开发者打造的 Arm PC。

[小米向 AOSP 提交代码：禁止通过 shell 直接导出 APK](#)

小米工程师在向 AOSP (Android Open Source Project) 提交的一个 commit 中指出，应禁止通过 shell 获取已安装的 APK，理由是某些 APK 可能包含私有资源或内容，因此不应该被允许直接从系统中拉取。

[Linux 5.19 新增近 50 万行新的图形驱动程序代码](#)

Linux 5.19 的 DRM 图形驱动程序更新包含对 AMD 和 Intel 最新版本显卡的完整驱动支持，因此代码数量有点超乎想象，更改共包含 1302 个文件，新增 495,793 行代码，删除 32,019 行代码。

[Palm OS 开发者开源 20 多年前的经典游戏](#)

Palm OS 开发者 Aaron Ardiri 在其 GitHub 主页发布了 20 多年前为 Palm OS 开发的多款游戏、GameBoy 模拟器和各种工具的源代码，总共有 12 个仓库。



C++ 23 将引入标准库模块支持，编译速度提升 10 倍

C++ 之父 Bjarne Stroustrup 表示，C++ 23 将包括一个标准库模块，称为 std。有了标准库模块，“Hello World” 就会变成下方示例这样：

```
import std:  
int main()  
{  
    Std::cout<< "Hello, World!\\n";  
}
```

Stack Overflow 年度调查：Rust/PostgreSQL 最受喜爱、PHP 薪水偏低

180 个国家/地区的 73000 多名开发者参与了 Stack Overflow 的年度开发者调查。结果显示 Rust 连续第七年成为开发者最喜欢的编程语言，其次是 Elixir、Clojure、TypeScript 和 Julia。JavaScript 仍然是最常用的语言，其次是 HTML /CSS、SQL、Python 和 TypeScript。开发者最常用的操作系统是 Windows、Linux 和 macOS，Windows 的 Linux 子系统的使用率从 2021 年的 3% 增加到了 14%。

WordPress 6.0 正式发布

经过 58 个国家共 500 多位贡献者的努力，WordPress 6.0 近日正式发布。此次更新将数百个增强和修复交付到一个稳定的版本中，证明了 WordPress 社区的力量和能力。

“古董级”诺基亚功能机跑 Linux 是怎样的画风？

下图这台古董机套了诺基亚手机的外壳，里面是作者全新设计的 PCB 主板，其尺寸与诺基亚的原始 PCB 完全相同，所以称之为“Notkia”。



TIOBE 6 月榜单：C++ 即将超越 Java

Jun 2022	Jun 2021	Change	Programming Language	Ratings	Change
1	2	▲	Python	12.20%	+0.30%
2	1	▼	C	11.81%	-0.64%
3	3	●	Java	10.47%	-1.07%
4	4	●	C++	9.63%	+2.20%
5	5	●	C#	6.32%	+17%
6	6	●	Visual Basic	5.42%	+1.42%
7	7	●	JavaScript	2.09%	-0.24%
8	10	▲	SQL	1.94%	+0.06%
9	9	●	Assembly language	1.89%	-0.21%
10	16	●	Swift	1.69%	+0.44%
11	11	●	Classic Visual Basic	1.32%	-0.45%
12	18	●	Delphi/Object Pascal	1.32%	+0.26%
13	8	▼	PHP	1.25%	-0.87%
14	23	▲	Objective-C	1.02%	+0.33%
15	20	●	Go	1.02%	+0.07%
16	14	▼	R	0.98%	-0.22%
17	15	●	Perl	0.78%	-0.41%
18	38	▲	Lua	0.76%	+0.05%
19	13	▼	Ruby	0.75%	-0.48%
20	26	▲	Prolog	0.74%	+0.38%

Angular 14 正式 GA

Angular 14 已正式 GA。值得关注的新特性包括：引入更加严格的类型化表单、Angular CLI 支持自动补全、独立组件发布开发者预览版、简化 page title 可访问性、Angular DevTools 支持离线使用，以及在 Firefox 中使用、引入实验性的 ESM 应用构建系统等。

Microsoft Store 新政策，禁止第三方对免费的开源软件收费

微软更新了它的应用商店规则，其中一条是“不要试图从开源或其他通常免费提供的软件中获利，也不要试图相对于产品提供的特性和功能定价过高”。

新手向 GitHub 仓库提交 PR，结果“轰炸”了近 40 万开发者

这名新手程序员或许无意间创造了一项世界纪录——他在 GitHub 创建的 PR 让 40 万开发者收到了邮件提醒，并被反复“轰炸”。

Deno 完成 2100 万美元 A 轮融资

Deno 运行时背后的同名开发商 Deno 宣布，它已完成了由红杉领投的 2100 万美元 A 轮融资，使其融资总额达到 2600 万美元。Deno 首席执行官 Ryan Dahl 表示，此轮融资将帮助他们用于构建 Deno 的商业产品——Deno Deploy，并同时开展营销和探索新的业务线。

RISC OS 诞生 35 周年

RISC OS 是面向 ARM 架构系统设计的基于图形用户界面的电脑操作系统，由命令行界面和桌面环境组成，其名字来源于它所支持的精简指令集 RISC 架构。RISC OS 最初由 Acorn Computers 开发，搭载他们于 1987 年推出的使用 ARM 处理器的 Archimedes 个人电脑。



惠普发布面向开发者的笔记本电脑：HP Dev One

HP Dev One 宣称完全面向开发者打造，比如其预装的系统是 Linux 发行版 Pop!_OS，以及提供了包含 Super 键的可定制 Linux 键盘。

Grafana 9 正式发布

Grafana 9.0 的主要重点是改善 Grafana 的用户体验，使可观察性和数据可视化更易用也更容易获得。无论是通过 Prometheus 和 Loki 可视化查询生成器还是面板和仪表板搜索功能，Grafana 9.0 都引入了更新的工作流程，使发现和调查数据变得更加容易和直观。

.NET MAUI 正式 GA：一套代码支持多平台

.NET MAUI 已正式 GA。.NET MAUI (.NET Multi-platform App UI) 是一个跨平台 UI 框架（前身是 Xamarin.Forms），通过 C# 和 XAML 创建原生移动和桌面应用。基于 .NET MAUI，开发者可在单个共享代码库中创建同时支持 Android、iOS、macOS 和 Windows 的原生应用。

HTTP/3 官宣，编号 RFC 9114

IETF（互联网工程任务组）宣布了 HTTP/3 标准，编号为 RFC 9114。HTTP/3 即 HTTP-over-QUIC，是采用了 QUIC 进行传输的新 HTTP 协议。QUIC (Quick UDP Internet Connections) 最初由谷歌开发，当 IETF 开始标准化 QUIC 时，它分成了两个层：传输和 HTTP。该传输协议能用于传输其它数据，不限于 HTTP 或类 HTTP 协议。

毕昇 JDK 登陆 Eclipse Adoptium 开源 JDK 发行版市场

毕昇 JDK 是华为开源的企业级 JDK 发行版，于 2022 年 05 月 26 日成功登陆了 Eclipse Adoptium 开源 JDK 发行版市场。

Apache Hop 2.0.0 正式 GA，开源数据编排平台

Apache Hop（Hop 是 Hop Orchestration Platform 的缩写）是一个灵活、元数据驱动的数据编排、工程和集成平台。该项目起源于二十多年前的 ETL 平台 Kettle，经过几年的重构，于 2020 年 9 月进入 ASF 孵化器。Apache Hop 2.0 提供了中文版本，这项工作由中国开发者 @Shi Xue 贡献。

Rust UI 框架 Tauri 发布 1.0 正式版，性能完爆 Electron

Tauri 是一个桌面 UI 框架，可让开发者使用每个平台的 Webview 技术栈为所有主要桌面操作系统构建应用程序，目前支持 Windows/macOS/Linux 等平台。开发者通过 Tauri 几乎可以使用任何编译为 HTML、JS 和 CSS 的前端框架来构建桌面 UI。Tauri 核心库采用 Rust 编写，使用 Tauri 开发的应用程序的后端是一个基于 Rust 的二进制文件，带有一个前端可以与之交互的 API，通过 JS API 调用后台接口。



从 0 到 1 掌握大数据技术

在人工智能时代，不论是否从事大数据开发，掌握大数据的原理和架构早已成为每个工程师的必备技能。如果未来是面向 AI 编程的，希望软件工程师不要把 AI 当作什么万能的东西。当机器学习结果出现问题的时候，我们既不要陷入某种不可知的“玄学”之中，也不要无谓地抱怨什么“人工智障”，而是应该积极参与到问题的讨论、分析和解决中去。

将来，数据会逐渐成为公司的核心资产和主要竞争力，公司的业务展开和产品进化也会朝着如何利用好数据价值的方向发展。如果你不懂大数据和机器学习，可能连最基本的产品逻辑和商业意图都搞不清楚。如果只懂编程，那么你的生存空间会越来越窄，发展也会处处受限。所以即使自己不做与大数据和机器学习相关的开发，每个程序员也应该懂大数据和机器学习。

OSCHINA 本期高手问答我们请来了李智慧老师和大家一起探讨关于大数据技术相关的问题。



李智慧 同程旅行交通首席架构师

曾任阿里巴巴、Intel 架构师，长期从事分布式系统与大数据开发，Apache Spark 代码贡献者，腾讯云 TVP，著有畅销书《大型网站技术架构：核心原理与案例分析》。

1、大数据开发和运维是分开的，还是开发就兼大数据运维的工作？你觉得是开发兼运维还是有必要分开来吗，原因是什么？

答：看团队规模吧，3-5 个人就没必要分开了，性能优化是运维吧，但是有些优化是需要开发的。人多的话，建议还是分开，人多手杂，这么多机器，都有管理权限肯定不行。

2、云计算与大数据是什么关系？

答：云计算和大数据都是分布式技术的分支体系，云计算一般指通过远程（云端）分布式技术提供弹性的数据计算能力，大数据一般指通过分布式技术解决大规模数据存储和计算的问题。两者结合，可以通过云端提供大数据存储和计算，就是大数据云，云计算可以包括大数据，大数据也可以在云端。

3、企业大数据如何起步？(企业到多大规模的数据后才适合开展大数据)

答：某种意义上，大数据是一种解决问题的思维方式，是一种把所有的数据关联起来发现其价值的技术体系。所以企业用不用大数据技术其实和数据规模没有关系，能不能用大数据解决问题，创造价值才是

关键。当然，前提是对自己数据，自己的业务场景，大数据技术体系有深刻的理解才能发现其中的价值。

4、大数据不断发展：从自建开源 Hadoop 数据湖架构，原始数据统一存放在 HDFS 系统上，引擎以 Hadoop 和 Spark 开源生态为主，存储和计算一体，到现在丰富的云上托管数据入湖，以及各种分析引擎。中小企业是否有必要自建大数据一套体系？

答：我的建议是中小企业开始的时候，没必要自建大数据体系，一方面是成本太高，另一方面就是不确定能不能通过大数据获益。云计算可以在初始阶段大幅降低成本，成熟的云厂商在业务落地方面也可以提供很多有益的实践和模式，帮助中小企业快速落地大数据。

5、程序员应该如何学好大数据技术？（1、从其业务程序转到大数据需要掌握哪些基础？2、是否需要掌握全套的，从采集到清洗到存储，从统计到分析挖掘，以及后期相关 AI 机器学习？）

答：首先程序员需要了解完整的大数据技术体系，知道在数据生产、应用的过程中有哪些环节，起什么作用，背后的思想和原理是什么，然后才能有的放矢，找到适合自己的定位，深入学习和实践，成功转型。

6、多大的数据才算大数据？

答：一般说来，人们提到大数据的时候，指的是包括数据、技术、平台、算法、应用场景在内的完整的生态体系，不止是数据本身。数据本身没有好坏之分，有的数据更容易从中挖掘出业务价值，有的数据更难一点。只有未发现价值的数据，没有无价值的数据。

7、当前大数据技术涉及的东西众多，例如 Hadoop、Spark&Flink、Hive&Trino、ClinkHouse 等等，每个组件从使用到对组件原理的深入都需要花非常多的时间才能算是到入门或者说能掌握其中的原理，而如果想要基于这些组件做好架构，那时间就更长了，

那么对于一个没有什么相关经验的人来说，怎样快速切入到大数据这个赛道？

答：大数据技术众多，但是核心思想和原理是有共同之处的，学好大数据一方面需要实践，一方面需要从技术原理的角度思考大数据技术之间的关系，这样可以在学习过程中起到事半功倍的效果。

8、您好，请问 JavaWeb 大数据需要学习什么内容？

答：学大数据有两个方向，一个是大数据开发，一个是算法应用，Java 背景的话，大数据开发更有优势一点，特别是大数据平台开发。

9、目前在考虑从前端转大数据方向，想请教下大佬，这条职业路径该如何走比较好？谢谢

答：大数据技术其实是分布式技术的一种，偏后端技术一点；此外还有大数据应用，就是通常说的机器学习、数据挖掘、数据分析。看你的的优势和兴趣，如果编程能力更强，就搞大数据技术方向；如果对业务理解更有洞察力，对数据更敏感，就搞大数据应用。

10、这几年大数据架构变化很快，从 SparkJar 开发到 SparkSQL，再到 FlinkJar、FlinkSQL，再到现在的大数据的实时计算。那么应该如何搭建项目架构？怎么选型比较好？Spark 还是 Flink？怎么不停地学习不同的新的大数据框架？

答：Spark 和 Flink 各有千秋，我觉得技术上差别不大，社区也都比较活跃，选哪个都可以。

各种大数据框架背后的架构原理其实相同的，其实不需要不停学各种大数据框架，或者说理解了大数据技术背后的原理，精通一个大数据框架，其他框架很快就可以轻车熟路了。

11、您好，大数据目前是更关注架构、技术核心还是应用？现在很多行业都号称用上了大数据，一看技术内容其实都差不多，着实有点难以分辨李逵李逵。

答：我觉得没必要分辨李逵还是李鬼，白猫黑猫抓住老鼠就是好猫。用不用大数据其实不重要，重要的还是看有没有价值。如果没用大数据，最后也没产生什么价值，却号称用了大数据，我觉得人艰不拆吧。

如果没用大数据，最后效果还不错，也号称用了大数据，我觉得是扩展的大数据的边界，应该祝贺和学习。

12、大数据目前确实是一个趋势，但是大数据的相关技术或工具非常多，开发者在选型的时候，真的是很复杂和很纠结的一件事情。请问老师，选型方面有那些指导意见或建议吗？

答：首先基于应用场景吧，批处理还是流处理，数据挖掘还是数据分析还是数据查询；其次看看社区活跃度，社区活跃、文档完善、网上资料多的表示应用广泛，使用中出了问题也容易找到答案。

13、什么是大数据的核心价值？对企业有什么益处？一般中小公司有啥好处？

答：大数据的核心价值就是通过技术手段发现数据中的价值。举个不恰当的例子，企业可以通过用户在其平台上的访问记录，进行用户画像，进而识别出价格不敏感的用户进行杀熟，就是大数据对企业的益处。企业当然也可以利用数据为用户和社会创造更好的价值。会不会利用数据的价值跟企业大小没有关系。

14、老师您好，目前公司也在尝试搭建大数据平台，目前遇到的困难是增量数据同步的问题，已有的旧系统表数据设计有的都不会存在时间戳这类字段，如何针对这种数据的来做数据同步呢？另外新系统的设计如何对表结构设计能更加方便？

答：可以定期做全量同步，也可以用 Canal 之类的工具记录增量操作。

15、你好，目前在搞深度学习，对于大数据还属于外行。我想问个外行问题：大数据核心要解决的问题是什么？解决这个问题的方法的理论基础是什么？以及在开展过程中，首先或者关键要面对的问题/困难是什么？

答：深度学习就是大数据的一部分了，内行了~ 大数据核心要解决的就是如何发现数据中的规律，利用好数据的价值问题。理论基础就是统计学，应该是你擅长的。开展过程中，很多困难要解决：如何获取

大规模的数据，如何存储大规模的数据，如何在大规模的数据上进行计算，如何优化计算方法，如何寻找合适的算法发现数据中的规律，如何寻找合适的算法解决现实的问题，如何对数据进行处理便于算法计算。这还是技术问题，法律商业问题更多。

16、您好，想请问一下针对海量数据的快速运算、统计分析，使用什么技术更好一些，数据使用更高效？

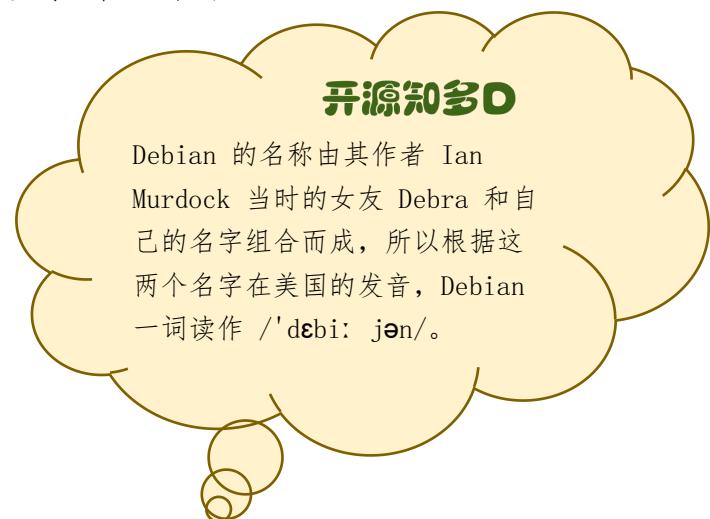
答：Flink, Spark Streaming。

17、李老师好，大数据技术栈太庞大了，Java 出身是不会有一丝优势，还有大数据和机器学习深度学习怎么更好结合呢？

答：大数据技术框架很多和 Java 相关，Java 出身会有一点优势。大数据技术栈是基础技术，机器学习是应用，可以类比 Java（或者 PHP, DotNet）和 Web 开发的关系，很多时候也可以把它们放在一起都叫大数据技术。

18、李老师您好，我最近了解到有基于 Go 语言构建的万亿级大数据平台 Pandora，请问您认为 Go 语言在大数据领域能够大展身手吗？

答：Go 语言内置的高并发特性对大数据处理有一定技术优势。不过对于构架大数据平台而言，扩展性和稳定性更重要一点，任务级的并发相对而言没那么重要。个人观点哈。



行业解决方案

服务客户

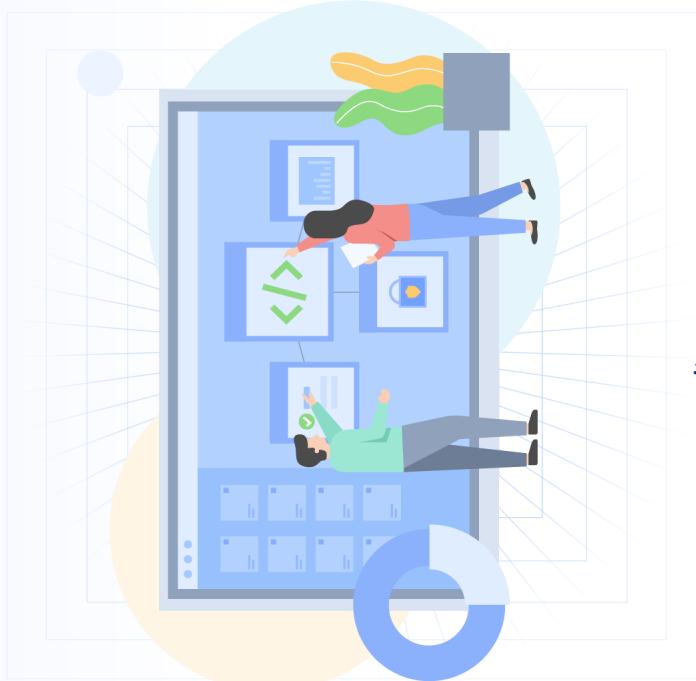
 招商银行 CHINA MERCHANTS BANK	Gitee 助力招商银行从 CC、SVN、RTC 等代码版本管理工具迁移到 Git，并在内部对接了看板平台、链路追踪系统、Jira、Track 等模块，以 Gitee 为中心打通了流水线平台的各个环节，为研发流程提速，优化配置管理体系，推进并优化度量体系建设，提高应用交付效率。
 中国光大银行 CHINA EVERBRIGHT BANK	光大银行选择 Gitee Code 作为其配置管理工具平台替换了原有的 Firefly 软件，利用 Gitee 的能力建立了内部统一的代码管理平台，并配合使用 Gitee Scan 代码扫描软件满足了光大银行开发团队对代码的规范、质量以及安全的特殊化需求。
 宁波银行 BANK OF NINGBO	宁波银行
 海通证券 HAI TONG SECURITIES	海通证券
 招商证券 CHINA MERCHANTS SECURITIES	招商证券
 招商银行 CHINA MERCHANTS BANK	招商银行
 华夏银行 CHINA CITIC BANK	华夏银行
 建设银行 CHINA CONSTRUCTION BANK	建设银行
 光大银行 CHINA EVERBRIGHT BANK	光大银行
 宁波银行 BANK OF NINGBO	宁波银行
 招商证券 CHINA MERCHANTS SECURITIES	招商证券
 华夏银行 CHINA CITIC BANK	华夏银行



私有云

码云 (Gitee) 研发管理平台

敏捷 · 高效



gitee.cn

深圳 北京 上海 郑州 广州 宁波 西安

联系电话: 13352947997
邮箱: client@oschina.cn



联系我们

 招商银行 CHINA MERCHANTS BANK	Gitee 助力招商银行从 CC、SVN、RTC 等代码版本管理工具迁移到 Git，并在内部对接了看板平台、链路追踪系统、Jira、Track 等模块，以 Gitee 为中心打通了流水线平台的各个环节，为研发流程提速，优化配置管理体系，推进并优化度量体系建设，提高应用交付效率。
 中华人民共和国海关总署 GENERAL ADMINISTRATION OF CUSTOMS OF CHINA	宁波银行选择了 Gitee 全流程 DevOps 平台，实现了全行研发管理的端到端 DevOps 全流程管控，支撑 2500+ 开发人员日常研发活动，Gitee 企业级研发效能管理平台，持续改进了研发过程质量，提升研发效能，为宁波银行数字化转型提供了基础平台支撑。
 比亚迪汽车 BYD AUTOMOTIVE	比亚迪选择 Gitee 产品作为全集团统一代码管理平台加强存储、安全访问等能力，成为海关源代码质量检测平台稳定运行的坚强基石。Gitee Scan 是代码准入管理的强有力工具，通过在测试环节中对代码质量的分析进行前置，将源代码质量管理贯穿与融入到软件开发的全流程。
 中国科学院 CHINESE ACADEMY OF SCIENCES	中国科学院
 中国电科 CHINA ELECTRONICS & TELECOMUNICATIONS CORP.	中国电科
 浪潮集团 WEIJIANG GROUP	浪潮集团
 比亚迪汽车 BYD AUTOMOTIVE	比亚迪汽车
 招商证券 CHINA MERCHANTS SECURITIES	招商证券
 华夏银行 CHINA CITIC BANK	华夏银行
 建设银行 CHINA CONSTRUCTION BANK	建设银行
 光大银行 CHINA EVERBRIGHT BANK	光大银行
 宁波银行 BANK OF NINGBO	宁波银行
 招商证券 CHINA MERCHANTS SECURITIES	招商证券
 华夏银行 CHINA CITIC BANK	华夏银行

开源中国是中国最大的开源技术社区，过去十几年一直为国内开发者提供了发现、使用和交流开源技术的公共平台，是开源技术在国内传播推广的重要平台。

- ★ 为金融、政府、军工、制造业、央企等行业提供 DevOps 工具链整体解决方案
- ★ 实战化的敏捷项目管理指导，为敏捷和数字化转型提供有力支持
- ★ 为客户提供业内最佳的研发管理实践和方法论视图
- ★ 深度的项目效率和度量分析，为研发效能提升提供可视化视图

Gitee 是开源中国运营的企业级 DevOps 研发管理平台，已经成为全球第二大的开发者平台，其中包含了代码管理、代码质量管理、持续交付、软件发布平台和项目协作管理等产品，并且支持公有云服务及私有化部署。

Gitee 个人版已经拥有超过 600 万名开发者和 1500 万个代码仓库，Gitee 企业版已经服务于全国 18 万家企业用户，Gitee 教育版已经支持所有 985 和 211 的一千三百多所高校，并获得国家工信部支持成为国家源代码托管平台，承担着国家信息技术产业自主创新的国家使命。



Gitee | 私有云

领先的 IT 研发管理服务提供商

- ★ 为大型研发团队提供高端平台和客户化服务
 - ★ DevOps 全流程研发管理平台及方法论
 - ★ 支持异地多中心统一部署架构
 - ★ 专业的技术支持团队和咨询服务
- | | | | | |
|------|------|--------|------|------|
| 大型金融 | 大型军工 | 大型政企国企 | 教育科研 | 10 + |
| 20 + | 30 + | | | |

