开源软件的商业价值

Erich S. Raymond & Linus Torvalds

”只要有足够的测试员及共同开发者，所有问题都会在很短时间内被发现，而且能够很容易被解决”

Linus Thorvalds 重新描述了一下

“只要有足够多的眼睛，就可以让所有问题浮现”

这个演讲是为这些人准备的

- 想了解开源产品如何运作的人群
- 想创建一个开源项目的人群
- 想参与、驱动、或创建一个开源项目分支的人群

此外，我们还将讨论

- 如何从开源产品上赚钱
- MySQL 和 MariaDB 在商业模式上的不同.

创建一个开源项目的最主要原因

- 解决某些人的个人问题/需求
  - 为了你的工作 (MySQL/PHP)
  - 作为一个研究项目，你业余爱好的一部分(LINUX)
- 为已经存在的闭源项目重新赋予开源首选，以便从开源中收益 (Netscape)
- 想赚钱并且同时做一件好事.
- 想给开源社区一些回馈.

让开发的软件开源是出于人的本性

- 在你解决你自己的问题时，从别人那里获得了免费的帮助和开发成果.
- 提升你的个人声誉（可以获得更好的工作？）
- 因为更低的费用（并且可以重复使用）而使用开源软件

.

- 把你的补丁回馈给社区因而不需要再去维护他们.

当然也有一些人参与开源是因为他们相信开源事业，或者想帮助别人，但是这是一小部分（大部分人依然从开源中获得收益）.

在开始一个新的开源项目之前

- 检查一下是不是已经有类似的活跃项目.
  - 参与（并且改进它）其中总是比创建一个新的项目或者创建一个新的分支好!
- 如果只有很老的已经死亡（不活跃）的项目，研究一下为什么他们失败了，并且从他们的错误中吸取教训.
  - Sourceforge, Github 以及 Launchpad 上到处是死亡的项目.
- 找一个希望和你一起工作，去定义这个项目适用范围的公司或用户组.
  - 你需要尽快有用户把这个项目用到生产环境!

开源不仅仅是软件

- 开源最重要的部分是创建一个活跃的社区!
- 你还需要(与社区相互作用):
  - 网页 (需要有人设计这些)
  - 一个社区或者知识库, 邮件列表, bug 系统.
  - 文档 & 本土化
  - 打包, 编译系统, 镜像(提供下载)
  - 源码库

在刚开始的时候你可以使用 github 或者 launchpad 来保存你的工程, 以及用 Open Build Service 来编译你的系统, 但是随着时间的推移你会需要更多的控制权, 然后就需要你自己来做了。

你需要一个优秀的团队和活跃的社区

- 一个特定的活跃的领袖（Linux）或者一群受到社区尊敬的活跃的领导者（MariaDB / PostgreSQL）.
- 一群优秀的开放的社区成员维护这个社区.
- 一群遵守良好的“旧时代”代码风格标准的人，并且热衷于教导其他人和参与社区.
- 围绕着你的产品有一群活跃、热情的用户，以及开发者社区.
- 在日常生产环境中使用产品的开发者.
- 为自己的需求而必须扩展产品的开发者.

通常产品质量由技术领袖决定

透明是保持成功的关键

- 一个开放的开发模式（所有的邮件列表，讨论架构，决策是开放的）.
- 对于如何把事情做完，以及什么事情要做有清晰的指导方针.
- 清晰的授权和商业模式.
- 海量的文档.
- 要清楚地了解你的路线图和发布计划.
- 有良好的/开放的补丁 review 流程来严格保证质量.
- 让你的计划透明，并且让用户去影响它们.
- 让你的 bug 开放，并且让人们知道何时可以解决.

履行自己的诺言！

与你的社区保持联系

- 参加会议并且讲讲你的产品.
- 听听什么是用户需要的，区分是你来做，还是帮助用户自己来做.
- 让人们更容易联系到你并且找到关于产品的信息!
- 确保大部分论坛和邮件组中的问题被解答了.
- 了解编写代码的人’拥有’这些代码
- 不要期望社区产出任何代码.

认识到从用户转到客户的路会很长，并且不要让短期的收益打乱了你长期的制胜策略

成为好的开源公民

- 公开你使用的外部代码 – 分享声誉
- 为贡献者提供快速的反馈（bug 的报告以及修复代码，代码优化，思路）
- 在论坛上帮助别人(跟你代码相关，以及你了解的代码相关的)
- 在邮件组和论坛里保持比较高的道德水准.
- 不要说竞争项目的坏话.
  - 基于事实的对比是可以的.
- 成为一个好的开源公民可以为你和你的产品积累信任，并且随着时间的推移从中受益.

在后期竞争中创建一个获得支持的社区是非常困难的 – 你应该从第一天开始

你需要把你的产品拿出来并让人使用!

- 早点发布 - 经常分发, 二进制包 + 源码.
- 每个发布的目标, 包括初期测试, 应该让 bug 少到足以让想尝试的人可以用到生产环境.
  - 人们开始只会少量使用你的产品，只有当他们信任你的产品，并且比转移到其他产品更容易地扩展他们的需求时，才会开始扩大使用你的产品.
- 与云提供商合作，让（想测试的人）可以在那获得产品（的体验）.
- 使用一个靠谱的开放工具集来开发(Sourceforge, Launchpad 或者你们自己的）让其他人更容易地参与你的项目.

做开源的生意

开源是一种哲学和发展模式.

开源不能保证你会获得足够得钱去开发或支持你的产品.

不同种类的开源产品

- 社区开发的产品.
  - 通常较大项目是许多公司开发用来解决他们商业需求或嵌入他们产品的项目.
    - Linux, PHP, Apache, MariaDB
- 公司作为从社区获得更多发展的工具而发布为开源的产品.
- 小团队和小公司开发并驱动一个开源产品作为他们主要提供的服务.
  - 这通常有一段很艰苦的时间要和闭源软件竞争(没有足够的钱做全职开发).
    - MySQL, JBoss, Wordpress, 大部分开源软件

创建一家公司时需要考虑的问题

- 你计划建一个虚拟的公司吗（“没有办公室”）?
- 你是在创建一家平等的公司吗? 公司应该被员工所拥有吗（黑客的商业模式)?
- 你是想做服务还是想开发产品?
  - 如果开发产品，你选择什么样的授权?
- 你计划拥有一个庞大的社区还是只和一些大公司合作?
- 你计划融资吗?
  - 如果是，那么你需要一个 ’退出’ 方案.

选择什么样的商业模式?

- 服务型企业
  - 主要提供服务（支持，培训，顾问）
  - 估值2 x 收益
- 软件企业
  - 提供授权，软件作为一种服务（SAAS），订阅服务
  - 估值10 x 收益+ X * 用户数

公司的终极目标是什么

- 在市场销售
  - 对产品/员工不可预知的未来
  - 公司所有者通常急于追求很高的利润
- 股票市场上市
  - 原始所有者依然可以部分控制公司
  - 收益稍微有点不确定（所有者通常只能在 6 个月后卖出股票）
- 创始人、员工持股（以及投资者）
  - 稳定可预见的未来
  - 所有者获得股息，员工获得奖金
- 创立一个开源基金会

为什么要开源?

- 产品更迅速的传播（更多用户）
- 在其他地方（开源社区）完成部分开发(更低的开销)
- 获得更多的测试和更多的 bug 报告(更高的质量)
- 可能在’非关键业务’方向上完成更多开发（更有用的产品）
- 更容易找到好的开发人员，合作者和客户

上述内容意味着更多的市场认可、反馈、指引、业务、合作伙伴和销售机会以及一个强大的商标.

通常开源项目会得到 更多反馈 和更多 bug 报告，相对于闭源项目.

用户信任一个开源厂商的原因

- 开源厂商更值得信赖，因为他们靠信任生存
- 没有厂商的捆绑. 你使用这个产品的投资是安全的， 即使:
  - 厂商歇业了
  - 厂商出人意料地彻底改变商业条款
  - 厂商停止支持你正在使用的产品版本

  如果这发生在一个很流行的产品上，会有人出来维护这些代码。
- 可以检查产品代码， 隐藏的陷阱门 风险更低

开发者使用开源厂商的好处

- 很容易地访问、查看和使用代码
- 自由地检查和更改代码，以满足你的业务需求，bug 修复或移植到其他任何系统.
- 自由地寻找一个人做上述事情
- 自由地使用（阅读，编译和修改）这些代码并且在开源环境中重新分发这些代码.

使用开源对大企业/国家的好处

- 你可以开发你自己的基础架构让软件按照你的意思来调整(语言，唯一的需求)
  - Facebook，Google
  - 巴西，冰岛
- 你可以从内部了解（产品）如何为你的业务所用
- 不依赖于外部供应商（对于小事情).
- 没有授权成本；低到很低的持有成本!

- It's in your interest to collaborate with the original community for long term sustainability
  - 从长远来看，（维护）一个完整的分支是很昂贵的!

什么时候开源?

如果你可以围绕开源创建一个可持续发展的商业模式，专有软件供应商将很难与你竞争

如今在很多商业部门, 闭源解决方案越来越难卖.

也用不着惊讶，为软件选择一个正确的授权会显著的影响你的战略，例如：

- 如何与你的用户社区合作
- 如何构建你的商业

如何选择一个开源授权?

- 关键问题:

- 什么是你开源产品的商业理念?
  - 服务，订阅 或者 授权?
- 当使用、修改以及可能地重新发布（你的产品）时，
  你为代码保留哪些权利?
- 你想要什么样的产品社区?

主要的开源授权
(简单介绍)

- Public domain
  - 给用户做任何事情的自由，包括改变你的授权和声称他们写了这个软件.
- BSD/Apache
  - 提供给用户完全免费的使用，但是需要在代码中保持授权.
- LGPL
  - 提供用户免费使用，但是如果他们分发其修改，需要把修改在 LGP 下分发。
- GPL
  - 提供用户免费使用，但是如果他们分发它（软件），则需要公布修改的部分并且把代码置于 GPL 之下
- AGPL (Optional addition to GPL V3)
  - 免费使用，但是用户需要公开代码，即使代码没有被分发（像 web server）

使用开源的商业模式

- Open-Core 模式 - 有一个开源的核心和在其之上售卖的闭源功能（例如 SugarCRM）
- Dual Licensing Model(双授权模式)- 一个产品/项目有 GPL 一样会传播的授权，以及一个商业闭源授权（例如 MySQL）
  - 另一个选项是 **“商业代码”** 或者延时开源
- Services Models(服务模式) - 你可以获得一个开源项目的产品化版本，并且为支持服务付一些费用，或者捐助一些功能. 你通常也可以为培训、功能需求等付费
- 订阅（通常混合了技术支持、延长产品生命周期并保证更新）
- 创建一个非营利基金会，以资助产品的发展(FSF, MariaDB 基金会)

Open Core

- 大概是目前企业第一次尝试做开源最流行的方式.
- 正是 Oracle 正在对 MySQL 做的，以及 EnterpriseDB 对 PostgreSQL 做的
- 不是一个开源的商业模式，因为它使用闭源元件，并且大部分开源开发者期望的好处都没了：
  - 你不能改变，修改，port 或重新分发代码
  - 你被锁定在一家供应商
- 你也许可以围绕产品创建一个小的开发者社区，但主要是不需要闭源扩展功能的人群.
- 对于社区开发者而言，可能出现的“最差”情况是当订阅到期时，在订阅授权下使用的开放核心或闭源代码停止工作了

双轨授权

- Ghostscript 第一次使用. MySQL 是第二个使用的.
- 只有当你对所有代码拥有全部权利时才能使用.
- 让同样的代码有两个授权, 例如 GPL 和普通商业闭源授权.
- 不能使用 GPL 的公司(因为他们不想公布他们的代码) 可以购买闭源版本.
- 只在基础架构上工作良好, 更容易嵌入的产品, 例如调用库和数据库.

商业代码

- 不是一个开源授权，但是提供社区类似的优势.
- 可以得到源码，任何人有权拷贝，修改&发布，但是在某些特定的条件下不可以在商业上使用。
- X 年之后代码自动转为某些开源/自由的授权. 日期在所有的代码文件中明确地写明了，以避免错误的理解
- 比 Open Core 好，因为没有了“单个供应商” 的问题并且代码最终会成为自由的代码.

商业代码

- 发布在 http://timreview.ca/article/691
- 综述在 http://monty-says.blogspot.com/
- Slashdot 的讨论
  http://developers.slashdot.org/story/13/06/26/1552215/monty-suggests-a-business-friendly-license-that-trends-open

- Unfortunately the second link is blocked in China, so you have to access it by a VPN.

销售许可的重要性

- MySQL 从来不能没有授权许可.
- 很难有公司为支持或（付费）开发买单
  - 公司用开源是期望免费
  - 例外是公司也在参与开发开源项目
- 订阅是不错的服务，但是也很难卖
- 对项目而言授权许可是 “free money”
  - 你需要一个实体对整个项目持有版权或者使用 SAAS (Software as a service)
  - 你的项目必须能够进行双轨授权
  - 你的项目是一个基础设施项目，通常嵌入在其他地方.

MySQL 和 MariaDB 之间商业模式的不同

- MySQL Ab 通过服务（支持，培训，付费开发）以及**双授权**挣钱. 设计能够接纳投资者.
- MySQL 主要由一个公司开发.
- MariaDB 有像 SkySQL 这样的公司围绕其提供服务.MariaDB 基金会获取捐助来驱动社区开发.
- 很多 MariaDB 的开发是 SkySQL 完成的，同时也有很多社区的开发.

The end