

# 开源软件概述

周明辉

[zhmh@pku.edu.cn](mailto:zhmh@pku.edu.cn)

[minghuizhou.github.io](https://minghuizhou.github.io)



北京大学  
PEKING UNIVERSITY

# 纲要

- 开源软件的定义
- 开源软件的历史和现状
- 开源生态的结构和治理
- 开源软件生态的研究

什么是开源软件？

# 开源软件

- 开源软件是一种**源代码可以自由获取**的计算机软件。
- 发布开源软件需要附带**开源许可证**：
  - 开源许可证是对开源软件的知识产权进行规范和约束的法律合同：甲方是软件版权所有者，乙方是软件用户。
  - 软件的版权持有人在开源许可证的规定之下允许用户使用、修改以及分发该软件。许可证定义了开源软件用户的权利和义务。
  - 软件知识产权：版权，专利，商标。
- 开源许可证通常符合**开源的定义**的要求。

# 开源的定义

- 开源不仅意味着可以访问源代码，开源软件的分发条款必须满足下述条件：

- (1) 自由再发行；
- (2) 程序必须包含或方便取得源代码；
- (3) 许可证必须允许更改和派生程序；
- (4) 保护作者源代码的完整性；
- (5) 无个人或团体的歧视；
- (6) 无领域歧视；

.....

- (8) 许可证不能限制其他软件；
- (10) 许可证需要是技术中立的。



<https://opensource.org/osd>

# The Open Source Definition

-- <https://opensource.org/osd>

## Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

### 1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

### 2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost, preferably downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

### 3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

### 4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

### 5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

### 6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

### 7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

### 8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

### 9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

### 10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

Last modified, 2007-03-22

# 开源许可证需要符合开源的定义的要求

- 开源社区存在大量不同类型的许可证，OSI 认证的开源许可证已有 126 个<sup>[1]</sup>，在开源项目中的使用率 > 80%。
  - 宽松型，Apache/MulanPSL，关键特点：分发可以闭源
  - 传染型，GPL/MulanPubL，关键特点：分发必须开源

木兰宽松许可证(宽松型许可证): <http://license.coscl.org.cn/MulanPSL2>

木兰公共许可证(传染型许可证): <http://license.coscl.org.cn/MulanPubL-2.0>

均采用中英文双语表述，以中文为准

【1】 [https://spdx.org/licenses/?ivk\\_sa=1024320u](https://spdx.org/licenses/?ivk_sa=1024320u), 2021/03/07

# 木兰许可证系列：中英文双语

## 大背景：

本土企业需求，中文开源社区的发展和成长需求。

## 具体需求：

中文解释权，少/无风险的开源项目和产品的发展需求。

## 第一个获得OSI认可的本土开源许可证：木兰宽松许可证MulanPSL2

5 Aug 2019

MulanPSL-  
1.0 发布

14 Feb 2020

MulanPSL-2.0  
OSI 认证

April 2020

Apache 基金会宣布  
Apache2.0 跟 MulanPSL-2.0  
兼容

23 March 2021

OSG-Japan 翻译并发布  
mulanPSL-2.0 日文版

国内社区广泛支持



**gitee**



确实社区



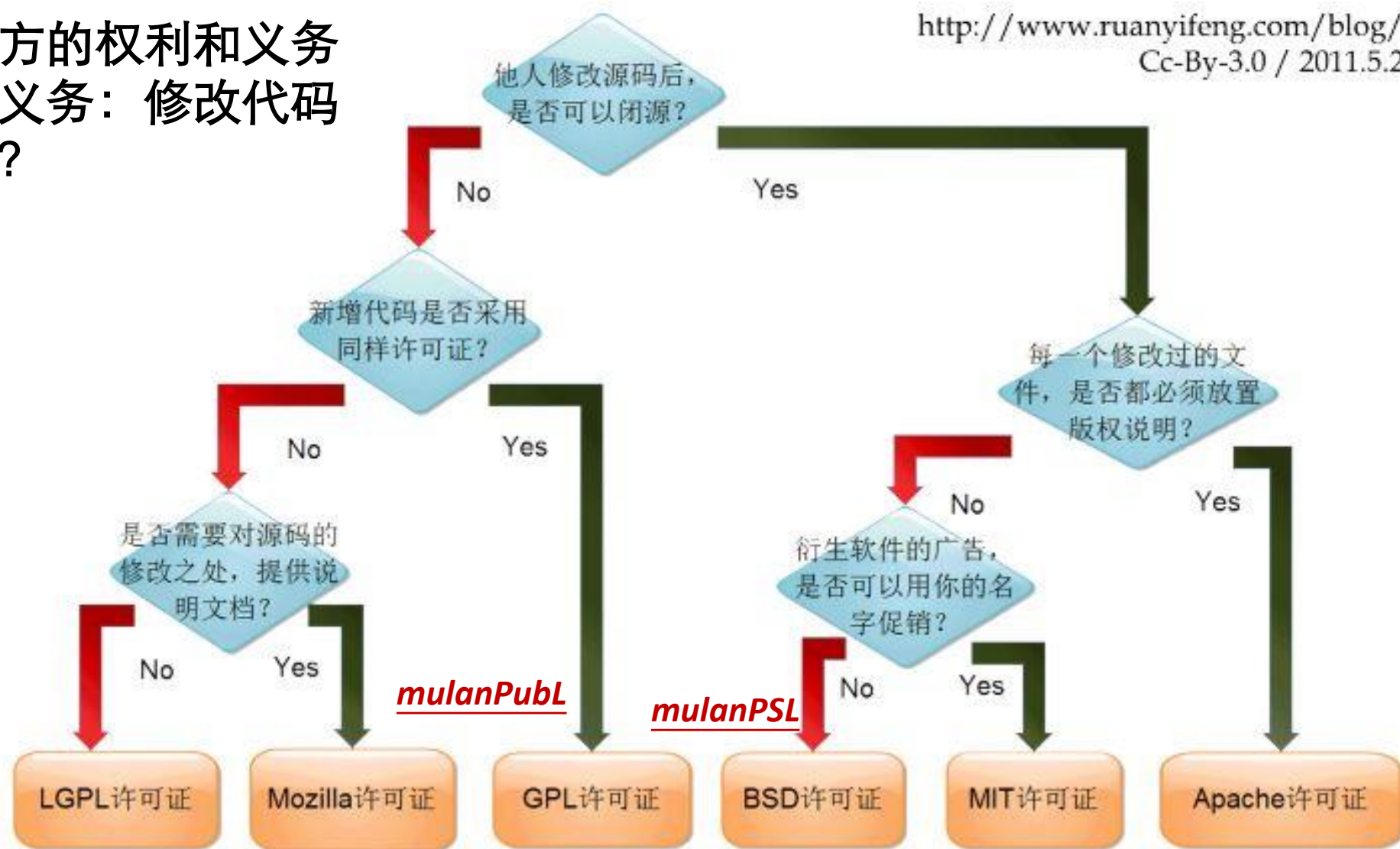
**iHub**



100k+ 项目采用了 MulanPSL-2.0，覆盖云计算、大数据、AI、OS 等，例如，openEuler，openGauss，香山(RISC-V处理器)。

# 不同开源许可证的使用

- 合同：定义乙方的权利和义务
- 主要区别在于义务：修改代码是否需要开源？



# 开源软件的历史和现状

# 开源软件的历史：创新的历史

大公司参与和主导开源社区，**混合模式**兴盛；开放众包、群智**创新**，开源延展到硬件、教育等领域

**开源 (Open Source)** 一词在1998年2月3日提出，在1998年开源峰会达成共识，开源**正式成为旗帜**，开源运动迅猛发展，影响力迅速扩大

Bill Gates推动**软件商业化**，软件单独售卖，源码不再免费可得；R.M. Stallman发起**自由软件free software**运动；各种开源社区兴起

软件商业化尚未出现或尚未成熟，软件跟硬件搭售，随源代码发布，学术共同体**自发开放和共享源代码**

多元生态发展阶段  
(商业模式多元化)

原始萌芽阶段



1980

多家争鸣阶段



1998

共识达成阶段



2005

融合发展阶段

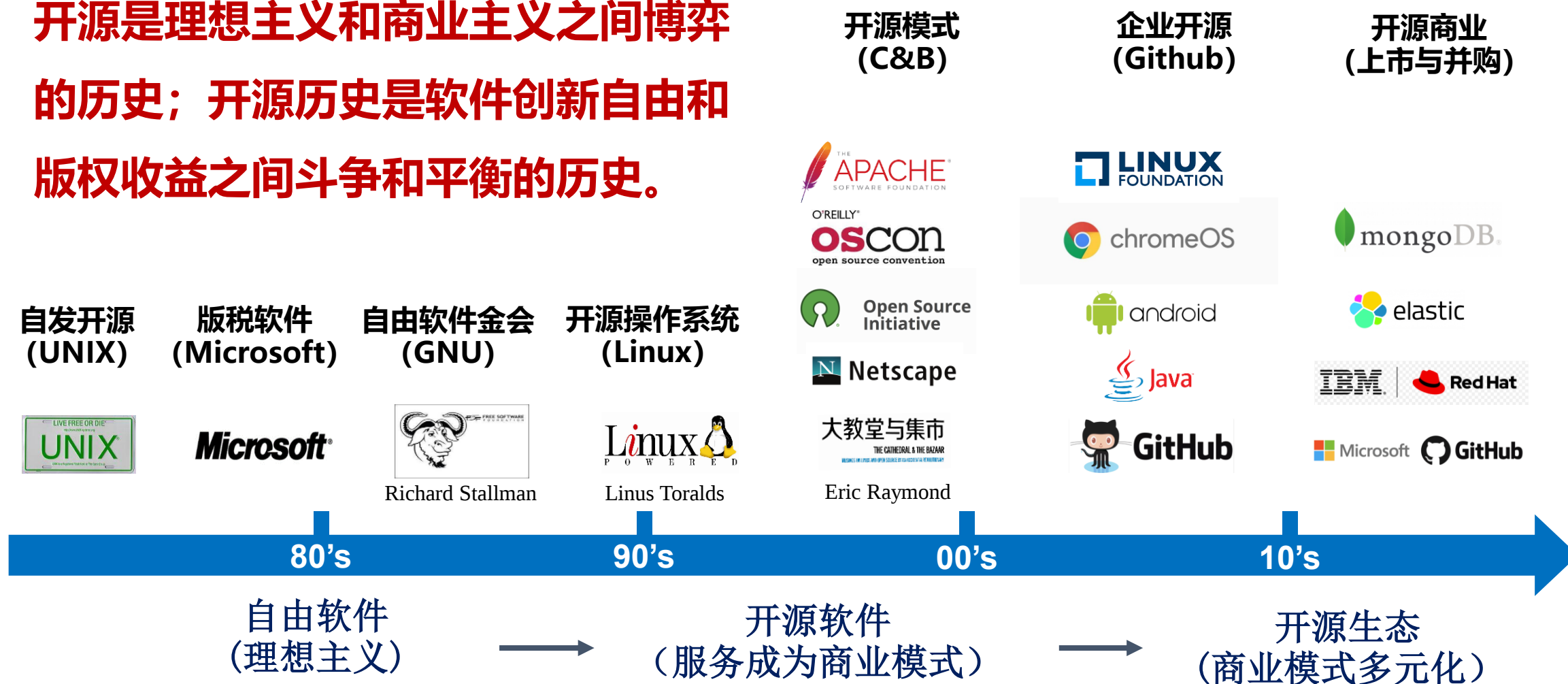


2021

软件从跟硬件搭售到单独售卖，是**专有**推动了创新；  
从商业许可发展到开源许可，是**共享**推动了创新；  
从开源发展到混合模式，是**共享/专有的融合**创新

# 开源软件的历史：自由和商业博弈的历史

开源是理想主义和商业主义之间博弈的历史；开源历史是软件创新自由和版权收益之间斗争和平衡的历史。



# 理想主义是开源的源起： 英雄们的历史



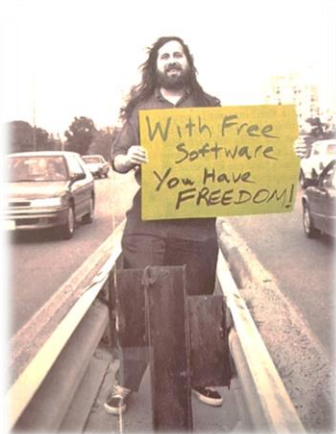
## 0. 1970年 启蒙年代

- Ken Thompson 和 Dennis Ritchie 缔造 Unix



## 1. 1984年 自由软件运动

- AT&T 将UNIX商业化
- Bill Gates的微软发布DOS和Windows
- Richard Stallman 发起GNU项目

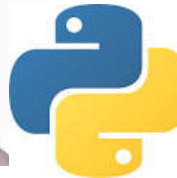


## 2. 1991年 Linux 内核系统

- Linus Torvalds 发起并发动社区
- 完善了GNU项目

## 3. 1998年 开源峰会

- Netscape 宣布开放Navigator 浏览器的源代码
- “开源软件”替代“自由软件”，并广泛传播



## 4. 著名开源软件项目

- Larry Wall, perl, 1987
- Guido V. Rossum, Python, 1991
- R. Ihaka & R. Gentleman, R, 1993

自由软件

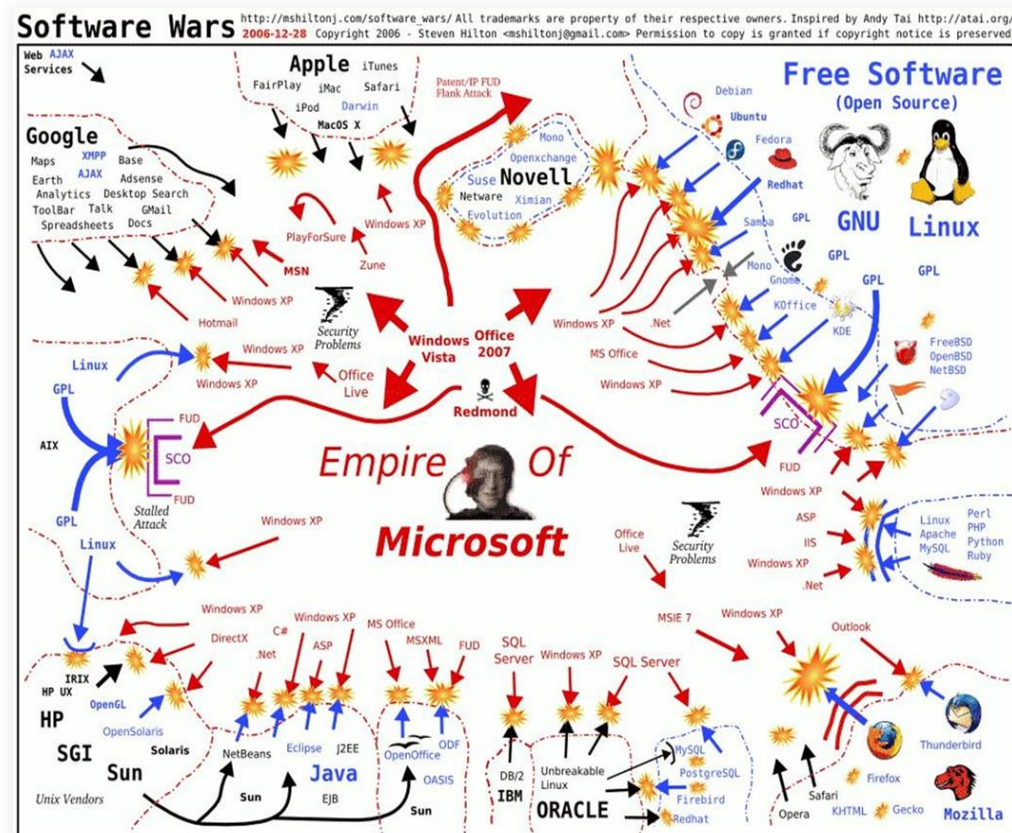
开源软件的诞生与发展

# 理想主义是开源的源起

□ 在每一个呈“垄断”性的软件领域，都有对应的开源版本

- 操作系统：Linux vs Windows
- 数据库：MySQL/PostgreSQL vs Oracle
- 浏览器：Firefox vs IE
- 办公系统：LibreOffice vs MS office
- AI大模型：ChatGPT 4 vs LLaMa 3

.....



# 开源软件的商业模式

| 商业模式                               | 简介                                                                                                                                           | 特点                                                                                                                                   | 代表企业                                                                                                                                                                       |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Support 支持服务</b>                | <ul style="list-style-type: none"><li>用户只需为技术支持及咨询服务买单</li></ul>                                                                             | <ul style="list-style-type: none"><li>人工外包作，利润率偏低</li><li>工作可复制性低，scale较难</li><li>客户转换率低，通常&lt;1%</li></ul>                          |      |
| <b>Hosting 托管</b>                  | <ul style="list-style-type: none"><li>供应商将其开源软件作为服务托管在云上，通过收取每月/每年的托管和服务费获利</li></ul>                                                        | <ul style="list-style-type: none"><li>该模式成为了部分云厂商打包开源项目赚取利润的途径</li></ul>                                                             |      |
| <b>Restrictive Licensing 限制性许可</b> | <ul style="list-style-type: none"><li>通过提供一个带有稍带限制的开源许可证来激励使用者进行付费</li></ul>                                                                 | <ul style="list-style-type: none"><li>许可证定义模糊，需要法院判决</li><li>部分公司禁止使用该商业模式下的开源软件</li></ul>                                           |      |
| <b>Open-core 开放核心</b>              | <ul style="list-style-type: none"><li>该模式下的大部分代码是开源的，而少数代码（针对企业用户）是专有的，需要收费</li><li>专有部分可以打包成与开源基础部分连接的单独模块或服务，或者在分叉版本中分发</li></ul>          | <ul style="list-style-type: none"><li>该模式可以避免云厂商打包开源项目赚取利润</li><li>难以拿捏开源范围的尺度</li><li>很难将代码中的开源与专有特性完全分离</li></ul>                  |      |
| <b>Hybrid Licensing 开放核心+混合许可</b>  | <ul style="list-style-type: none"><li>最新的模式，在开放核心基础上进行了改进</li><li>混合许可在同一个代码库中混合了开源代码和专有代码</li><li>用户可以选择只使用开源代码，或者同时使用开源代码和专有软件代码</li></ul> | <ul style="list-style-type: none"><li>代码在同一个代码库中，使管理和开发变得更容易</li><li>允许用户方便升级到付费模式</li><li>允许外部社区（比如GitHub）成员对专有软件功能模块进行改进</li></ul> |   |

目前，开放核心+混合许可逐渐成为主流的商业模式，其原因在于：

- 开源软件商能够轻松管理代码库而不必拿捏开源的尺度
- 客户能够方便的从免费开源模式切换到付费模式（不需要额外部署，也不需要和销售沟通）
- 外部的开源社区也能对专有付费模块代码进行改进，降低了开发成本

来源：云启资本

# 国际国内开源发展的对比



## 平台

GitHub开发者数量超过1亿，年增长26%；  
公开仓库数量2.84亿，年增长9800万

Gitee共1200万开发者，年增长20%；  
总仓库数量3000万，年增长500万

## 项目

GitHub 上top 20活跃AI项目中，一部分是个人开发者拥有的；  
GitHub 上80%贡献来自私有仓库

国际top10活跃项目无中国项目，top50仅有两个；  
国内活跃项目多由大厂支持

## 开发者

GitHub上印度、巴西、日本开发者增速最快；  
法国在政府推动吸引科技初创企业后，实现开发者数量22%的年增长

GitHub上中国开发者数量排第三，2022年被印度超过，预计2028年将被巴西超过；  
在生成式AI领域，中国开发者数排第九，略高于第十的新加坡

### 共同趋势：

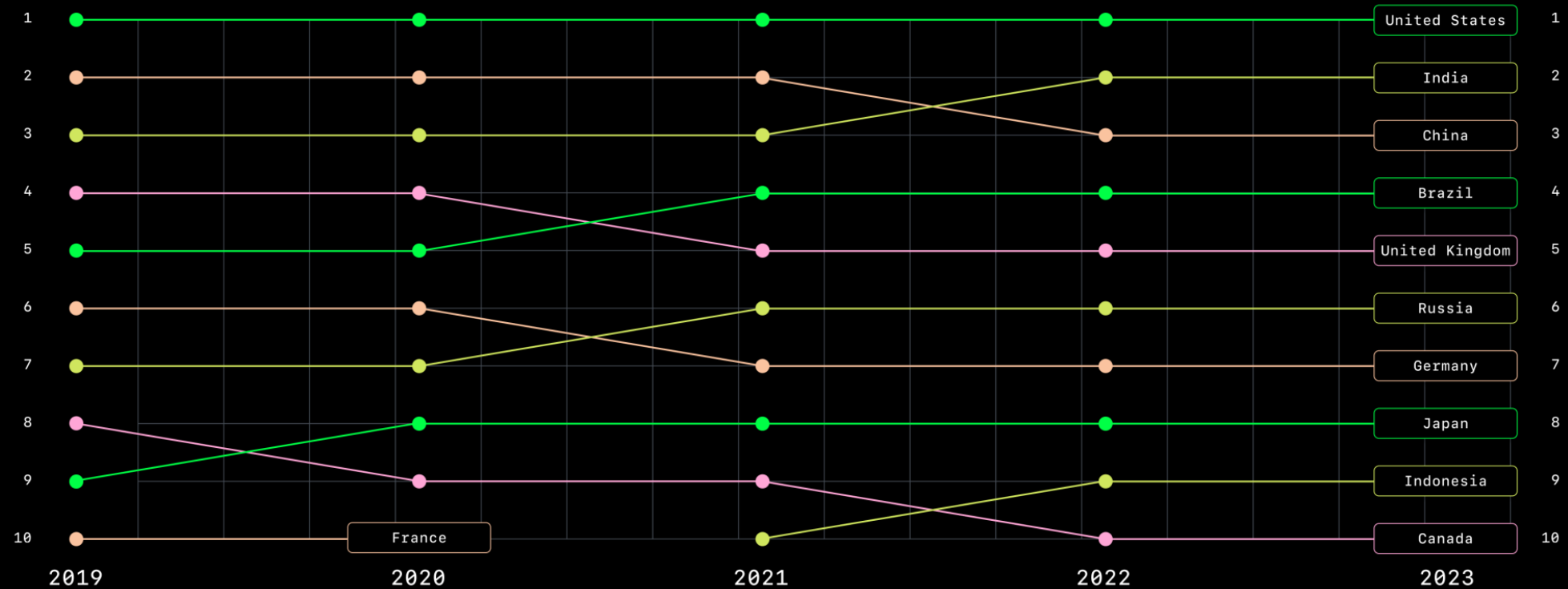
- 开源版本控制系统Git 被93% 的开发人员用于构建和部署软件
- 由个人开发者自发组建的开源组织影响力在逐步提升
- 开发者正在大量使用生成式人工智能进行开发，云原生构建多模态AI应用

数据来源：

GitHub. 《Octoverse: The state of open source and rise of AI in 2023》. <https://github.blog/2023-11-08-the-state-of-open-source-and-ai>.

Gitee. 《2023 中国开源开发者报告》. <https://talk.gitee.com/report/china-open-source-2023-annual-report.pdf?fr=wx>.

# The top 10 developer communities on GitHub from 2019-2023

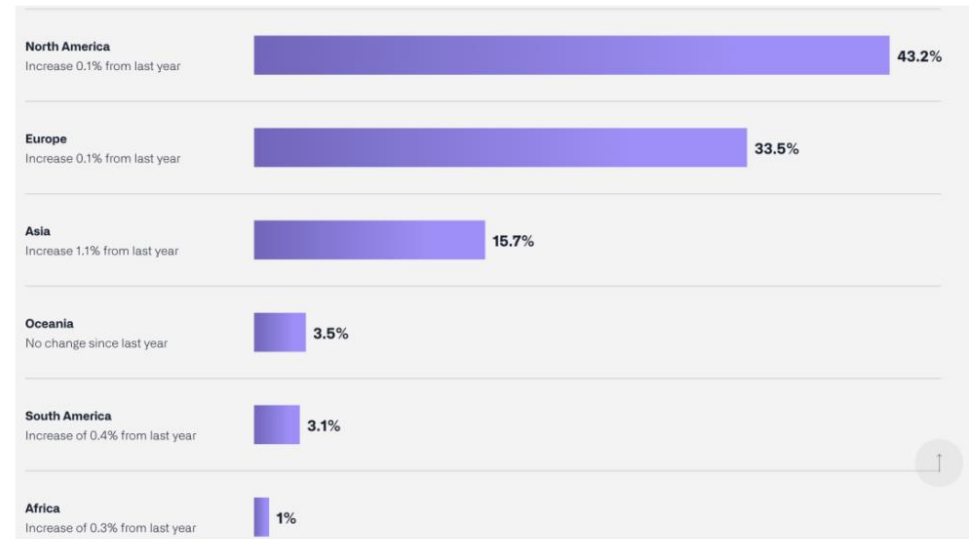


# 中国的开源

- 受益开源，贡献国际项目，开源自有项目
  - 操作系统：openEuler, openHarmony,
  - 数据库：TiDB, OceanBase, openGauss,
  - AI框架：paddlepaddle, mindSpore,
  - 应用技术：Vue.js, echarts
- 开源基础设施逐步发展和成熟
  - 代码托管平台：Gitee, code.china, Gitlink
  - 第一个开源基金会：开放原子开源基金会
  - 第一个OSI approved的中英文开源许可证：MulanPSL-2.0
- 开源商业化
  - 一批开源软件独角兽或准独角兽创业企业，如PingCAP等
- 蓬勃发展的开源社区和促进会
  - 学术共同体和民间：CCF开源发展委员会、木兰社区、开源社
  - 企业：开源雨林、腾源会、蚂蚁开源
  - 部队：红山开源社区

GitHub 2021年度报告发布：中国755万开发者排名全球第二！

新智元 新智元 2021-11-17 12:49



# 中国的开源：机会和挑战并存

- 国家战略： 开源创新被列入十四五规划和2035远景目标
  - “支持数字技术开源社区等创新联合体发展，完善开源知识产权和法律体系，鼓励企业开放软件源代码、硬件设计和应用服务。”
- 本土的优势： 市场大（多样化需求）、人才多（人才储备强）、产业（供应链）体系完整。
- 存在的不足：
  - volunteering文化缺乏，用户创新缺乏，协作共赢欠缺，难以形成生态。
  - 冗余建设很多，以竞争、主导为出发点的“开源”仍然不少。
  - 缺乏商业模式。

# 开源生态的结构与治理

# 一个案例：开源操作系统

- openEuler：旨在通过开源社区构建一个支持多处理架构和多芯片的生态体系。

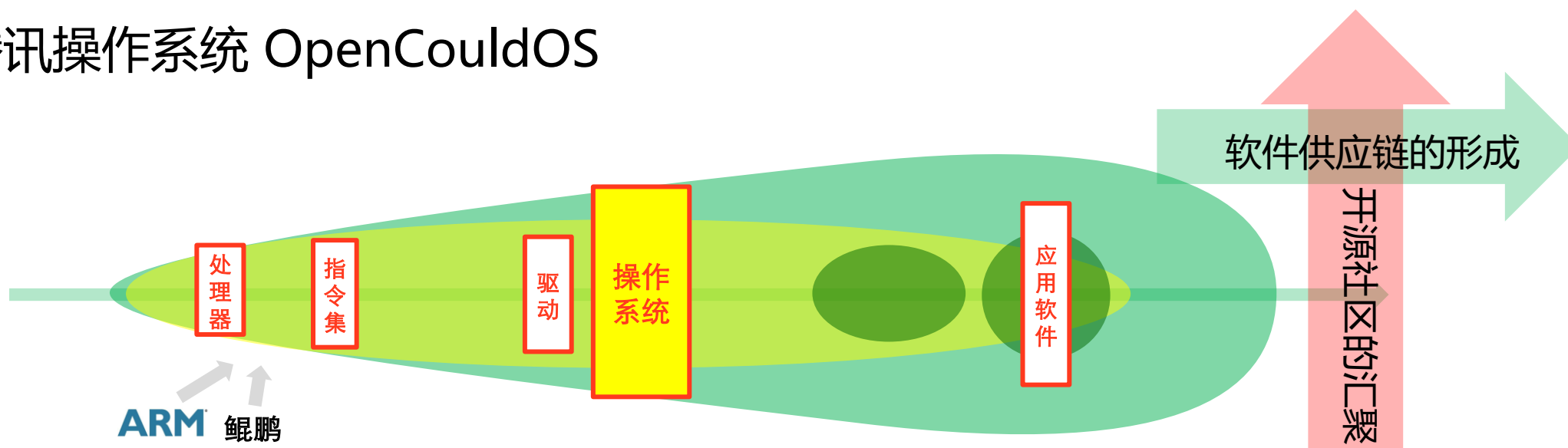
- 软硬件供应链：鲲鹏，Linux kernel，openEuler，基于openE的OS发行版  openEuler

- 开源社区：华为，Euler OSV，个体用户，.....



- 龙蜥操作系统 AnolisOS

- 腾讯操作系统 OpenCouldOS

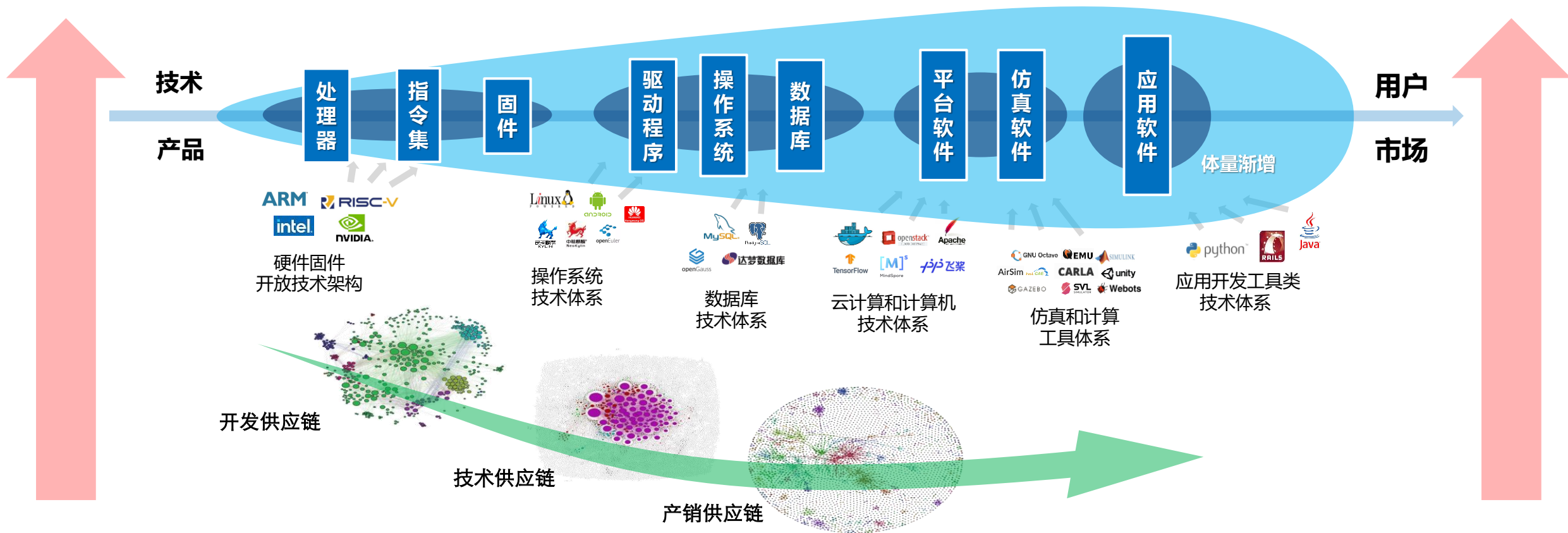


# 开源项目运作的核心框架

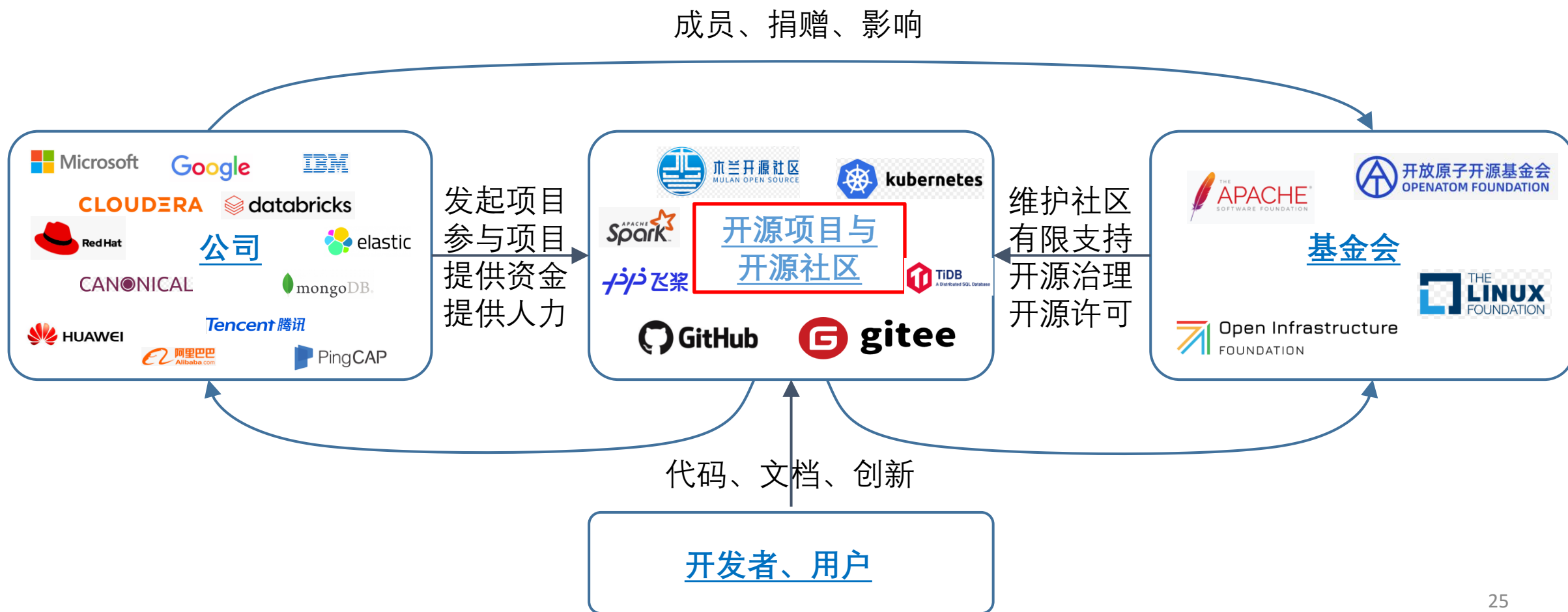
- **代码托管**：选择许可证，选择托管平台，（选择基金会），开放源代码
  - 软件的生态定位
- **社区使能**：定义系统化贡献指南，经营多样化社区
  - 定义指南：CONTRIBUTING.MD，为新手任务打标签
  - 运营：生态的形成需要有三类角色：软件/产品/服务的生产者、提供者和消费者
- **版本发布和迭代**：新特征开发和发布、老版本持续维护的周期
- **项目可持续**：有用户，有社区，有市场

# 开源软件项目的核心挑战：生态的形成和持续

- 开源成功的核心表征是生态，生态有两个维度：
  - 水平横向：软硬件全栈，各类供应链关系网络
  - 垂直纵向：项目社区的汇聚、协作、可持续



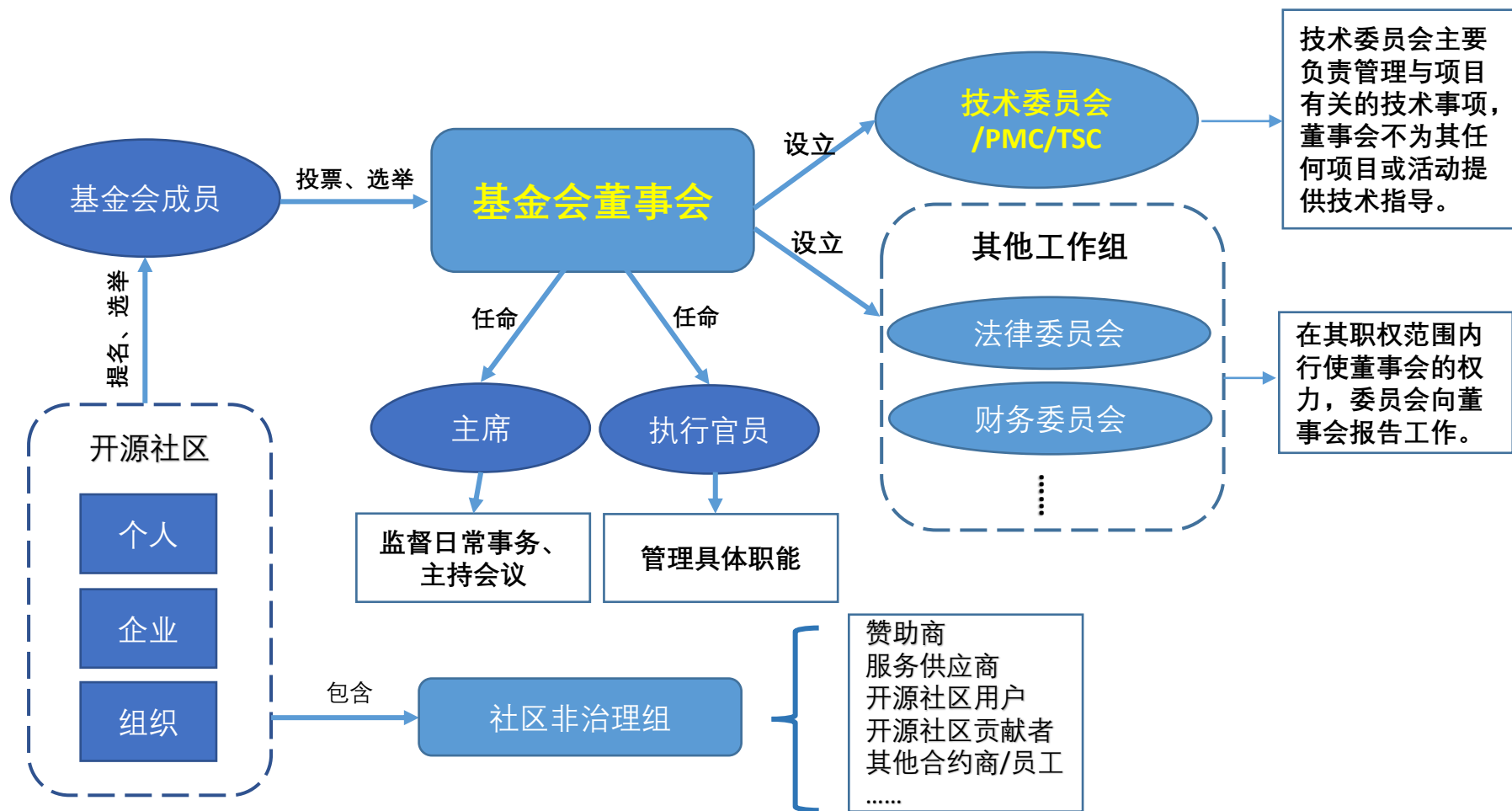
# 开源生态的基本结构与要素



# 开源基金会的角色

- 任何人都可以在Internet上发布项目的源代码，但是要构建一个可持续的项目社区，仅靠代码是不够的，而基金会就扮演了“社区看门人”的角色。
- 开源基金会作为管理和推广开源项目的非盈利机构，为开源项目提供基础设施、活动、培训以及法律、商业、技术等服务。

# 开源基金会的组织架构



# Apache基金会(Apache Software Foundation, ASF)



成立于1999年

涉及大数据、云、搜索和CMS、DevOps和构建管理、物联网和边缘计算、机器人和深度学习、服务器、Web框架等领域。



一个项目要进入Apache基金会，需要按照基金会的要求进行“孵化”。Apache孵化器就是为那些想要进入Apache基金会的项目提供服务的，孵化一般需要一年半的时间，满足一系列质量要求之后方可毕业，通过孵化毕业的项目要么成为顶级的ASF项目，要么成为其他顶级项目的子项目。



# 中国计算机学会开源发展委员会：CCF ODC



## 平台研发

以开源模式打造开放共建的CCF开源服务平台，包括开源项目开发孵化平台、开源人才培养服务平台，为产教研深度融合提供全方位支持

## 社区建设

探索建立CCF开源项目孵化机制，为产学研领域成果孵化提供全方位服务  
推进开源人才培养，形成产学研联动培养模式

## 人才培养

联接汇聚科教资源、产业资源和社会资源等，形成产、教、研联动的人才培养模式，推进开源课程资源建设与共享，培养优秀创新人才

# 木兰开源社区

木兰开源社区：对国家科研成果开源托管、产业界中小微开源项目开展孵化运营

● 汇聚了专项中22个项目的170余项开源成果

|                                                                                                                       |                                                                                                                      |                                                                                                                                 |
|-----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <div>首页 开源软件 资讯 活动 开源许可证 软件工程服务 社区内容推荐 木兰确实托管平台 大家都在搜...</div>                                                        |                                                                                                                      |                                                                                                                                 |
| <div>开源社区信息监控与推荐系统 (Primrose) - 东南大学<br/>开源生态系统监控与推荐平台<br/>云计算 MulanPSL-2.0 python</div>                              | <div>智能云开发辅助机器人 (Cosine Robot) - 东南大学、南京大学、山东...<br/>贯穿软件开发全生命周期的辅助机器人<br/>MulanPSL-2.0 云计算 python</div>             | <div>AppPop: 软件流行度预测工具 - 西北工业大学<br/>针对影响软件流行度的多种因素建模, 预测软件的流行度<br/>数据处理 MulanPSL-2.0</div>                                      |
| <div>X-ADMM: 软件情景自适应工具 - 西北工业大学<br/>根据终端设备的情景信息, 对于软件模型通过剪裁、分割等方法进行动态调整, 从而实现软件与设备的自适应...<br/>数据处理 MulanPSL-2.0</div> | <div>CompEvo: 构件级在线更新系统 - 南京大学<br/>面向Java web应用的可独立为软件提供服务的构件级级在线更新系统<br/>云原生 MulanPSL-2.0</div>                     | <div>Hanabi: 软件缺陷自动修复工具软件 - 北京大学<br/>给定一个有缺陷的程序及揭示缺陷的测试, 本项目全自动化地尝试修复缺陷<br/>数据处理 GPLv3</div>                                    |
| <div>TreeGen: 面向功能增强的程序代码自动生成工具 - 北京大学<br/>给定一串自然语言描述, 本项目通过人工智能相关技术自动化生成程序代码。<br/>数据处理 MulanPSL-2.0</div>            | <div>Relax: 资源竞争检测和消解工具 - 国防科技大学<br/>以配置为出发点, 通过监控系统资源状态并调整与资源相关的配置项, 从而消解资源冲突, 提高软件执行效率<br/>数据处理 MulanPSL-2.0</div> | <div>MBCGO: 基于模型转换的可成长自主无人系统 &amp; 配置项成长寻优 - 北京航空航天大学<br/>基于模型基的可成长自主无人系统, 自动调整配置项参数, 适应任务、资源、环境的要求<br/>数据处理 MulanPSL-2.0</div> |
| <div>ciru: 容器Checkpoint/Restore机制 - 中国科学院软件研究所</div>                                                                  | <div>Ripple: 大规模容器集群配置系统 - 中国科学院软件研究所</div>                                                                          | <div>Javelus: Java软件在线更新系统 - 南京大学<br/>基于OpenJDK实现的工业级Java软件在线更新系统</div>                                                         |

● 重点孵化12项科教界和产业界开源项目

| 项目名称             | 贡献单位         | 项目介绍                                                                                                     | 许可证          | 时间         | 类别   |
|------------------|--------------|----------------------------------------------------------------------------------------------------------|--------------|------------|------|
| Kube-OVN         | 灵雀云          | Kube-OVN是一款由灵雀云自主研发的开源企业级云原生Kubernetes容器网络编排系统。                                                          | Apache 2.0   | 2020.12.30 | 企业成果 |
| PiFlow           | 中科院计算机网络信息中心 | PiFlow是一个基于分布式计算框架技术开发的大数据流水线处理与调度系统。该系统将大数据采集、清洗、存储与分析进行抽象和组件化开发, 以所见即所得、拖拽配置的简洁方式实现大数据处理流程化配置、运行与智能监控。 | Apache 2.0   | 2021.6.9   | 科技成果 |
| PostMan          | 华中科技大学       | PostMan是一个网络功能中间件, 通过高效的按需组包与批处理, 能够快速地缓解突发流量所引起的后端服务性能骤降问题。                                             | MulanPSL-1.0 | 2021.6.9   | 科技成果 |
| skyline          | 浪潮           | Skyline是对标OpenStack社区Horizon项目, 在易用性、页面性能等方面进行深度优化, 提供简单、易用、高效的OpenStack控制台。                             | MulanPSL-2.0 | 2021.6.9   | 企业成果 |
| OceanBase-Client | 蚂蚁集团         | OceanBase Client (简称 OBClient) 是一个基于 MariaDB 开发的客户端工具。您可以使用 OBClient 访问 OceanBase 数据库的集群。                | GPL          | 2021.6.9   | 企业成果 |
| SRS              | 杨成立          | SRS是一个简单高效的实时视频服务器, 支持RTMP/WebRTC/HLS/HTTP-FLV/SRT/GB28181, 应用于直播和WebRTC等互联网视频场景。                        | MIT          | 2021.6.9   | 个人   |
| zCore            | 清华大学         | zCore操作系统是基于Rust语言编写的新一代操作系统。                                                                            | MIT          | 2021.7.13  | 科技成果 |
| DADI             | 阿里云          | DADI 是 Data Accelerator for Disaggregated Infrastructure 的缩写, 旨在为计算存储分离架构提供各种可能的数据访问加速技术。                | Apache2.0    | 2021.7.13  | 企业成果 |
| LinkWeChat       | 江冬勤          | LinkWeChat 是一款基于人工智能的企业微信 SCRM 系统, 为企业构建私域流量营销系统的综合解决方案, 助力企业提高社交客户运营效率。                                 | GPL          | 2021.7.13  | 个人   |
| Furion           | 百小僧          | Furion 是基于 .NET5/6 平台开发的 C# 底层开发框架, 支持 Web、控制台、IoT等领域开发, 框架设计理念是让 .NET 开发更简单, 更通用, 更流行。                  | MulanPSL-2.0 | 2021.8.10  | 个人   |
| 建木               | 九州云          | 建木自动化平台以触发器、流程编排、任务分发等功能为平台核心, 可以用在各类使用场景下, 包括但不限于, CI/CD、DevOps、自动化运维、多业务系统集成等使用场景。                     | MulanPSL-2.0 | 2021.9.9   | 企业成果 |
| Open-Digger      | 华东师范大学       | OpenDigger 是由 X-lab 发起的一个开源数据分析报告开源项目, 这个项目旨在凝聚全球开发者的智慧共同对开源相关数据进行分析统计, 以使开发者可以更好的理解和参与开源。               | Apache2.0    | 2021.11.9  | 科技成果 |



涉及音视频、云原生、网络、大数据、操作系统等领域

# 人工智能开源创新生态

AI开源开放标准



AI开源工具体系



AI资源共享平台



AI生态运营组织



人工智能开源创新生态  
包括**开源软件**、**开源硬件**、**开放标准**、**开放数据**以及**开源运营**等

# 开源生态的研究

# 典型开源生态构建形成经典模式但面临挑战

## ■开源发展40年形成的经典生态构建方法面临挑战

■开源生态典型特征：**边界开放、主体多样、依赖复杂**，而且愈演愈烈

### 经典方法

|               |                                                                                             |
|---------------|---------------------------------------------------------------------------------------------|
| 自由式<br>开源生态构建 | <ul style="list-style-type: none"><li>个体参与者为主体</li><li>以优秀项目为核心</li><li>自组织协作</li></ul>     |
| 垄断式<br>开源生态构建 | <ul style="list-style-type: none"><li>大企业主导</li><li>企业自身项目开源/主导已有开源</li><li>强组织模式</li></ul> |

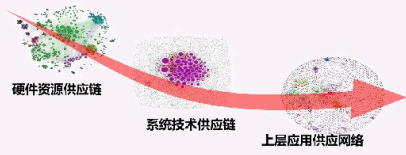
### 典型特征



开源生态边界开放且内部协作快速演化



社区参与主体多样且行为高度不确定

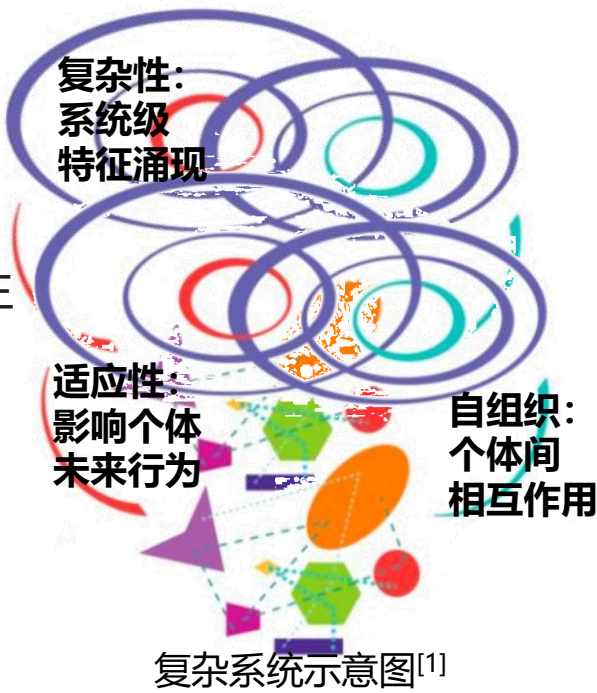


软件制品间依赖复杂且持续动态变化

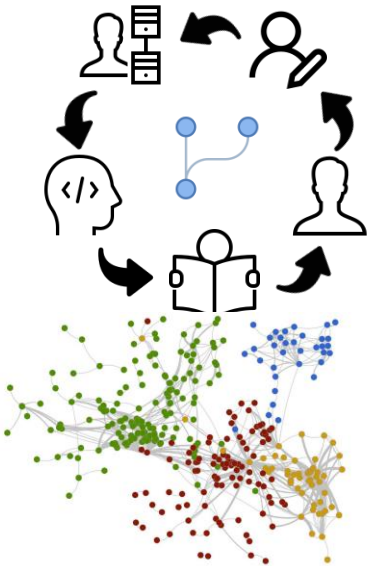
# 开源生态是个复杂系统

## 复杂系统

- 定义：由大量不存在中央控制的相互作用的个体组成的，通过简单运作的规则会产生出复杂的集体行为，并能够通过学习和进化产生适应性的系统<sup>[1]</sup>。
- 特征<sup>[2]</sup>：
  - 自组织：个体一般遵循相对简单的规则，不存在中央控制或者领导者。
  - 复杂性：个体的集体行为可以产生出复杂、不断变化而且难以预测的行为模式。
  - 适应性：个体可以通过学习改变自身行为，以增加生存或成功的机会。



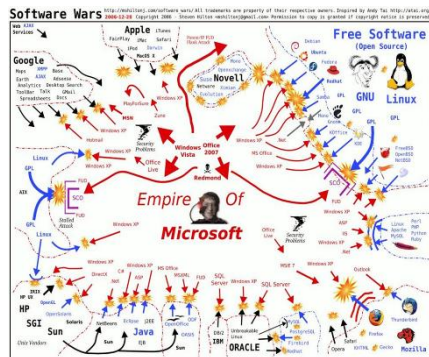
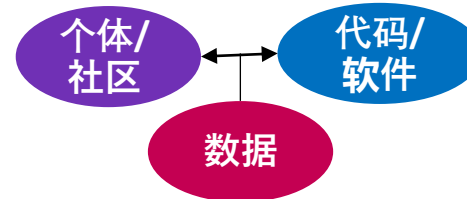
## 开源生态 复杂系统



| 复杂系统                 | 挑战                                                                                                      |
|----------------------|---------------------------------------------------------------------------------------------------------|
| 开源社区<br>(汇聚与协同)      | <ul style="list-style-type: none"><li>开源社区结构与演化的度量</li><li>从开发者的开发效率与留存，到开源社区的开发效率与可持续性的度量与支持</li></ul> |
| 开源软件供应链<br>(培育与风险防控) | <ul style="list-style-type: none"><li>开源供应链结构与演化的度量</li><li>从软件包风险到软件供应链风险的度量与防控</li></ul>              |

[1] 约翰·H·霍兰. 周晓牧等译. 隐秩序——适应性造就复杂性[M]. 上海: 上海科技教育出版社, 2000: 37, 76, 82.  
[2] Bar-Yam Y. General features of complex systems[J]. Encyclopedia of Life Support Systems (EOLSS), UNESCO, EOLSS Publishers, Oxford, UK, 2002, 1.

# 数据驱动的复杂系统 度量、预测和智能化推荐



开源生态大数据



海量成功案例和最佳实践



开源生态构建的  
机制机理、方法  
技术和支撑工具

生态运作模式

开源数字社会学/  
开源动力学

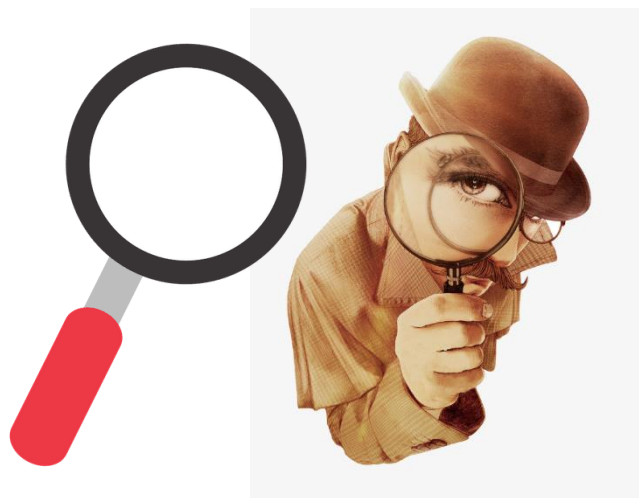
北京大学开源分析实验室:  
<https://osslab-pku.github.io>

# 研究路径：精细度量，以求理解，进而控制

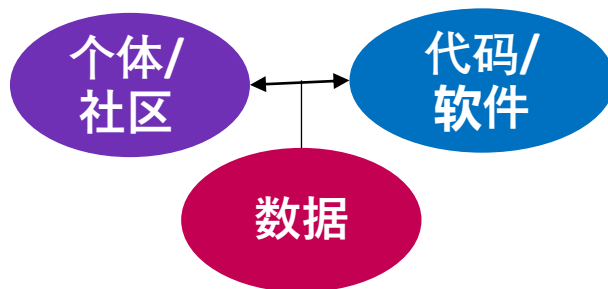
Open the "black box"



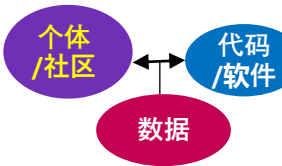
Understand the complexity



Control risks



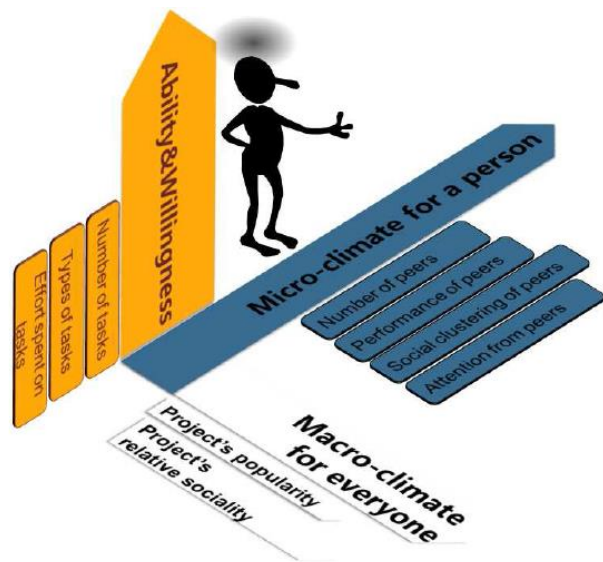
# 个体的度量：度量开源贡献者的行为—意愿、能力和机会



## ■如何吸引贡献者？

- 人们初始进入社区时呈现不同的行为模式，可以据此预测他是否会成为长期贡献者；
- 贡献者进入社区困难，什么是新进者合适的任务(good first issue)?

$$\text{isLTC} \sim \text{nUsr} + \text{RS} + \text{GotFix} + \text{BtE} + \text{nCmt} + \text{nPeer} + \text{pShared} + \text{LckAttn} + \text{PeerPerf} + \text{prj}$$



| Measure               | Predictor            | Odds Ratio    |               | Direction |
|-----------------------|----------------------|---------------|---------------|-----------|
|                       |                      | Mozilla       | Gnome         |           |
| Ability & Willingness | got at least one fix | 2             | 2             | ↑↑        |
|                       | comment/not BB       | 1.5           | 3             | ↑↑        |
|                       | number of comments   | 2             | 1.5           | ↑↑        |
| Micro env             | lack of attention    | $\frac{2}{3}$ | $\frac{2}{3}$ | ↓↓        |
|                       | peers' productivity  | 1.2           | 2             | ↑         |
|                       | peers' soc. clust.   | 1.5           | 1.2           | ↑         |
|                       | number of peers      | 1.14          | 0.94          | ↕         |
| Macro env.            | number of users      | 0.85          | $\frac{1}{2}$ | ↓↓        |
|                       | relative sociality   | 1.07          | 0.73          | ↕         |

Response: {not-LTC, LTC} for Mozilla/Gnome (130,472/125,665 observations)

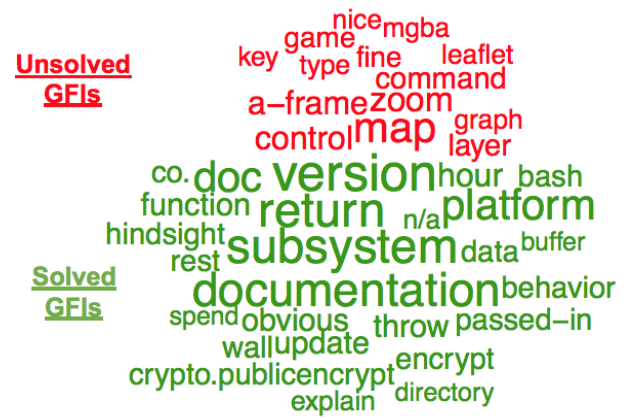
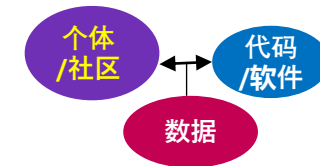


Figure 10: Comparison of solved GFIs and unsolved GFIs. The top 50 keywords with the highest frequency are shown.

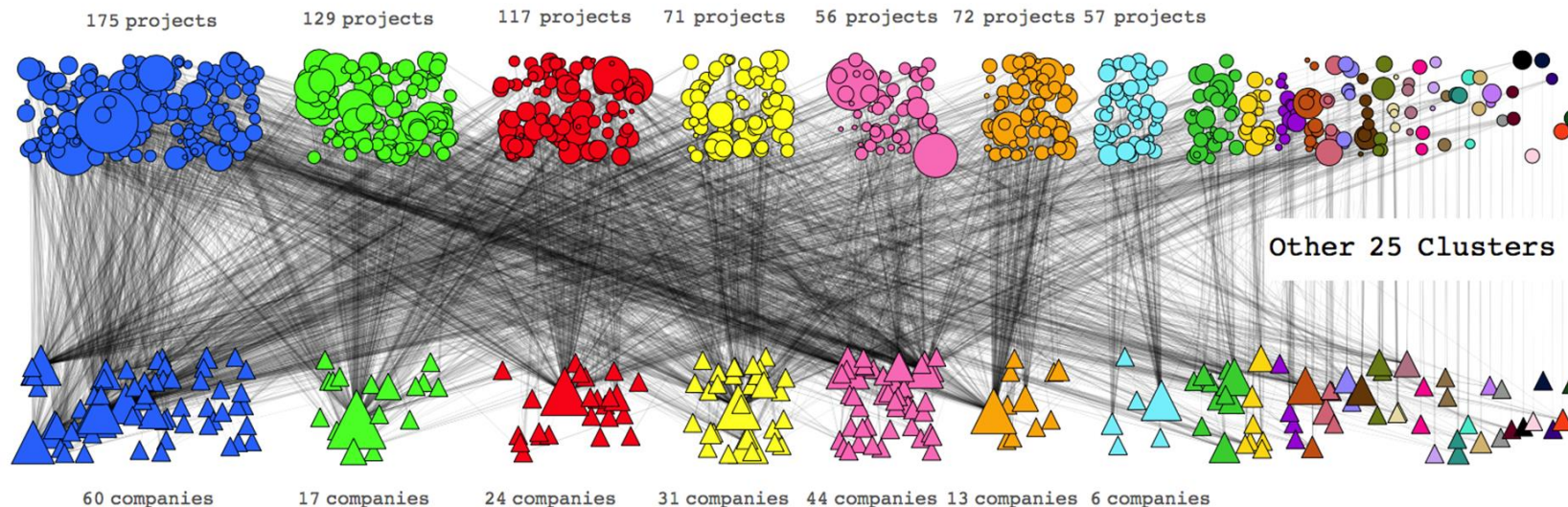


# 开源社区的度量：公司的协作模式



## OpenStack 250个公司:

- 提供完整解决方案: Rackspace, Huawei, 99cloud
- 提供部分解决方案: SwiftStack, 高鸿信安 gohighsec
- 特定业务集成: Intel
- 提供补充服务: Codethink



1,067 nodes: 250 companies + 817 projects.

### Intentional collaboration:

- Supply and Consumption
- Distribution-oriented ally
- Service delegation

Rackspace

Ansible

Walmart

### Passive collaboration:

- Out of own interest
- Most common

HP

Magnum

CERN

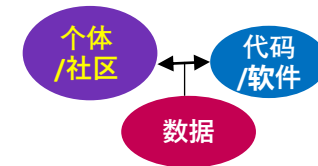
### Isolated collaboration:

- Plugins/drivers
- New services

Citrix

fuel-plugin-xenserver

# 智能化支持：新手任务推荐—GFI-Bot



- 场景：为开源项目的open issue自动打标签，判断其是否“新手”任务
- 度量新手和新手任务的特征，以做精准推荐

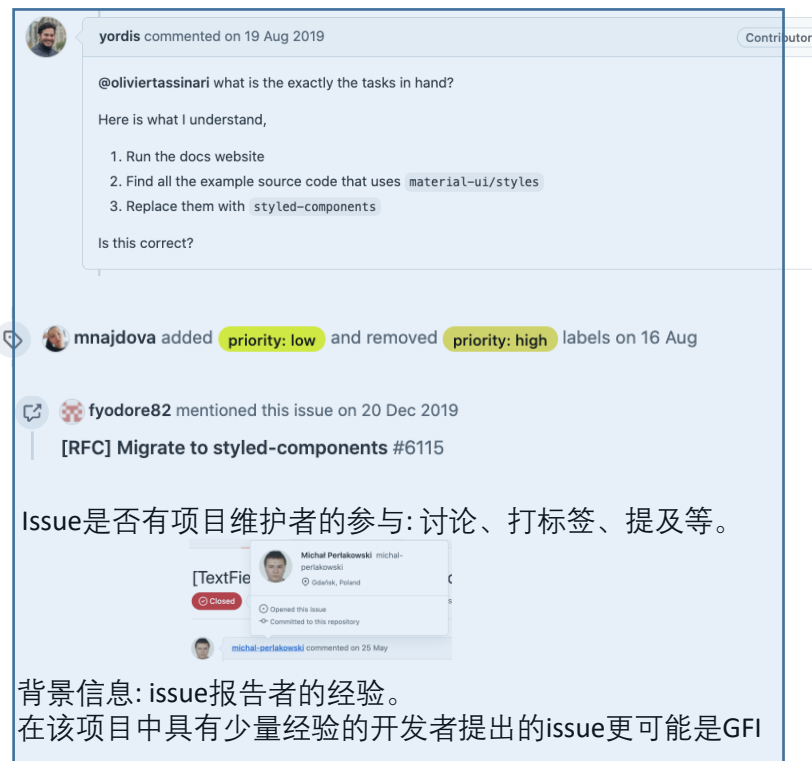
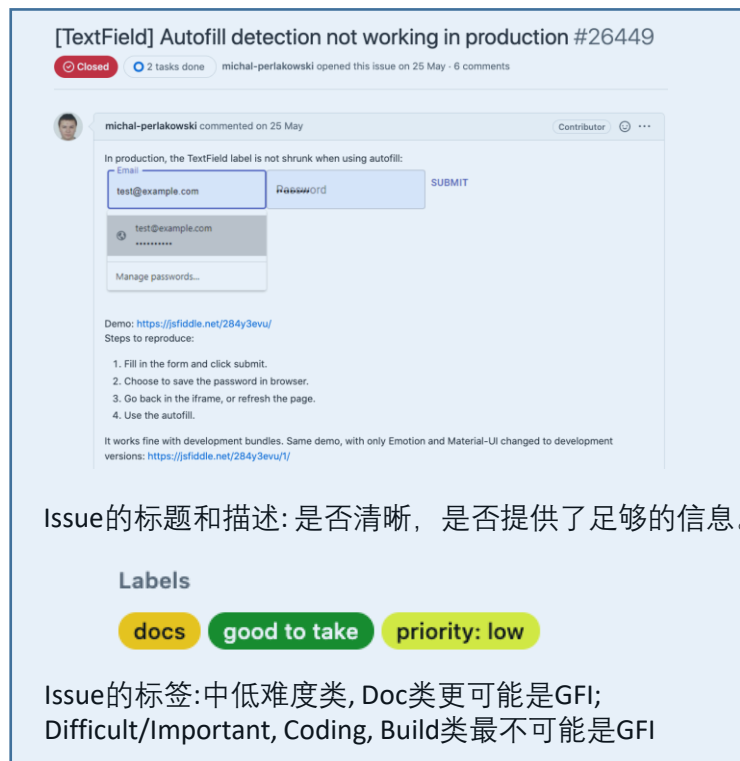
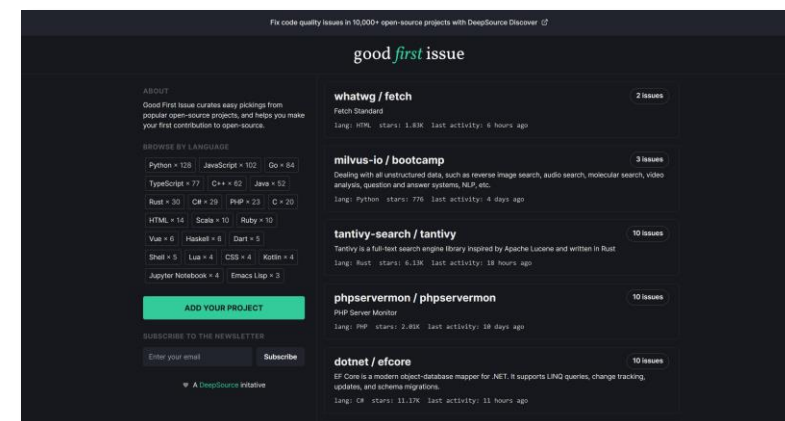


Figure 10: Comparison of solved GFIs and unsolved GFIs. The top 50 keywords with the highest frequency are shown.



在线工具:  
<https://github.com/osslab-pku/gfi-bot>

模型( $f$ ): 回归模型(XGBoost等):  $p = f(x)$

训练集: 历史issue, 令被新人解决的issue的 $p=1$ , 其余issue的 $p=0$

测试点: 任一未被分配的open issue

输出( $p$ ): 一个issue是GFI的概率

Xiao et al. Personalized First Issue Recommender for Newcomers in Open Source Projects. ASE'2023  
He et al. GFI-Bot: Automated Good First Issue Recommendation on GitHub. Tool@ESEC/FSE'2022  
Xiao et al. Recommending good first issues in GitHub OSS projects. ICSE'2022  
Tan et al. A First Look at Good First Issues on GitHub. FSE'2020

# Gitee Compass\*



# 研究路径：精细度量 -> 以求理解 -> 进而控制

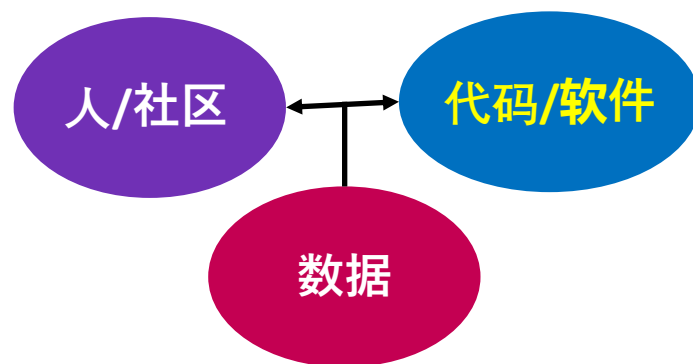
Open the "black box"



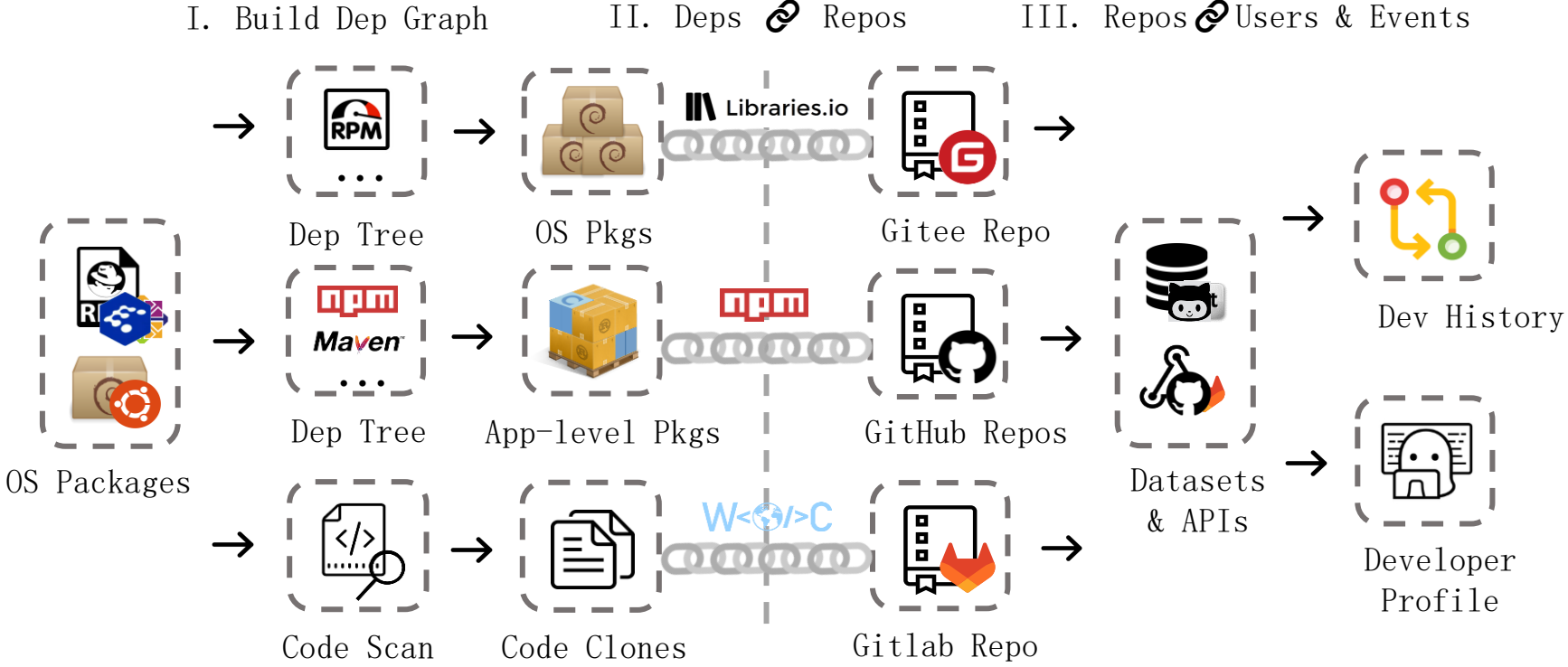
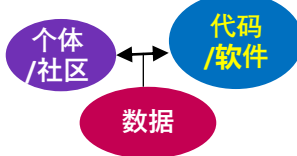
Understand the complexity



Control risks



# 软件的度量：SBOM驱动的可信开源软件供应链



可视、  
可管、  
可控。



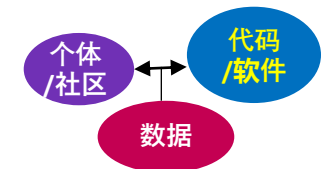
SBOM生成

软件包详情

| SBOM 元数据                                                                                                                                                               | License | 漏洞 | 软件包依赖 |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----|-------|
| <b>General</b>                                                                                                                                                         |         |    |       |
| 软件包名称: coreutils                                                                                                                                                       |         |    |       |
| 版本(epochversion-release): 0:9.0-3.oe2203                                                                                                                               |         |    |       |
| 来源信息: acquired package info from repodata DB: repodata/f344a955efaed1bd8fffc33ce0e                                                                                     |         |    |       |
| 漏洞: 致命 0 高 0 中 0 低 0 未知 0                                                                                                                                              |         |    |       |
| <b>Source Control</b>                                                                                                                                                  |         |    |       |
| Home Page: <a href="https://www.gnu.org/software/coreutils/">https://www.gnu.org/software/coreutils/</a>                                                               |         |    |       |
| Download Location: <a href="https://gitee.com/src-openeuler/coreutils/tree/openEuler-22.03-LTS">https://gitee.com/src-openeuler/coreutils/tree/openEuler-22.03-LTS</a> |         |    |       |
| Supplier: Organization: <a href="http://openeuler.org">http://openeuler.org</a>                                                                                        |         |    |       |
| <b>Description</b>                                                                                                                                                     |         |    |       |
| These are the GNU core utilities. This package is the combination of the old GNU fileutils, sh-util                                                                    |         |    |       |
| <b>Summary</b>                                                                                                                                                         |         |    |       |
| A set of basic GNU tools commonly used in shell scripts                                                                                                                |         |    |       |

- 基本属性
- advanced关键属性:
  - 源码仓库
  - 长依赖链合规性
  - 社会工程学社区属性
  - .....

# 智能化支持：软件供应链全局性合规性风险消解



- 构建复杂软件供应链合规性风险消解模型，提出使用SMT-Solver求解依赖图的版本约束与许可证兼容约束的全局最优合规性风险消解方法

## 挑战性问题

- 依赖图结构复杂
- 一个节点变动可引起依赖图整体变动

## 关键突破

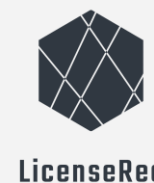
- 首个合规性风险自动消解工具
- 消解方案与 19 个真实案例相匹配，被 5 个流行 PyPI 软件包的开发者采纳

### Algorithm 1: The SILENCE Approach

**Input:**  $\langle p, v \rangle$ ,  $\mathcal{G}(p, v)$ , and the PyPI dataset (Section V-A)

**Output:**  $\mathbf{G} = \{\mathcal{G}'_1, \dots, \mathcal{G}'_N\}$ ,  $\mathbf{L} = \{l_1, \dots, l_M\}$

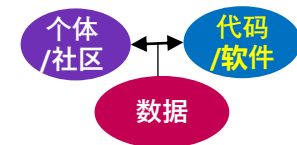
```
1  $\mathbf{G} \leftarrow \emptyset$ ,  $\mathbf{L} \leftarrow \emptyset$ 
2 foreach  $l \in \mathcal{L}$  do # find compatible licenses
3   if  $\langle l(p', v'), l \rangle \notin \mathcal{I}$  for all  $\langle p', v' \rangle \in N(p, v) \setminus \langle p, v \rangle$  then
4      $\mathbf{L} \leftarrow \mathbf{L} \cup \{l\}$ 
5 Keep only top- $M$  licenses in  $\mathbf{L}$  ordered by their popularity
6  $vars \leftarrow \{p, \text{plus all packages reachable from } deps(p, v)\}$ 
7  $clauses \leftarrow build\_constraints(p, v, vars)$ 
8 while  $\mathcal{G}' \leftarrow find\_solution(vars, clauses, objective)$  do
9   if  $|\mathbf{G}| \geq N$  or  $\mathcal{G}' = unsat$  then break
10   $\mathbf{G} \leftarrow \mathbf{G} \cup \{\mathcal{G}'\}$ 
11  Add new constraints to exclude solutions similar to  $\mathcal{G}'$ 
12 return  $\mathbf{G}, \mathbf{L}$ 
```



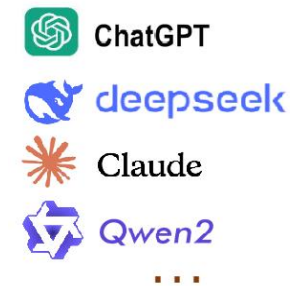
<https://licenserec.com>

合规性风险自动消解技术

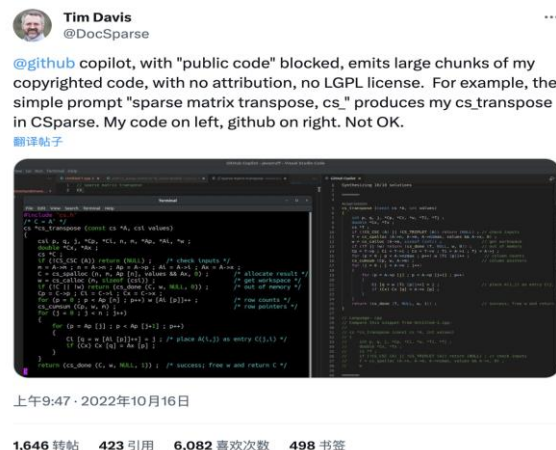
# ongoing research: 大模型合规性能力评估



- 大语言模型（LLMs）利用广泛的开源代码和指令训练，在各种软件工程任务中都表现出非凡能力。
- 由于LLMs具有一定的记忆能力，它们在特定提示下生成的代码片段可能会与训练数据中的内容相似，甚至完全相同<sup>[1]</sup>。
- 代码生成任务中已有大量的工作来评估LLMs的性能（准确性、稳健性、安全性和隐私等方面），但在许可证合规性方面仍然是空白。



开源作者投诉Copilot输出其代码，但并未附带许可信息<sup>[2]</sup>



We've filed a lawsuit challenging GitHub Copilot, an AI product that relies on unprecedented open-source software piracy. **Because AI needs to be fair & ethical for everyone.**

Copilot因输出许可证保护的内容却不遵循许可证条款，正在面临诉讼<sup>[3]</sup>

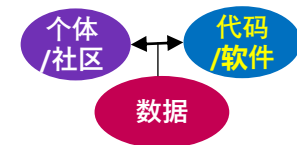
|                                                                                                       |    |
|-------------------------------------------------------------------------------------------------------|----|
| VII. FACTUAL ALLEGATIONS.....                                                                         | 12 |
| A. Introduction.....                                                                                  | 12 |
| B. Codex Outputs Copyrighted Materials Without Following the Terms of the Applicable Licenses .....   | 13 |
| C. Copilot Outputs Copyrighted Materials Without Following the Terms of the Applicable Licenses ..... | 18 |

[1] Yang Z, Zhao Z, Wang C, et al. Unveiling memorization in code models[C]//Proceedings of the IEEE/ACM 46th International Conference on Software Engineering. 2024: 1-13.

[2] <https://twitter.com/DocSparse/status/1581461734665367554>

[3] <https://githubcopilotlitigation.com/>

# LiCoEval 评估大模型合规性能力



挑战：区分潜在的侵权产出与独立创作的内容

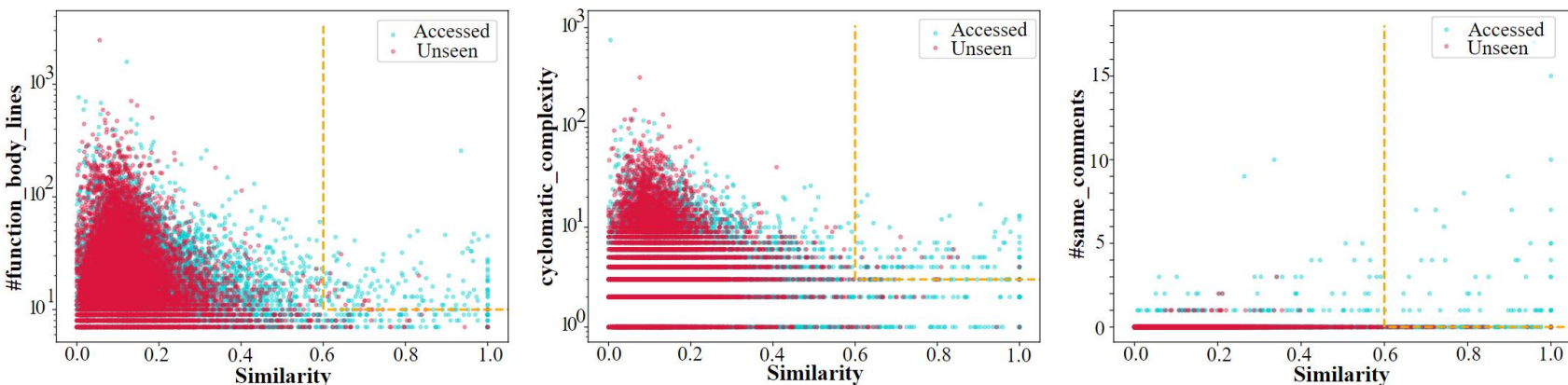
国际上普遍认可的两种证明实质性抄袭原创作品的方法<sup>[1-2]</sup>：

- 原告可以选择提供证据，证明被告有“接触”到受版权保护的作品，且两部作品“实质相似”。
- 另一方面，当无法证明接触时，原告可以提供证据，表明所涉及的作品具有“惊人相似性”（足以排除独立创作的可能）。



实证研究：

在大型语言模型（LLMs）生成代码的场景中，合理的惊人相似标准可能在哪里？

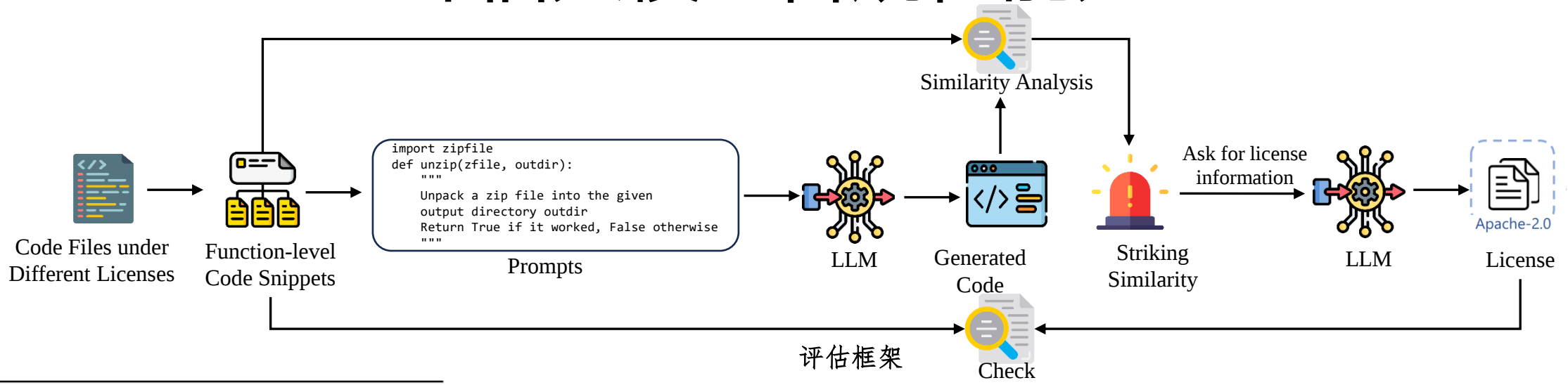


函数体代码行数 > 10  
圈复杂度 > 3  
文本相似度 > 0.6  
相同注释数 > 0

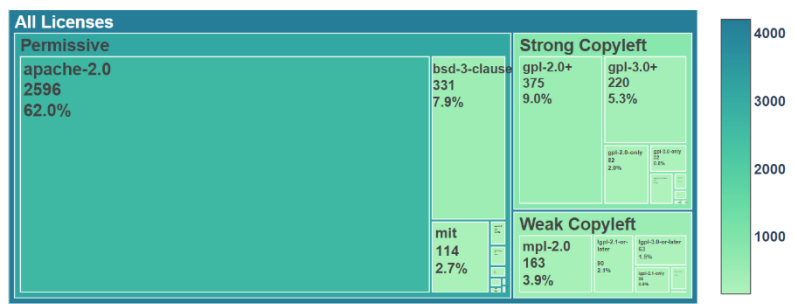
[1] <https://www.ce9.uscourts.gov/jury-instructions/node/274>

[2] Scheffler S, Tromer E, Varia M. Formalizing human ingenuity: A quantitative framework for copyright law's substantial similarity[C]//Proceedings of the 2022 Symposium on Computer Science and Law. 2022: 37-49.

# LiCoEval 评估大模型合规性能力



| Metric                | Min | Median | Mean  | Max    | Distribution* |
|-----------------------|-----|--------|-------|--------|---------------|
| #prompt_lines         | 2   | 25     | 28.5  | 119    |               |
| #prompt_tokens        | 25  | 265    | 310.3 | 964    |               |
| #body_lines           | 11  | 28     | 36.8  | 398    |               |
| #body_tokens          | 64  | 356    | 479.8 | 5418   |               |
| #project_reuse        | 16  | 54     | 245.0 | 42,476 |               |
| #comments             | 1   | 4      | 5.7   | 159    |               |
| cyclomatic_complexity | 4   | 7      | 9.1   | 95     |               |



Benchmark

|             | Model                        | HumanEval | Weight | #striking_sim | Acc  | #permissive | Acc <sub>p</sub> | #copyleft | Acc <sub>c</sub> | LiCo  |
|-------------|------------------------------|-----------|--------|---------------|------|-------------|------------------|-----------|------------------|-------|
| General LLM | GPT-3.5-Turbo [83]           | 72.6      | ×      | 29 (0.69%)    | 0.72 | 26          | 0.81             | 3         | 0.0              | 0.373 |
|             | GPT-4-Turbo [83]             | 85.4      | ×      | 25 (0.60%)    | 0.72 | 22          | 0.82             | 3         | 0.0              | 0.376 |
|             | GPT4o [83]                   | 90.2      | ×      | 47 (1.12%)    | 0.74 | 41          | 0.85             | 6         | 0.0              | 0.385 |
|             | Gemini-1.5-Pro [84]          | 71.9      | ×      | 41 (0.98%)    | 0.59 | 39          | 0.62             | 2         | 0.0              | 0.317 |
|             | Claude-3.5-Sonnet [85]       | 92.0      | ×      | 84 (2.01%)    | 0.69 | 79          | 0.71             | 5         | 0.4              | 0.571 |
|             | Qwen2-7B-Instruct [86]       | 79.9      | ✓      | 20 (0.48%)    | 0.95 | 20          | 0.95             | 0         | -                | 0.985 |
|             | GLM-4-9B-Chat [87]           | 71.8      | ✓      | 0 (0.0%)      | -    | -           | -                | -         | -                | 1.0   |
|             | Llama-3-8B-Instruct [88]     | 62.2      | ✓      | 1 (0.02%)     | 0.0  | 1           | 0.0              | 0         | -                | 0.714 |
| Code LLM    | DeepSeek-Coder-V2 [89]       | 90.2      | ✓      | 37 (0.88%)    | 0.0  | 36          | 0.0              | 1         | 0.0              | 0.142 |
|             | CodeQwen1.5-7B-Chat [90]     | 83.5      | ✓      | 17 (0.41%)    | 0.24 | 17          | 0.24             | 0         | -                | 0.781 |
|             | StarCoder2-15B-Instruct [60] | 72.6      | ✓      | 13 (0.31%)    | 0.23 | 13          | 0.23             | 0         | -                | 0.780 |
|             | Codestral-22B-v0.1 [91]      | 61.5      | ✓      | 91 (2.17%)    | 0.73 | 87          | 0.77             | 4         | 0.0              | 0.360 |
|             | CodeGemma-7B-IT [92]         | 56.1      | ✓      | 3 (0.07%)     | 0.33 | 3           | 0.33             | 0         | -                | 0.809 |
|             | WizardCoder-Python-13B [61]  | 64.0      | ✓      | 27 (0.64%)    | 0.04 | 26          | 0.04             | 1         | 0.0              | 0.153 |
|             |                              |           |        |               |      |             |                  |           |                  |       |

评估结果

# 愿景：产学研深度融合建设全球化开源创新生态

## 产学研深度融合的开源生态支撑**基础设施、国际标准和新型社区**

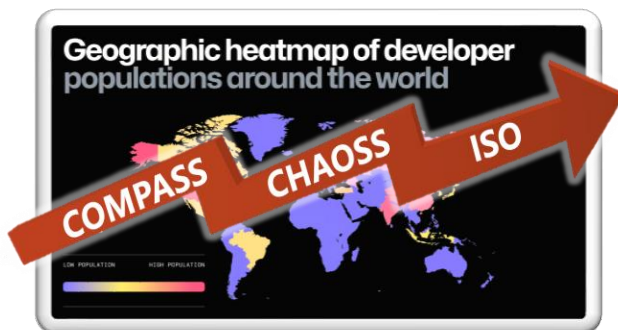
开源产业

开源研究

开源教育

### 构建全球开源生态度量体系 建立国际影响力

- 建立国际度量体系标准
- 联合关键开源社区
- 支持全球开源生态建设



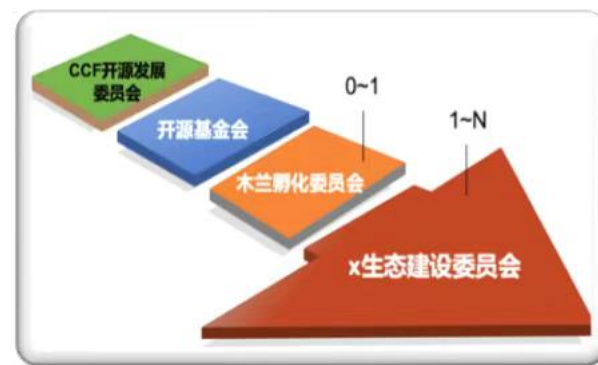
### 1 全网开源数据资源化 支持协同创新研究

- World of Code: 开源数据资源化
- HuggingData: 数据利用社区化



### 3 产学研开源科教基地 服务创新人才培养

- 立足CCF开源发展委员会建立和发展开源基金会
- 推动0~1创新孵化(木兰孵化委员会)和开源教育(头歌平台)
- 推进1~N的生态建设: 基础软硬件, AI技术, 人机物融合技术



# 建议阅读的文献

- *The Cathedral & the Bazaar* (大教堂与集市). Raymond, E.S. (1999). O'Reilly Retrieved from <http://www.catb.org/~esr/writings/cathedral-bazaar/>
- 人月神话. 弗雷德里克·布鲁克斯. 出版社: 清华大学出版社. 译者: 汪颖. 出版年: 2002-11. ISBN: 9787302059325
- 开源的成功之路 (The Success of Open Source), 史蒂文 (美国), 外语教学与研究出版社, 2007-6, ISBN: 9787560066363.
- A. Mockus, R. T. Fielding, and J. Herbsleb, “Two case studies of open source software development: Apache and Mozilla,” ACM Trans. Softw. Eng. Methodol. vol. 11, no. 3, pp. 1–38, Jul. 2002.
- Minghui Zhou and Audris Mockus. Who Will Stay in the FLOSS Community? Modelling Participant's Initial Behaviour. IEEE Transactions on Software Engineering. vol.41, no.1, pp.82-99, Jan. 1 2015.
- 周明辉,张伟,尹刚. 开源软件的量化分析.中国计算机学会通讯.第12卷,第2期. 2016年2月.
- 周明辉,张宇霞,谭鑫. 软件数字社会学. 中国科学. 2019

# 开源是现在和未来，机会和挑战并存

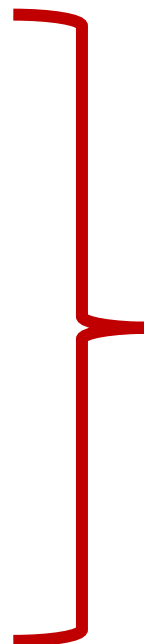
- 人人有责，从我做起



~7300万开发者  
~2亿代码仓库



~ 600万开发者  
~1500万代码仓库



开源产业

开源研究

开源教育

# 开源生态的形成

- **多样性**：至少三类角色：软件/产品/服务的生产者、提供者和消费者。
- **组织性**：社区的层次化组织保障有效劳动和任务分工。层次结构以贡献/能力为参数分配角色和职责。
- **协作性**：通过协作突破个体能力的局限，但同时可能降低个体的效率（在协作网络中，个体能力大约是其协作者数量的1/2次幂[1]）。
- **主次性**：存在一小部分核心贡献者，使能号召更广泛的外围参与者。



开源软件生态：

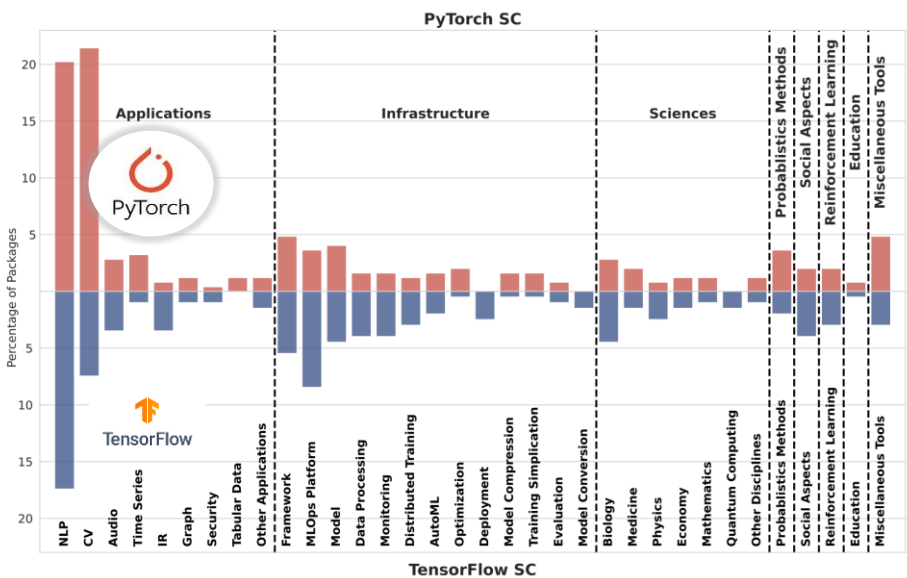
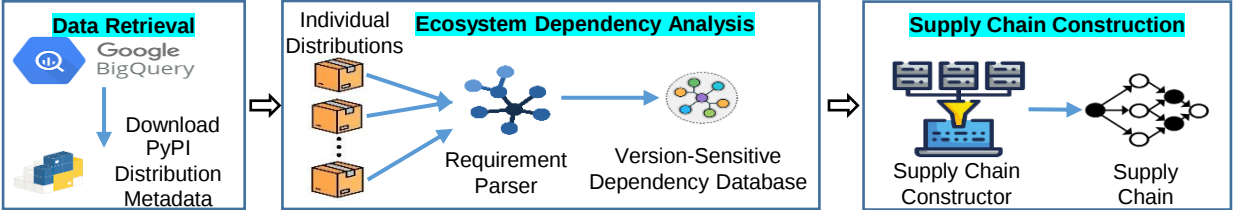
- **开放性**：开放的代码和贡献者准入机制，鼓励吸引更多参与者。
- **透明度**：社区的行为和决策过程具有高度透明性，确保所有参与者了解并乐于参与。
- **平衡性**：自治与集中控制的平衡；志愿者与企业行为的平衡。
- **多赢性**：零频零和博弈，实现共赢多赢。

- **凝聚力**：需要核心软件（即项目）形成凝聚力。
- **组装性**：软件需要模块化，便于合理分工。

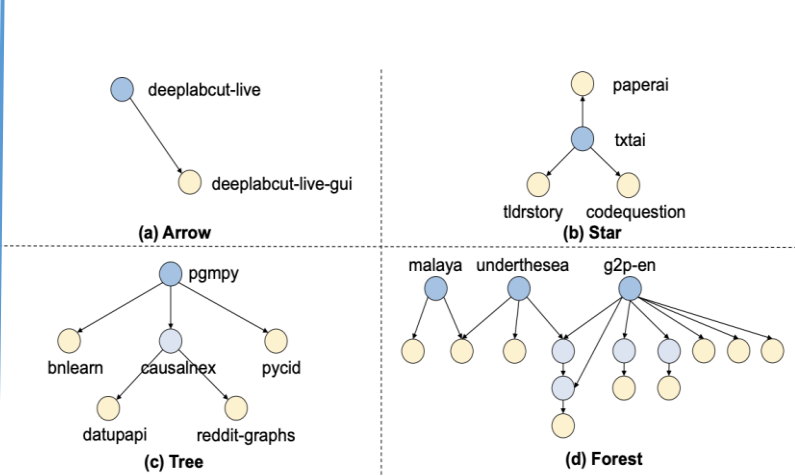
- **工具性**：基础设施提供有效高效工具集助力开发效率提升。
- **群智性**：基础设施支持高效率的群体协作，汇聚群智，放大个体能力；支持个体学习和能力提升。

# 开源软件供应链的度量: domain, cluster, and disengagement

- DL供应链: 以TensorFlow或PyTorch为原点构建--通过安装依赖形成的--PyPI软件包供应链
  - TensorFlow SC: 2567个软件包, 19116个版本
  - PyTorch SC: 3278个软件包, 26563个版本



- TensorFlow和PyTorch供应链分别在通用开发工具和特定应用领域上形成了自己的核心竞争优势。
- TensorFlow SC提供更高比例的Infrastructure类型的软件包, 如MLOps Platform API和部署工具。PyTorch SC提供更高比例的Application类型的软件包, 如NLP和CV。



- TensorFlow和PyTorch SC的package聚类呈现4种形状: Arrow, Star, Tree和Forest, 依赖结构复杂性递增。
- 大多数package集中在Tree和Forest集群中。

Table 1. Reasons for Package Disengagement.

| Reason                | TensorFlow SC <sup>1</sup> | PyTorch SC <sup>2</sup> |
|-----------------------|----------------------------|-------------------------|
| <b>Dependency</b>     | 38 (45.78%)                | 13 (29.55%)             |
| Incompatibility       | 23 (27.71%)                | 10 (22.73%)             |
| Bloated               | 15 (18.07%)                | 4 (9.09%)               |
| <b>Functionality</b>  | 27 (32.53%)                | 24 (54.55%)             |
| Performance           | 16 (19.28%)                | 7 (15.91%)              |
| Simplification        | 9 (10.84%)                 | 13 (29.55%)             |
| Framework-Independent | 3 (3.61%)                  | 5 (11.36%)              |
| <b>Installation</b>   | 25 (30.12%)                | 14 (31.82%)             |
| Flexibility           | 15 (18.07%)                | 1 (2.27%)               |
| Size Trimming         | 10 (12.05%)                | 13 (29.55%)             |

<sup>1</sup> 83 disengaged packages in total in TensorFlow SC.

<sup>2</sup> 44 disengaged packages in total in PyTorch SC.

- 软件包脱离DL供应链主要是出于依赖问题、功能改进和安装方便等原因。
- TensorFlow供应链上最常见的脱离原因是依赖不兼容; PyTorch供应链上最常见的脱离原因是功能简化和减少安装大小。