



# 走进开源 & 开源案例分析与实践

— 以 CloudWeGo 为例

— 罗广明 / CloudWeGo 开源负责人

2023/9/13



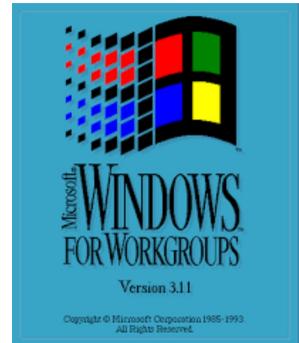
# CONTENTS

## 目录

01. 开源概述
02. 字节跳动开源介绍
03. CloudWeGo 介绍
04. 开源社区建设与运营
05. 开源实践入门

# 01 开源概述

## 1.1 自由软件运动



早期，软件是硬件的附属品，随硬件输出，硬件供应商会提供额外的软件开发。软件可以被任意分发和修改。

但是，随着硬件利润的下降，以及软件复杂度的上升，越来越多厂商开始单独售卖软件，软件也不再开放。专有软件售卖以微软为盛，也包括 AT&T 对 UNIX 的商业化。

1983 年，怀揣着软件被日益束缚的忧虑，**Richard Stallman** 发起了 **GNU** (GNU is Not Unix) 项目。GNU 的创立，标志着自由软件运动的兴起。次年，**ESR** 从 MIT 辞职全职加入了 GNU，并于 1985 年创立了 **Free Software Foundation (FSF)**。

自由软件运动是一个社会运动，旨在打破厂商对软件的封锁，让软件可以被每个人使用、分发和修改。

## 1.2 自由软件定义

自由软件基金会 FSF 对自由软件是这样定义的：“自由软件是指那些授予用户自由共享，学习和修改权利的软件。”

在 FSF 的定义中，自由软件必须遵守如下四个“自由支柱”（注意这些是权利而不是义务）：

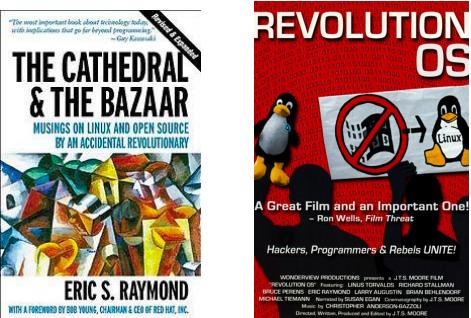
- 可以在任何使用环境中**自由地部署**软件，而没有任何限制。例如，某个程序的许可证会在30天后过期，那么它就不是自由软件。
- 可以**自由地研究**软件的工作方式，并能够根据实际需要和偏好进行**修改**。
- 可以**自由地二次分发**（re-distribute）某个软件，以帮助有需要的人。此处的二次分发既可以是有偿的，也可以是无偿的。
- 可以**自由地增强**软件的性能，并发布增强的功能，进而让社区（各种程序员或非程序员）能够从中受益。此类行为既可以是有偿的，也可以是无偿的。

## 1.3 大教堂与集市及开源软件的兴起

[自由软件](#)并不意味着不能收费，核心是指用户具有软件使用、修改和分发的权利。自由软件的这一倡导常常收到诟病，因为它损害了著作者的权益。另一方面，英文中 free 既有自由的含义，也有免费的含义，这为自由软件运动带来了不必要的解释成本。



《[大教堂与集市](#)》是 Eric S. Raymond 的文集，写于 1997 年。ESR 通过对 Linux 开发模式的思考，加之在个人项目 Fetchmail 上的实践，指出了开源对生产力提高有极大的提升。ESR 将基于闭源的传统开发模式比作「大教堂」，将基于开源的开放式协作模式比作「集市」，并认为「集市」的模式将会是未来。ESR 提出了极富洞察力的观点，使 ESR 一度成为开源的代名词。



1998 年，在《大教堂与集市》的影响下，Netscape 开源了 Mozilla 项目，至今仍具有很强的影响力。同年，ESR 与 Bruce Perens、Michael Tiemann 创造了「开源（Open Source）」一词，并共同创建了 [Open Source Initiative](#)。至此，开源逐渐走向了正规化。

## 1.4 开源软件定义

非盈利组织 Open Source Initiative (OSI) 极力倡导，任何开源软件都必须遵循如下标准：

- 可以被免费进行二次分发。
  - 源代码应当公开、可用。
  - 可以与原始软件不同的格式进行修改和分发。
  - 软件本身不应歧视任何个人或团体。
  - 软件本身不应限制其他软件的使用或调用。
- 
- ✓ 开源不仅仅意味着源码开放，同时还需要支持衍生物发行、符合传播规范、满足非歧视原则等要求。
  - ✓ OSI 让开源具有了教育和倡导开放式开发流程的优势。（社会价值）
  - ✓ 开源软件提供了一种与潜在的软件用户和开发人员相互动的宝贵方法。通过一个具有互动参与性的社区，大家可以创建新的或改进原有的源代码。（项目价值）

Question: 开源 = 开放源代码？

## 1.5 自由软件 & 开源软件 & 免费软件

- 所谓“Free Software” 中的“free”一词强调的是自由，而不是价格上的免费。
  - **自由软件有比开源软件更严格的概念**，因此所有自由软件都是开放源代码的，但**不是所有的开源软件都能被称为“自由”**。
  - 免费软件就是不要钱的软件，但“不要钱”的定义往往是模糊的：是指人们取得该软件时无需付费，还是说人们在使用的过程中都无需付费，亦或是指该软件的发行者不从中获取利益？实际情况往往是复杂而黑暗的。
- 
- ✓ 自由软件更多的体现的是一种**信仰**，可以看作为计算机的言论自由运动，象征的是互联网的分享与交流精神；
  - ✓ 开源更为**具体**，是一种开放源代码共同开发的协作模式；开源软件有真有假，并非开源就一定有意义；
  - ✓ 免费软件虽“不要钱”，但用户还需小心谨慎。

# 1.6 开源软件的意义

从软件分发而言

- 任何人可以修改源代码，以满足使用需求
- 打破专有软件垄断，根据许可证要求再分发
- 降低软件总拥有成本，促进软件行业快速发展

从行为动机而言

- 礼物文化：付出热情、智慧和努力，得到认可
- 行动中学习，教与学的过程，例如 Code Review

从经济效益而言

- 开源对经济的促进，例如增加对开源项目的投入，将使欧盟的国内生产总值每年增加950亿欧元。

从技术发展而言

- 标准化快速落地，打破软件烟囱，技术生态繁荣发展
- 沟通协作和技术场景复杂，促进代码和架构的模块化
- 开源引领技术创新，成为新技术的摇篮

从组织管理而言

- 打造开放式组织，最大化知识工作者的效益
- Private Collective Model，即成员私有投入，产出公开可见
- 民主与决策，Community over code 下的社区治理实践

## 1.7 开源的健康发展

### 企业的角度

- 遵守开源许可证要求，支持开源项目的基本协同原则，同时避免企业面临法律诉讼
- 提升外部开源贡献，形成公司业务发展和开源推动的正循环
- 重视开源安全风险，推动代码审查和漏洞治理机制，反哺开源软件的安全加固
- 选择合适的开源技术路线，支持开源标准设立，同时避免企业长期技术债务

### 社区的角度

- 重视开源治理和运营，规范开发者行为准则，建立开发者沟通渠道，维护良好的社区环境
- 完善开发环境和基础设施，降低准入门槛，创造开放、兼容的协作氛围
- 有效平衡开源的使用者、贡献者，以及商业公司的利益，避免损害开源软件的长期发展空间

02

## 字节跳动开源介绍

## 2.1 字节跳动开源理念

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，对于开源的态度一直是开放、鼓励的，愿与全球合作伙伴共同实现生态繁荣。

Inspire Creativity, Embrace Open Source

经历三个阶段：从使用、到参与、再到主动开源

使用开源

参与开源

主动开源

快速推动业务发展

结合自身业务优化，保持和社区一致，经验共享

共建社区  
推动技术进步

## 2.2 字节跳动开源策略

- 以北极星指标代替短期KPI
- 打造精品项目优于广泛覆盖
- 用户价值优于商业变现
- 安全和合规是底线

### 制定公司级开源战略

明确开源的价值、目标与策略，确保有体系、有目标地开展开源工作



### 明确开源项目评审标准

筛选聚焦出高价值的重点项目，确保重点开源项目的资源支持与效果产出



### 强化开源管理规范

优化审批流程，确保使用/贡献/自研开源等各类开源实践的合法、合规、合理



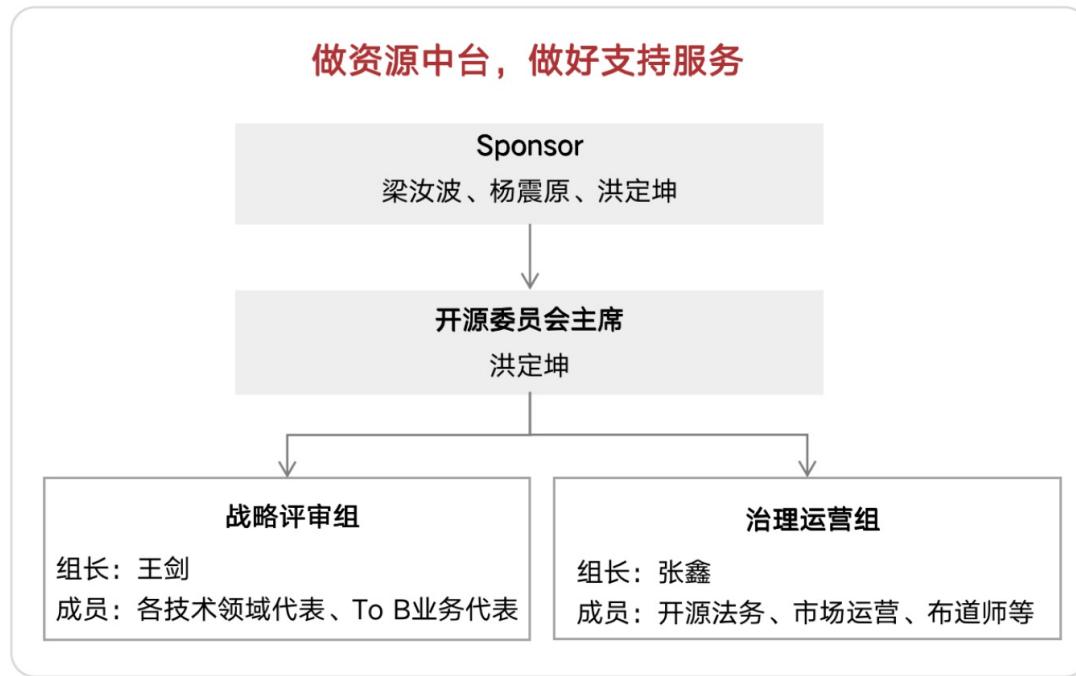
### 做好内外部开源布道

对内提升开源认知与教育，对外展现开源成果、打造社区影响力



## 2.3 字节跳动开源历程

- 目前，字节已经开源了 50 多个技术类开源项目，包括 CloudWeGo、KubeWharf 等多个基于字节内部技术沉淀的开源项目。
- 开源的管理，经历了从个体驱动到体系化治理的过程。
- 2022年成立了开源委员会 OSPO，由公司 CEO 以及两位技术 leader 作为 sponsor。



# 03

## CloudWeGo 项目介绍

### 3.1.1 背景介绍 – 云原生

- 云原生（Cloud Native）是一种构建和运行应用程序的方法，是一套技术体系和方法论。
- 云原生从字面意思上来看可以分成云和原生两个部分。
  - 云是和本地相对的，传统的应用必须跑在本地服务器上，现在流行的应用都跑在云端，云包含了 IaaS、PaaS 和 SaaS 等。
  - 原生就是土生土长的意思，我们在开始设计应用的时候就考虑到应用将来是运行云环境里面的，要充分利用云资源的优点，比如云服务的弹性和分布式优势。

### 3.1.1 背景介绍 – 云原生

- Matt Stine 提出的 Cloud Native (13、15、17)
  - 单体架构在向 Cloud Native 迁移的过程中需要文化、组织、技术共同变革。
  - 把 Cloud Native 描述为一组最佳实践：
    - 十二因子: [https://12factor.net/zh\\_cn/](https://12factor.net/zh_cn/) ;
    - 微服务;
    - 自服务敏捷基础设施;
    - 基于 API 的协作;
    - 反脆弱性。

——《Migrating to Cloud Native Application Architecture》

### 3.1.1 背景介绍 – 云原生

- CNCF 重新定义的 Cloud Native (2018)：
  - 基于容器、服务网格、**微服务**、不可变基础设施和声明式 API 构建的可弹性扩展的应用；
  - 基于自动化技术构建具备高容错性、易管理和便于观察的松耦合系统；
  - 构建一个统一的开源云技术生态，能和云厂商提供的服务解耦。



**CLOUD NATIVE  
COMPUTING FOUNDATION**

2015年由 Linux 基金会发起了一个 [The Cloud Native Computing Foundation \(CNCF\)](#) 基金组织，CNCF 基金会的成立标志着云原生正式进入高速发展轨道，[google](#)、[Cisco](#)、[Docker](#)各大厂纷纷加入，并逐步构建出围绕 Cloud Native 的具体工具，而云原生这个的概念也逐渐变得更具体化。

## 3.1.2 背景介绍 – 微服务

- 微服务架构是以一组小型服务的方式来开发一个独立的应用系统，每个服务都以一个独立进程的方式运行，每个服务与其他服务使用**轻量级（RPC & HTTP）通信机制**。
- 这些服务是围绕业务功能构建的，可以通过全自动部署机制独立部署，同时服务会使用最小规模的集中管理能力，也可以采用**不同的编程语言和数据库**，实现去中心化的服务管理。



### 3.1.3 背景介绍 – 字节微服务的特征

1

规模大、增长快

2

容器化程度高、云原生

3

Go 使用最广  
Rust 正在加大投入

4

打造下一代高性能  
服务网格

## 3.2 CloudWeGo 简介



CloudWeGo 是字节跳动基础架构团队开源出来的项目，它是一套可快速构建企业级云原生架构的中间件集合，它专注于**微服务通信与治理**，具备**高性能、高可扩展、高可靠**的特点，且关注易用性。

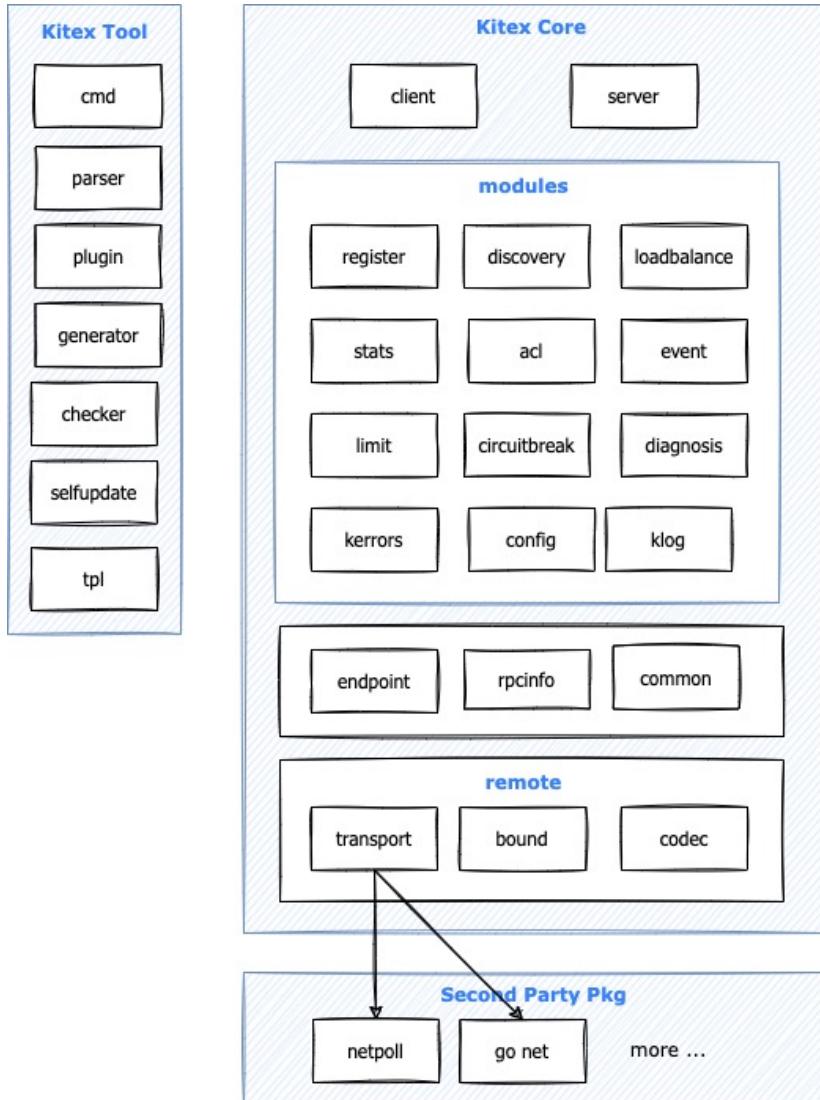
**“内外一致”，持续迭代！**

### 3.3 项目全景图



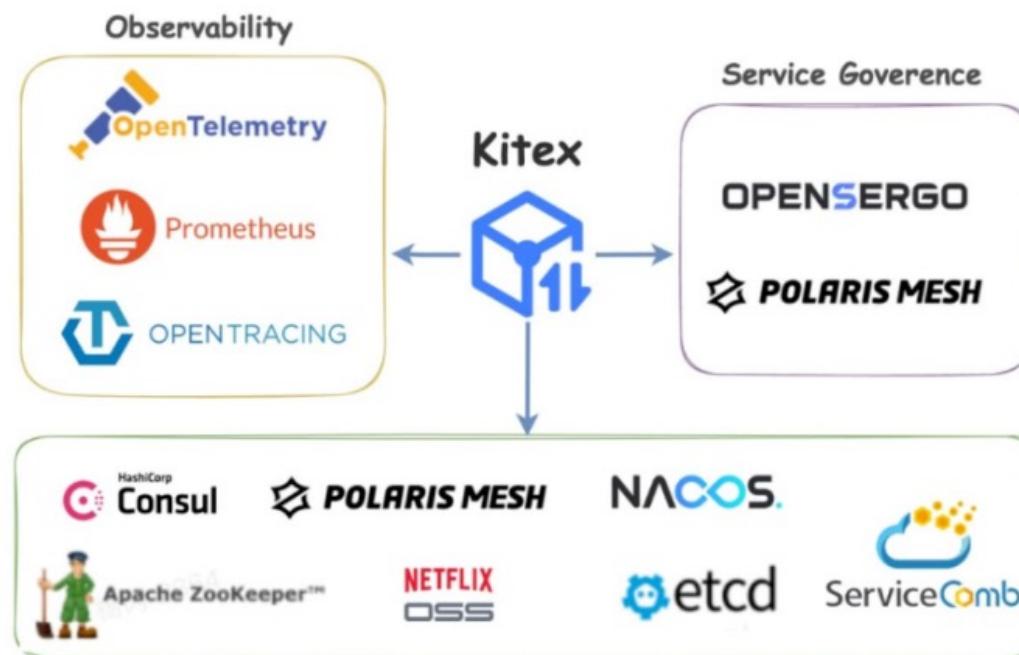
\* 其中 Sonic 和 Monoio 项目目前托管在 bytedance 开源 org 下。

### 3.3.1 重点开源项目 - Kitex

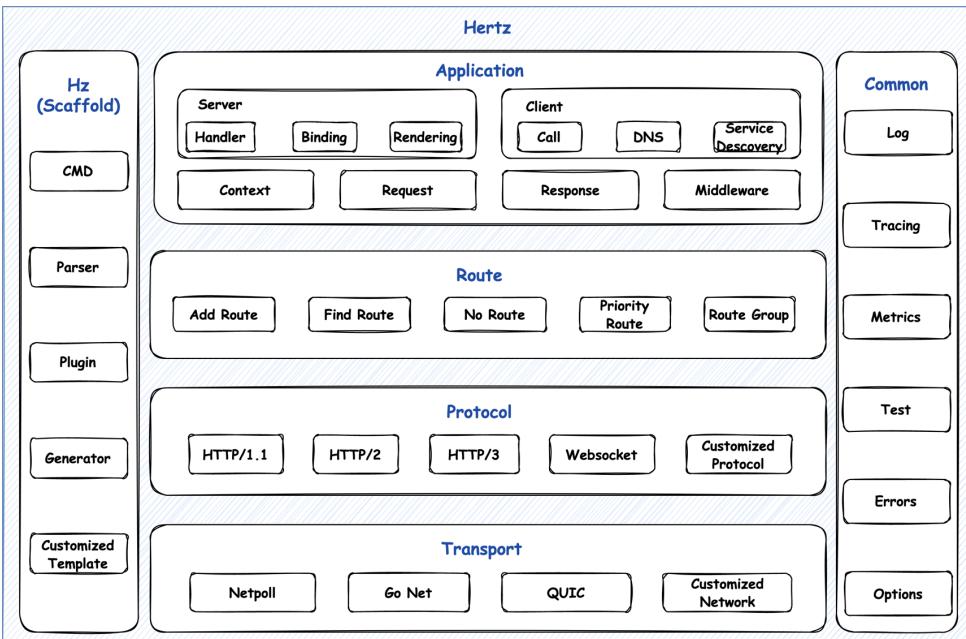


Kitex 是 Golang 微服务 RPC 框架，具有高性能、高可扩展的特点

- 多消息协议：**默认支持 Thrift (Bufferd & Framed)、Kitex Protobuf、gRPC
- 多消息类型：**PingPong、Oneway (Thrift)、双向 Streaming (gRPC)
- 服务治理：**服务注册/发现、负载均衡、熔断、限流、重试、监控、链路跟踪、日志、诊断等
- 代码生成：**内置代码生成工具可支持生成 Thrift、Protobuf 以及脚手架代码

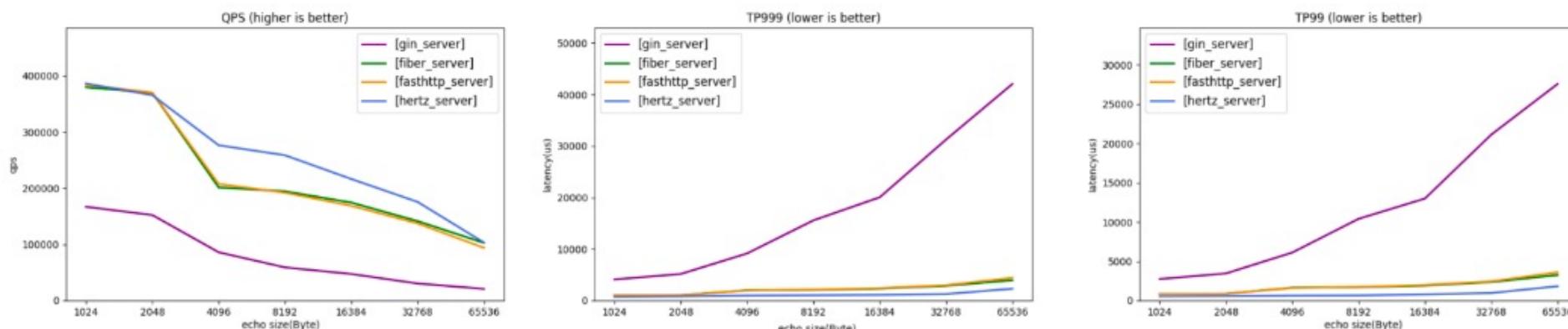


## 3.3.2 重点开源项目 – Hertz

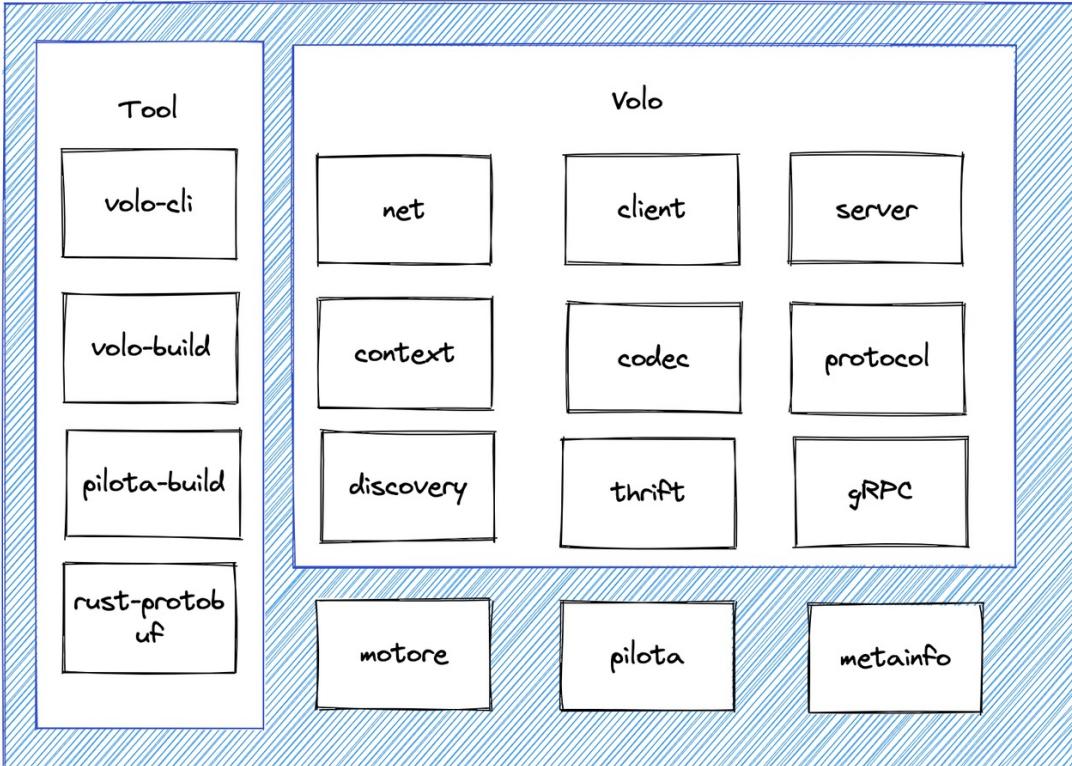


Hertz 是 Golang 微服务 HTTP 框架

- 多协议**: 默认支持 HTTP1.1、HTTP/2、HTTP/3、WebSocket
- 网络层**: Netpoll & Go net 针对不同场景按需切换
- 中间件**: CORS、Recovery、Basic Auth、JWT、Gzip、i18n、Session、Pprof、KeyAuth、Swagger、RequestID、Secure、Sentry、CSRF
- 服务治理**: 监控、链路追踪、OpenTelemetry、服务注册与发现、Sentinel
- 代码生成**: 内置代码生成工具可基于 Thrift 和 Protobuf 的 IDL 生成 Hertz 项目的脚手架。



### 3.3.3 重点开源项目 – Volo



Volo 是 Rust 微服务 RPC 框架

- **多协议:** 默认支持 Thrift 和 gRPC，并拆分为两个独立的框架
- **中间件抽象:** 业界首个基于 GAT + TAIT 设计
- **高性能:** 较 Tower 提升较大，框架部分开销基本可以忽略不计
- **代码生成:** 内置代码生成工具可支持生成 Thrift、Protobuf 以及脚手架代码
- **扩展性强:** 服务发现、负载均衡等服务治理功能，都可以以 Service 形式进行实现，而不需要独立实现 Trait。

### 3.3.4 重点工具项目 – cwgo

cwgo 是 CloudWeGo All in one 代码生成工具，整合了各个组件的优势，提高开发者提体验。cwgo 工具支持 thrift 和 protobuf 的 IDL，可以方便生成工程化模版，其主要功能特点包括：用户友好生成方式、支持生成 MVC 工程化模板、支持生成 Server & Client 代码、支持生成数据库代码等。

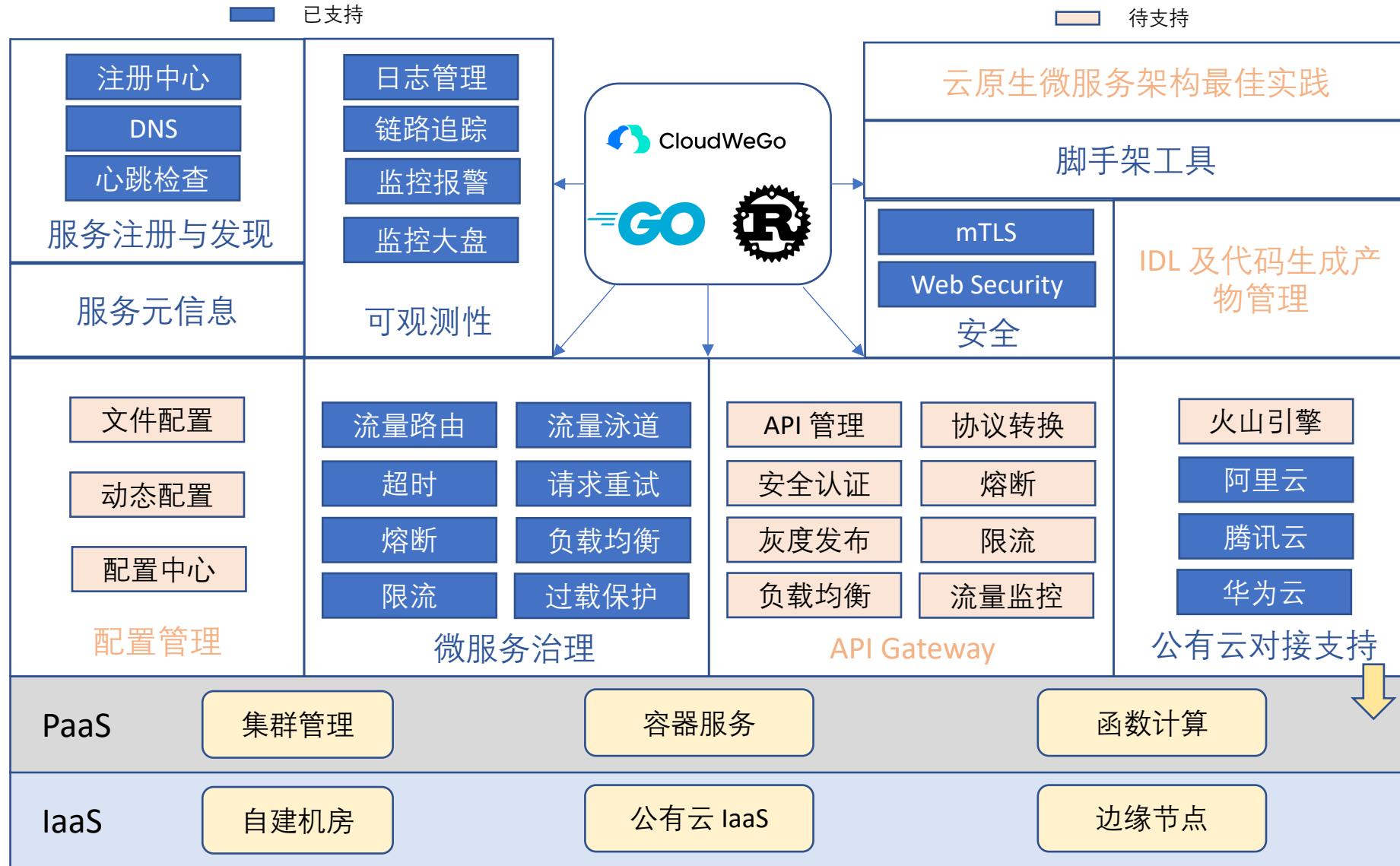
```
└── biz // 业务逻辑目录
    ├── dal // 数据访问层
    │   └── init.go
    ├── mysql
    │   └── init.go
    └── redis
        └── init.go
    └── handler // view 层
        └── hello
            └── example
                ├── hello_service.go // handler 文件，用户在该文件里实现 IDL service 定义的方法，update 时会查找当前文件已有的 handler
                └── hello_service_test.go // 单测文件
    └── router // idl 中定义的路由相关生成代码
        └── hello
            └── example // hello/example 对应 thrift idl 中定义的namespace；而对于 protobuf idl，则是对应 go_package 的最后一级
                ├── hello.go // cwgo 为 hello.thrift 中定义的路由生成的路由注册代码；每次 update 相关 idl 会重新生成该文件
                └── middleware.go // 默认中间件函数，hz 为每一个生成的路由组都默认加了一个中间件；update 时会查找当前文件已有的 middleware
    └── register.go // 调用注册每一个 idl 文件中的路由定义；当有新的 idl 加入，在更新的时候会自动插入其路由注册的调用；勿动
    └── service // service 层，业务逻辑存放的地方。更新时，新的方法会追加文件。
        ├── hello_method.go 具体的业务逻辑
        └── hello_method_test.go
    └── utils // 工具目录
        └── resp.go
    └── build.sh // 编译脚本
    └── conf // 存放不同环境下的配置文件
        └── ...
    └── docker-compose.yaml
    └── go.mod // go.mod 文件，如不在命令行指定，则默认使用相对于GOPATH的相对路径作为 module 名
    └── hertz_gen // IDL 内容相关的生成代码
        └── ...
    └── idl
        └── hello.thrift
    └── main.go // 程序入口
    └── readme.md
    └── script // 启动脚本
        └── bootstrap.sh
```

```
└── biz // 业务逻辑目录
    ├── dal // 数据访问层
    │   └── init.go
    ├── mysql
    │   └── init.go
    └── redis
        └── init.go
    └── service // service 层，业务逻辑存放的地方。更新时，新的方法会追加文件。
        ├── HelloMethod.go
        └── HelloMethod_test.go
    └── build.sh
    └── conf // 存放不同环境下的配置文件
        └── ...
    └── docker-compose.yaml
    └── go.mod // go.mod 文件，如不在命令行指定，则默认使用相对于GOPATH的相对路径作为 module 名
    └── handler.go // 业务逻辑入口，更新时会全量覆盖
    └── idl
        └── hello.thrift
    └── kitex.yaml
    └── kitex_gen // IDL 内容相关的生成代码，勿动
        └── ...
    └── main.go // 程序入口
    └── readme.md
    └── script // 启动脚本
        └── bootstrap.sh
```

### 3.3.5 其它重点开源项目

- [bytedance/sonic](#): A blazingly fast JSON serializing & deserializing library, accelerated by JIT and SIMD.
- [bytedance/monoio](#): A thread-per-core Rust runtime with io\_uring/epoll/kqueue.
- [Netpoll](#): A high-performance non-blocking I/O networking framework focusing on RPC scenarios.
- [Thriftgo](#): An implementation of thrift compiler in go language.
- [Frugal](#): A very fast dynamic Thrift serializer & deserializer.
- [Fastpb](#): A faster Protobuf serializer & deserializer.
- [Pilota](#): A thrift and protobuf implementation in pure rust with high performance and extensibility.
- [Motore](#): Async middleware abstraction powered by GAT and TAIT.
- [Dynamicgo](#): Dynamically and efficiently operate data for Go.
- [Shmipc](#): A high performance inter-process communication library built on Linux's shared memory technology.

## 3.4 项目能力域与定位



# 04

## 开源社区建设与运营

## 4.0 开源社区的重要目标

1. 生态建设

4. 影响力

2. (企业) 用户

3. 开发者/贡献者

## 4.1.1 生态建设与合作

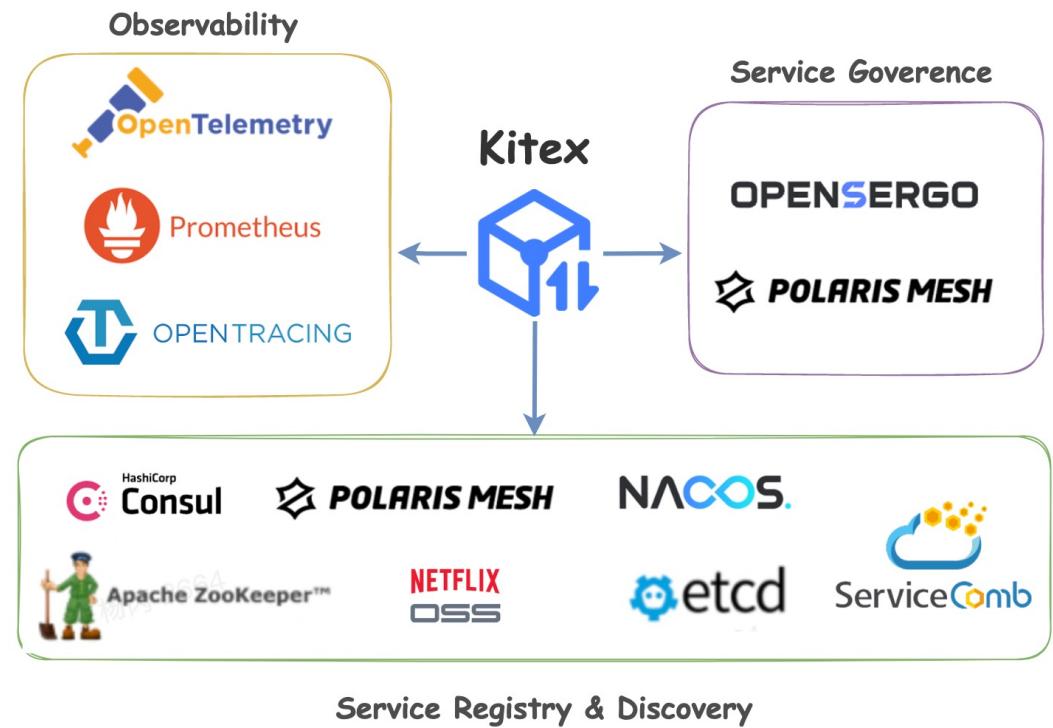
- 2021年11月，加入 CNCF Cloud Native Landscape
- 2022年3月，加入 NextArch 基金会微服务 SIG，积极推进开源微服务治理标准建设
- 2022年4月，启动 OpenSergo 开源共建，与阿里云、Bilibili 共同推进开源多语言微服务治理标准建设
- 2022 年4月，CloudWeGo-Kitex 正式支持了腾讯云 微服务引擎 TSE 的接入
- 2022 年5月，CloudWeGo-Kitex 正式支持了阿里云 MSE 注册中心的接入以及 ARMS 链路追踪的接入
- 2022 年9月，CloudWeGo-Kitex 正式支持了华为云 微服务引擎 CSE 的接入
- 2022年9月，通过信通院发起的可信开源社区测评，通过后正式加入可信开源社区共同体

## 4.1.2 生态建设与合作

- Kitex：积极和上下游开源项目深度合作与集成，建设完整基于开源的微服务解决方案体系

The screenshot shows the GitHub repository page for 'kitex-contrib'. The repository is described as a place for various extensions in the Kitex ecosystem. It has 70 followers and a public URL: <https://github.com/cloudwego/kitex>. The 'Pinned' section displays six projects:

- monitor-prometheus** (Public): Prometheus monitoring for your kitex services. (Go, 14 stars, 3 forks)
- registry-etcd** (Public): Etcd as service registry for Kitex. (Go, 16 stars, 13 forks)
- registry-nacos** (Public): Nacos as service registry for Kitex. (Go, 18 stars, 13 forks)
- obs-opentelemetry** (Public): Observability integration with OpenTelemetry. (Go, 16 stars, 11 forks)
- opensergero** (Public): Service governance integration with OpenSergo. (Go, 4 stars, 3 forks)
- xds** (Public): XDS (eXtensible Data Services) integration. (Go, 23 stars, 2 forks)



<https://github.com/kitex-contrib>

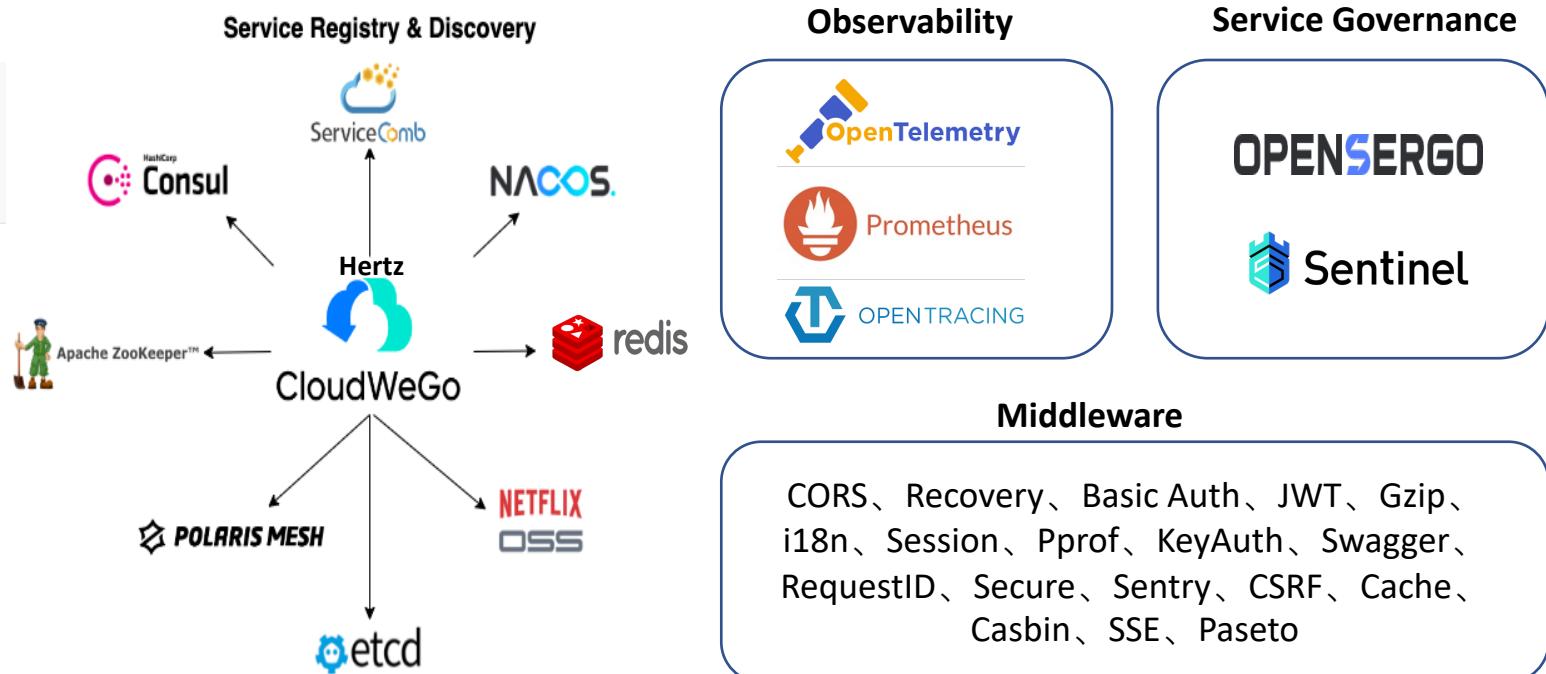
## 4.1.3 生态建设与合作

- Hertz: 积极和上下游开源项目深度合作与集成，建设完整基于开源的微服务解决方案体系

The screenshot shows the GitHub repository page for `hertz-contrib`. It displays a list of popular repositories that extend the Hertz ecosystem:

- `websocket`: Websocket for Hertz (Public)
- `registry`: The service registration & discovery extensions for Hertz (Public)
- `jwt`: JWT middleware for Hertz (Public)
- `obs-opentelemetry`: Opentelemetry for Hertz (Public)
- `swagger`: Swagger for Hertz (Public)
- `gzip`: gzip for hertz (Public)

Key statistics for the repository include 83 followers, 36 repositories, 23 people, and various commit counts.

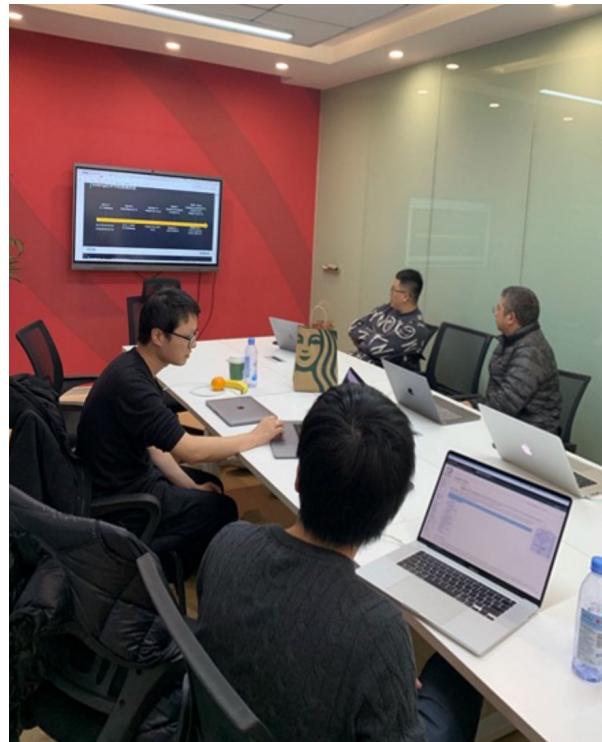


<https://github.com/hertz-contrib>

## 4.2.1 企业用户支持机制

1. 通用问题 --> GitHub Issue
2. 功能需求 & bug 提报 --> 飞书项目: <https://project.feishu.cn/cloudwego>
3. 技术交流沟通 & 紧急问题处理 --> 飞书群
4. 大型技术交流与培训 & 疑难杂症排查 --> 现场支持

Open	Closed	Author	Label	Milestones	Assignee	Sort
28	232					
0	(Hertz v0.5.1) 基于 thrift 生成代码, import 路径生成错误	zttaki	question	#586 opened 2 days ago by zttaki		2
0	The hertz(v 0.5.1) project cannot be build on Windows 32	Hang-che	bug	question		19
0	[QUESTION] - How to get Listener from server.Default() so that I could use it in different libraries?	sujit-banija	enhancement	question		1
0	Proposal(hz): Add configuration for modifying the layout structure provided by hz by default	Skycerought	proposal			4
0	hz 命令接口的讨论	errocks	question	#529 opened 3 weeks ago by errocks		8
0	need ETag / If-None-Match	ultraf	enhancement	good first issue		7
0	docs: optimize logger doc	L2ncE	good first issue	#488 opened on Dec 23, 2022 by L2ncE		
0	docs: add documentation of access_log	L2ncE	good first issue	#487 opened on Dec 23, 2022 by L2ncE		
0	Request help: Write a business demo code use Hertz and Hz	li-jin-gou	good first issue	#486 opened on Dec 23, 2022 by li-jin-gou	3 tasks	4
0	Add a more detailed example for service discovery.	li-jin-gou	good first issue	#485 opened on Dec 23, 2022 by li-jin-gou	7 tasks	4
0	HTTP cache - RFC compliant with many other features	darkweak	enhancement	#483 opened on Dec 22, 2022 by darkweak		1
0	Feature preparation: support HTTP/3 & QUIC for Hertz	welkeyever	enhancement	#458 opened on Dec 6, 2022 by welkeyever	8 tasks done	8
0	set server context default timeout		good first issue	question		7

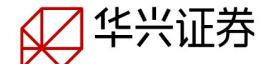


-  CloudWeGo-方正证券用户交流群(28) [外部](#)  
群消息更新于 5月29日
-  CloudWeGo - 小米用户交流群(7) [外部](#)  
群消息更新于 5月10日
-  CloudWeGo-浙江中核信息科技企业用户交流群(11) [外部](#)  
群消息更新于 5月25日
-  CloudWeGo-创梦天地用户交流群(10) [外部](#)  
群消息更新于 12:46
-  CloudWeGo - 贪玩游戏用户交流群(24) [外部](#)  
群消息更新于 15:03
-  CloudWeGo-中科类脑用户交流群(9) [外部](#)  
群消息更新于 昨天
-  CloudWeGo-MiniMax 用户交流群(41) [外部](#)  
群消息更新于 4月26日
-  CloudWeGo-星阑科技用户交流群(5) [外部](#)  
群消息更新于 5月4日
-  CloudWeGo-随申行用户交流群(12) [外部](#)  
群消息更新于 5月18日

## 4.2.2 CloudWeGo 落地企业用户 & 用户案例



谁在用



<https://www.cloudwego.io/zh/cooperation/>

<https://www.cloudwego.io/zh/>

## 4.3.1 SIG 划分 & 贡献者成长路径

### Community membership

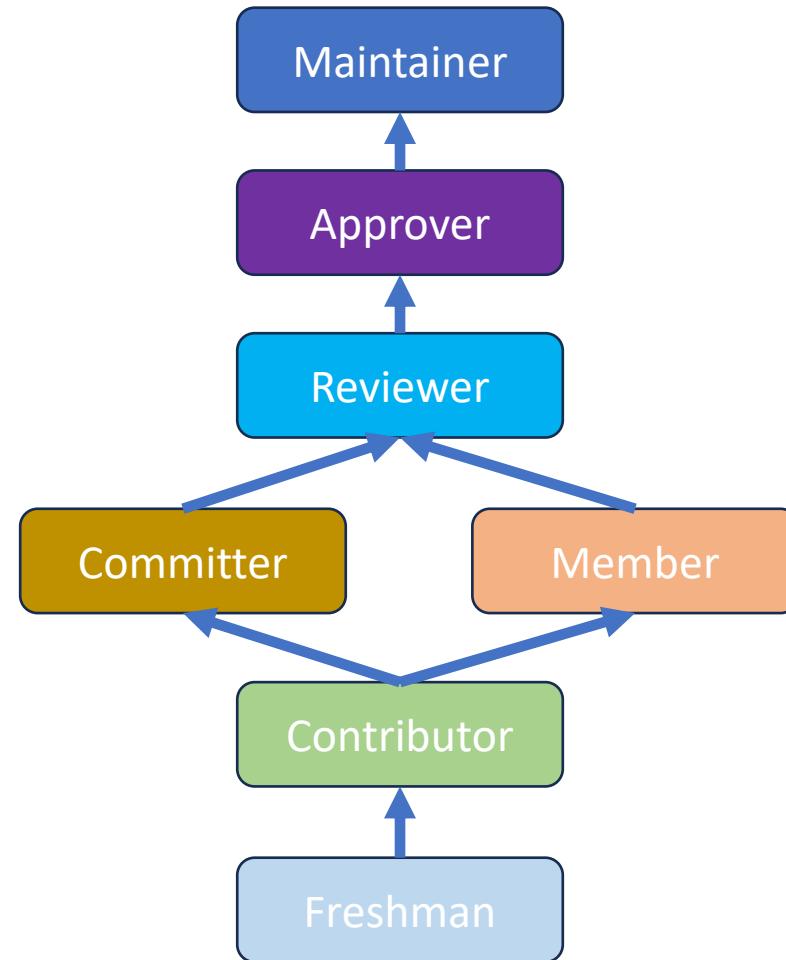
This doc outlines the responsibilities of contributor roles in CloudWeGo.

The CloudWeGo project is subdivided into subprojects under:

- Kitex (Kitex & Kitex ecosystem & kitex-contrib)
- Hertz (Hertz & Hertz ecosystem & hertz-contrib)
- Volo (Volo & Volo ecosystem & volo-rs & Motore & Pilota)
- Netpoll (Netpoll & Netpoll ecosystem)
- Monoio (Monoio & Monoio ecosystem)
- Serdes (Thriftgo & Frugal & Fastpb & Sonic & thrift-gen-validator)
- Shmipc (shmipc-spec & shmipc-go)
- Website & Docs (cloudwego.github.io & community)
- Others if may.

Responsibilities for roles are scoped to these subprojects (repos) defined by GitHub team.

Role	Responsibilities	Requirements	Defined by
Member	Active contributor in the community.	Sponsored by two approvers or maintainers. Multiple contributions to the project.	GitHub org member
Committer	Active code contributor and/or issue reply in the subproject.	Sponsored by two approvers or maintainers. Multiple code contributions to the project.	GitHub subproject committer Team
Reviewer	Review contributions from other members and give feedback and instructions.	Continuous history of review and authorship in a subproject.	GitHub subproject reviewer Team
Approver	Approve accepting contributions.	Highly experienced and active reviewer and contributor to a subproject.	GitHub subproject approver Team
Maintainer	Set direction and priorities for a subproject	Demonstrated responsibility and excellent technical judgement for the subproject.	GitHub subproject maintainer Team



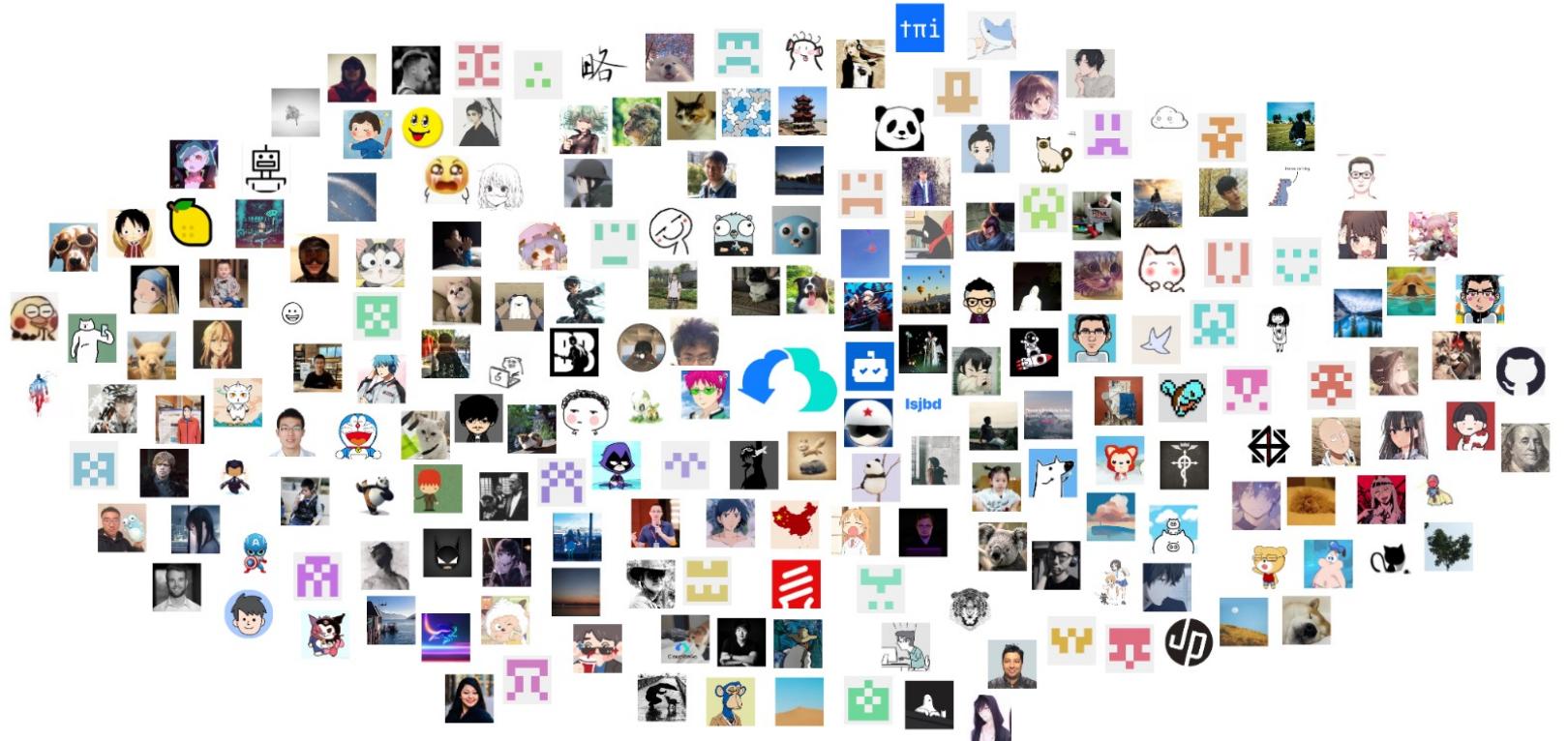
## 4.3.2 社区贡献者

社区的主体是人；

社区成员分为两大类，贡献者和用户；

贡献者即包括代码贡献的人，也包括非代码贡献的人；

代码/文档贡献很好度量，非代码贡献的角色也应当值得肯定。



感谢以上 200 多位 CloudWeGo 核心仓库的「代码/文档」贡献者

统计截止 2023.6.1

### 4.3.3 社区贡献者



## 4.3.4 活跃&核心贡献者的激励

### 精神激励

1. Committer/Reviewer 证书
2. 年度贡献者勋章
3. 参与社区讨论和决策
4. 在社区相关活动进行分享
5. 公众号曝光宣传
6. 大会布道机会



#### CloudWeGo 社区新晋 Reviewer 诞生

CloudWeGo 2023-04-25 17:30 发表于浙江



Hi all, 很高兴告诉大家，CloudWeGo 再次新晋一位 Reviewer 的同学：[rogerogers \(姜建齐\)](#)，已经正式通过提名！

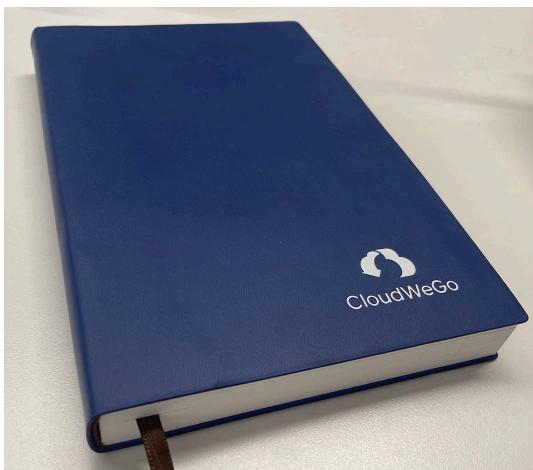
rogerogers (姜建齐) 不仅是一名优秀的后端研发工程师，还擅长前端和网站建设。成为 CloudWeGo Committer 7个月以来，持续致力于官网建设、文档优化、Hertz&Kitex Zap logger 扩展等方向，与社区长期间陪，积极参与社区 Issue 回复和交流。

希望未来 rogerogers (姜建齐) 可以和社区继续结伴前行，收获更多成果。欢迎更多同学参与到 CloudWeGo 社区建设中来，一起打造优质开源云原生微服务架构平台。

## 4.3.5 活跃&核心贡献者的激励

### 物质激励

1. CloudWeGo 多种周边礼品  
定期发放
2. 线下面对面聚会与聚餐
3. 节日红包
4. 职业发展指导 & 内推



## 4.4.1 社区成熟度能力 – 三先进



## 4.4.2 可信开源星云象限 – 领导型



2023.4.21，中国信通院发布可信开源项目星云象限，从项目可持续性与可信性两个维度，将可信开源共同体项目分为四个阶段，分别为孵化型、成长型、挑战型、领导型。

第一批可信开源项目星云象限共选取人工智能、数据库、中间件、操作系统、大数据、云原生平台6个技术领域共16个项目进行测评，**其中 CloudWeGo 位于领导型象限。**

## 4.4.3 外部荣誉



CloudWeGo 入选 OSCHINA/ segmentfault / CSDN 2022 年度开源榜单

## 4.4.3 外部荣誉

### 中国云原生技术生态图谱

极客邦科技双数研究院



4

CloudWeGo 核心子项目齐齐入选 中国云原生技术生态图谱 (2022)

## 4.4.4 知名开源大赛获奖



## 4.5 Community Over Code

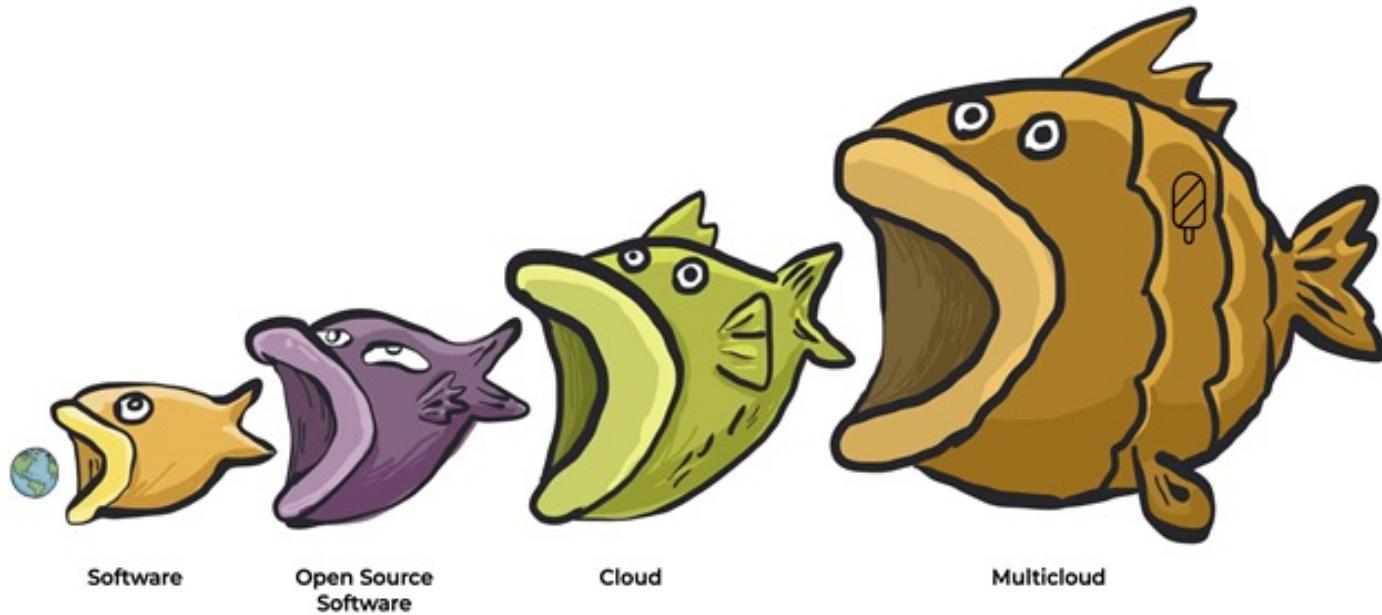
The most successful long-lived projects value a broad and collaborative community over the details of the code itself.



# 05 开源实践入门

## 5.1 开源初衷

软件吞噬世界，开源吞噬软件



Open Source For Good !

## 5.2 为什么参与开源

### 收益

获得开源技能、专业知识、编程能力等硬实力的提升

管理、沟通、协作等软实力的提升

结交朋友、扩展人脉、提升视野

获得荣誉、获得大会演讲机会，提升个人影响力

获得成就感和幸福感

更多、更好的工作机会

Just for fun !

## 方法和建议

### 5.3 如何参与开源

善用官方文档

善用搜索引擎

使用 GitHub Issue & Discussion

动手实操

保持礼貌

积极主动

通过贡献赢得尊重、地位和影响力

## 5.3.1 提问的智慧

### How To Ask Questions The Smart Way

- Copyright © 2001,2006,2014 Eric S. Raymond, Rick Moen

#### 在提问之前

在你准备要通过电子邮件、新闻群组或者聊天室提出技术问题前，请先做到以下事情：

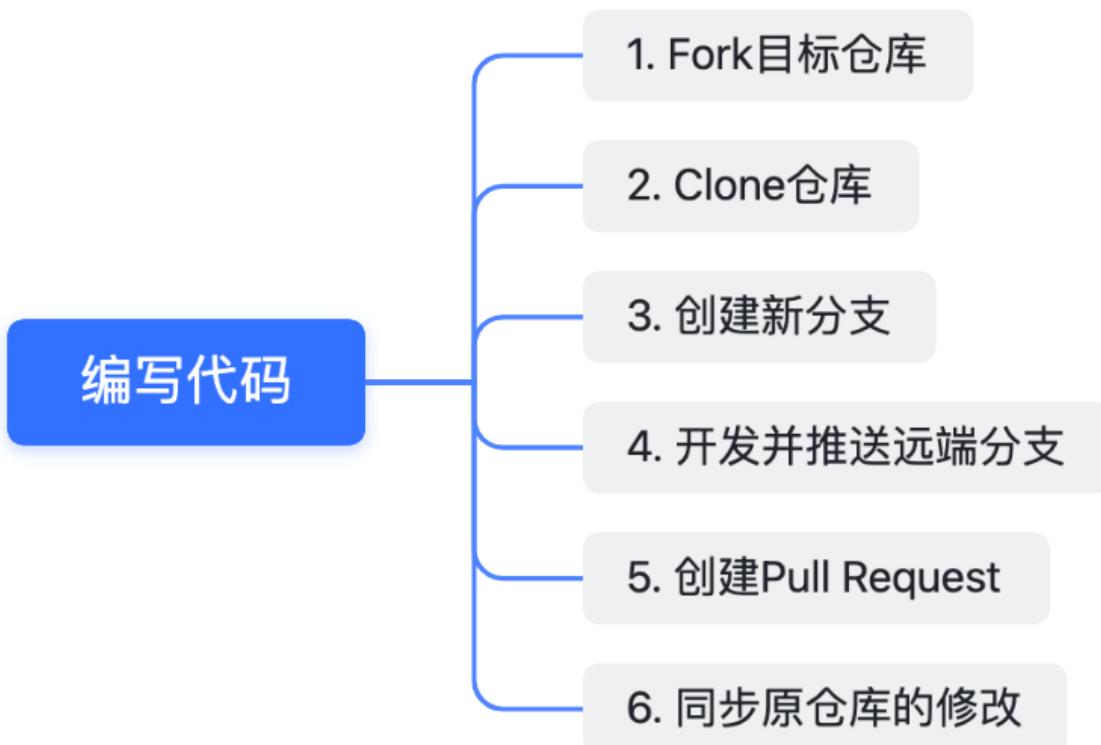
1. 尝试在你准备提问的论坛的旧文章中搜索答案。
2. 尝试上网搜索以找到答案。
3. 尝试阅读手册以找到答案。
4. 尝试阅读常见问题文件（FAQ）以找到答案。
5. 尝试自己检查或试验以找到答案。
6. 向你身边的强者朋友打听以找到答案。
7. 如果你是程序开发者，请尝试阅读源代码以找到答案。

#### 当你提问时

- 
1. 使用有意义且描述明确的标题
  2. 使用清晰、正确、准确且合乎语法的语句
  3. 准确地表述问题并言之有物
  4. 话不在多而在精
  5. 别动则声称找到 Bug
  6. 描述问题症状而非你的猜测
  7. 按发生时间先后列出问题症状
  8. 礼多人不怪，而且有时还很有帮助

[https://github.com/ryanhanwu/How-To-Ask-Questions-The-Smart-Way/blob/main/README-zh\\_CN.md](https://github.com/ryanhanwu/How-To-Ask-Questions-The-Smart-Way/blob/main/README-zh_CN.md)

## 5.3.2 如何提交 PR



1. git checkout -b [BRANCH NAME]
2. git status
3. git add [YOUR FILE]
4. git commit -m [YOUR COMMIT MESSAGE]
5. git push origin [YOUR BRANCH]

Add PR description, 通常情况下我们需要描述以下信息

- 1.这个 PR 是什么类型的，是 feature 还是fix 等
- 2.这个 PR 是做什么的
- 3.这个 PR 是否解决了某些issue

参考来源：[开源101之如何给开源库\(Hertz\)提PR](#)

### 5.3.3 同步原仓库的修改

在你开发的过程中，可能会有其他开发同学已经把自己的PR merge到main分支了。那么这时候我们就把修改merge到自己的开发分支。

1. 添加原始仓库作为upstream仓库

```
git remote add upstream [HTTPS]
```

2. 获取原始仓库的变更

```
git fetch upstream
```

3. 合并更改

```
git rebase upstream/main
```

4. 解决冲突并 push 到自己的开发分支

```
git push origin [YOUR BRANCH]
```

参考来源：[开源101之如何给开源库\(Hertz\)提PR](#)

## 更多开源和技术内容

CloudWeGo 官网

<https://cloudwego.io>

GitHub 仓库地址

<https://github.com/cloudwego>

CloudWeGo 微信公众号



## 课堂练习

### CloudWeGo 开源贡献实操

- <https://github.com/cloudwego/pilota/issues/192>
- <https://github.com/cloudwego/pilota/issues/190>
- <https://github.com/cloudwego/pilota/issues/152>
- <https://github.com/cloudwego/volo/issues/225>
- <https://github.com/cloudwego/volo/issues/226>

THANKS

