



高性能 RPC 框架 Kitex 内外统一的开源实践

杨芮

August, 2022

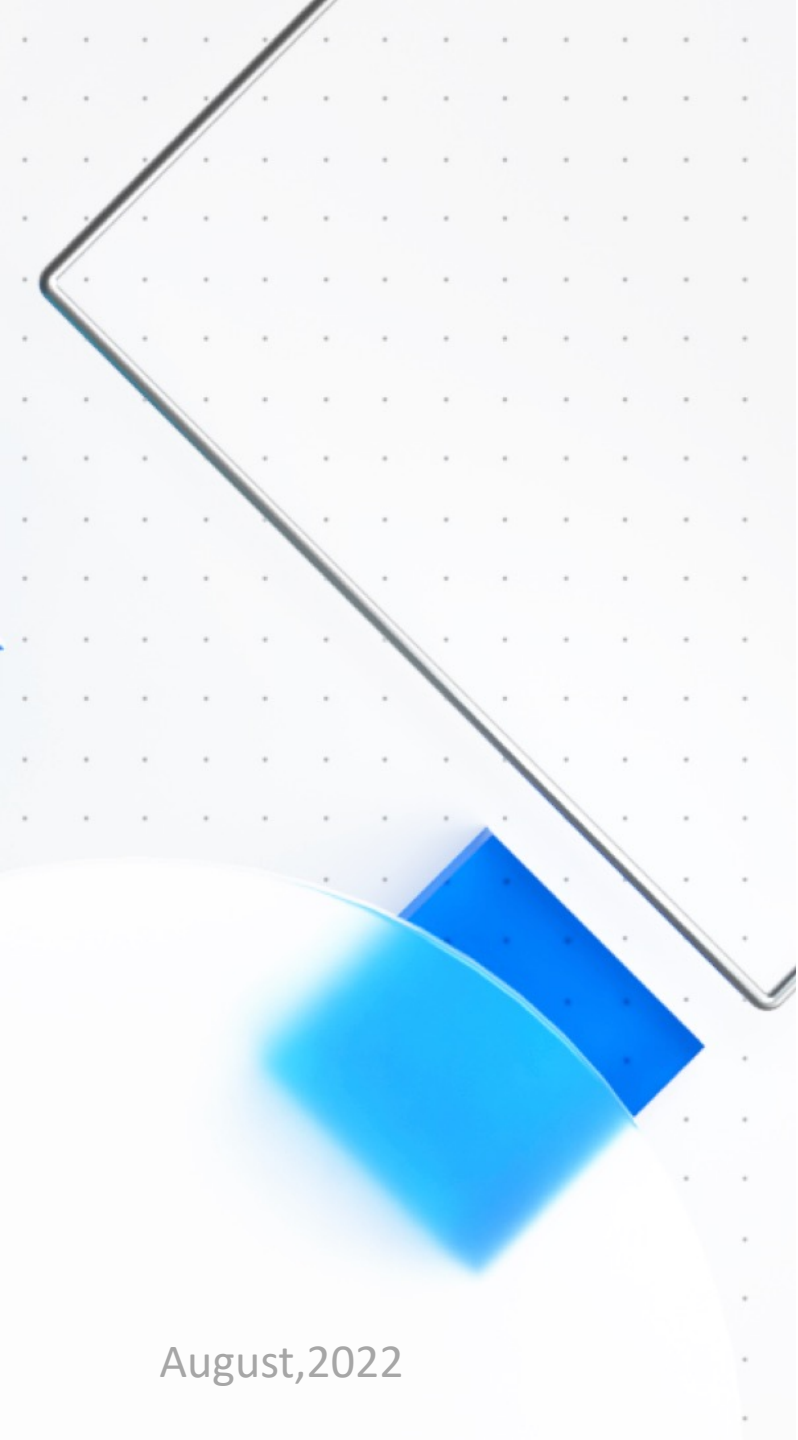
自我介绍



杨芮

Kitex 项目负责人

目前主要负责字节跳动 Golang 微服务框架的设计开发。
之前在美国负责 Java 服务框架设计开发。
QCon 2021 明星讲师。

- 
- 01 由内至外 – 开源过渡
 - 02 开源一年变更回顾
 - 03 社区共建完善生态及企业落地
 - 04 总结和展望
 - 04 版本发布和质量保障

01

由内至外 – 开源过渡

01_CloudWeGo 和 Kitex



www.cloudwego.io

Kitex 是 CloudWeGo 开源的第一个服务框架

服务框架



Kitex



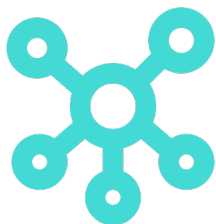
Hertz



Rust 框架

...

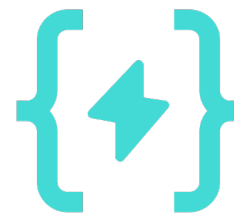
基础库



Netpoll



Thriftgo



Sonic

...

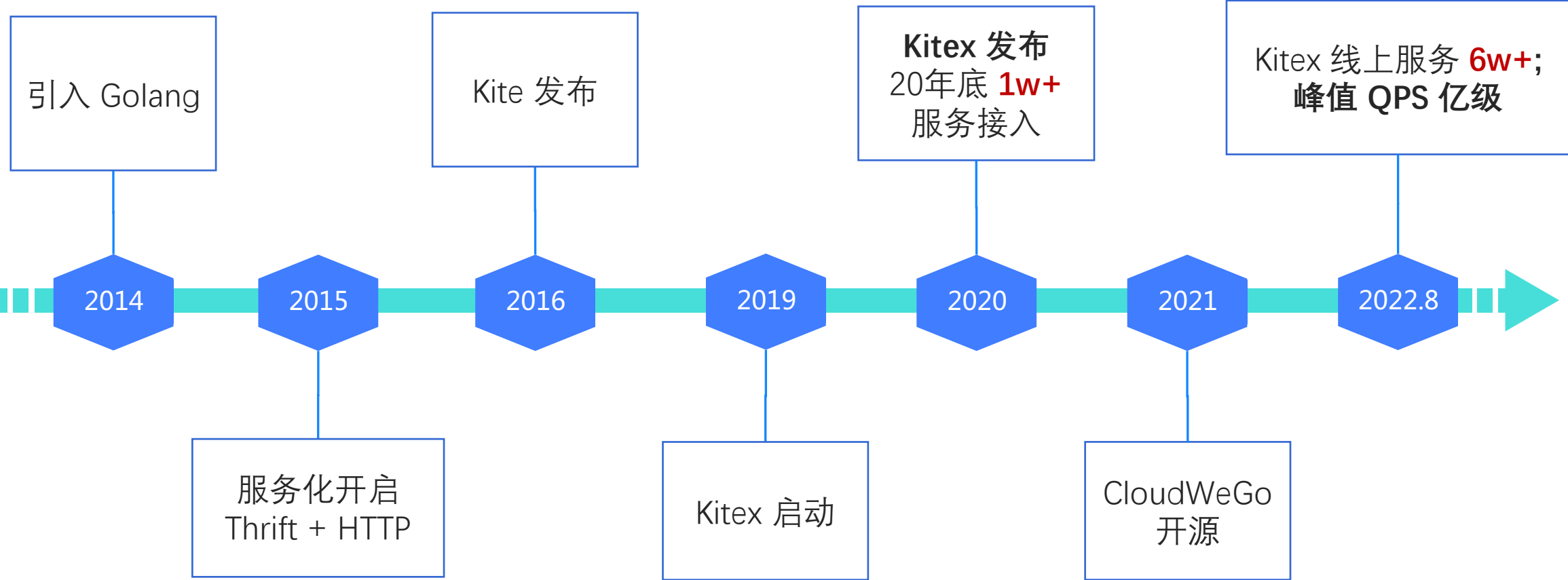
Frugal

Kitex 会不会是一个 KPI 项目？

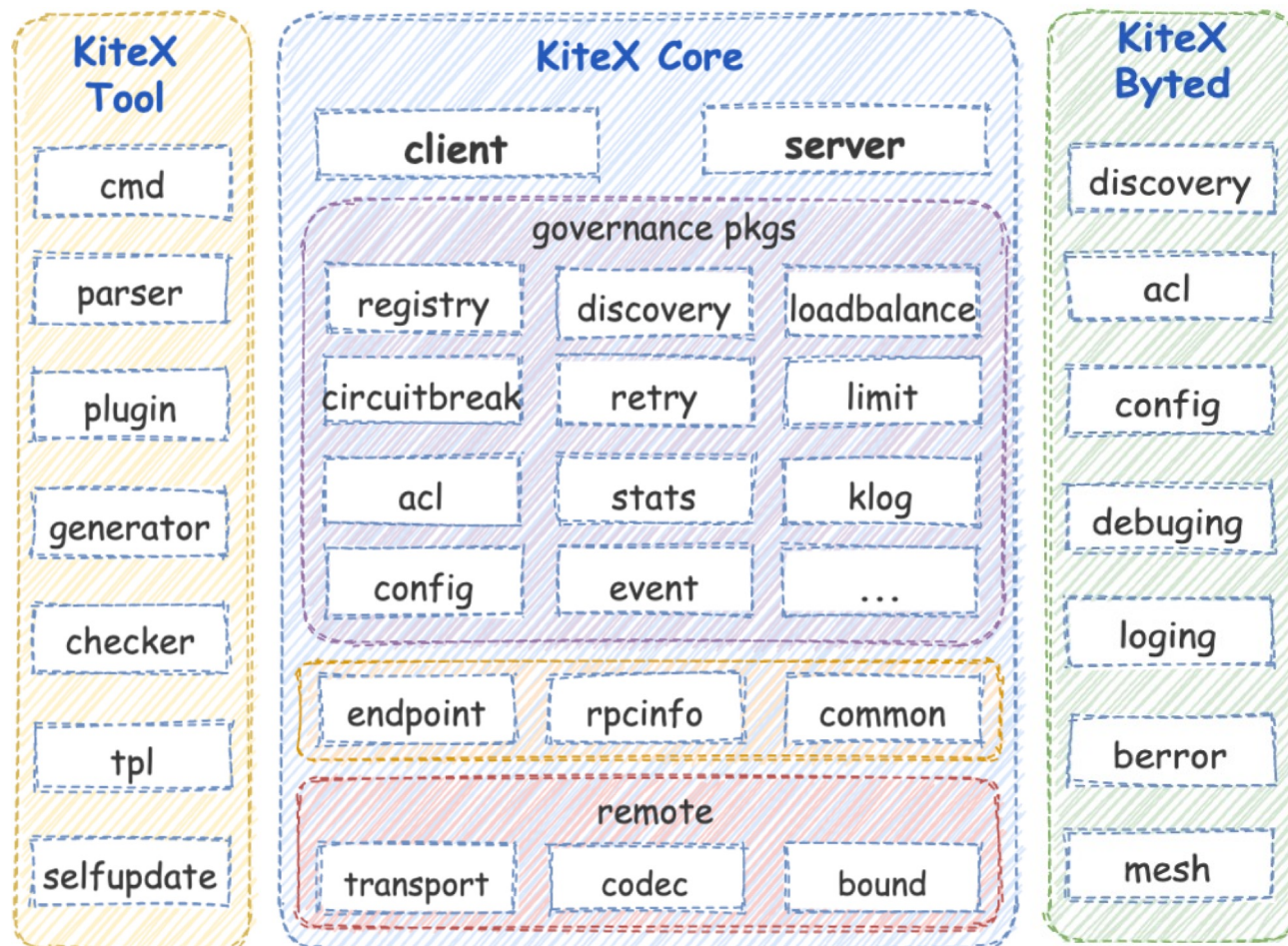


不是 KPI 项目！！
企业实践、内外统一、持续迭代

01_由内至外 – 开源过渡



01_由内至外 – 开源过渡

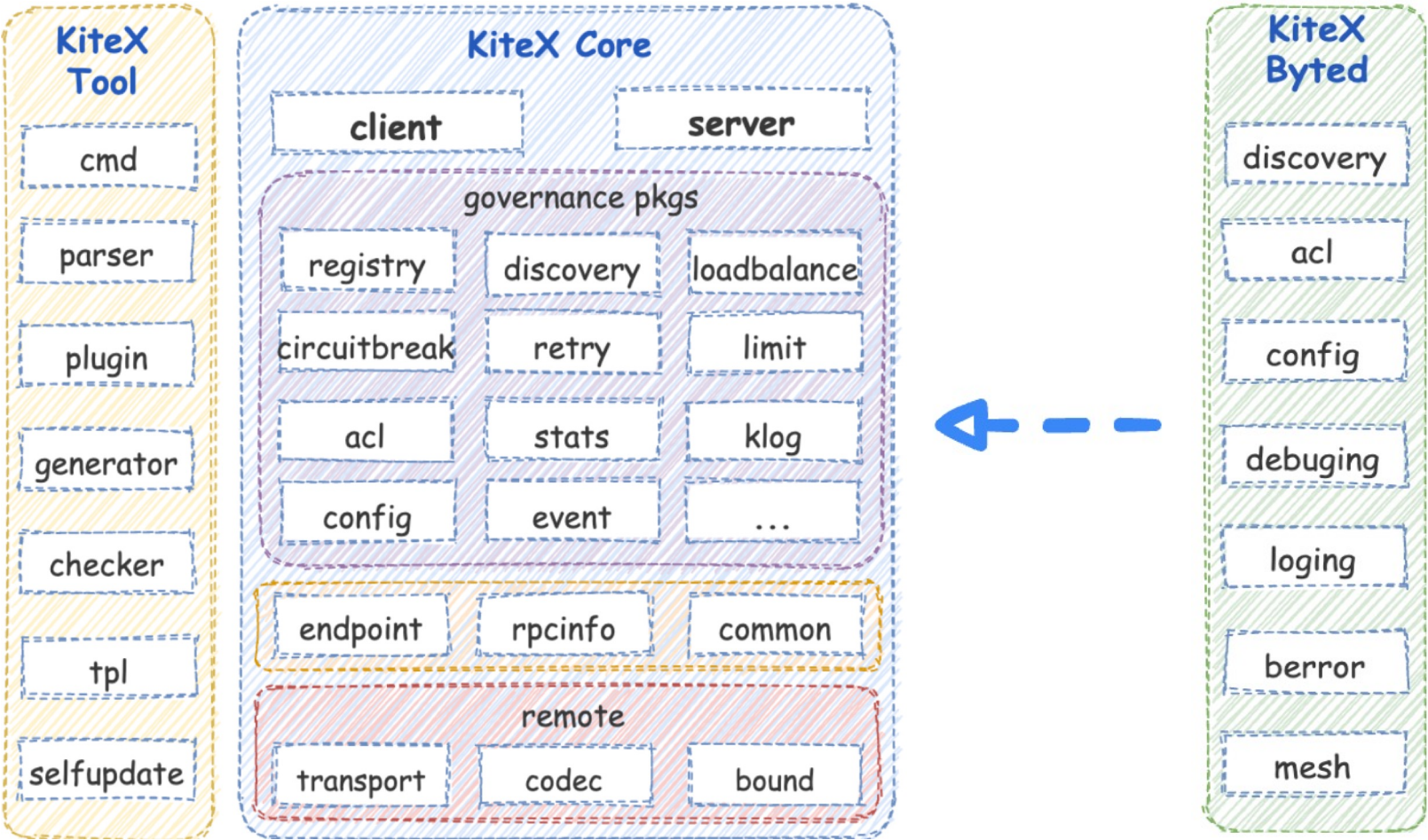


- **Kitex 核心**
API 定义、核心逻辑实现以及对各 API 的默认扩展
- **Kitex Byted**
与字节内部基础能力集成的扩展实现
- **Kitex Tool**
IDL 解析和代码生成器

01_由内至外 – 开源过渡

✈️ 开源部分 cloudwego/kitex

✈️ 内部扩展部分



01_由内至外 – 开源过渡

CloudWeGo



2021 年 2 月筹备开源

- 明确开源目标和项目集合
- 依赖库梳理，通用能力开源 bytedance/gopkg
- 调整代码，开源适配，代码拆分



2021 年 7 月正式开源

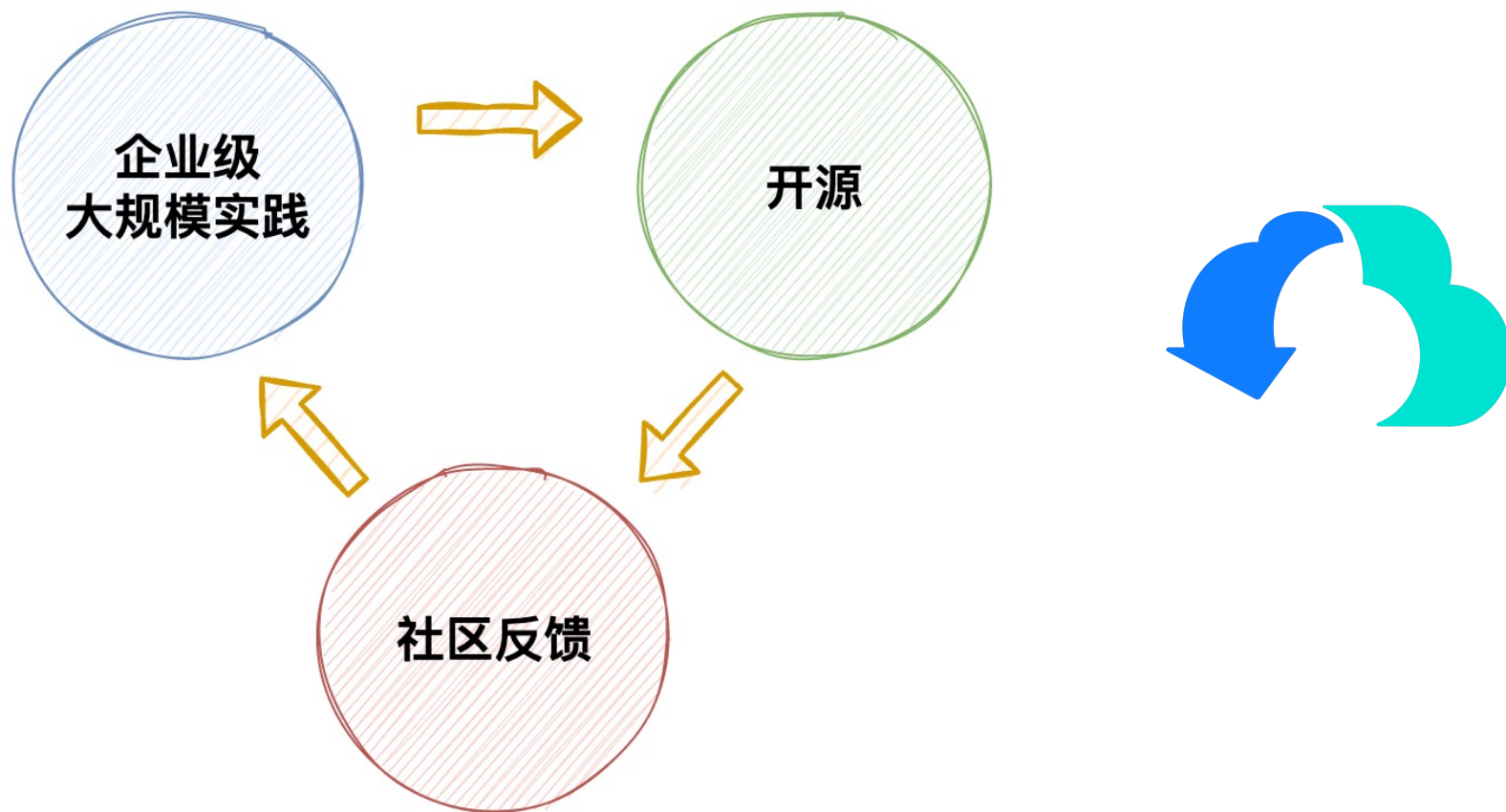
- 内部发布中版本，使用开源库
- 确保内部平滑过渡



2021 年 9 月官宣

01_由内至外 – 开源过渡

Kitex 不是为了开源而实现的，但它的实现是面向开源的~



02

开源一年变更回顾

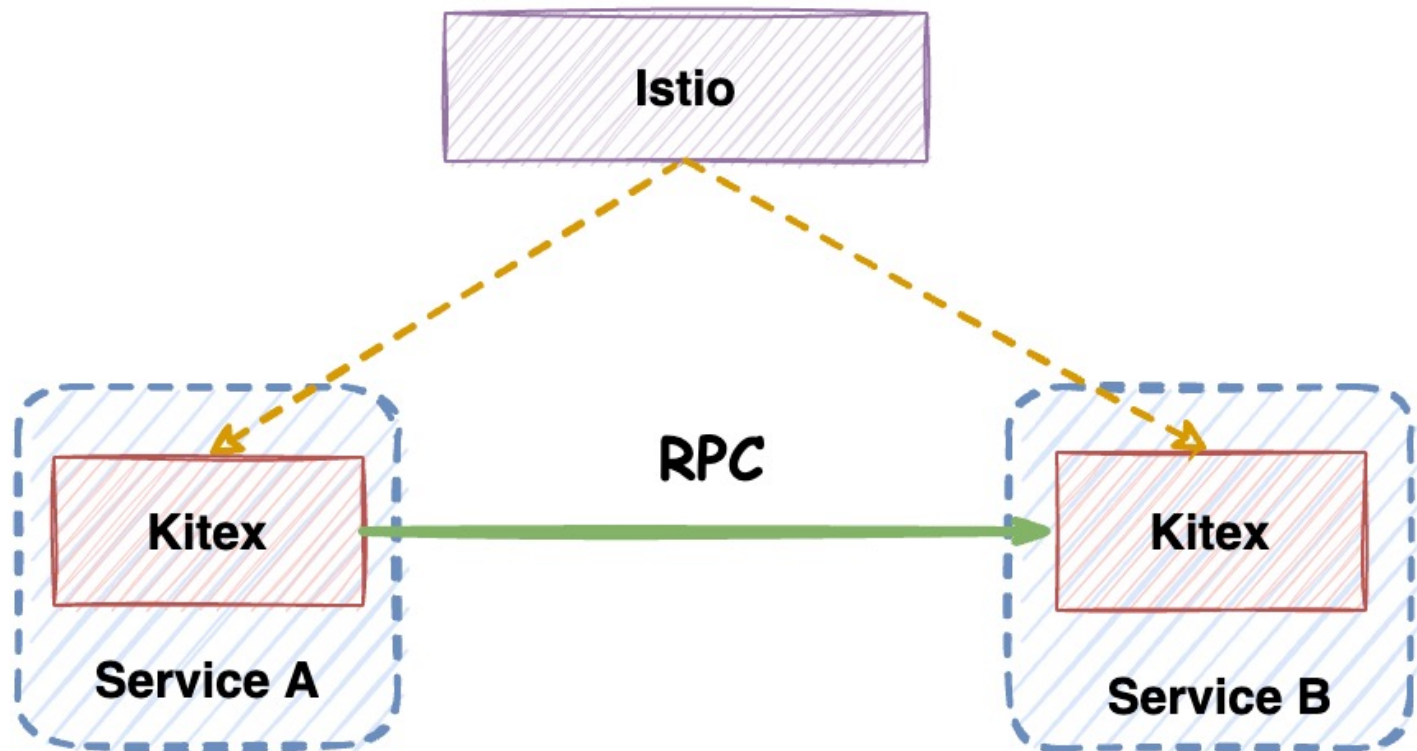
August, 2022

02_ 框架的衡量指标



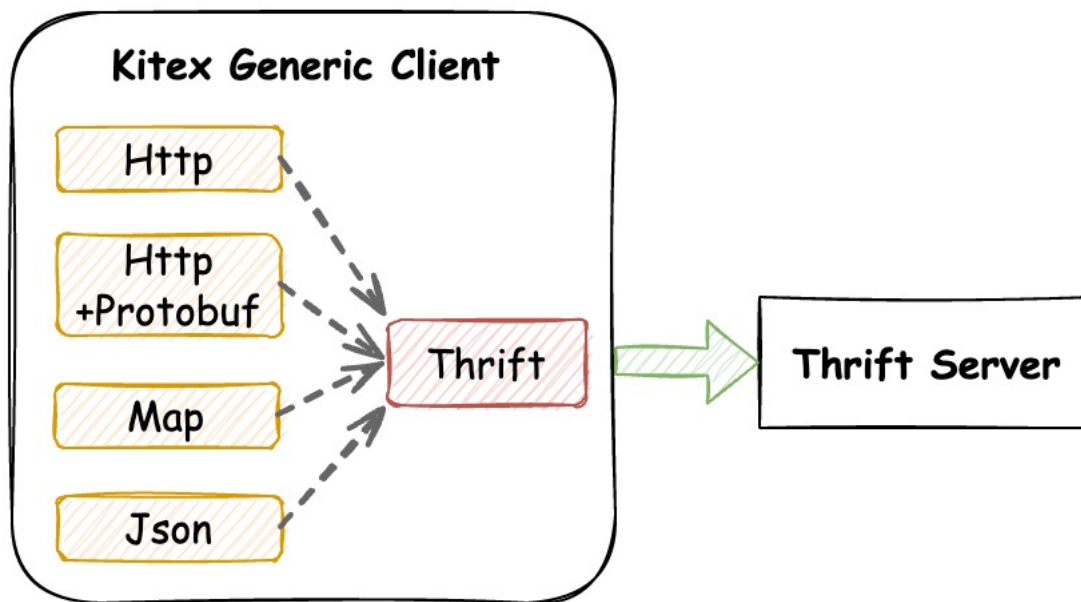
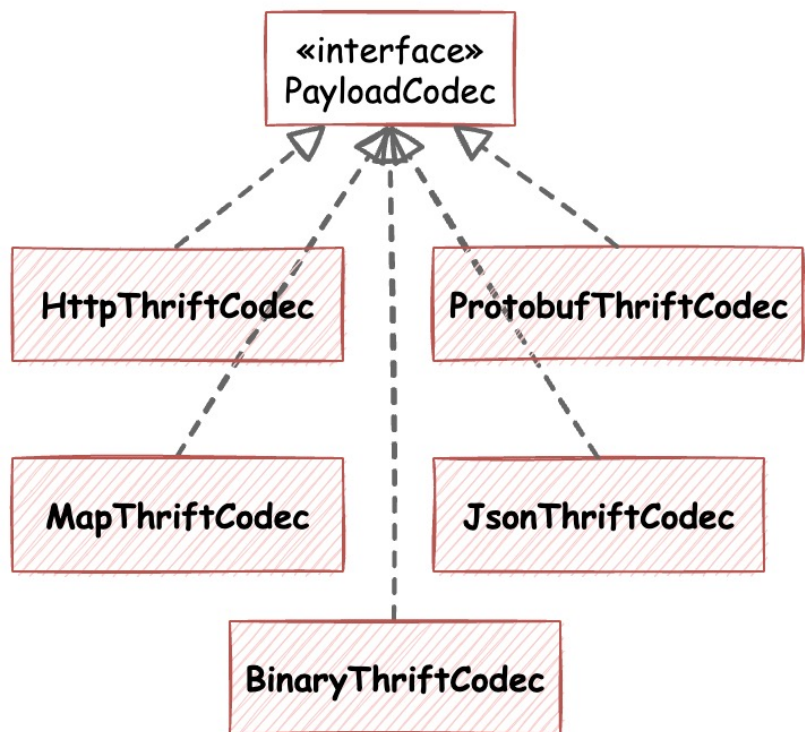
02_ 功能特性 — Proxyless

- 面向开源场景提供的支持



02_功能特性 — 新增 JSON 和 Protobuf 泛化调用支持

➤ 泛化调用相关的扩展实现



02_功能特性 — 重试能力增强

➤ 支持指定结果重试

```
var opts []client.Option
opts := append(opts, client.WithSpecifiedResultRetry(yourResultRetry))
xxxCli := xxxservice.NewClient("serviceName", opts...)
```

➤ 支持请求粒度重试设置

```
// demo1: call with failure retry policy, default retry error is Timeout
resp, err := cli.Mock(ctx, req,
callopt.WithRetryPolicy(retry.BuildFailurePolicy(retry.NewFailurePolicy()))
// demo2: call with backup request policy
resp, err := cli.Mock(ctx, req,
callopt.WithRetryPolicy(retry.BuildBackupRequest(retry.NewBackupPolicy(10)))
```

02_功能特性 — Thrift Validator

➤ thrift-gen-validator

thriftgo 工具的插件，根据 Thrift IDL 中定义的注解描述约束给对应的 struct 生成 IsValid() error 方法，校验值的合法性。

```
$ kitem --thrift-plugin validator -service a.b.c youridl.thrift
```

```
enum MapKey {  
    A, B, C, D, E, F  
}  
  
struct DemoTestRequest {  
    1: required string Message (vt.max_size = "30", vt.prefix = "Debug")  
    2: i32 ID (vt.gt = "1000", vt.le = "10000")  
    3: list<double> Values (vt.elem.gt = "0.25")  
    4: map<MapKey, binary> KeyValues (vt.key.defined_only = "true", vt.min_size = "1")  
}  
  
struct DemoTestResponse {  
    1: required string Message (vt.in = "Debug", vt.in = "Info", vt.in = "Warn", vt.in = "Error")  
}
```

02_性能优化/易用性提升 — Thrift 高性能编解码

➤ Frugal

github.com/cloudwego/frugal

无需生成编解码代码，基于 JIT 的高性能动态 Thrift 编解码器~

结合 Kitex 在大部分场景性能表现优于生成代码的编解码

	A	B	C	D	E	F	G
1	数据结构	连接类型	并发数	包大小	极限 TPS	延迟 TP99	延迟 TP999
2	复杂	[KITE-X-MUX]	100	1024	169131.15(+0.5%)	1.63(-0.6%)	2.13(-0.9%)
3		[KITE-X-MUX]	200	1024	187606.33(-0.7%)	2.69(+1.1%)	3.58(+3.8%)
4		[KITE-X]	100	1024	167613.9(+20.8%)	1.39(-22.3%)	7.92(+26.9%)
5		[KITE-X]	200	1024	165659.22(+18.2%)	2.83(+0.4%)	12.14(-13.3%)
6	简单	[KITE-X-MUX]	100	1024	315973.82(+0.2%)	1.07(+0.0%)	1.4(+0.0%)
7		[KITE-X-MUX]	200	1024	364755.89(+0.3%)	1.65(+0.6%)	2.11(+1.0%)
8		[KITE-X]	100	1024	232710.84(+7.7%)	0.69(-34.3%)	0.99(-66.1%)
9		[KITE-X]	200	1024	238802.41(+10.3%)	1.93(-12.7%)	2.75(-16.9%)

02_性能优化 – Protobuf 高性能编解码

➤ FastPB

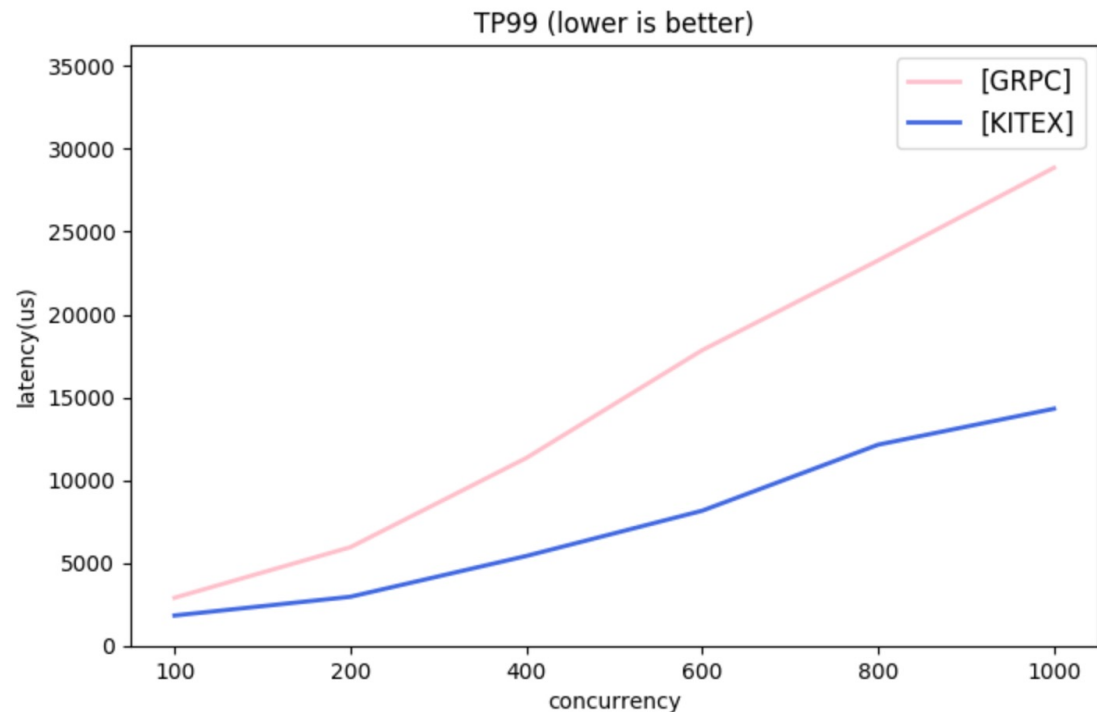
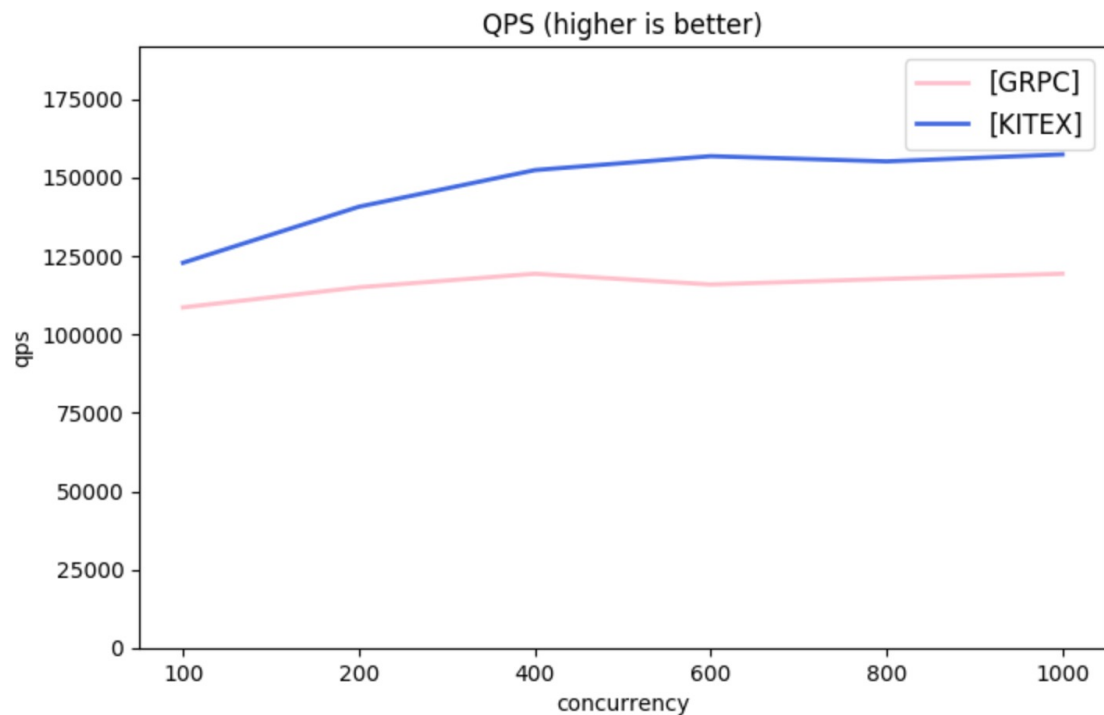
github.com/cloudwego/fastpb

参考 Thrift 的 FastWrite/Read 实现，重新设计了 Protobuf 生成代码，以独立开源

- Protobuf Benchmark 官方对比测试
 - FastWrite: (ns/op) ↓41.5% , (B/op) ↓4.5%
 - FastWrite: (ns/op) ↓67.8% , (B/op) ↓83.9%

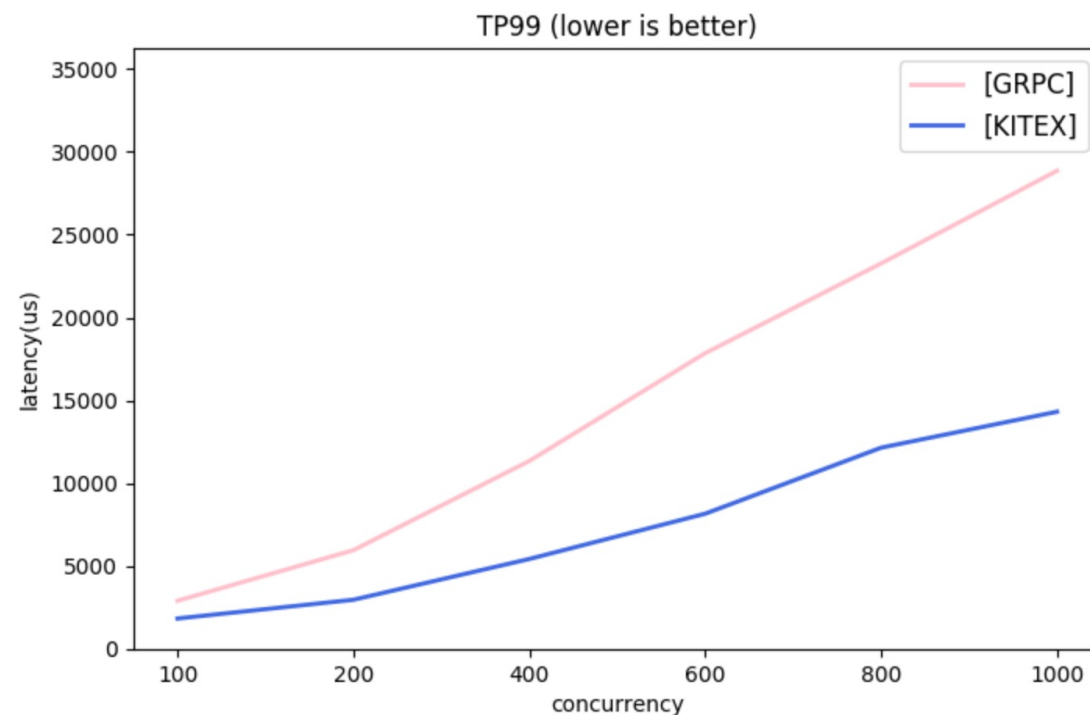
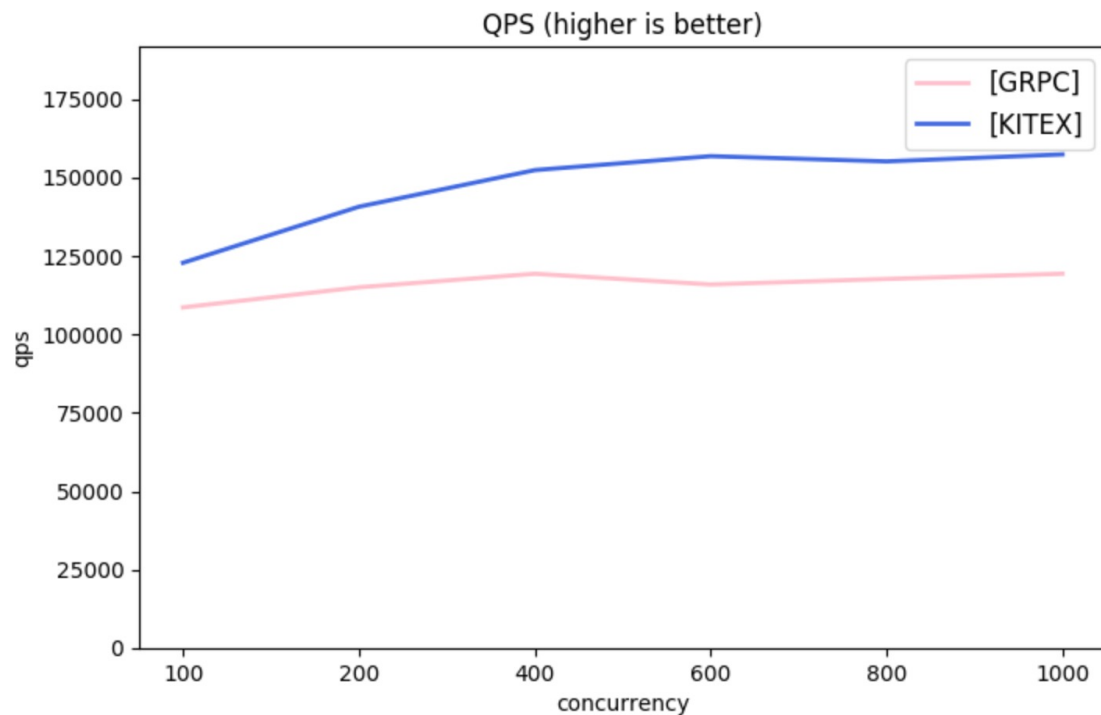
02_性能优化 — gRPC 性能优化

➤ gRPC 协议 Unary 请求压测对比



02_性能优化 — gRPC 性能优化

➤ gRPC 协议 Streaming 请求压测对比 TODO



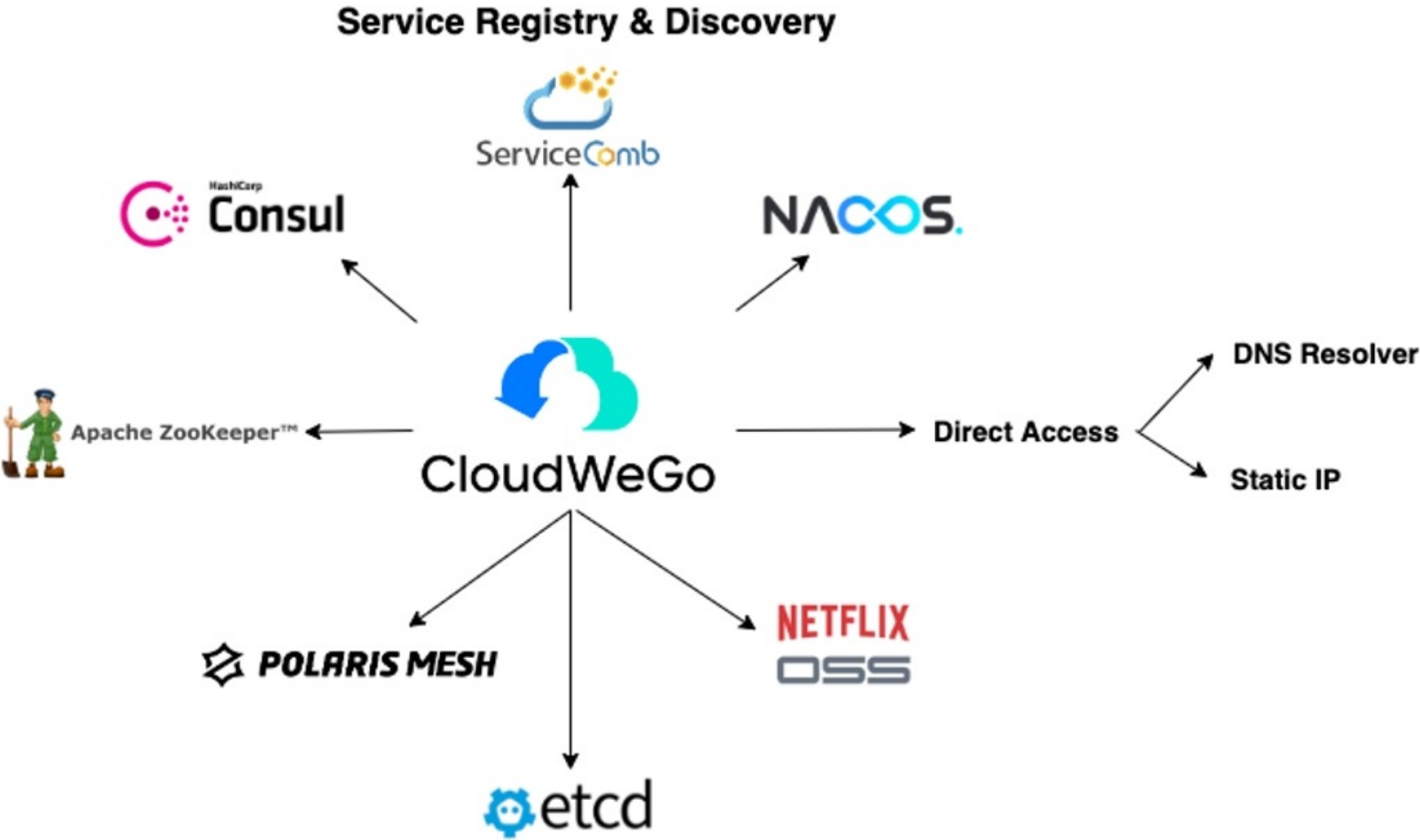
03

社区共建完善生态及企业落地

03_社区共建的 Kitex 扩展生态

感谢积极参与 CloudWeGo 社区建设同学们!

➤ kitex-contrib TODO 重画



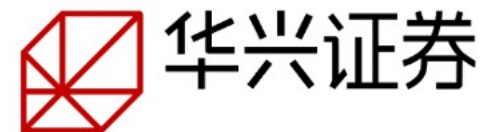
Observability



Service Governance



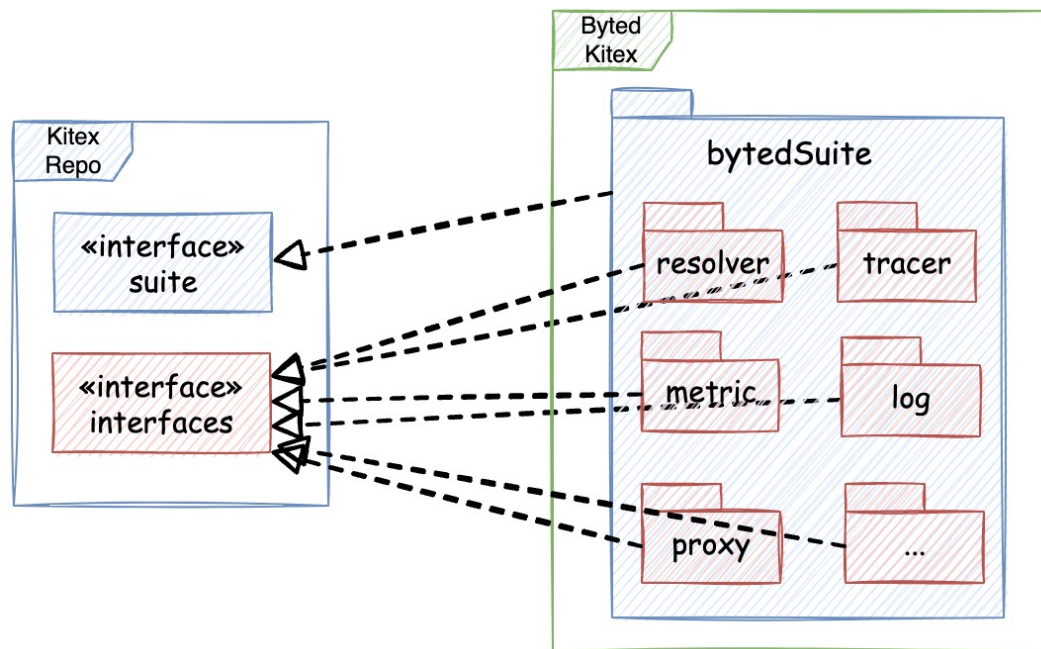
03_ 对接外部企业，协助落地



其他企业用户若有需求都可以联系我们~

03_ 如何使用 Kitex 与内部基础设施集成

➤ 以字节内部为例



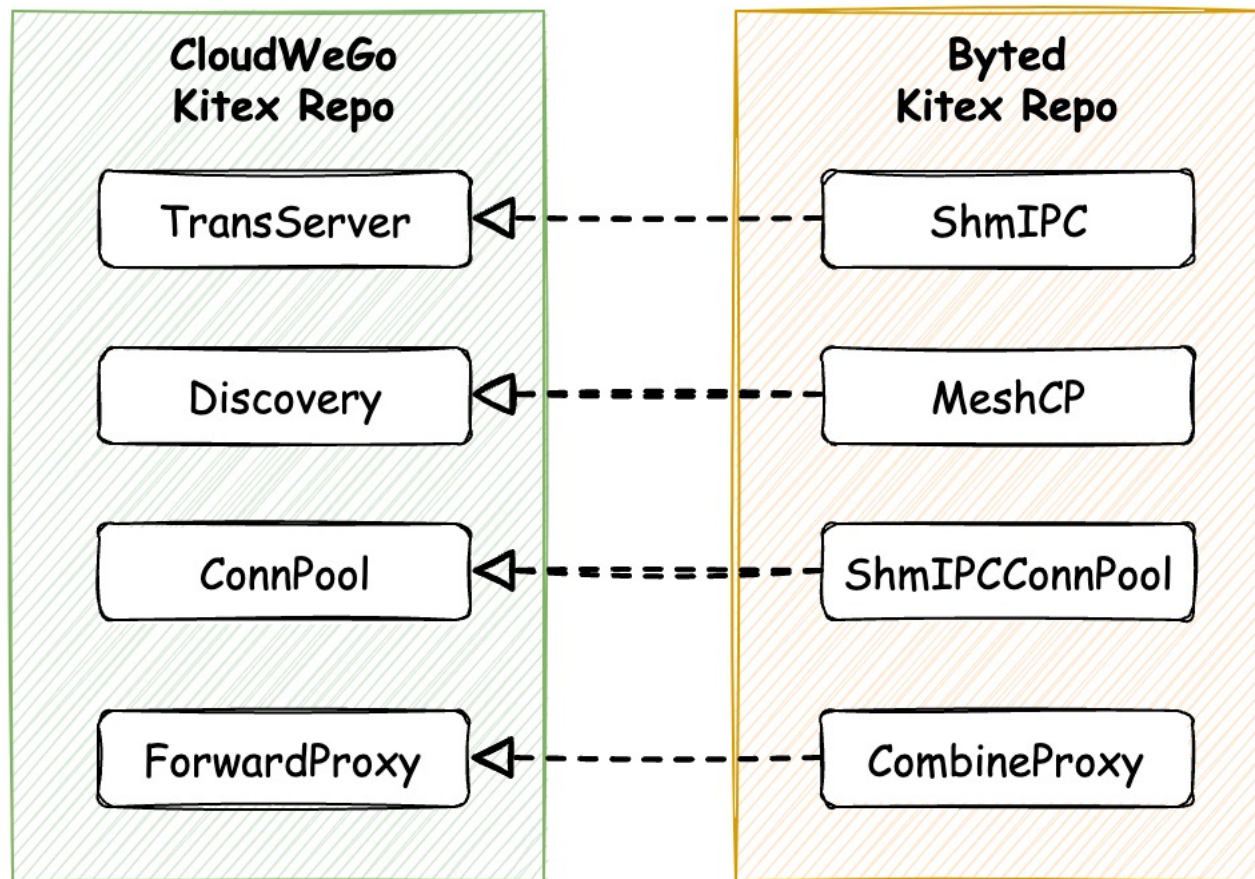
```
// Client 初始化 option
options = append(options, byted.ClientSuiteWithConfig(serviceInfo(), config))

// Server 初始话 option
options = append(options, byted.ServerSuiteWithConfig(serviceInfo(), config))
```

03_ 如何使用 Kitex 与内部基础设施集成

➤ 微服务合并部署

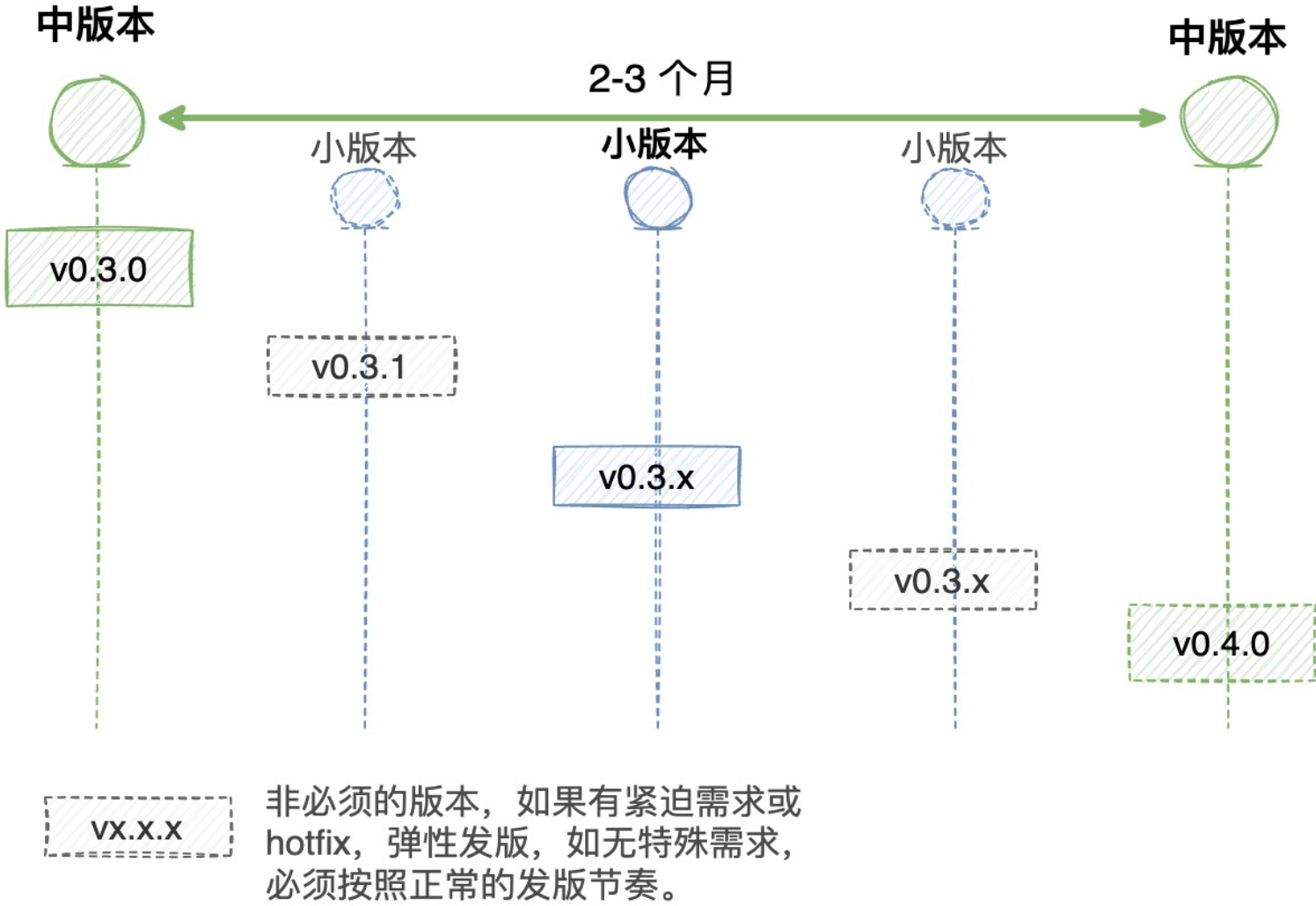
- 初始化条件判断 - 合并模式
- 监听多个地址
- 单独的服务发现
- 单独的连接池策略 (ShmIPC)
- 多通信方式: Shm、UDS、TC



04

版本发布和质量保障

04_版本发布



04_质量保障

➤ 稳定性保障（变更触发）

- ✓ 单元测试、集成测试
- ✓ 多场景混合的稳定性测试
- ✓ 异常测试

➤ 性能保障

- ✓ 多场景的性能压测（每日）
- ✓ 性能指标大盘

➤ 版本发布

- 发版前一周封禁 develop 非修复类变更，进行版本测试
- 输出版本测试报告

➤ 功能试点

- ✓ 测试服务验证充分后与需求方先行试点

05

总结和展望

05_总结

- Kitex 如何保持内外统一地从内部应用较广的框架转为开源框架？
- 开源一年以来发布了哪些重要功能特性？
- Kitex 的周边生态建设如何？
- Kitex 的发版节奏以及质量保障机制是什么？

05_展望

- 与社区同学共建，持续丰富社区生态
- 结合工程实践，为微服务开发者提供更多便利
- 完善好 BDThrift 生态，同时也会优化 Protobuf/gRPC
- 更多特性支持或开源，ShmIPC、QUIC、Protobuf 泛化…



THANKS

欢迎关注 CloudWeGo



August, 2022