百度APP Go 语言实践

陈肖楠

百度
chenxiaonan01@baidu.com

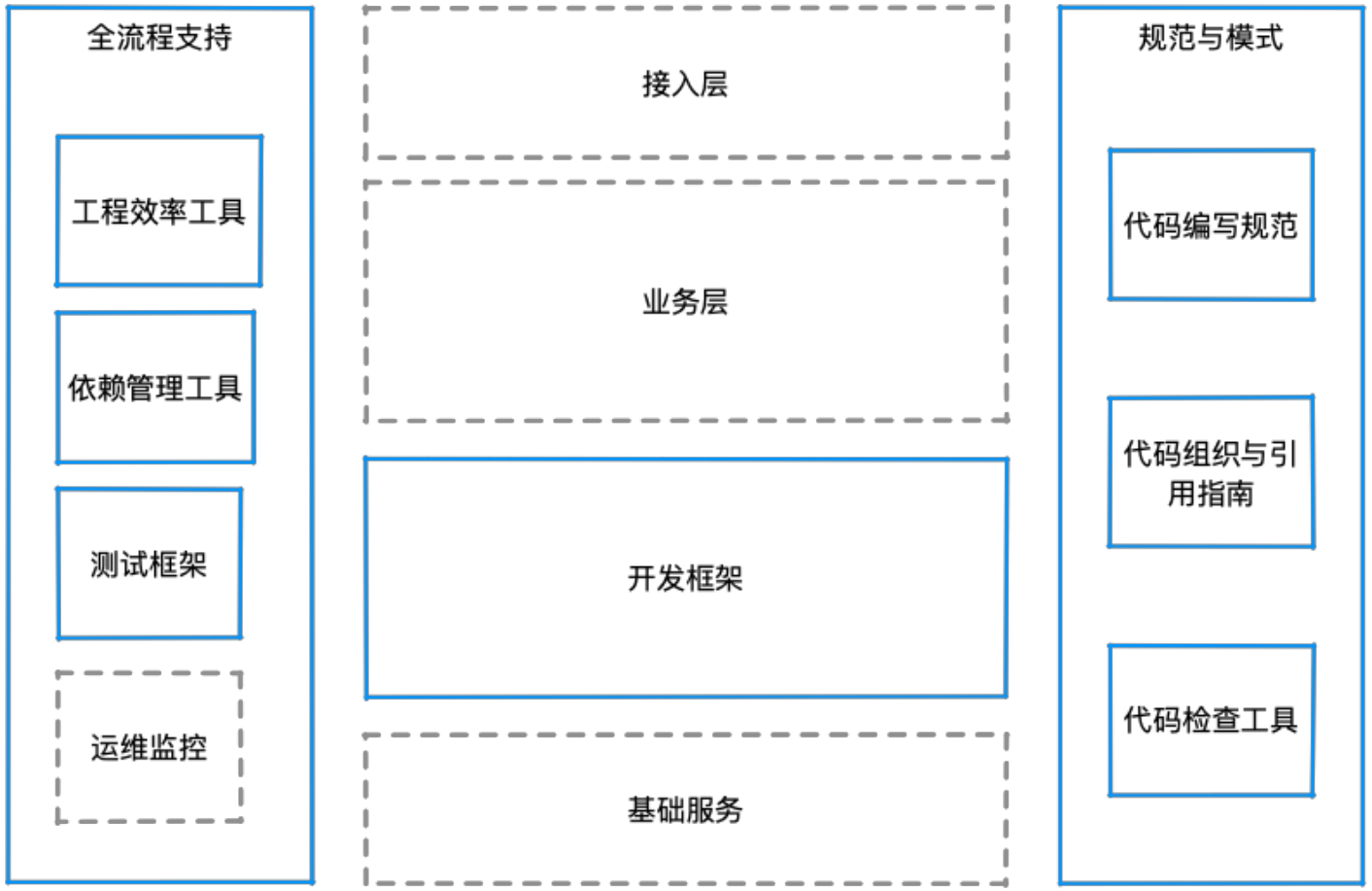
目录

开发规范


工程能力 Average
数据每日10:00更新
2019-04-16
64.0/100
工程能力总分
6.92/15
协作开发能力
37.99/55
质量保证能力
9.89/20
部署变更能力
9.22/10
流水线自动化能力
Weak
Poor
Average
Good
Excellent
怎样变更好
需求
需求管理
迭代管理
Bug管理
开发
本地编码规范
本地可维护性检查
本地重复性检查
本地测试
分支规范
提交规范
关联需求
代码准入
增量源码安全扫描
百度编码规范
CodeReview
增量静态代码扫描
云端编译
P0级自动化回归测试
Unit Test
可维护性检查
重复文件检查
测试
全量源码安全扫描
自动化回归测试
测试用例管理
服务安全扫描
压力测试
环境基础能力
全量静态代码扫描
性能测试
DIFF 测试
异常测试
容量测试
上线&验证
包管理规范
自动检查能力
分级部署能力
流水线自动化
主线成功率
执行效率
失败恢复时间

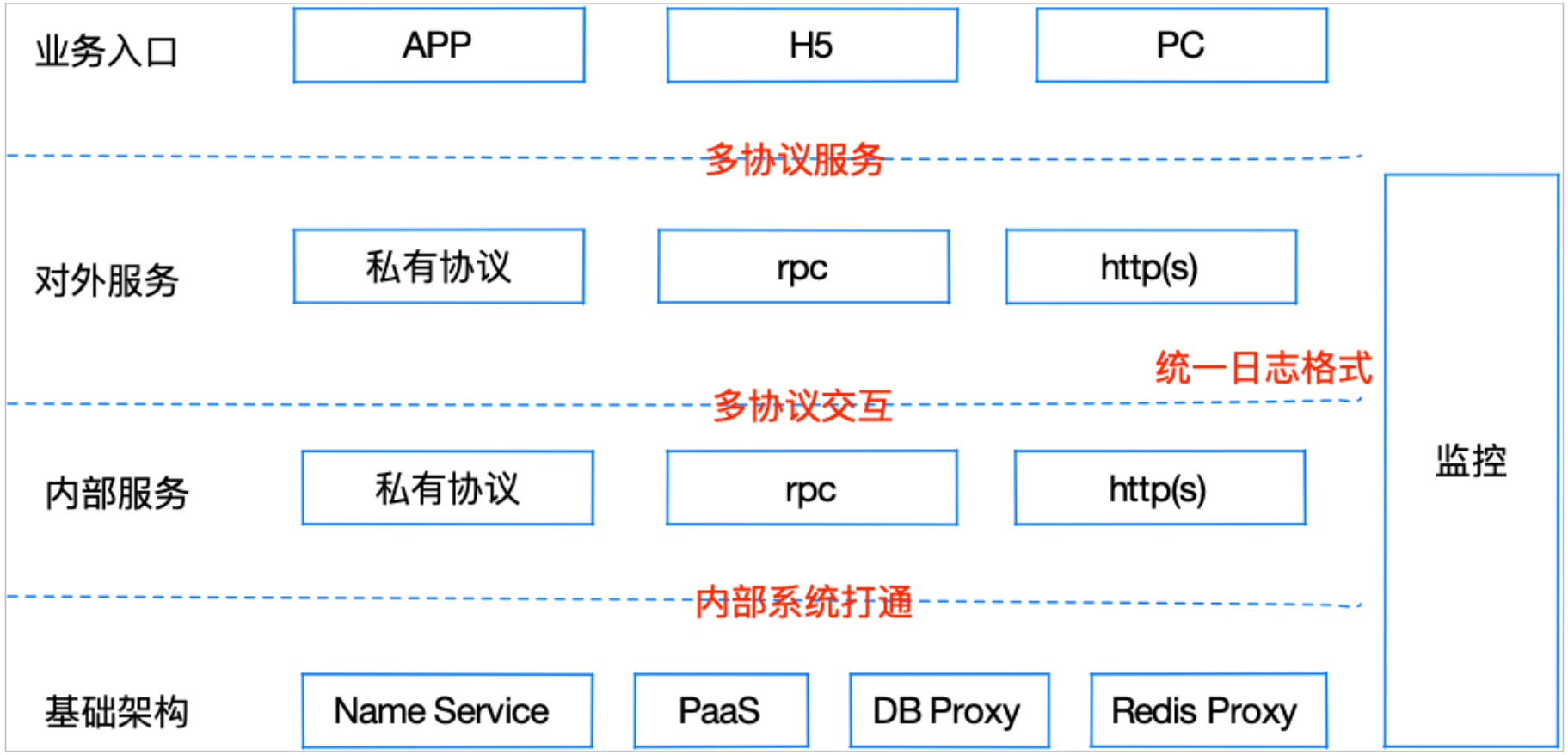
目录

Go语言体系

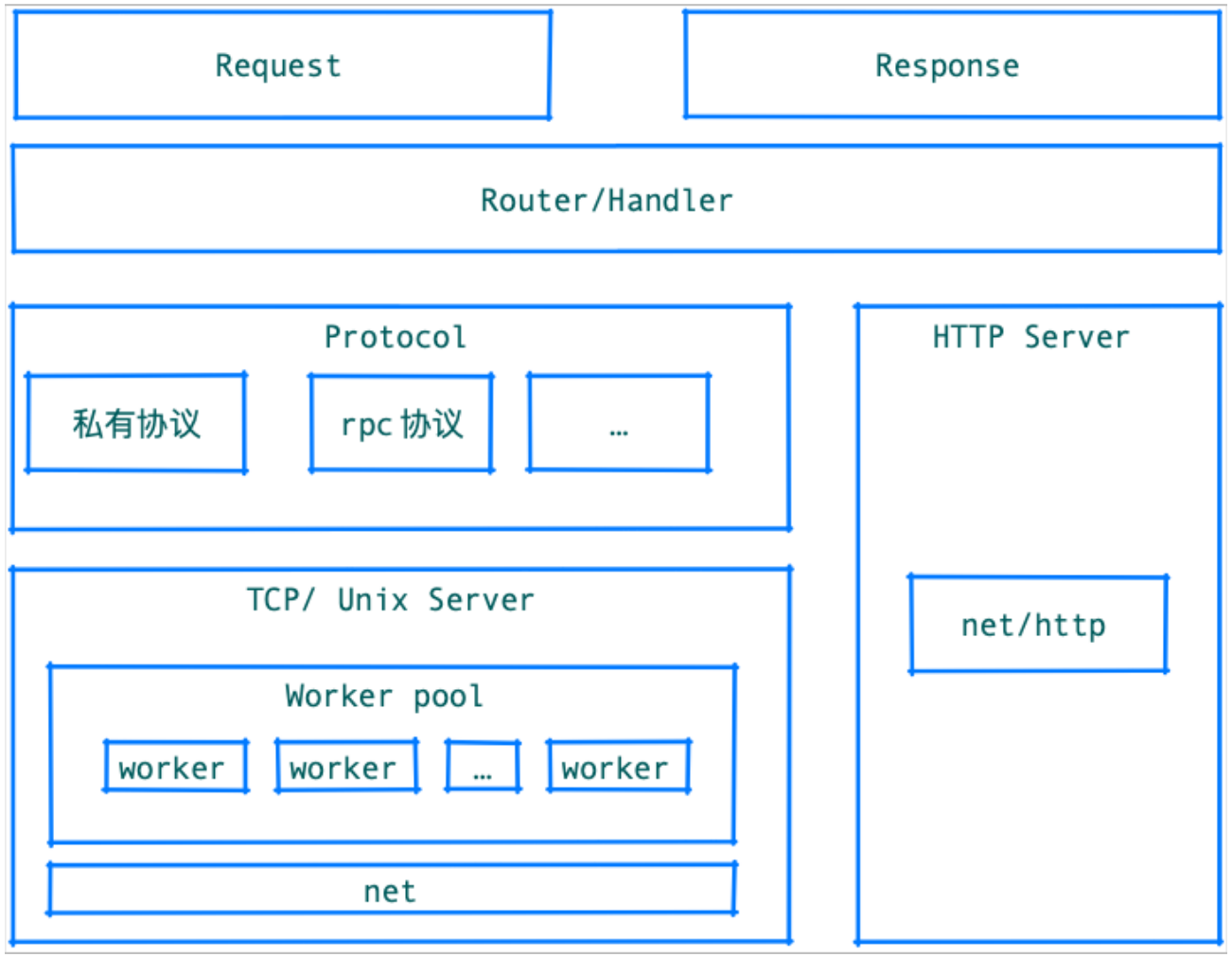

全流程支持
工程效率工具
依赖管理工具
测试框架
运维监控
接入层
业务层
开发框架
基础服务
规范与模式
代码编写规范
代码组织与引用指南
代码检查工具

目录

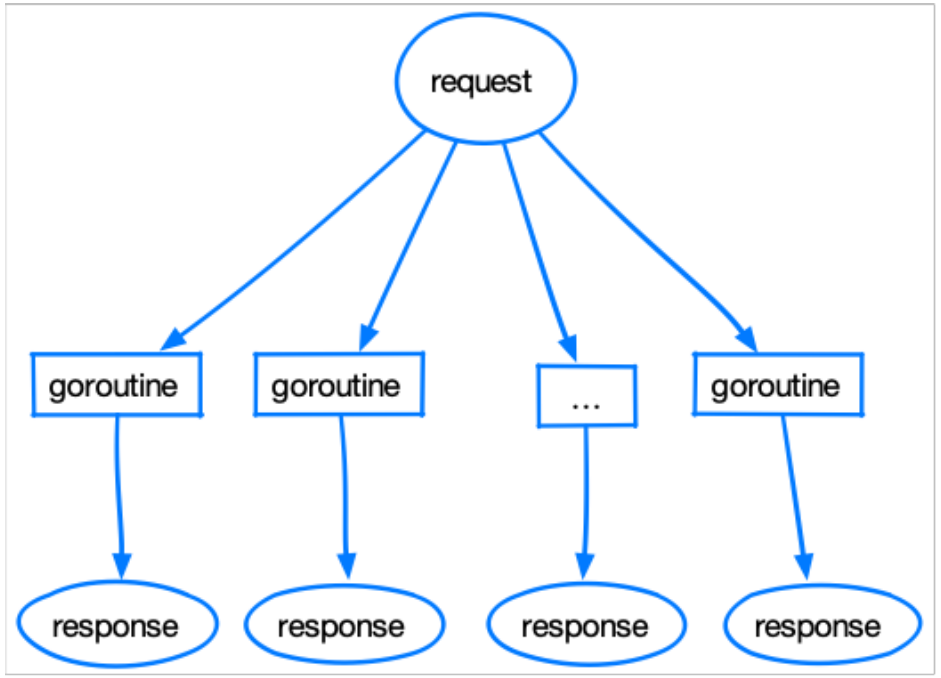
为什么需要内部开发框架

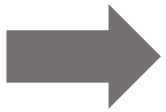


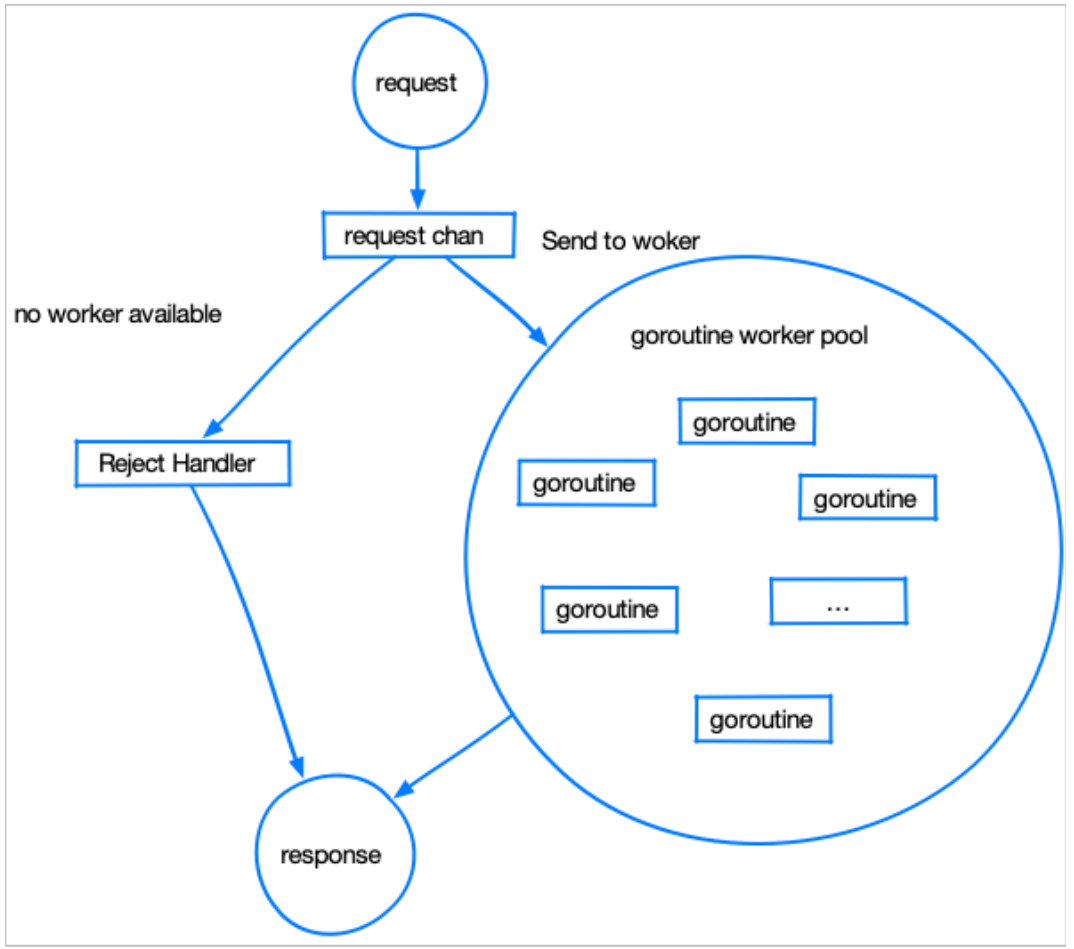
开发框架 Server 设计



Server 实现遇到的问题及方案







开发框架 Client 设计

目的:

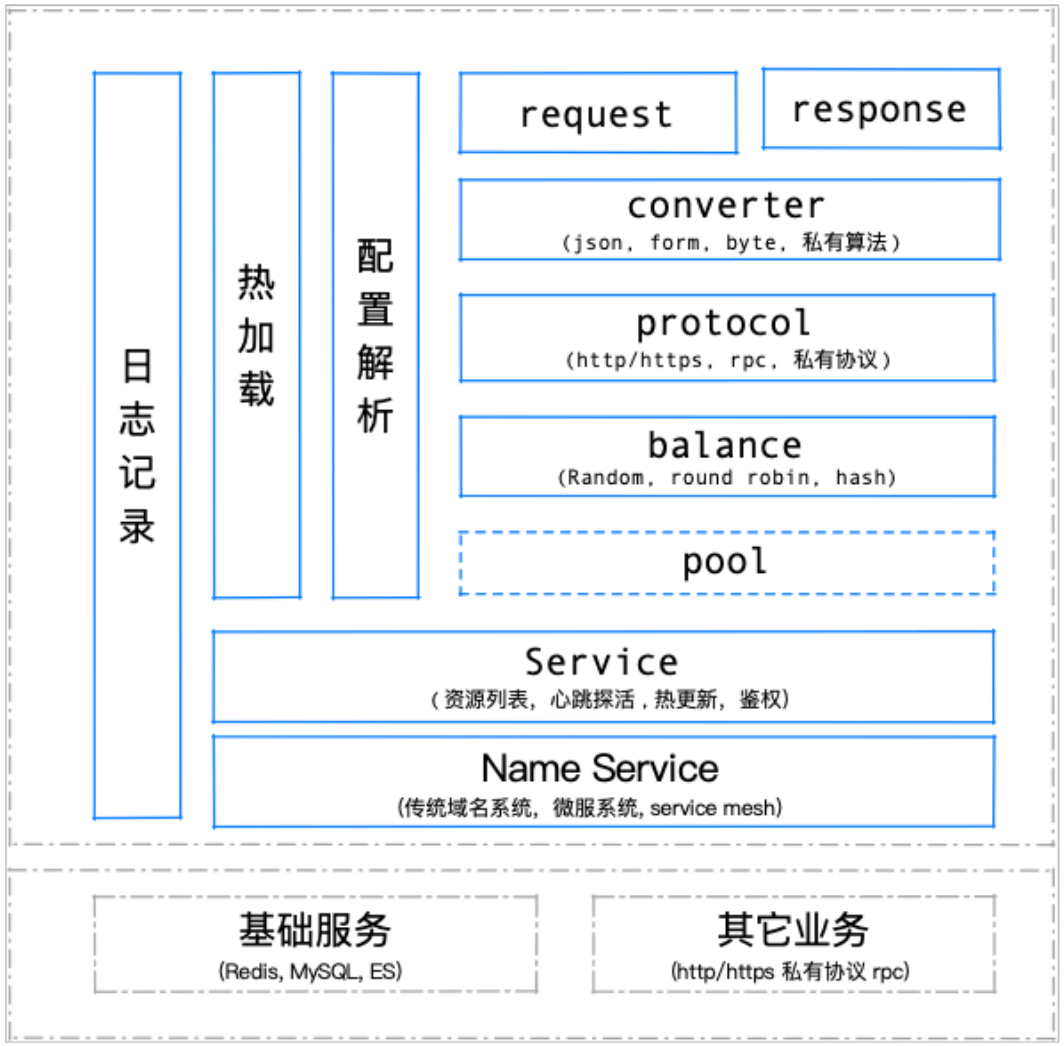
✓屏蔽网络请求的细节

功能:

✓支持多协议，多数据格式

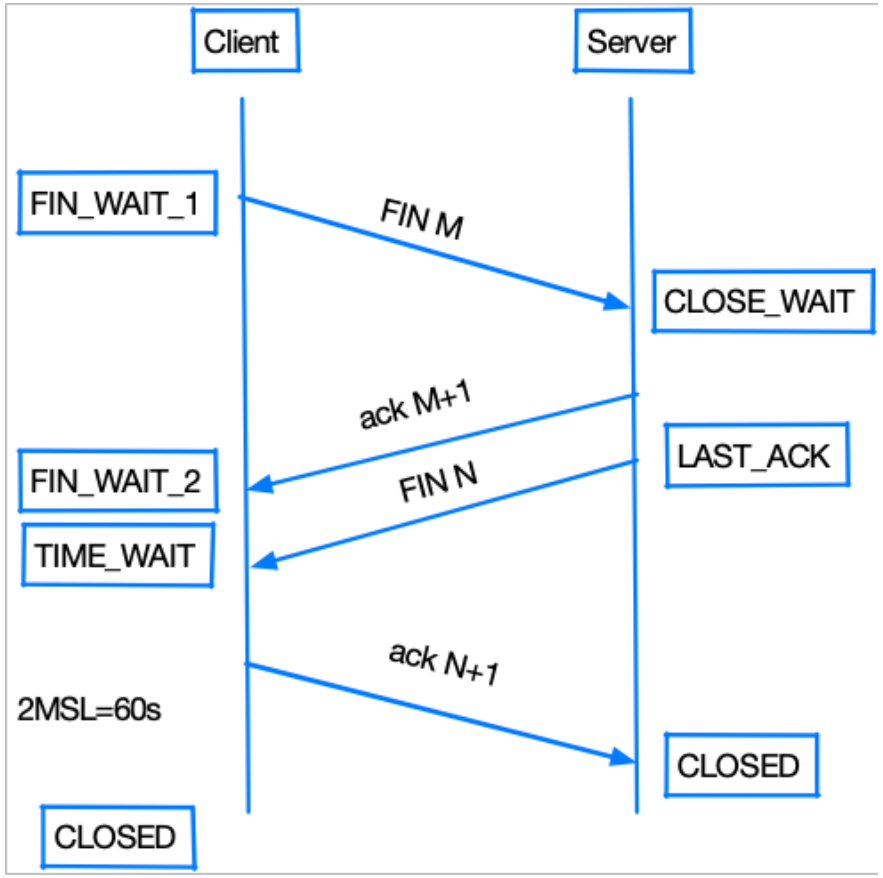
✓长连接提高性能

✓服务发现SDK，支持多 Name Service, 多负载均衡策略



Client 实现的一些坑

- 问题:
  - Server端出现TIME_WAIT 过多，使服务器连接不足而拒绝一部分请求
- 原因:
  - 客户端使用默认 http.Client 时，keep-alive默认开启，但是没有复用http.Client，导致服务端超时主动关闭，出现 TIME_WAIT



解决方案 1：Client 端主动关闭

DisableKeepAlives = true

无效果

Server 端发送完数据后会主动关闭连接，导致 TIME_WAIT 还是会在 Server 出现

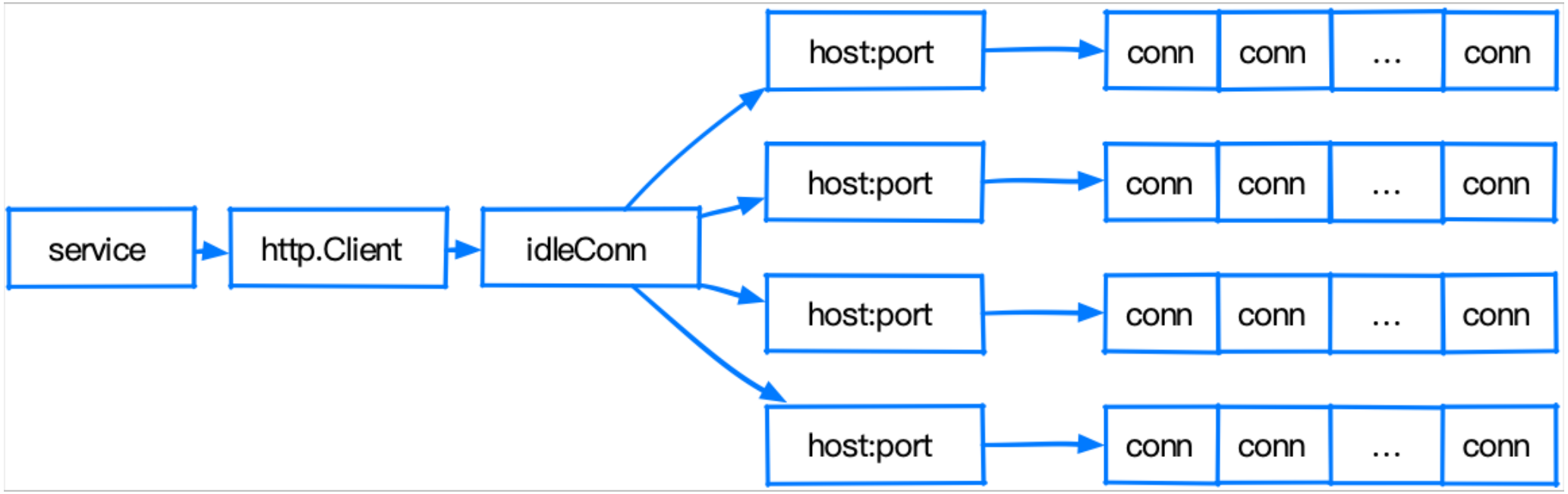
DisableKeepAlives = false
&&
MaxIdleConnsPerHost < 0

有效果

Client 端会主动关闭连接

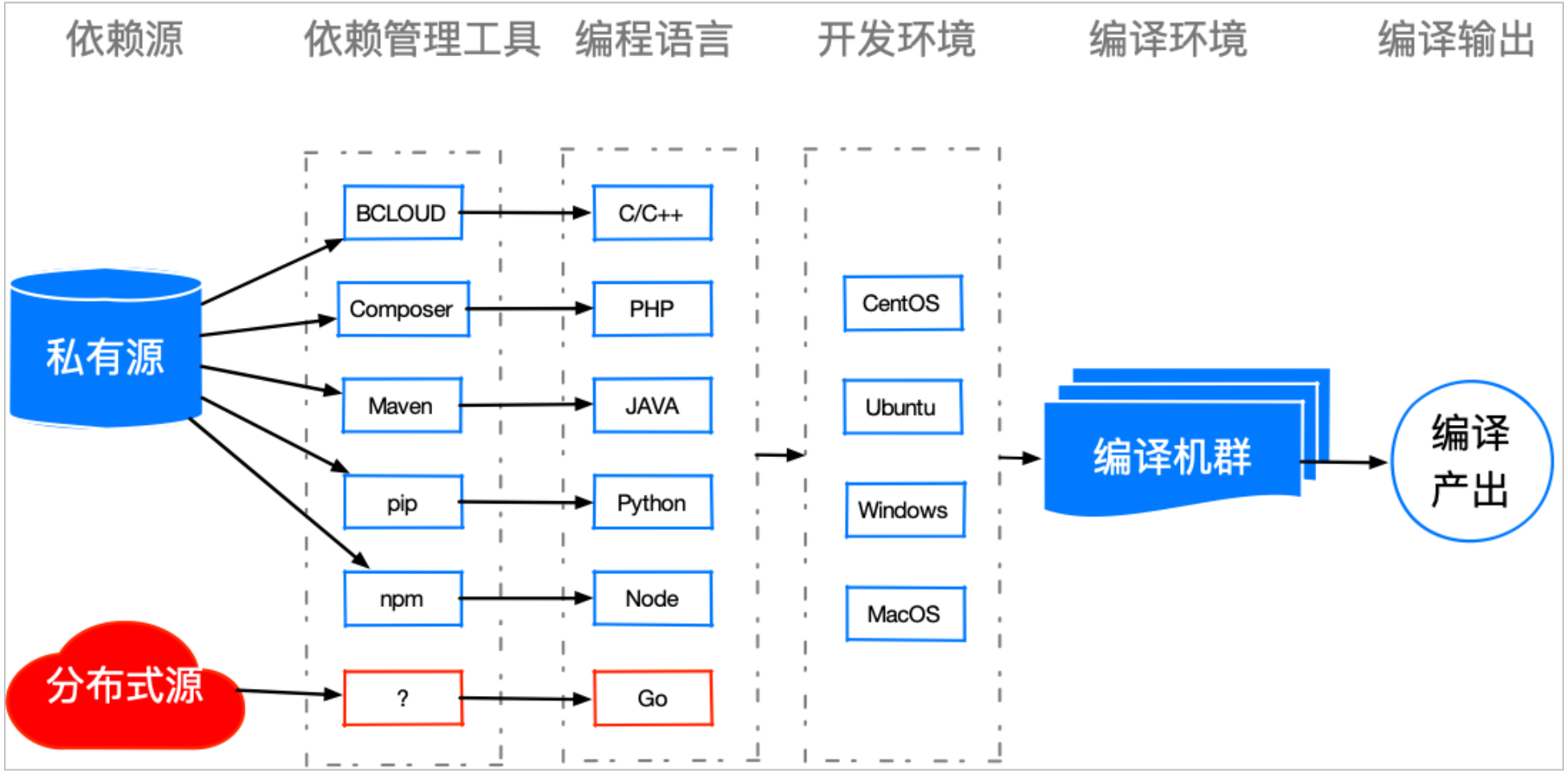
server 端可以实现快速回收

解决方案 2：连接复用


service
http.Client
idleConn
host:port
host:port
host:port
host:port
conn
conn
…
conn
conn
conn
…
conn
conn
conn
…
conn
conn
conn
…
conn

目录

百度构建系统


依赖源
依赖管理工具
编程语言
开发环境
编译环境
编译输出
私有源
分布式源
BCLOUD
Composer
Maven
pip
npm
?
C/C++
PHP
JAVA
Python
Node
Go
CentOS
Ubuntu
Windows
MacOS
编译机群
编译
产出

面临现状

编译机群无法访问外网

只有 GitHub mirror 可以利用

都是用 git 进行管理

公司内部依赖不符合规范

开源方案

Godep

- ✗ import 地址与下载地址绑定无法下载外网依赖

Glide

- ✓ 可以指定 mirror
- ✗ mirror不随版本控制，环境改变无法使用
- ✗ mirror 需要一个一个指定，过程比较繁琐

Go mod

- ✓ 可以使用 replace
- ✓ replace 信息可以随版本控制
- ✗ replace需要一个一个指定比较繁琐
- ✗ go proxy 需要适配外网，对内网模块不支持

之前的方案

依赖打包

把依赖代码打包, 编译时下载解压, 绕过了公司内部安全检查机制

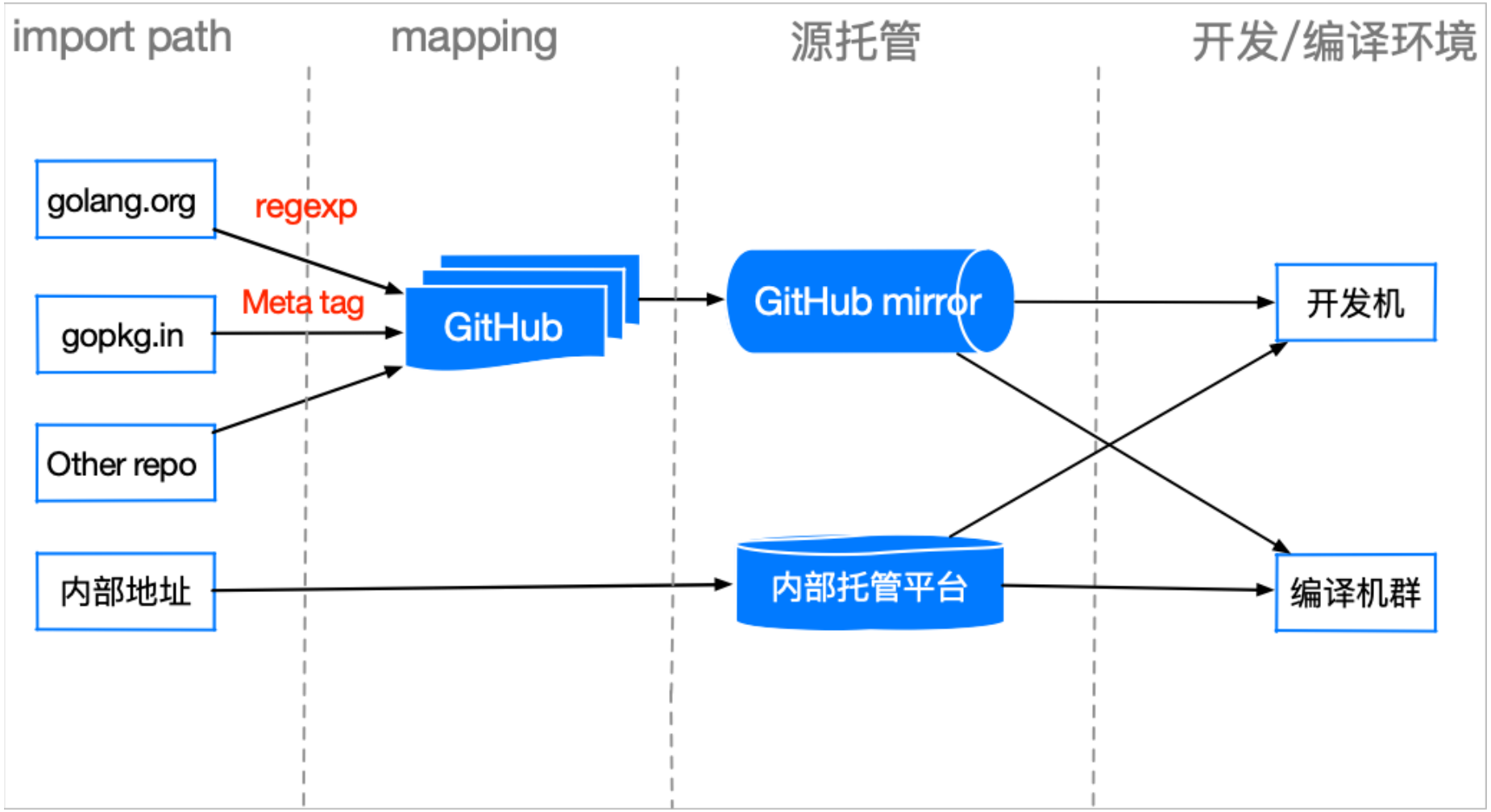
使用 Vendor

把依赖代码放到vendor中会产生大量的重复代码，造成公司资源浪费

自建镜像

把依赖代码全部放到公司内部git仓库中，存在版本更新等问题，维护成本高

目前解决方案


import path
mapping
源托管
开发/编译环境
golang.org
gopkg.in
Other repo
内部地址
regexp
Meta tag
GitHub
GitHub mirror
内部托管平台
开发机
编译机群

方案的优缺点

优点

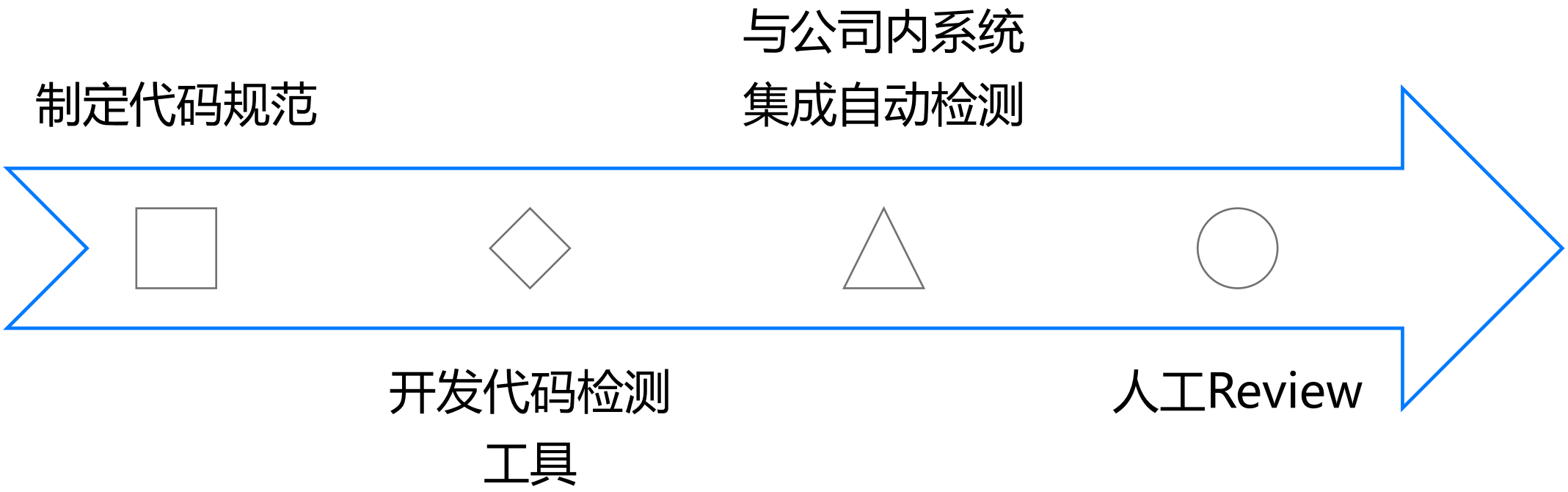
- ✓ 编译环境无需拥有外网访问权限，保证工作环境的安全性
- ✓ 提供go get功能可以直接下载外部依赖和内部依赖
- ✓ 自动生成依赖文件和对应的映射关系，无需开发者手动填写

缺点

- ✗ 外部依赖只支持使用git及托管在 github 上有的依赖
- ✗ 还未兼容go 1.11后不使GOPATH的情况

目录

代码质量保证



Go代码规范

命名规范

控制流规范

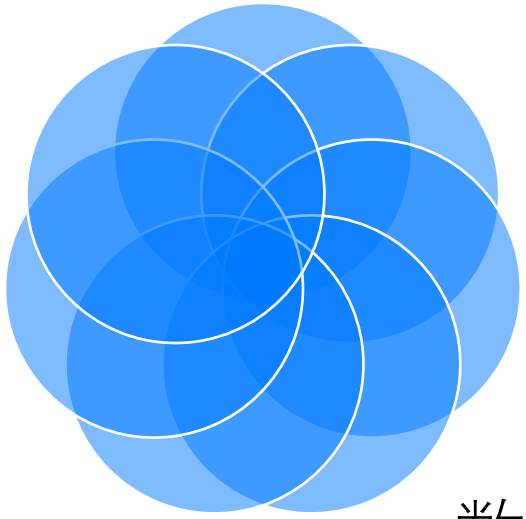
注释规范

包引用规范

……

变量声明规范

数据流规范



代码规范等级划分

ADVICE

建议级别的规则，
追求更好的质量，
可豁免

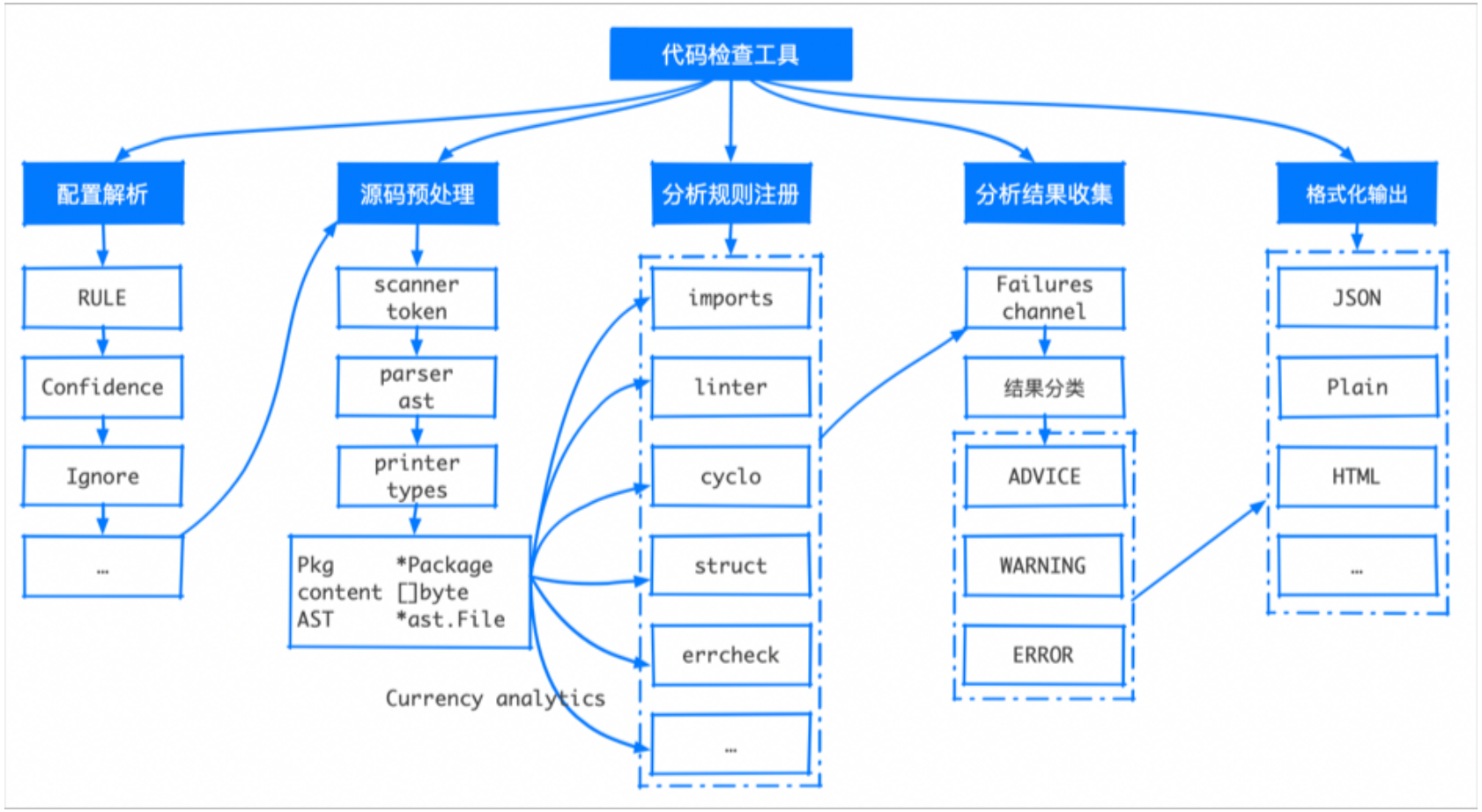
WARNING

警告级别的规则，
应该尽量去遵守，
可豁免

ERROR

严格要求的规则，
会阻塞代码入库，
不可豁免

代码检查工具设计方案


代码检查工具
配置解析
源码预处理
分析规则注册
分析结果收集
格式化输出
RULE
Confidence
Ignore
…
scanner
token
parser
ast
printer
types
Pkg *Package
content []byte
AST *ast.File
imports
linter
cyclo
struct
errcheck
…
Failures
channel
结果分类
ADVICE
WARNING
ERROR
JSON
Plain
HTML
…
Currency analytics

代码检查实现举例

```
func (w ruler) Checker(n ast.Node) ast.Visitor {
    node, ok := n.(*ast.FuncDecl)
    if ok {
        num := 0
        if node.Type.Results != nil {
            num = node.Type.Results.NumFields()
        }
        if num > w.max {
            w.Fail(linter.Failure{
                BDRULE:     "041",
                BDTYPE:     "WARNING",
                Confidence: 1,
                Failure:    fmt.Sprintf("max %d but got %d", w.max, num),
                Node:       node.Type,
                URL:        "http:/[redacted]com/display/GOLANG/041",
            })
            return w
        }
    }
    return w
}
```

传入 AST 节点

推断为函数声明

解析函数签名

构造检查结果

百度内部Go语言使用

业务流量大

BFE: 百度流量入口

春晚抢红包项目

安全要求高

自动驾驶

……

迭代快

百度智能小程序

……

用户多

百度APP

……

百度APP Go 语言实践

- 陈肖楠
- chenxiaonan01@baidu.com
- 百度资深研发工程师
- 15年加入百度
- 百度Go规范委员会成员，目前主要从事Go语言体系建设工作