



百度APP Go 语言实践

陈肖楠

百度

chenxiaonan01@baidu.com



探探 Gopher China 2019

目录

① 开发规范

② Go 语言体系

③ 开发框架

④ 依赖管理

⑤ 代码检查

开发规范



目录

① 开发规范

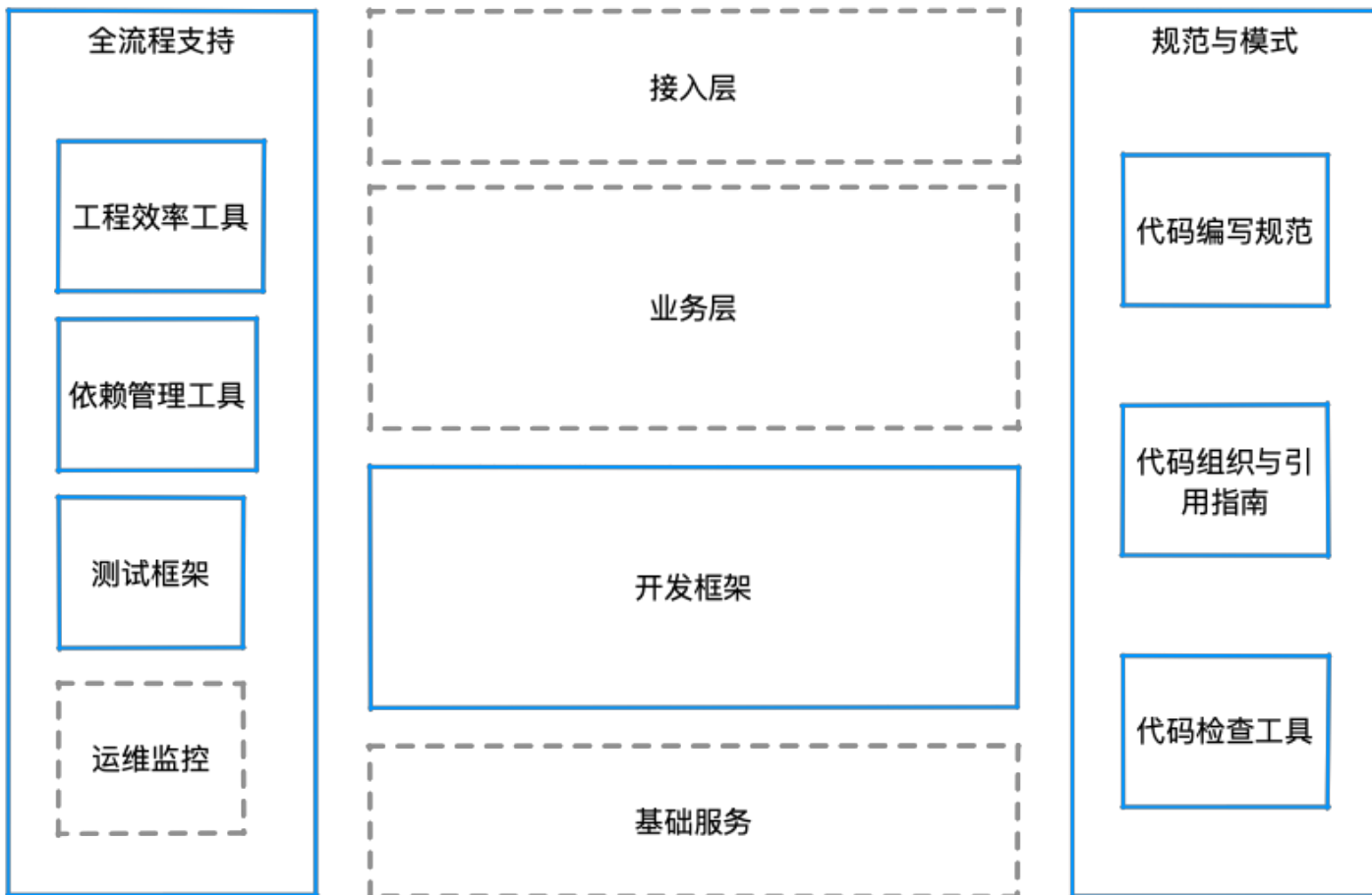
② Go 语言体系

③ 开发框架

④ 依赖管理

⑤ 代码检查

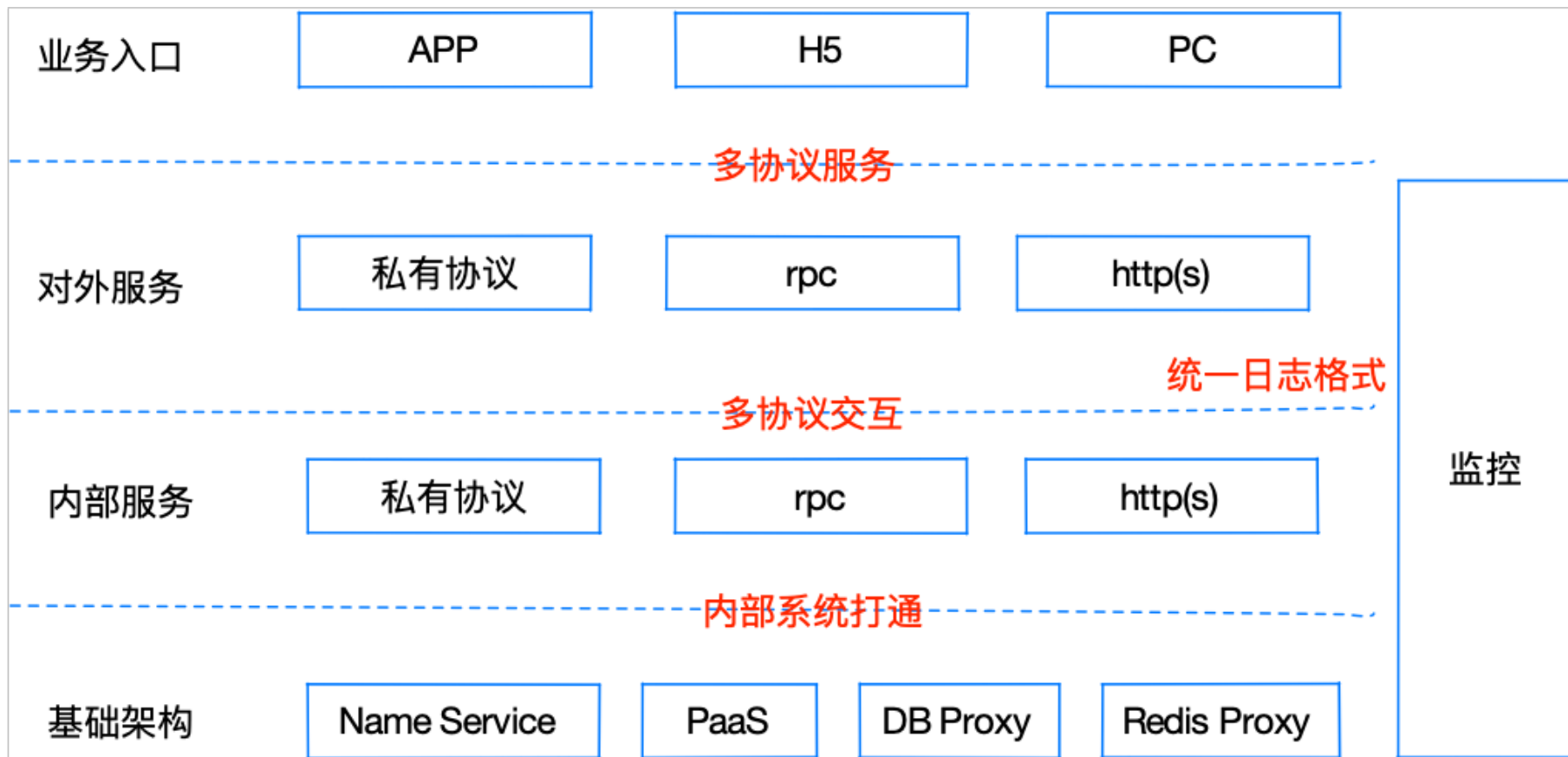
Go语言体系



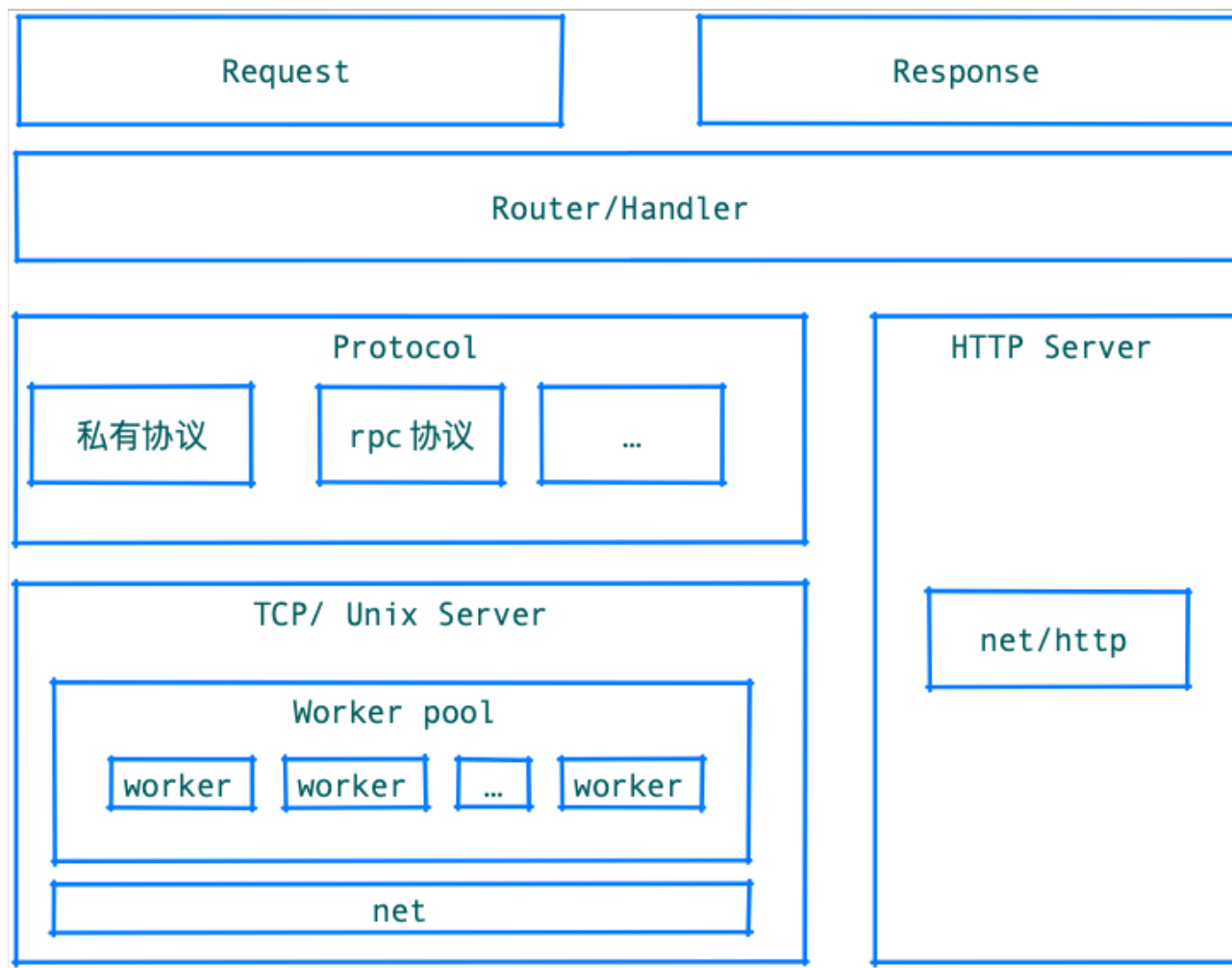
目录

- ① 开发规范
- ② Go 语言体系
- ③ 开发框架**
- ④ 依赖管理
- ⑤ 代码检查

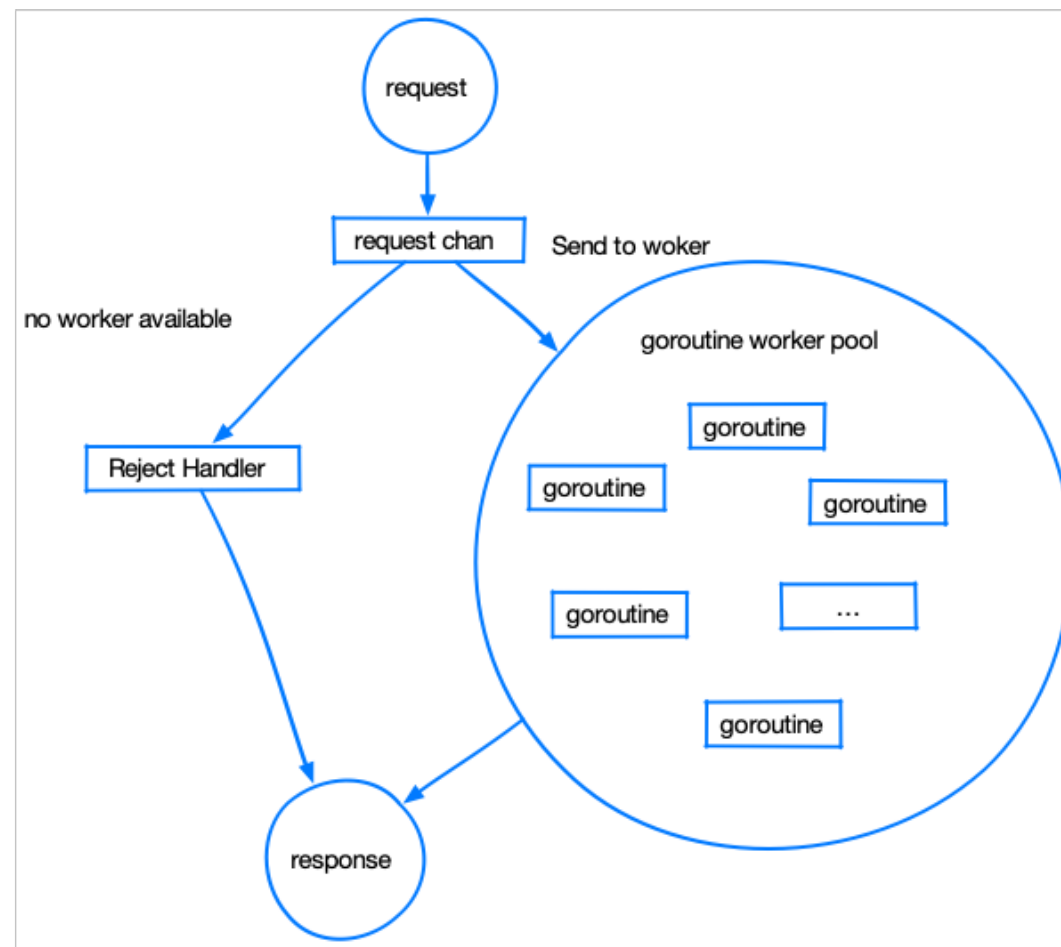
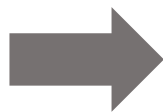
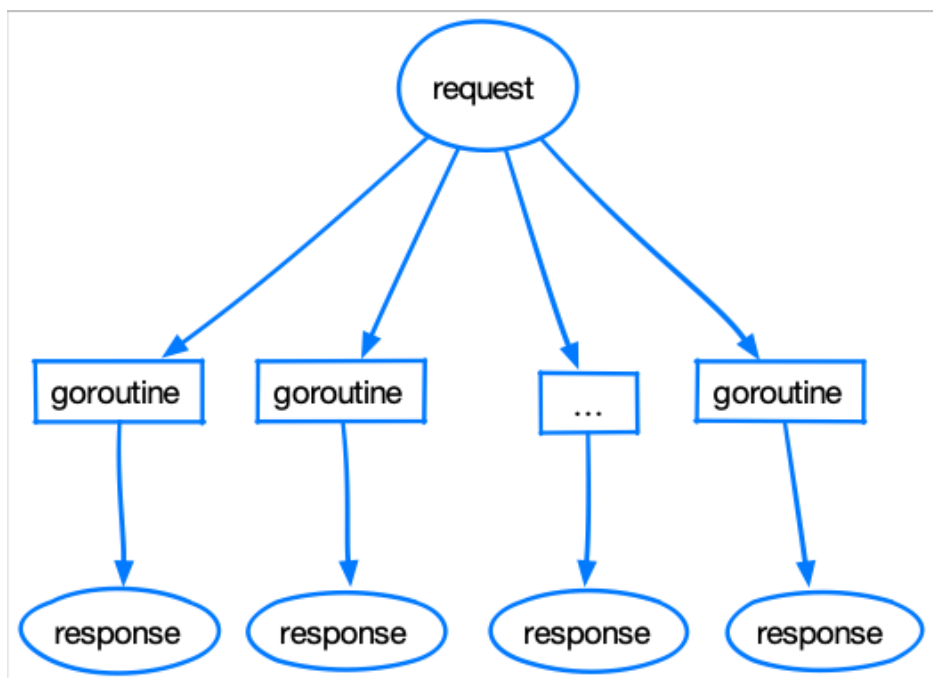
为什么需要内部开发框架



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:

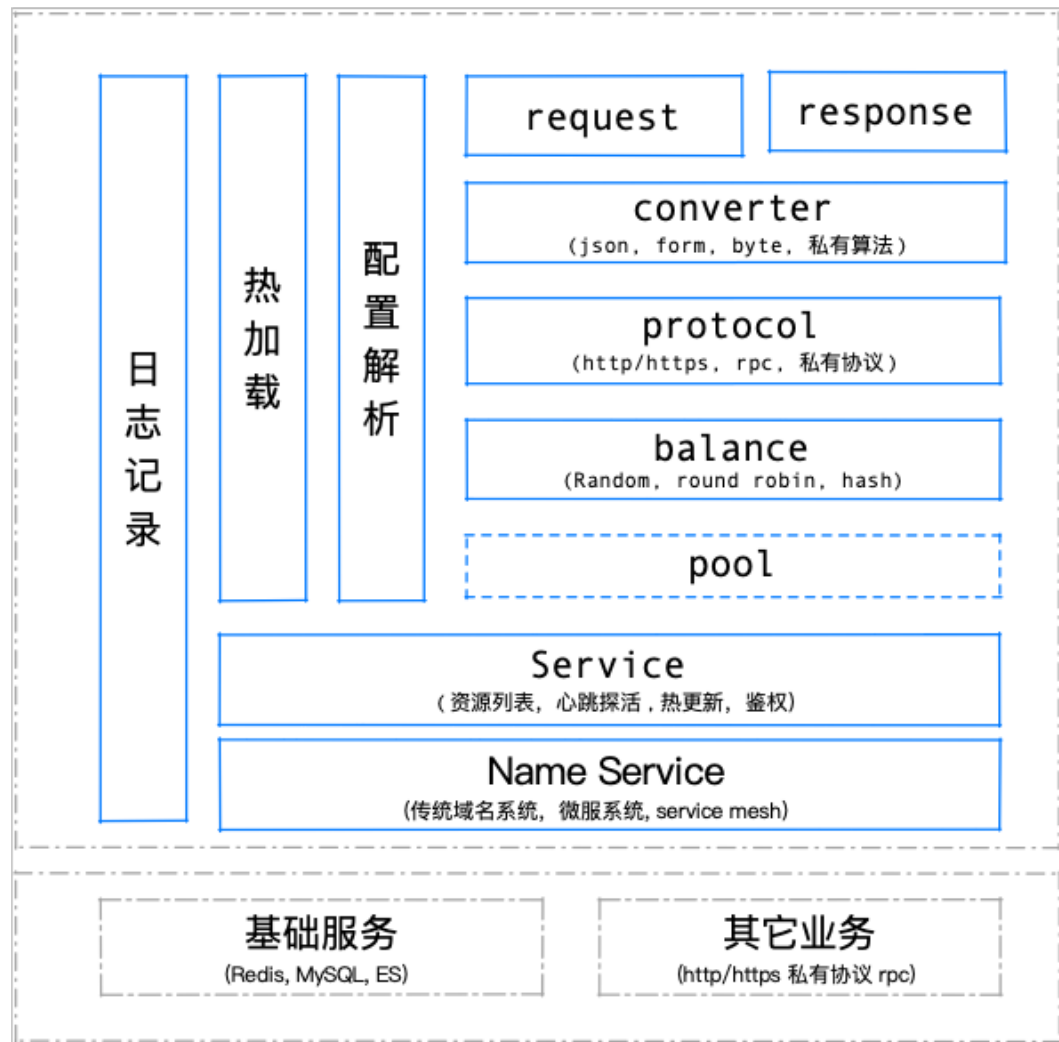
✓ 支持多协议，多数据格式

✓ 长连接提高性能

✓ 服务发现SDK，支持多

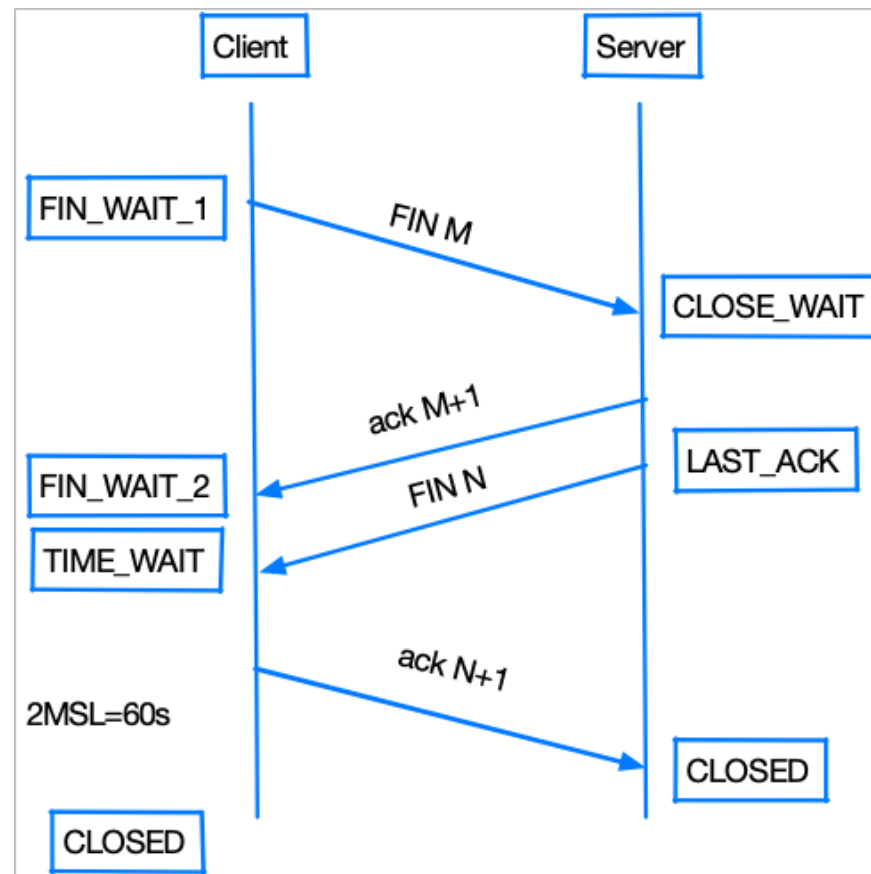
Name Service, 多负载均

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

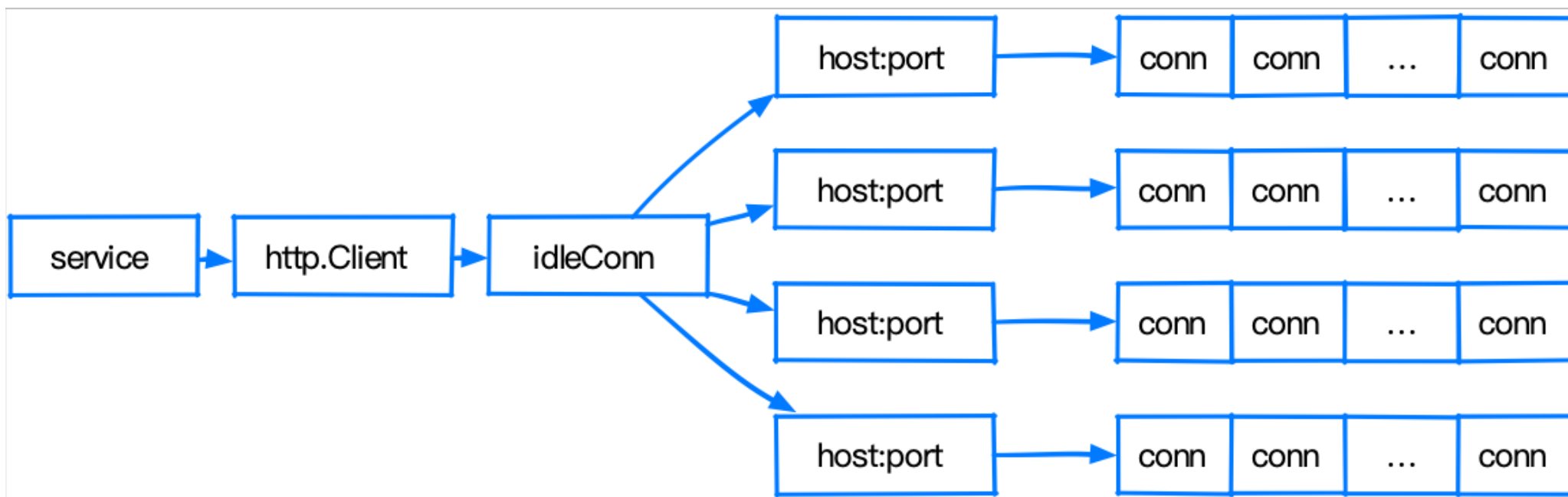
Server 端发送完数据后会主动关闭连接，导致 `TIME_WAIT` 还是会在 Server 出现

`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

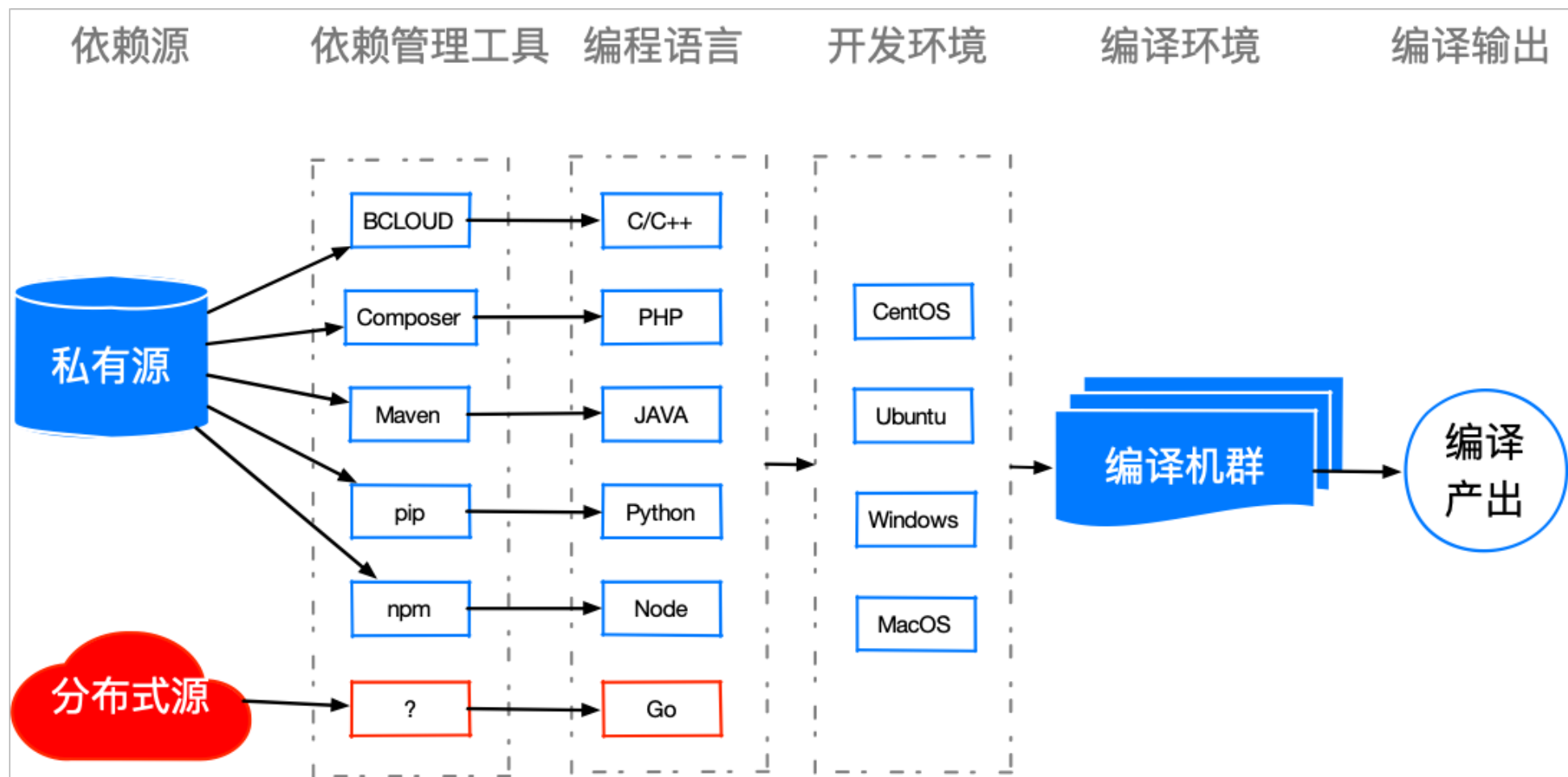
解决方案 2：连接复用



目录

- ① 开发规范
- ② Go 语言体系
- ③ 开发框架
- ④ 依赖管理**
- ⑤ 代码检查

百度构建系统



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

Glide

Go mod

✗ import 地址与下载地址绑定无法下载外网依赖

✓ 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

✓ 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



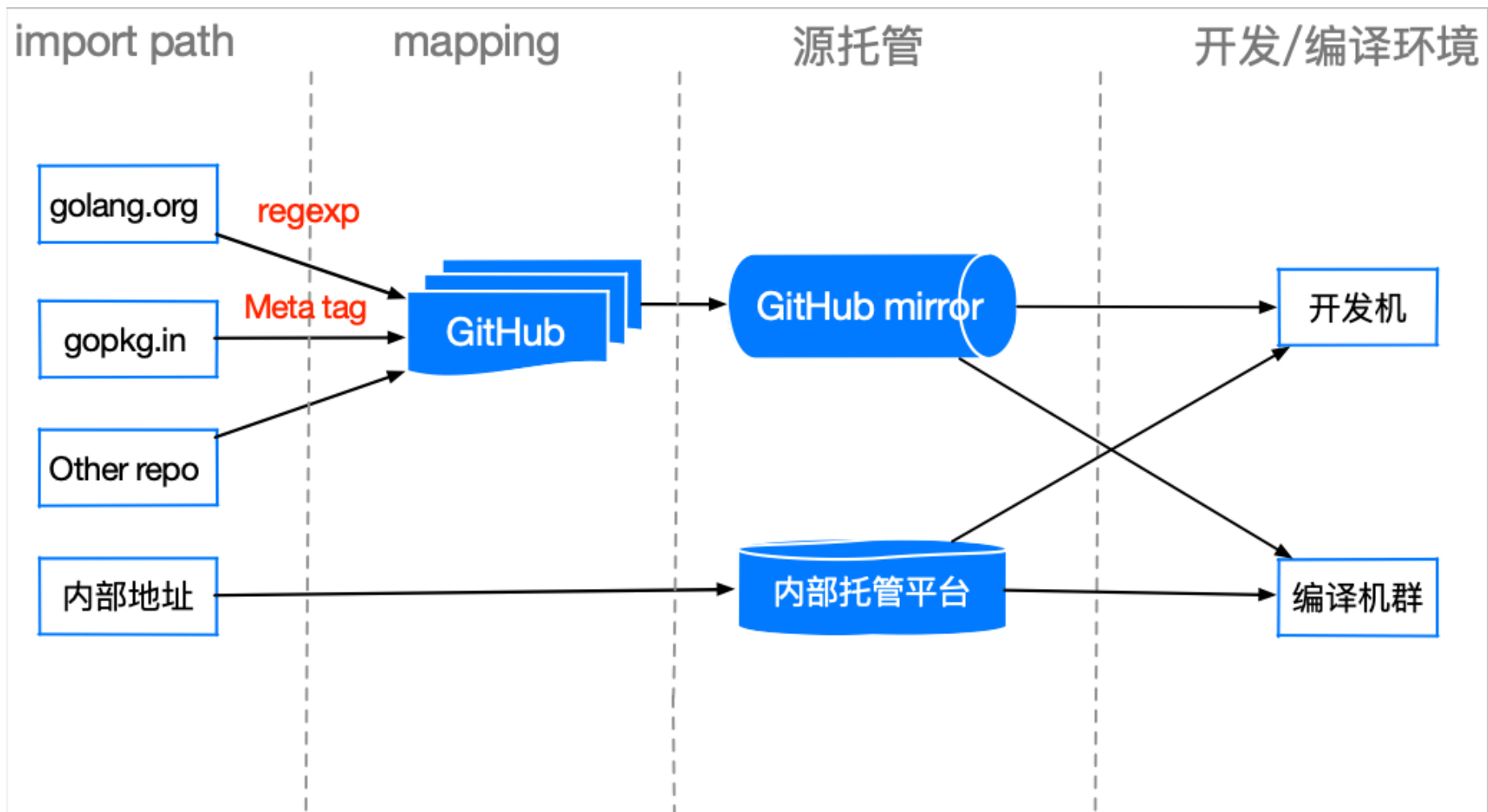
使用 Vendor

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



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

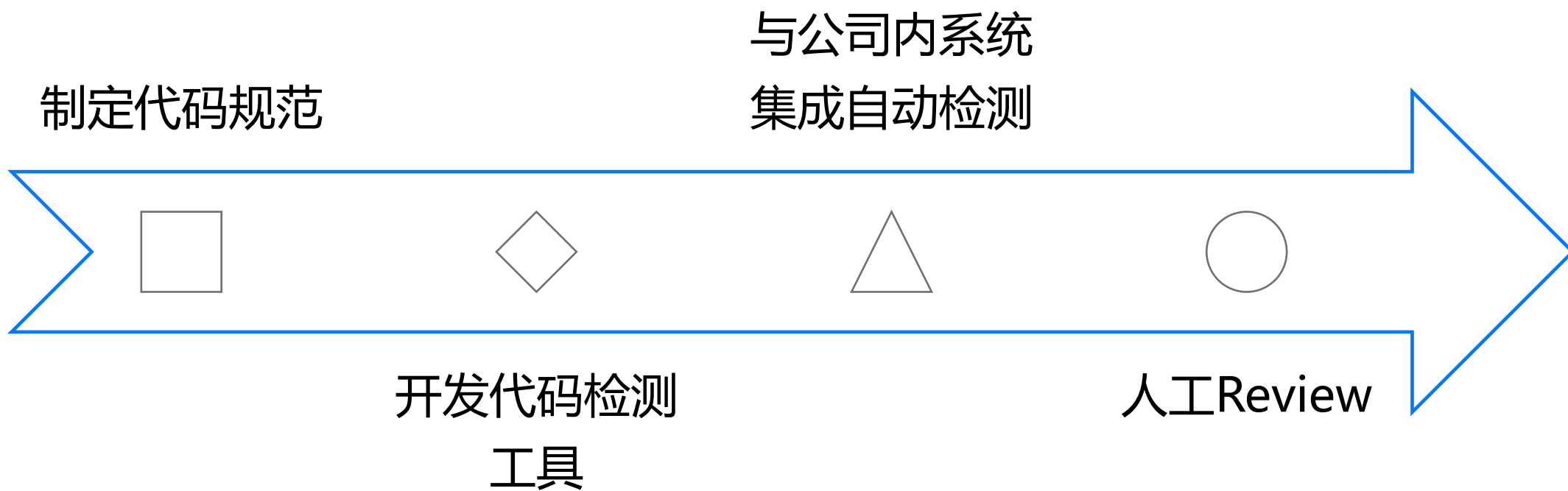
缺点

- ✗ 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后不使用GOPATH的情况

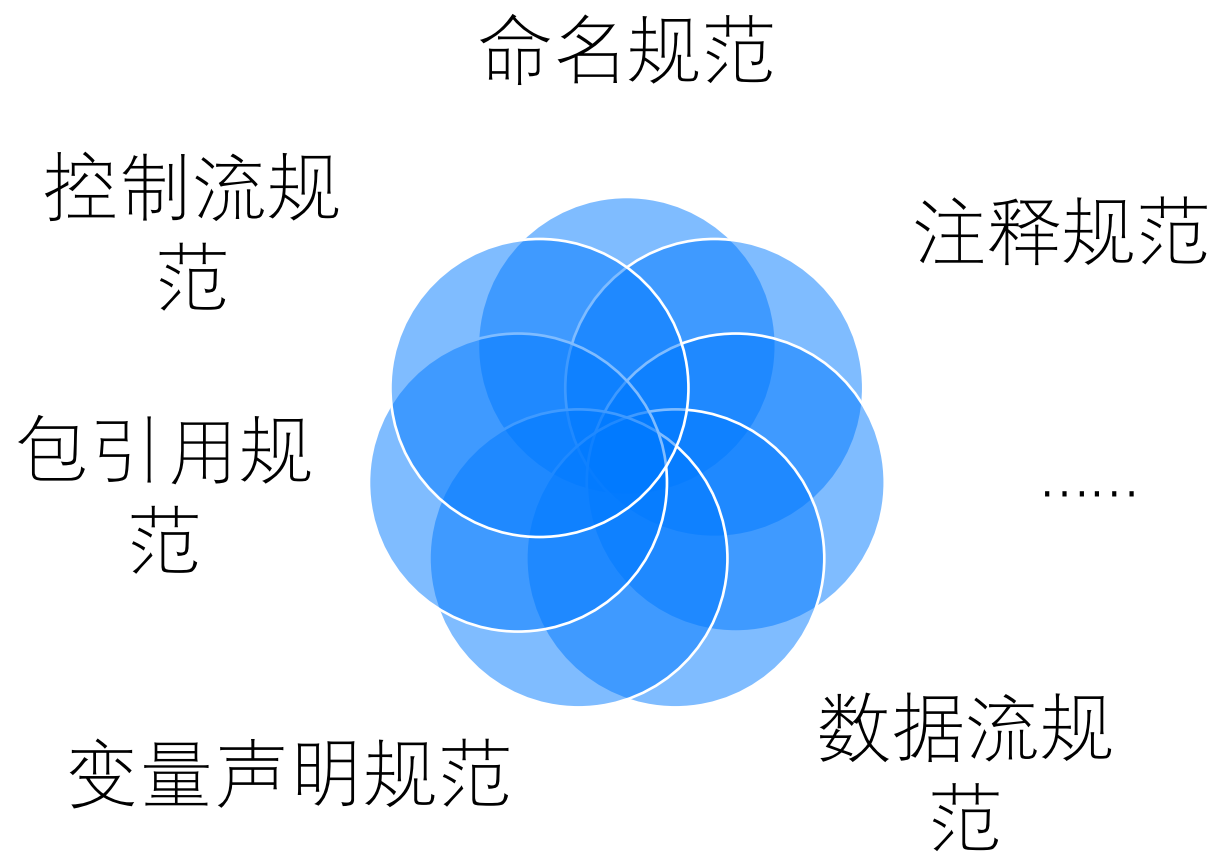
目录

- ① 开发规范
- ② Go 语言体系
- ③ 开发框架
- ④ 依赖管理
- ⑤ 代码检查**

代码质量保证



Go代码规范



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

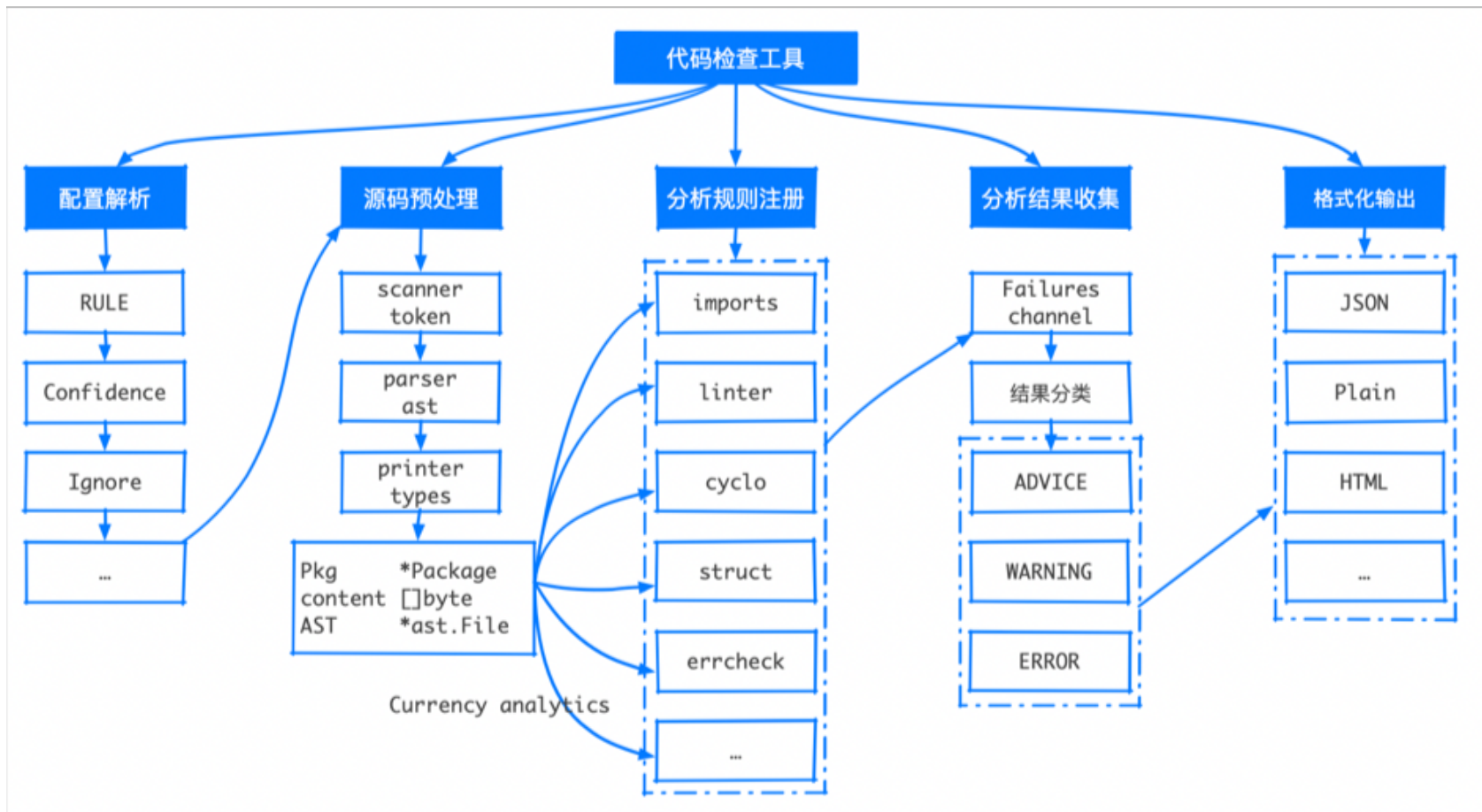
警告级别的规则，
应该尽量去遵守，
可豁免



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



代码检查实现举例

传入 AST 节点

```
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://golang.org/doc/faq/041",
            })
        }
        return w
    }
    return w
}
```

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

