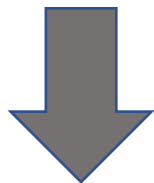


不是很系统! 想到哪讲到哪



# “漫谈” 技术选型

## vs 保守

难以言喻的自由：汇编语言。

极端自由：Perl、Ruby、PHP，脚本。

非常自由：JavaScript、Visual Basic、Lua。

自由：Python、Common Lisp、Smalltalk/Squeak。

温和自由：C、Objective-C、Scheme。

温和保守：C++、Java、C#、D、Go。

保守：Clojure、Erlang、Pascal。

非常保守：Scala、Ada、Ocaml、Eiffel。



# 构 时候



传统主义者



反传统主义者



我们想达到的状态

图 3-1 开发人员对遗留代码的态度图谱

你是什么 格?

克制的激进派

# 架构

- 演进式 系统的 随着系统实现的增 而增 敏捷
- 计划式 在 目标构建开始之前 就 常 细地制定各种 计划 桥梁施工
- 最小 计划式 的 计划式 的演进式

不同项目  
或者同一个项目的不同阶段,  
采取的策略不一样!

没有最完美的设计  
只有最合适的设计

- 四板斧:

需求分析  
方案调研  
方案对比  
方案确定

# 分析

- 求是什么?
- 约束是什么?
- 一定不能
- 是什么?



## 分析：动型

运用最小的架构技术集合去降低最紧迫的风险，以求事半功倍

- 步
  - 别，优先（从出发）
  - 一
  - 估低

## 分析：what & how

- 1.我不知道要做什么
- 2.我好像知道要做什么
- 3.我明确知道不能做什么
- 4.我明确知道要做什么
- 5.我明确知道要怎么做

你现在已知的: **Keywords**

1. 资料收集
2. 明确方向
3. 确认细节

# 方 : 料收

- 搜索: Google & Github
- 书/Github star/日常 /笔 库
- 现有已知的 目是否有类似的方案?

What you should do daily? 没吃过猪肉还没见过猪跑?

"# 看各种架构方案

\$# 看各种% '()\*+&, '%实践! 例如-. /&0 &' (1\*+&, ' 2-)33&+4 5 &' (1\*+&, ' 2673.8' .+. 0(&' (1\*+&, '

9# 看各种网上的文章:碎片化;

## 方向：方向

- 业界竞品：主 看文档 - 能力版图/ 域知
- 开源：看实现
- 类似方案：看机制



## 方 : APIGateway例子

- 业界: AWS/ 云/ 云/左耳朵耗子MegaEase APIgateway(文档)
- 开源:
  - Openresty: Kong ( )
  - GO: Tyk / KrakenD / Janus ( )
  - 国内 发 : **Apache APISIX** <https://github.com/apache/apisix>
- 类似方案 网关 traefik / Go版本的翻墙代理 (机制)

# 方 : APIGateway例子

## CNCF Cloud Native Interactive Landscape



The Cloud Native Trail Map (png, pdf) is CNCF's recommended path through the cloud native landscape. The cloud native landscape (png, pdf), serverless landscape (png, pdf), and member landscape (png, pdf) are dynamically generated below. Please [open](#) a pull request to correct any issues. Greyed logos are not open source. Last Updated: 2020-08-08 00:24:03Z

You are viewing 13 cards with a total of 58,150 stars, market cap of \$1.14T and funding of \$132.38M.

Landscape

Card Mode

Serverless

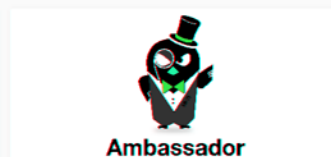
Members

1186

Orchestration & Management - API Gateway (13)



3Scale  
Datawire  
★ 202  
MCap: \$111.29B



Ambassador  
Datawire  
★ 2,874  
Funding: \$4.25M



APIOAK  
APIOAK  
★ 320



APISIX  
Apache Software Foundation  
★ 3,077



Gloo  
Solo.io  
★ 2,497  
Funding: \$13.5M



Kong  
Kong  
★ 26,561  
Funding: \$69.1M



krakenD  
Docker  
★ 2,960



Mia-Platform  
Mia-Platform



MuleSoft  
Salesforce  
★ 139  
MCap: \$181.15B



Reactive Interaction  
Gateway  
Accenture  
★ 430  
MCap: \$147.49B



Sentinel  
Alibaba Cloud  
★ 13,232  
MCap: \$695.69B



Tyk  
Tyk  
★ 5,682  
Funding: \$5.03M



WSO2 API Microgateway  
WSO2  
★ 176  
Funding: \$40.5M

方：到什么度？

你必须完全知道  
每一个、**关键细节**、  
是如何实现的！

方：到什么度？

对于权限中心来说, 策略表达式是`核心`  
对于APIGateway来说, 如何`代理`并`管理连接`是核心

例子< 1=>?)+. @)A 技术选型的时候! 怎么做代理B 如何同时支持@. 30, \*C. +B开源的几个项目! 各自如何实现的B

D)' 70<(E++F7+&G#-. H. 80. =8, lA

- `httputil.ReverseProxy` was updated in Go 1.12 to support websockets automatically. [issues](#) / [对应commit](#) / [源码](#), `httputil` 对 websocket 转发处理, 参考源码的 `handleUpgradeResponse` 方法

[net/http/httputil](#)

Go 1.15

`ReverseProxy` now supports not modifying the X-Forwarded-For header when the incoming `Request.Header` map entry for that field is `nil`.

When a Switching Protocol (like WebSocket) request handled by `ReverseProxy` is canceled, the backend connection is now correctly closed.

没有最完美的方案  
只有最合适的方案

合适 = 现实 \* 期望

1. 每个方案的`现实`是什么?
2. 你的`期望` (需求) 是什么?



## 方 对 : 合 原则

合适原则: 合适优于业界领先

简单原则: 简单优于复杂

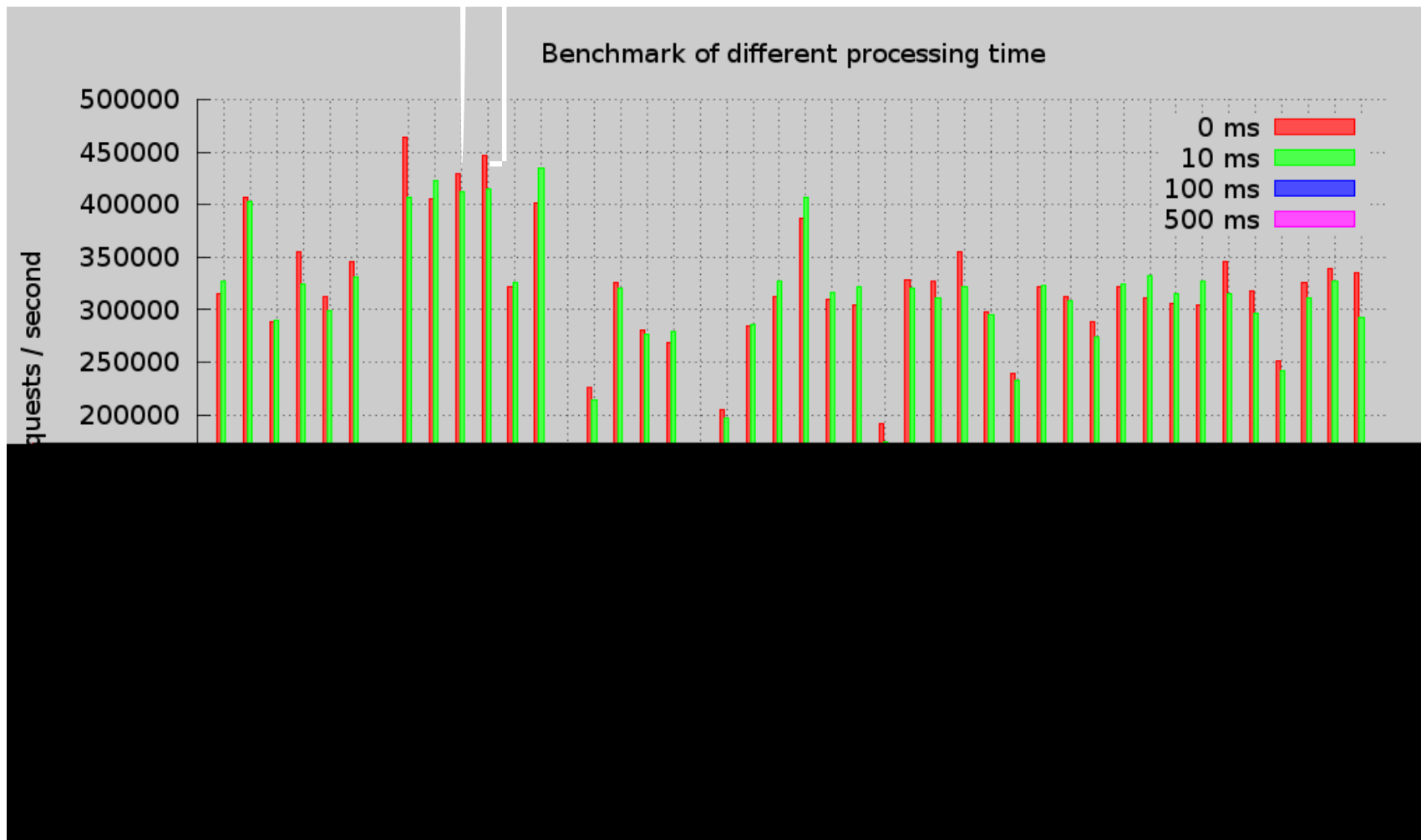
演化原则: 演化优于一步到位

# 复杂度在哪?

- 务实:

- 合 但 也 不 太 后
- 单 但 在 以 及 可 围 内
- 化 但 子 好, 在

# 方 对 ： 择开发机架



[E++F0<22J&+E73#\\*, K20K\)GG'. 0+2J, L@. 3LM8\)K. @, 8CL3. ' \\*EK\)8C](#)

## Why APIGateway choose chi?

\*E& \&() (G&JE+@. &JE+!(&/&, K)+&\*())' /(\*, KF, 0)3G. 8, 7+. 8(M, 8(37&G/&' J(? , (N00=(0. 8H&\* . 0#

### Features

- **Lightweight** - cloc'd in ~1000 LOC for the chi router
- **Fast** - yes, see [benchmarks](#)

http • **100% compatible with net/http** - use any http or middleware pkg in the ecosystem that is also compatible with net/http

ounting • **Designed for modular/composable APIs** - middlewares, inline middlewares, route groups and subrouter m

- **Context control** - built on new `context` package, providing value chaining, cancelations and timeouts

- **Robust** - in production at Pressly, CloudFlare, Heroku, 99Designs, and many others (see [discussion](#))

- **Doc generation** - `docgen` auto-generates routing documentation from your source to JSON or Markdown

- **No external dependencies** - plain ol' Go stdlib + net/http

## 方 对 : Why gin?

# Why IAM choose gin?

?&' (&0())(@. 3(MB) K. @, 8C(@8&++. ' (&' (? , (:?, G)' J;#(>(M. )+78. 0()(K)8+&' &LG&C. (1=>(@&+E(F. 8M, 8K)' \*. (+E)+(&0(7F( +, (PQ(+&K. 0(M)0+. 8(+E)' C0(+, E++F8, 7+. 8#(>M(A, 7(' .. /(F. 8M, 8K)' \*. ()' /(J, , /(F8, /7\*+&H&+A!(A, 7(@&GG(G, H. (?&' #

Router

Validation

Middleware & Response

## 方 对 : 定依 原则

稳定依赖原则: 依赖关系必须要指向更稳定的方向. 预期会经常变更的组件都不应该被一个难以修改的组件所依赖, 否则这个多变的组件也将会变得难以被修改.

稳定性指标=出依赖/(出依赖+入依赖)

活跃度如何? 发布频率? 最后一个稳定版之后的 issue 有哪些关键的尚未解决?



## 定方：技术 务

风险驱动模型 意味着某段时间内资源有限的情况下  
只能关注某个核心;  
如果做取舍, 就会欠债!

什么时候还! 有没有能力还?

定方 : 可 是不可

可逆的决策, 需要理性看待, 主要考虑变更的成本!

不可逆的决策, 需要谨慎评估!

开弓没有回头箭! 决定了就只能一路趟到黑

定方 : 时刻关 更新

- . ) / ( + E . ( \* E ) ' J . ( G , J R

Feature + Bugfix + 性能

难以决策的时候的一种选择

## 定方：推 决

- 一个 中 人力 什么 中存在 合 其 些 做出  
不 决 合 些 决 与 业务 例 关 可以  
决 务器 具 依 入 助
- 一个 好 不 依 于 些 可 地 些  
决 力于 产 响 到 低
- 未来会 到的 提前考 方案扩展性, 但是不做决策. 例如Redis  
缓存未来单实例抗不住, 用sharding? Cluster?



协议上支持, 但是不实现  
接口上支持, 但是不开放  
存储上支持, 但是不使用

看得足够远, 就不担心需求频繁变更  
保留未来的`可能性`

## 无论大小/无关难易

- 为什么我们 弃了 而 向sqlx(最后一个版本2018)?
- 为什么我们后端用Go, 前端用Django(而不是GO?)
- 为什么CMDB是微服务框架而我们不是?
- 为什么APIGateway的go-redis ratelimit没有升级到最新?
- .....

技术选型是需要在日常项目中不断`磨练`的一种意识

Q & A

END