微服务研究报告

Wen Zhenglin
2016-4-28

概览

- 背景介绍
- 什么是微服务
- 微服务的特点
- 微服务解决什么问题
- 微服务带来的挑战

背景介绍

动机

减少修改带来的高成本

应对市场的快速变化

简化开发人员的负担（长远来看）

什么是微服务

Microservices is a software architecture style in which complex applications are composed of small, independent processes communicating with each other using language-agnostic APIs.

These services are small, highly decoupled and focus on doing a small task, facilitating a modular approach to system-building.

Ref: https://en.wikipedia.org/wiki/Microservices

微服务的特点

Componentization via Services (encourage reusability and independence)

Organized around Business Capabilities (Conway's Law)

Products not Projects (you build, you run it)

Smart endpoints and dumb pipes (better performance/more resilience)

Decentralized Governance (self-direct team)

微服务的特点(续)

Decentralized Data Management (remove the dependence/avoid impact)

Infrastructure Automation (for more efficiency)

Design for failure (resilience)

Evolutionary Design (service refactoring / adding service)

微服务的原则

Hide Internal Implementation Details (API access only)

Independently Deployable (upgrade, update, migrate, refactoring)

Isolate Failure (avoid wide range of impact, reduce cost/time)

Highly Observable (better monitoring)

微服务解决什么问题

提供架构层面问题的解决思路，同时提供指导作用，通过现有相关技术以解决架构问题

一种更系统性的指导(基于其它科技公司的经验)

微服务带来的挑战

核心问题

1. 数据接口定义
2. 数据通信控制

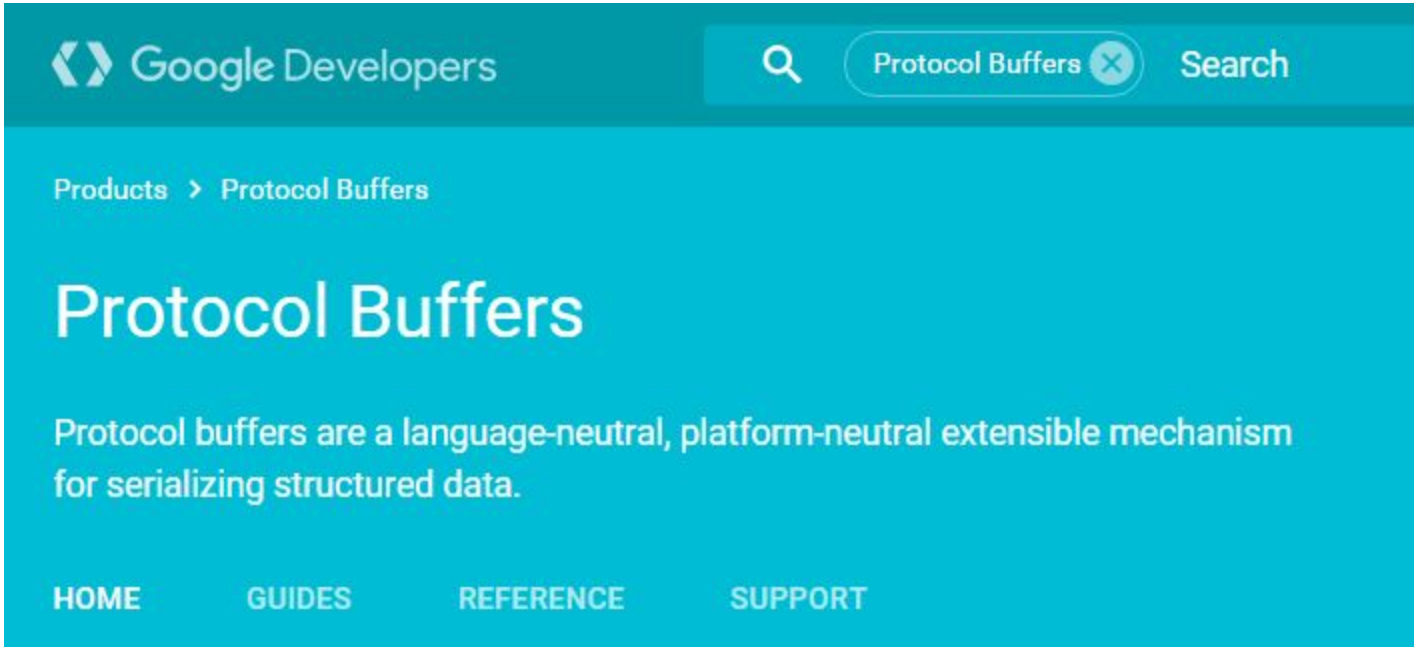
须满足:”语言无关性，高性能“

数据接口定义

通用数据格式描述（基本类型）

二进制（比特级最小编码）高效传输

数据接口定义(续1)


Google Developers
Protocol Buffers
Search
Products > Protocol Buffers
Protocol Buffers
Protocol buffers are a language-neutral, platform-neutral extensible mechanism for serializing structured data.
HOME
GUIDES
REFERENCE
SUPPORT

数据接口定义示例

```
syntax = “proto3”;

message Person {
  string name = 1;
  int32 id = 2;
  string email = 3;

  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }

  message PhoneNumber {
    string number = 1;
    PhoneType type = 2;
  }
  repeated PhoneNumber phone = 4;
}
```

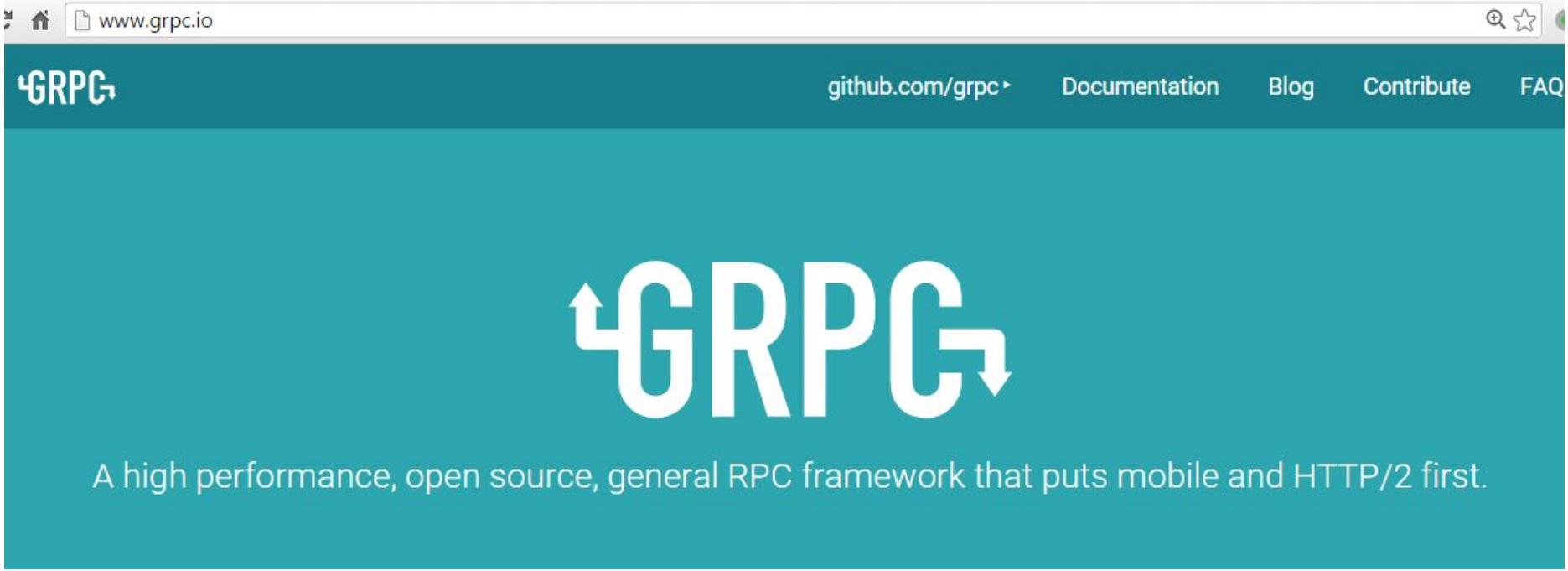
数据通信控制

基本的数据流控制(服务终止，取消，超时控制)

四种基本的流通信方式

1. 单一请求
2. 客户端流请求
3. 服务端流请求
4. 双向流

数据通信控制(续)


www.grpc.io
GRPC
github.com/grpc
Documentation
Blog
Contribute
FAQ
GRPC
A high performance, open source, general RPC framework that puts mobile and HTTP/2 first.

数据通信控制(续1)

Libraries in ten languages

Automatically generate idiomatic client and server stubs for your service in a variety of languages.

gRPC has libraries in:
C, C++, Java, Go, Node.js, Python, Ruby, Objective-C, PHP and C#.

Get started

Overview Guide
API Reference Docs

服务定义示例（gRPC）

服务(rpc)与接口(proto3)相结合，用此定义生成对应语言的服务端与客户端代码

```
service HelloService {
  rpc SayHello (HelloRequest) returns (HelloResponse);
}

message HelloRequest {
  required string greeting = 1;
}

message HelloResponse {
  required string reply = 1;
}
```

核心问题--解决方案

已单独介绍:

“gRPC boilerplate to high performance scalable APIs.pdf”

服务拆分

参见:

- 微服务架构之服务拆分_by_wen_2016-3-28.pdf
- A technique to identify microservices on monolithic systems_by_wen_2016-3-29.pdf

服务发现

https://demo.consul.io/ui/#/nyc3/services/consul

SERVICES | NODES | KEY/VALUE | ACL | NYC3

Filter by name | any status | EXPAND

consul	9 passing
proxy	4 passing
redis	12 passing
web	12 passing

consul

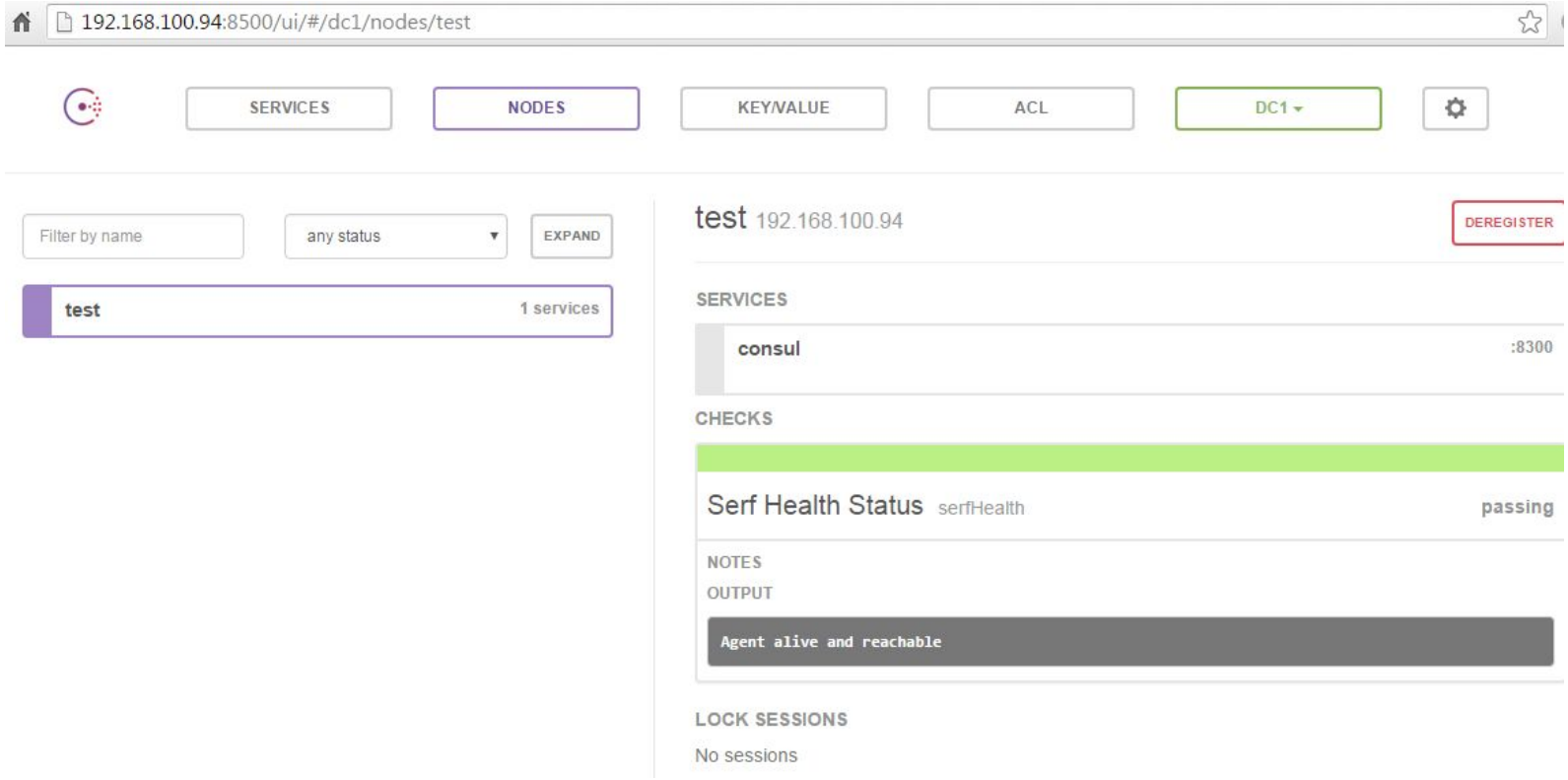
TAGS

No tags

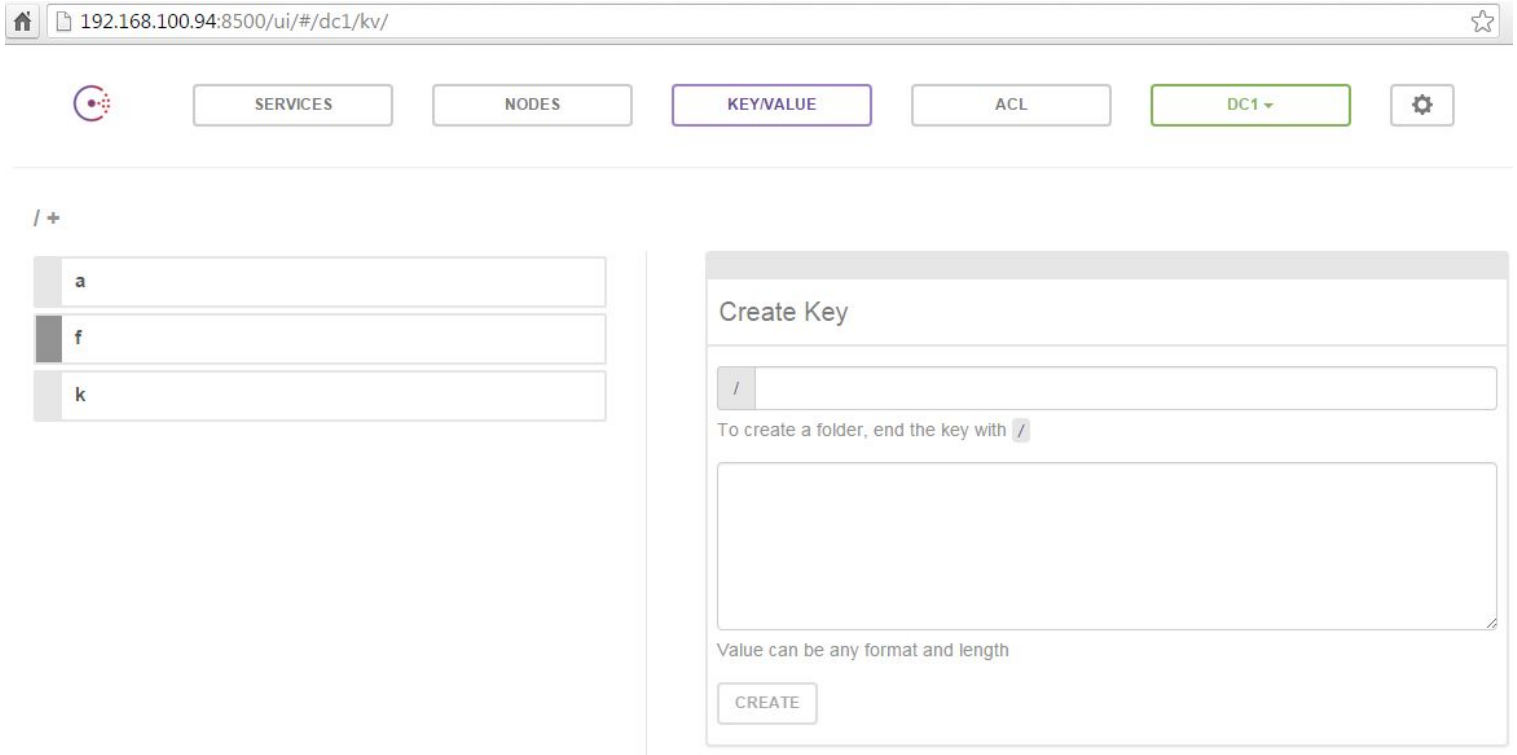
NODES

consul-server-nyc3-1	104.236.126.18	3 passing
Disk Util Disk Util		passing
Load Avg Load Avg		passing
Serf Health Status serfHealth		passing

服务发现(续1)


192.168.100.94:8500/ui/#/dc1/nodes/test
SERVICES
NODES
KEY/VALUE
ACL
DC1
Filter by name
any status
EXPAND
test
1 services
test 192.168.100.94
DEREGISTER
SERVICES
consul
:8300
CHECKS
Serf Health Status serfHealth
passing
NOTES
OUTPUT
Agent alive and reachable
LOCK SESSIONS
No sessions

服务发现(续2)


192.168.100.94:8500/ui/#/dc1/kv/
SERVICES
NODES
KEY/VALUE
ACL
DC1
/ +
a
f
k
Create Key
/
To create a folder, end the key with /
Value can be any format and length
CREATE

API文档管理

petstore.swagger.io/#!/pet/addPet

pet : Everything about your Pets

Show/Hide | List Operations | Expand Operations

POST /pet — Add a new pet to the store

Parameters

Parameter	Value	Description	Parameter Type	Data Type
body	(required)	Pet object that needs to be added to the store	body	Model / Example Value

Parameter content type: application/json

```
{
  "id": 0,
  "category": {
    "id": 0,
    "name": "string"
  },
  "name": "doggie",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
```

https://demo.consul.io/ui/#/nyc3/services/consul

Response Messages

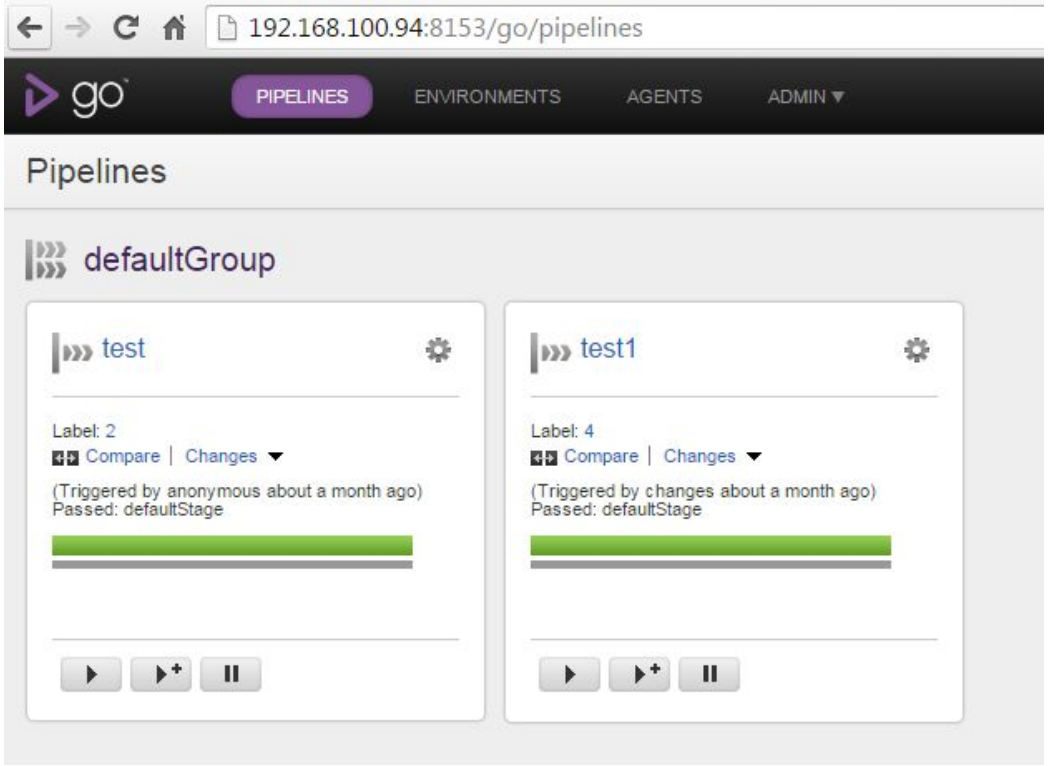
HTTP Status Code	Reason	Response Model	Headers
405	Invalid input		

Try it out!

PUT /pet — Update an existing pet

GET /pet/findByStatus — Finds Pets by status

服务持续交付(测试和部署自动化)

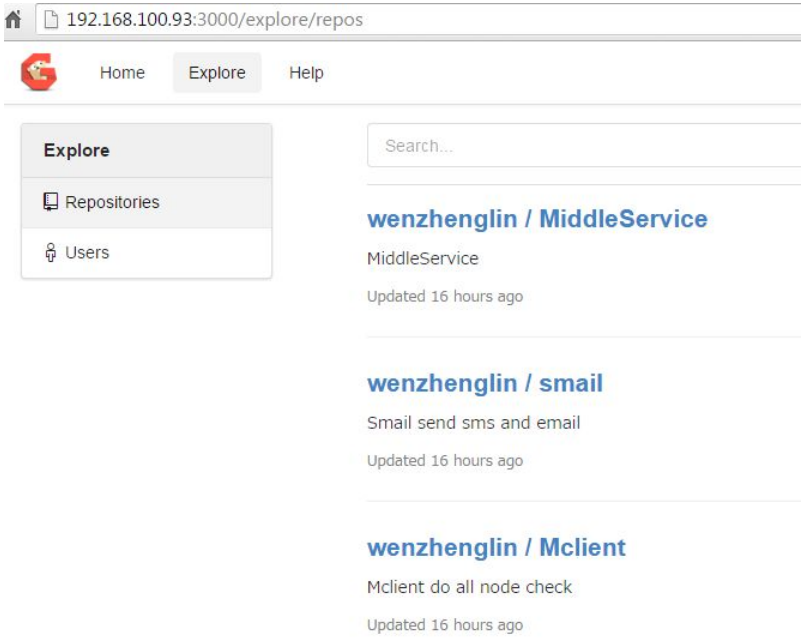


服务监控

监控平台

可视化性能监控

服务监控--监控平台



MiddleService

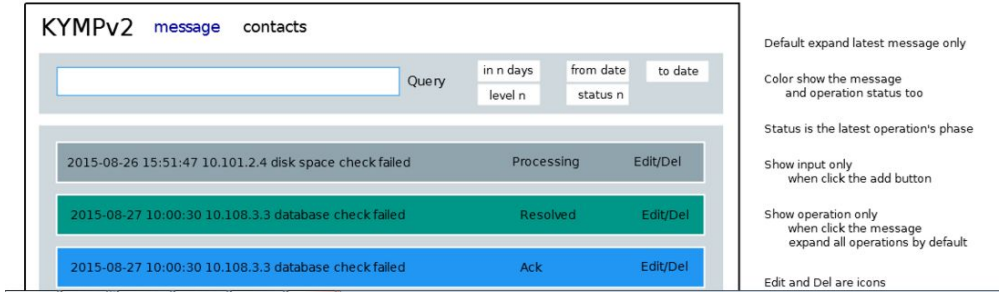
为监控平台的中间服务（检测结果接收服务），处理Mclient发送的检测结果

提供问题操作流程，对消息进行一定的控制和过滤后通过Smail进行告警的发送

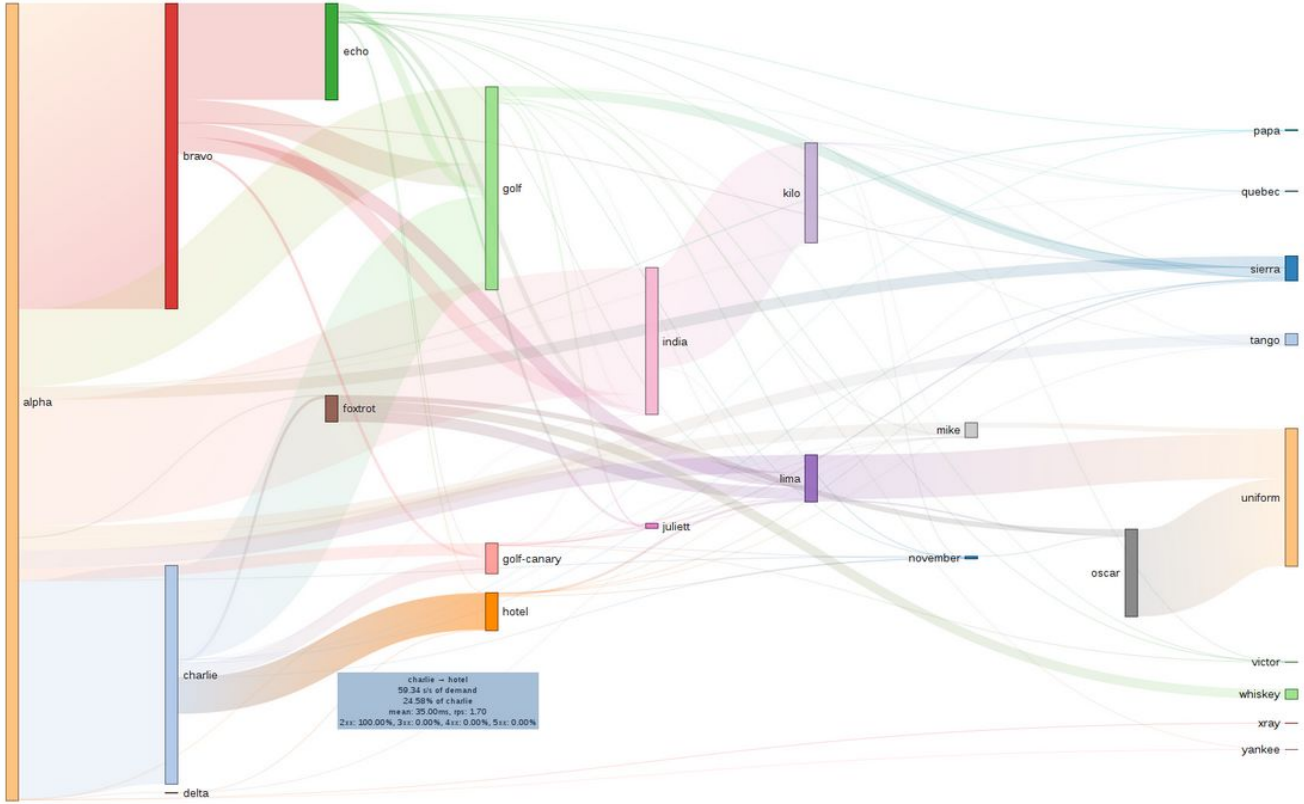
> 目前用户界面暂未实现，主要功能已投入使用

界面预览

消息界面1



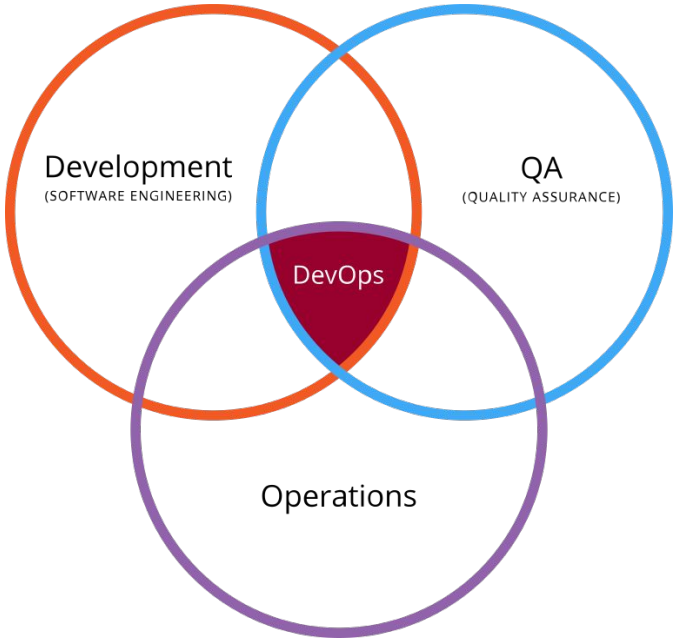
服务监控--可视化性能监控



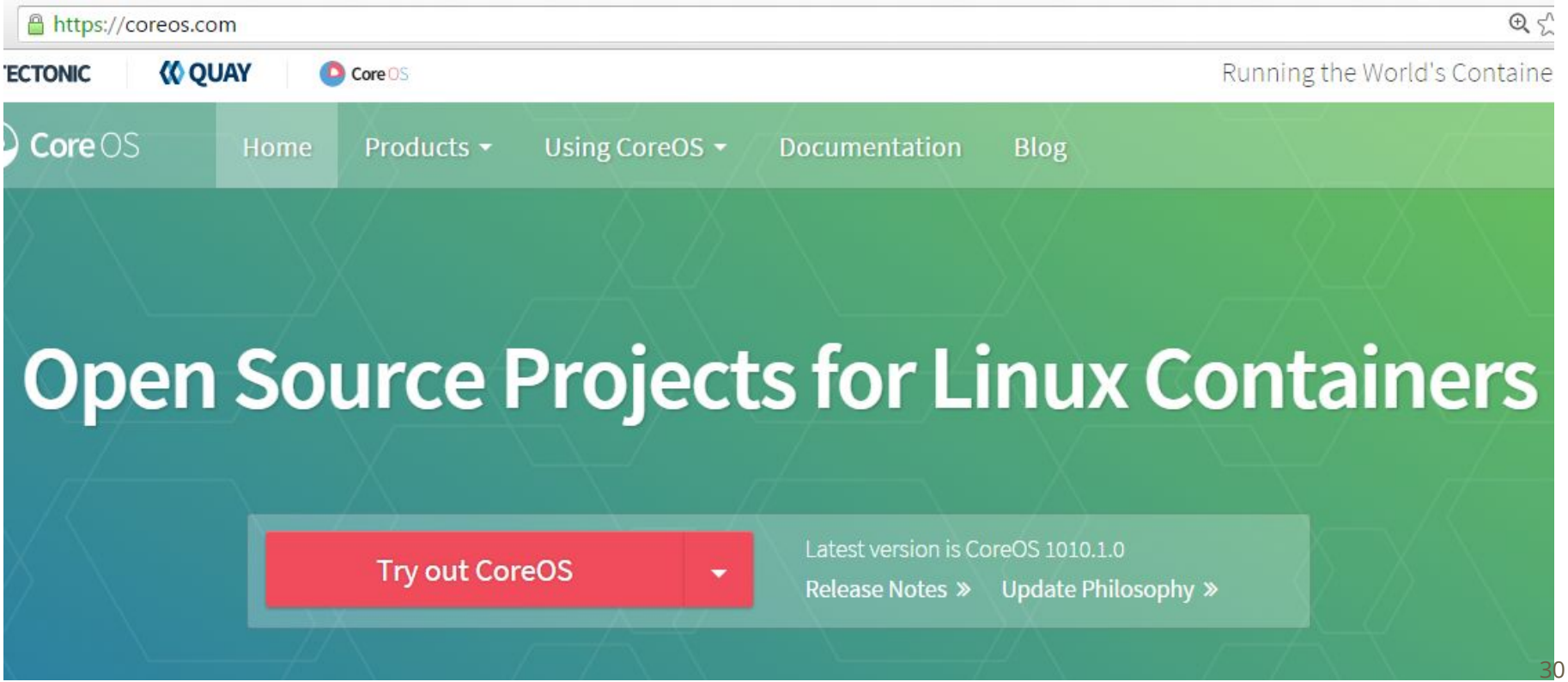
基础设施自动化管理

服务器操作系统快速配置(虚拟化，私有云平台)

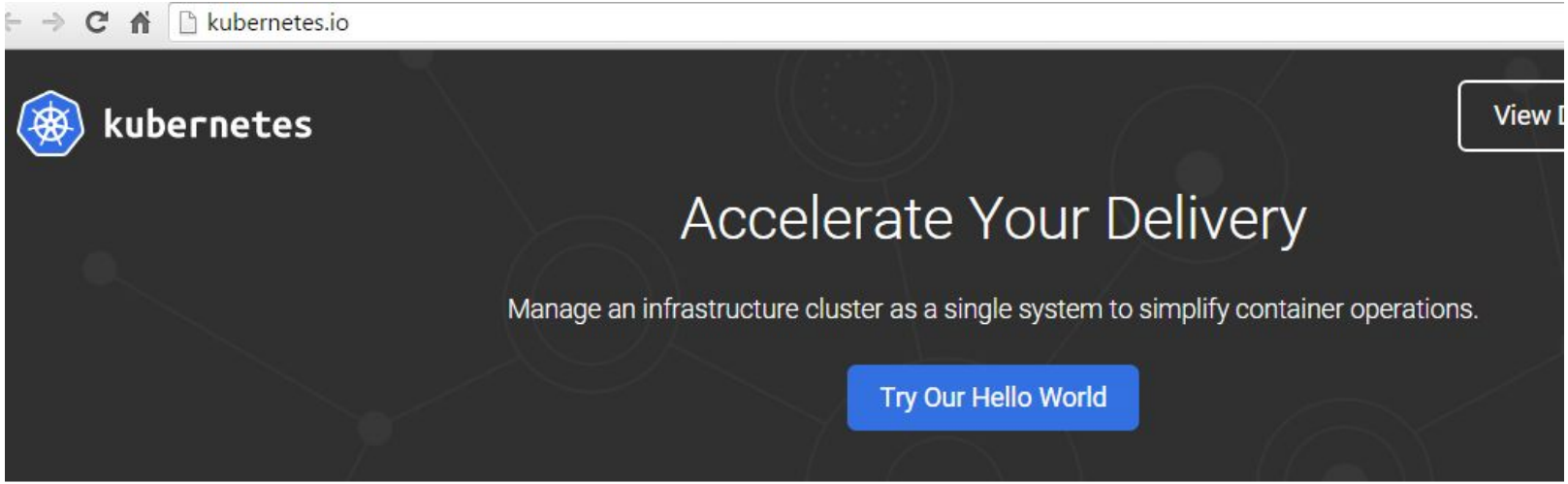
DevOps的运用



基础设施——CoreOS

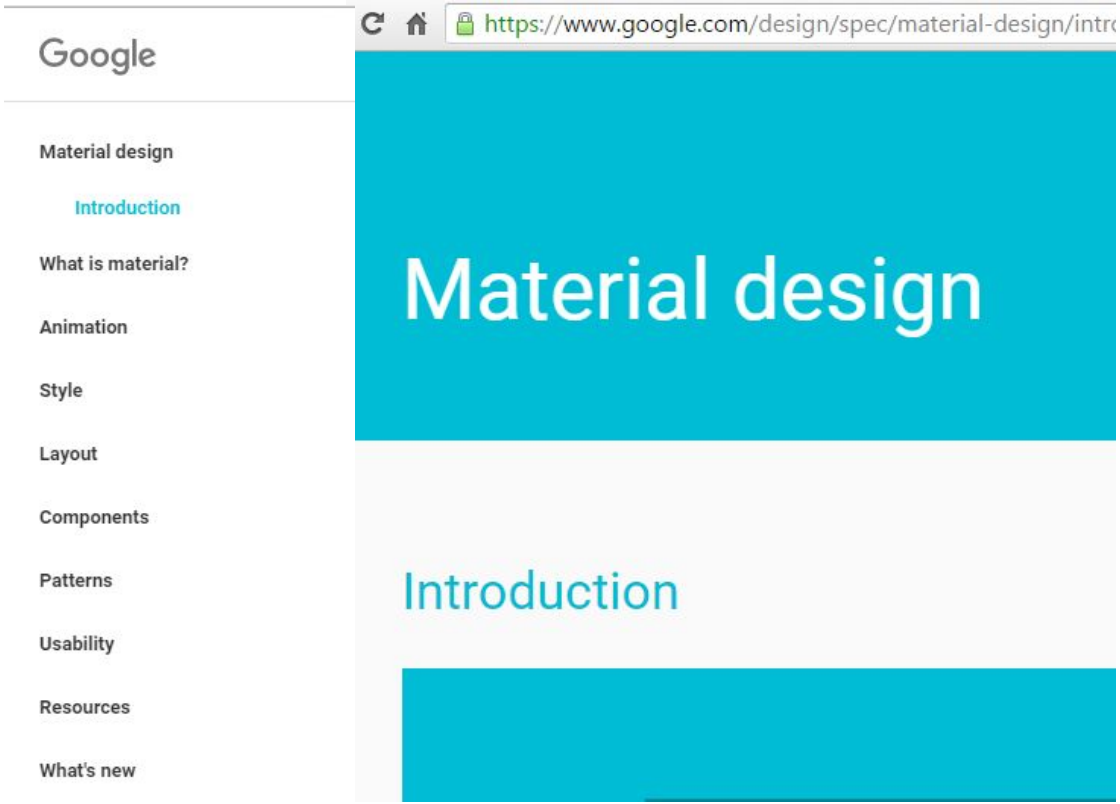

https://coreos.com
ECTONIC
QUAY
CoreOS
Running the World's Containe
CoreOS
Home
Products
Using CoreOS
Documentation
Blog
Open Source Projects for Linux Containers
Try out CoreOS
Latest version is CoreOS 1010.1.0
Release Notes »
Update Philosophy »

基础设施——Kubernetes



Kubernetes is an open-source system for automating deployment, operations, and scaling of containerized applications.

用户体验


Google
Material design
Introduction
What is material?
Animation
Style
Layout
Components
Patterns
Usability
Resources
What's new
https://www.google.com/design/spec/material-design/intro
Material design
Introduction

用户体验(续1)

https://www.polymer-project.org/1.0/

polymer

Get Started　Guides & Resources　Element Catalog　Blog

Production ready

Polymer 1.0 has been rebuilt from the ground up for speed and efficiency. The new, leaner core library makes it easier than ever to make fast, beautiful, and interoperable web components. If you haven't used Polymer before, it's time to try it out. If you haven't tried it recently, time to take another look.

GET POLYMER　VIEW ON GITHUB

用户体验(续2)

https://material.angularjs.org/latest/

Angular Material

Documentation Version

LATEST RELEASE (1.1.0-RC4)

GETTING STARTED

DEMOS

Introduction

What is Angular Material?

For developers using AngularJS, Angular Material is both a UI Component framework and a reference implementation of Google's Material Design Specification. This project provides a set of reusable, well-tested, ar accessible UI components based on Material Design.

GETTING STARTED

DEMOS

CUSTOMIZATION

API REFERENCE

组织结构的问题

Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.

— M. Conway

Ref: http://www.melconway.com/Home/Conways_Law.html

组织结构的问题(续)

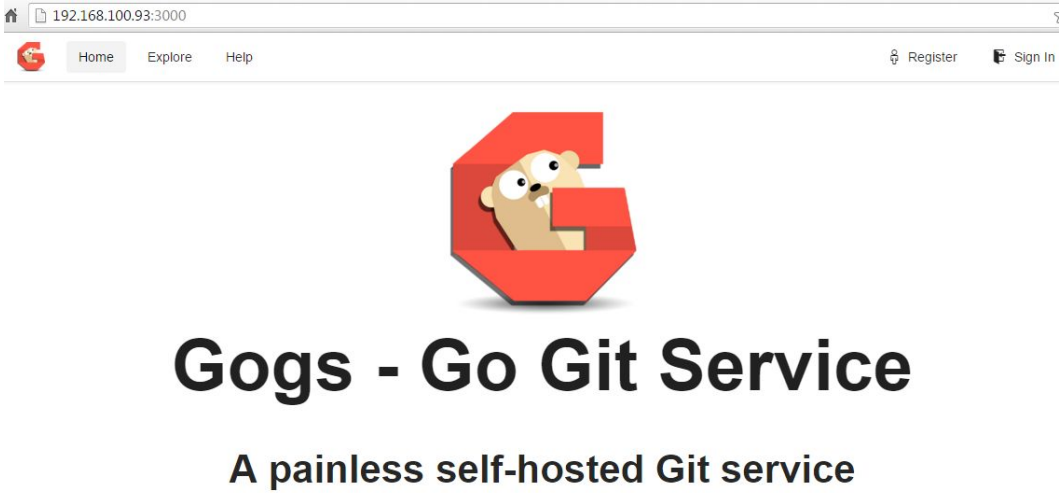
根据 ”Conway’s Law“

意味着:可能需要改变组织通信结构来适应微服务通信结构

组织结构的问题--内部开源

已单独介绍:

Internal Open Source_by_wen_2016-4-21.pdf



业务本身的复杂

历史原因和铁路运输本身导致的业务复杂

微服务并不会减少业务本身的复杂性

微服务是一种技术层面的解决方案

经验问题

没有微服务架构相关经验，进而增加了难度

其实并没有太多新的技术(rpc技术早已出现)

服务本身的实现或许变得更容易(根据业务的复杂度不同)

反而上是服务管理上增加了难度

总结

总结

微服务架构对效率(特别是服务管理的效率)要求变得更高，尽管有很多问题有待解决，基于核心问题的解决，我相信按照目前的研究程度已可用于实战

根据规模不一样考虑做到对应的程度，我们可以从非常根本(核心)的地方开始，正如微服务所说是一个持续改进的过程(演进式架构)

谢谢！

Wen Zhenglin

wenzhenglin@bjjdsy.com.cn

2016-4-28