论微服务架构及应用

历年真题

近年来，随着互联网行业的迅猛发展，公司或组织业务的不断扩张，需求的快速变化以及用户量的不断增加，传统的单块（Monolithic）软件架构面临着越来越多的挑战，已逐渐无法适应互联网时代对软件的要求。在这一背景下，微服务架构模式(Microservice Architecture Pattern)逐渐流行，它强调将单一业务功能开发成微服务的形式，每个微服务运行在一个进程中；采用 HTTP 等通用协议和轻量级 API 实现微服务之间的协作与通信。这些微服务可以使用不同的开发语言以及不同数据存储技术，能够通过自动化部署工具独立发布，并保持最低限制的集中式管理。

请围绕“论微服务架构及其应用”论题，依次从以下三个方面进行论述。

1．概要叙述你参与管理和开发的、采用微服务架构的软件开发项目及在其中所担任的主要工作。

2．与单块架构相比较，微服务架构有哪些特点？请列举至少 4 个特点并进行说明。

3．结合你参与管理和开发的软件开发项目，描述该软件的架构，说明该架构是如何采用微服务架构模式的，并说明在采用微服务架构后，在软件开发过程中遇到的实际问题和解决方案。

我的范文

摘要

正文

目的

充分地分解应用程序以促进敏捷开发和部署。以拆分和服务化为基础，将海量用户产生的大规模的访问流量进行分解，采用分而治之的方法，达成用户需要的功能指标，并同时满足用户对高可用、高性能、可伸缩、可扩展和安全性的非功能质量的要求。

特点

1.微服务把每个职责单一的功能放在一个**独立的服务**中。

2.每个服务允许在一个**独立的进程**中。

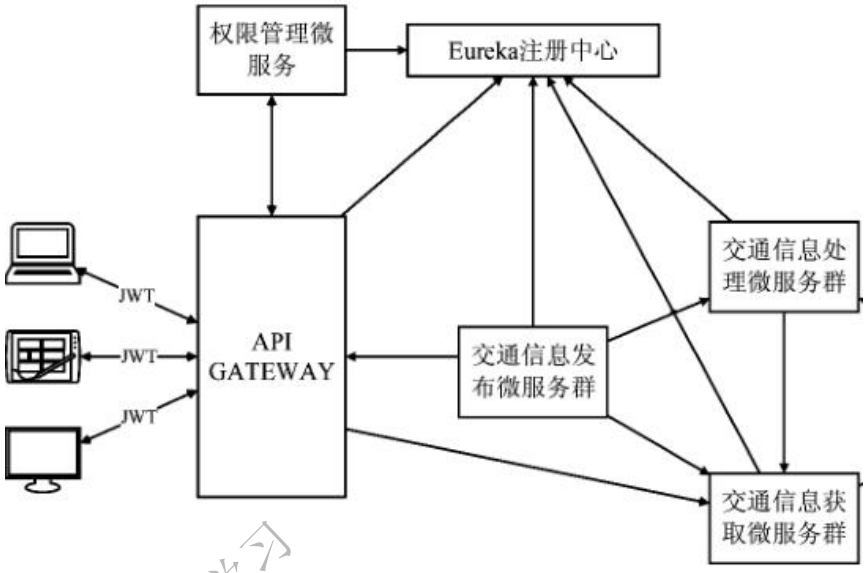
3.每个服务有自己的**数据存储**，实际上，每个服务应该有自己独享的数据库、缓存、消息队列等资源。

4.每个服务有**多个实例**在运行，每个实例可以运行在**容器化平台**内，达到平滑扩展伸缩的效果。

5.每个服务应该有自己的**运营平台**，以及独享的**运营人员**，这包括技术运维和业务运营人员。每个服务都高度自治，内部的变化对外透明。

6.每个服务都可根据**性能需求**独立的进行**水平伸缩**。

整体架构



在逻辑上，交通信息平台后端分为信息采集，信息处理，信息发布三个模块, 每个模块按处理数据的不同, 又包含多个微服务, 因此我们将系统分为交通数据采集、交通部门信号采集、路况数据处理、车位数据处理、路况数据发布、车位数据发布以及权限管理共 7 个微服务, 同时将项目团队划分为 3 个小组. 根据功能的轻重缓急和工作量，安排各个微服务的研发。每个小组负责一个或多个组件完整的生命周期，即服务谁开发，服务谁运营。最后各个服务组件通过 RESTful HTTP 协议和消息路由功能进行服务组装。

治理框架. 选用的微服务治理框架为 Spring Cloud，其为一系列框架的有序集合。它利用 Spring Boot 的开发便利性巧妙地简化了分布式系统基础设施的开发，如服务发现注册、配置中心、消息总线、负载均衡、断路器、数据监控等，都可以用 Spring Boot 的开发风格做到一键启动和部署。(在负载均衡方面, 我们主要支持随机、轮询、最少链接数的策略将来自网络的请求随机分配给内部中的多个服务器)

Eureka 注册中心. 在每个微服务启动时，向 Eureka 注册中心进行注册，并维持心跳连接，这样系统的维护人员就能够通过 Eureka Server 来监控系统中的各个微服务是否正常运行，Spring Cloud 的一些其他模块就可以通过 Eureka Server 来发现系统中的其他微服务，并执行相关的逻辑。/// 因为各个微服务部署时每次使用的 IP 并不一定固定，并且同一个微服务可能有多个实例来实现同样的功能。使用注册中心可以将 IP 隐藏起来，让其他的微服务通过固定的名字(服务名称)去调用这个微服务，就能够实现负载均衡和动态部署，当这个微服务承受的压力过大时，再启动一个相同的实例并注册就能实现动态的横向扩容。

权限管理. 在微服务框架中，需要面向用户提供服务的是信息发布模块，其他模块只需要满足微服务架构内部的调用。所以权限的管理统一放到网关(API GATEWAY)中，API 网关通过 Zuul 实现，所有的外部请求都要经过网关，在网关中判断本次请求的合法性，并作出响应。/// 由于微服务采用 RESTFUL 的通信模式，是无状态的(适用于传统单体应用的 shiro 此时不好使)，所以引入 JWT (JSON WEB TOKEN) 认证协议解决权限问题.

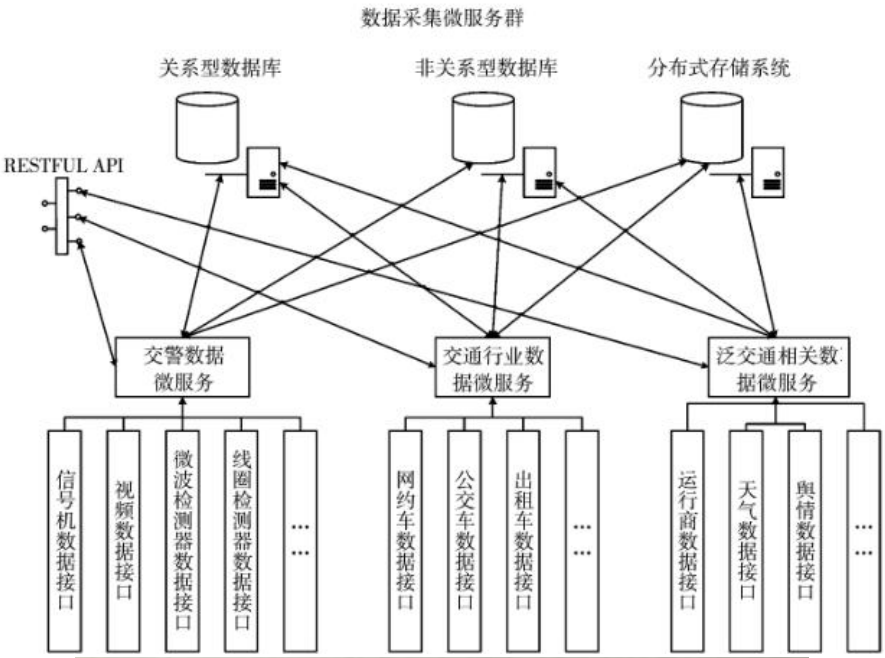
~~JWT 是一种用于双方之间传递安全信息的简洁的、URL 安全的表述性声明规范。JWT 鉴权流程如下，客户端在提交用户名密码后，服务端调用权限微服务验证用户是否存在，并获取用户的角色信息，加入到 JWT 的负载中，服务端返回签名后的 JWT。客户端在之后的访问中带上 JWT，服务端对 JWT 进行解析后，对其进行验签，验签通过则调用权限微服务，根据 JWT 中带有的用户信息获取本次访问是否有对应的权限，并做出相应的响应。~~

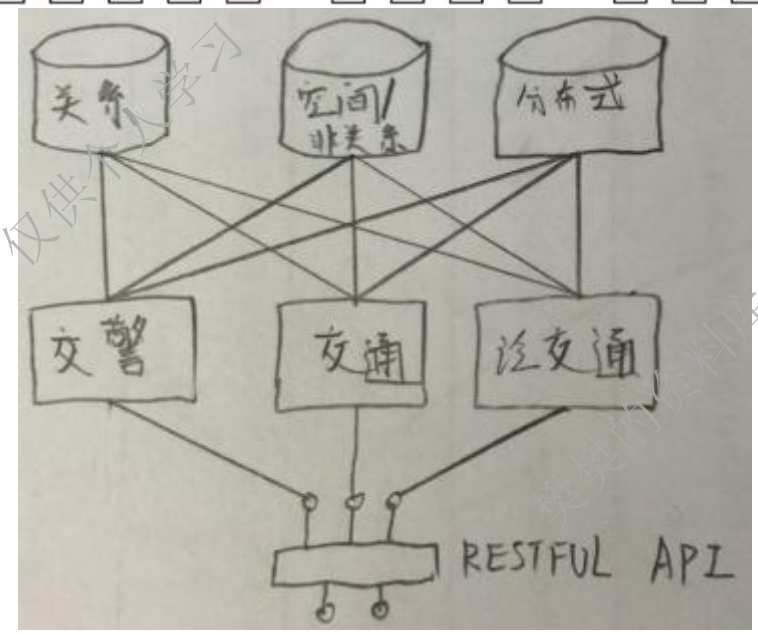
前后端通信. 对于交通信息系统而言，最终目的是为道路交通参与人员提供交通信息服务，获取相关信息的方式有多种，比如桌面客户端，移动客户端，浏览器端。对于这些不同的用户终端而言，其数据展示形式是不同的，但是这些不同的数据展示形式都对应着同样的信息或者若干种信息的有限种组合。这就要求后端只提供数据，而数据的视图在客户端生成。所以对于不同的终端而言，后端要提供统一的数据格式与同一组数据内容，终端根据服务的对象获取需要的数据，并进行展示。采用 RESTful API 进行前后端数据交换，使用 JSON 格式传递数据，由于 JSON 的自解释性，能够减少前后端之间的耦合。

数据采集微服务群

交通数据由于来源的不同可以分为交警数据(视频卡口、微波、线圈和信号机等设备检测运行产生的数据)、交通行业数据(出租车、网约车和公交车等的 GPS 数据)和泛交通相关数据(运营商数据、天气和舆情等数据等).

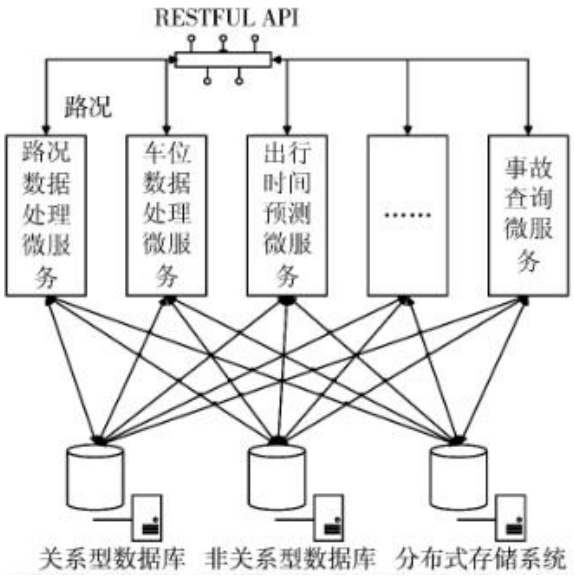
交通信息采集的目的在于监测道路运行状况，通过采集的数据可以分析路段和交叉口运行状态，预测交通事件发生为组织和诱导提供数据支持。对于多元的信息来源带来的数据异构化问题可以将信息来源进行分类和归纳，拆分为不同的模块，使用不同的独立微服务来承接数据，实现数据的接收与存储。/// 对于数据量较小的服务可以放到传统的数据库中，在做好硬盘 raid 的情况下可以保证数据的安全。传统数据库分为关系型和非关系型数据库(还有空间数据库)，可以根据存储数据的类型选择合适的数据库产品。对于数据量比较大的数据，可以采用分布式文件系统来进行存储。通过低成本的机器集群，组成一个大容量，高可用，具有容错能力的分布式文件系统。比如目前使用比较普遍的，开源的 hadoop 项目提供的 HDFS 分布式文件存储系统。





数据处理微服务群

在数据处理层面，可以按照业务场景和处理数据的种类拆分为微服务，在每个微服务之间留下接口，进行微服务之间的调用。数据处理微服务通过调用一个或多个数据采集微服务获取原始数据，再根据具体业务规则进行计算，得到面向用户需求的一些具体指标和数据.根据功能需求，我设计了路况数据处理微服务、车位数据处理微服务、出行时间预测微服务、事故查询微服务等.



信息发布

信息发布是交通信息系统的最后一个环节，信息发布有两种形式，推送和请求。推送是当一条信息进入消息队列之后，对每一个曾经订阅过这条消息的终端进行推送。而请求则是通过客户端对服务端 API 的资源请求，在经过请求合法性验证之后，由服务端向客户端返回请求的数据。

微服务集群的部署

当传统的单体服务被拆分为微服务以后由于实例数量也从一个变成了多个，对项目的部署与启动带来了挑战，所以引入容器化的方法来部署项目。

通过 docker 将每一个微服务打包成一个能独立运行的容器，并且通过 docker-compose 描述这些镜像之间的依赖关系，最终通过一个统一的入口，一步启动整个微服务集群。/// 同时可以在保证后端微服务之间调用通畅的情况下不对外暴露端口，只将需要对外提供服务的端口映射到物理机端口，实现将后端微服务隐藏起来的效果，从而保证微服务集群的安全性。

问题和解决方案

在微服务实践中，我们主要遇到三个问题，一运维开销及成本增加，因为每个微服务需独立运行，还可能采用多种语言环境，这将导致资源开销大(服务划分应尽量合理，不要划分太细太多)；二代部分代码重复，某些底层功能需要被多个服务所用(采用公共模块的方式提供底层服务)；第三个问题是难以可视化及全面测试，在动态环境下服务间的交互会产生非常微妙的行为(通过监控发现生产环境的异常，进而快速回滚，弥补可测性不足的问题)。

端口对外暴露太多: docker 容器, 隐藏内部端口, 对外映射

服务太多启动麻烦: docker 容器, 一键部署