**微服务架构**

个人作业 郑宇-2022141461089

**1. 主要叙述**

微服务架构通过将应用程序分解为一系列小型服务，每个服务运行在独立的进程中，并使用轻量级通信机制相互通信，以实现业务逻辑。这种架构支持分布式部署、高可用性和可伸缩性，同时运维智能化也是其优势之一。微服务架构允许前后端分离，加快系统响应速度，满足互联网化环境下的业务需求。

**2. 应用场景**

微服务架构适用于需要快速迭代、高可用性和可扩展性的现代互联网应用，特别是在电商平台、金融服务、物流、互联网服务等行业中表现突出。例如，在金融领域，微服务架构通过分布式的方式实现灵活的业务处理和容错机制，支付宝、PayPal等支付系统广泛采用微服务架构。在物流行业，顺丰、UPS等物流公司采用微服务架构管理订单处理、仓储、配送、路线优化等业务流程。

**3. 优点和缺点**

优点：

- 灵活性和可维护性： 服务独立部署和更新，便于维护和升级。

- 可扩展性： 可以根据需求独立扩展服务，提高资源利用率。

- 容错性： 单个服务的故障不会导致整个系统的崩溃。

- 技术多样性： 团队可以选择最适合的技术栈开发各自的服务。

- 高可用性：服务可以水平扩展，以应对高流量和高并发请求。

- 更好的可移植性：微服务架构允许服务在不同的服务器、容器甚至是云环境中部署，支持弹性扩展和高可用性。

缺点：

- 复杂性： 管理多个服务的部署、协调和监控增加了复杂性。

- 数据一致性： 需要额外的工作来保证跨服务的数据一致性。

- 网络延迟： 服务间的通信可能导致额外的网络延迟。

- 分布式系统的挑战：微服务架构引入了网络通信、服务发现、负载均衡、分布式追踪等新的复杂性。

- 更多的工作量和更高的运维成本：管理多个服务的部署、协调和监控增加了工作量和运维成本。

**4. 技术栈**

- 编程语言： Java、Node.js、Python、Go等。

- 框架： Spring Boot、Express.js、Django、Flask等。

- 服务发现与注册： Eureka、Consul、Zookeeper。

- 配置管理： Spring Cloud Config、Consul。

- API网关： Zuul、Nginx、Kong。

- 消息队列： Kafka、RabbitMQ。

- 数据库： MySQL、PostgreSQL、MongoDB、Cassandra等。

- 容器化和编排： Docker、Kubernetes。

- 服务网格：Istio、Linkerd，为微服务提供服务发现、负载均衡、故障恢复等。

- 监控和日志：Prometheus、Grafana、ELK Stack，用于监控服务状态和日志管理。

- 持续集成/持续部署（CI/CD）：Jenkins、GitLab CI、GitHub Actions，支持自动化部署和测试。

**5. 现阶段哪些较为知名的系统和第三方软件、库使用了这个架构，并简单评价**

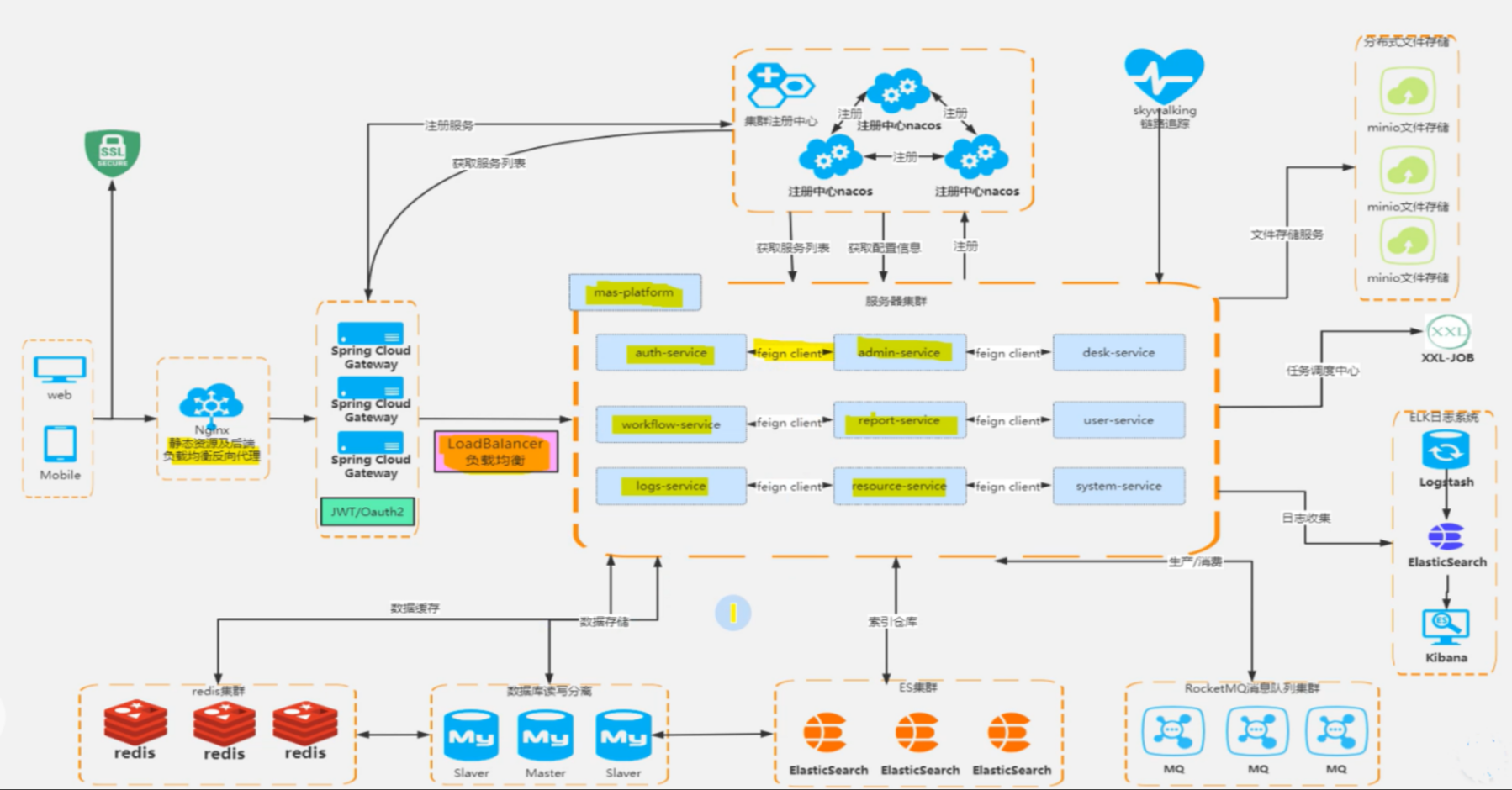
- Netflix： 广泛使用微服务架构，开发了包括Eureka、Hystrix在内的多个微服务支持组件，是微服务架构的先驱之一。

- Amazon： 其电商和云计算服务背后有大量的微服务支持，展示了微服务架构在大规模分布式系统中的可行性。

eBay：通过采用微服务结构模式解决了单体应用的问题，将应用分解为小的、互相连接的微服务，每个微服务都是微型六角形应用，都有自己的业务逻辑和适配器。

评价： 这些公司的成功案例证明了微服务架构在构建可扩展、高可用的现代应用程序方面的强大能力。然而，这也要求公司有足够的技术实力来管理微服务架构带来的复杂性。

1. **微服务架构设计中常用组件**



1.流量入口Nginx

在上图中可以看到，Nginx作为整个架构的流量入口，可以理解为一个外部的网关，它承担着请求的路由转发、负载均衡、动静分离等功能。作为一个核心入口点，Nginx肯定要采用多节点部署，同时通过keepalived来实现高可用，从而保障整个平台的高可用。

2.网关

网关是在Nginx后的另外一个核心组件。它承担着请求鉴权，路由转发，协议转换，流量监控等一系列功能，上图中网关是采用spring Cloud Gateway来实现业务网关的功能，在网关选型中，还有其他的选择，比如Zuul1,Zuul2,Kong等等，这些方案都有自己的优势和局限性，可以根据自己他们的特点来抉择到底选用哪一个方案。

上图中，Spring Cloud Gateway下面有jwt和OAuth2，这两个就是基于token的认证鉴权，一般互联网项目中，在登录模块都是支持微信或者qq登录，这就是用到OAuth2的授权登录。

3.业务组件

从上面的架构图中可以看到，网关之后就是我们的业务组件了，可以理解就是拆分之后的微服务了，比如电商平台常见的账号服务、订单服务、发票服务、收银台服务等等。服务组件之间通过Feign来进行http调用，Feign集成Ribbon来实现客户端侧负载均衡。

4.服务注册中心

不管是基于Dubbo实现的SOA，还是基于Spring Cloud拆分的微服务架构，服务注册中心都是必须的，把所有的服务组件都注册到注册中心，进而实现服务的动态调用。常见能实现注册中心功能的有Zookeeper,Eureka,Nacos。

5.缓存和分布式锁

在图中左下角，我们可以看到Redis组件，我们可以把Redis作为缓存来使用，把一些查询慢，使用率高的热点数据做缓存处理，能快速提高接口响应时间。同时redis在微服务中的一大使用场景就是分布式锁，传统的Sychronized和显示Lock锁显然是不能解决分布式并发问题。

为了保障Redis的高可用，可以采用哨兵部署，不是三个redis节点，一主二从，同时部署三个哨兵节点，来实现故障转移，避免单点问题，如果Redis存储的数据量很大，达到了单节点的Redis的性能瓶颈，也可以用Redis集群模式来实现分布式存储。

6.数据持久层

不管单体服务，还是微服务，数据持久层都是必须的，选用互联网项目经常使用的mysql作为DB，为了保证服务读写效率以及高可用性，主从分离模式，同时实现读写分离，来保障mysql的读写性能。

随着业务量增长，单表的数据量达到性能瓶颈之后，就要采用分库分表来对数据库表进行水平拆分和垂直拆分了，具体如何进行合理的拆分，以及技术选型，这些和项目现有的表结构设计是息息相关的，要考虑后续的可拓展性。

7.结构型数据存储

mysql比较擅长存储关系型数据，项目中有需要存储结构性数据的场景，比如存储JSON字符串，这种场景通过mysql来存储显然事不合适的。一般我们会采用Elasticsearch或者MangoDB来进行存储，如果业务中需要检索功能，更建议使用Elasticsearch。Elasticsearch支持DSL，有比较丰富查询检索功能，甚至能实现GIS空间检索功能。

8.消息中间件

前面说到，微服务架构中，服务之间同步调用是通过Feign来实现的，那服务间的异步解耦就要通过MQ来实现了。虽然我们可以通过多线程来实现异步调用，但是这种异步调用不支持持久化，可能会造成消息丢失，所以一般都集成RabbitMq或者RocketMq。

9.日志收集

在微服务架构中，通过一个组件，比如说订单服务都是多节点分布式部署，每个节点的log日志都是存储在节点本地，如果要查询日志，我们难道要登录到各个节点找到对应的日志信息？这种查看日志肯定是不行的。所以一般会引入ELK来做日志收集，和可视化展示查询。

- Logstash 用来做日志收集工作，通常在Logstash前会加一个Filebeat，由Filebeat来收集日志，Logstash做数据转换工作。

- Elasticsearch做数据存储，以及生成索引数据，便于Kibana做检索。

- Kibana做数据的展示，以及查询检索功能，我们通过检索关键词就能快速的查询到想要日志信息。

10.任务调度中心

项目中经常会用到定时功能，单体应用中，我们使用sping自带的Schedule，或者使用Quartz即可，在分布式应用中，我们就要集成分布式定时器，比如Quartz(Quartz配合数据库表也是支持分布式定时任务的)，还有Elastic-Job、XXL-JOB等等。

Elastic-job 当当网基于quartz 二次开发的弹性分布式任务调度系统，功能丰富强大，采用zookeeper实现分布式协调，实现任务高可用以及分片。Elastic-Job是一个分布式调度的解决方案，由当当网开源，它由两个相互独立的子项目Elastic-Job-Lite和Elastic-Job-Cloud组成，使用Elastic-Job可以快速实现分布式任务调度。

XXL-JOB 是一个分布式任务调度平台（XXL是作者徐雪里姓名拼音的首字母），其核心设计目标是开发迅速、学习简单、轻量级、易扩展。将调度行为抽象形成“调度中心”公共平台，而平台自身并不承担业务逻辑，“调度中心”负责发起调度请求。将任务抽象成分散的JobHandler，交由“执行器”统一管理，“执行器”负责接收调度请求并执行对应的JobHandler中业务逻辑。因此，“调度”和“任务”两部分可以相互解耦，提高系统整体稳定性和扩展性。

11.分布式对象存储

项目中经常会有文件上传功能，比如图片，音频视频。在分布式架构中，我们将文件存储在节点服务器上显然是不行的，这时候，我们就需要引入分布式文件存储。常见方案有MinIo、阿里的OSS（收费），阿里FastDFS等等。

MinIO 是一款基于Go语言发开的高性能、分布式的对象存储系统。客户端支持Java,Net,Python,Javacript, Golang语言。

FastDFS是一个开源的轻量级分布式文件系统，它对文件进行管理，功能包括：文件存储、文件同步、文件访问（文件上传、文件下载）等，解决了大容量存储和的问题。特别适合以文件为载体的在线服务，如相册网站、视频网站等等。