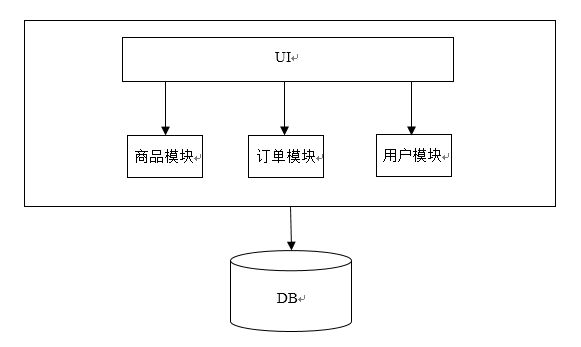
# DR Cloud平台技术架构概述

## 背景

目前公司的主要产品，采用的技术架构是以单体架构设计开发的。

单体架构如图所示：



单体架构的特点是：

1所有的功能集成在一个应用中。

2所有的功能基于相同框架

3应用打包成一个war包部署到服务器上。

4应用与数据库分开部署。

5通过部署应用集群和数据库集群来提高系统的性能。

单体架构的缺点是：

1 可靠性差：单体架构的应用的崩溃，会导致整个应用的崩溃。影响所有业务用户。

2 性能制约：应用的用户群访问单体架构的应用时，web容器，数据库连接池、数据库由于其负载上限，面对请求高峰时容易发生阻塞，存在性能制约。

3 技术债务：当某些功能模块或者框架由于业务和技术的不成熟造成问题，往往整体的修改会带来较大的成本。比如前端框架发生问题，转化新的框架之后，往往也会造成后端应用的大量修改。

4 扩展能力受限：随着整体的扩展，越来越多的功能耦合在一起，单体架构内部形成蛛网状，代码难以阅读，代码与业务域的映射越来越复杂，造成了扩展的难度

公司之前采用的单体架构，已经经历了上述问题的阵痛。

1 mes上线和m3p现场部署时都曾发生的一个业务功能问题造成整个应用崩溃的问题，进而使得用户所有的功能无法正常使用。（可靠性）

2 mes发生的数据库连接池负载上限的问题。（性能）

3 m3p在前端框架选择失误后造成现在诸多前端问题无法修改，选择重构后后端服务也需要大量修改，造成返工量的成本问题（技术债务），前端问题详见测试结论------历史遗留问题

4 m3p由于之前架构的问题，造成现在扩展的难度。MES和EAP新业务的扩展同样存在相同问题（扩展问题）

综上所述，公司目前需要采用一套适合公司当下和未来的技术架构，来解决公司实际的面临的技术问题。

## 微服务架构

微服务架构是一种将单一应用开发为一组小型服务的方法，每个服务运行在自己的容器中，服务之间通过消息中间件MQ、RPC、Rest API调用进行通信，这些服务围绕小粒度业务构建并且可通过全自动部署机制独立部署。这些服务共用一个最小型的集中式管理，服务可用不同的语言开发，使用不同的数据存储技术。

微服务架构如图所示：



微服务架构的优点：

1一个微服务只关注一个特点的业务功能，独立部署单独运行出现问题时并不影响整体的应用。（可靠性）

2 一个微服务运行在一个容器中，请求相对单体架构的应用进行了分流，web容器负载减少，提高了性能。一个微服务拥有一个独立的数据库连接池，连接池负载减少，提高了性能。（性能）

3 微服务技术栈不受限，可以根据业务和项目团队的特点，合理选择技术栈。即时单个微服务即使出现技术或者业务的失误，返工量较小。。（技术债务）

4微服务按需伸缩，可以基于实际需求，实现细粒度的扩展，或者选择增加服务节点。（扩展）

5微服务只关注整个系统功能中单独、有界限的一部分。单一职责可以帮我们优雅的开发、更便捷的交付。

## 技术体系

目前在行业内，已经有一些有影响力的公司和组织提供了对微服务的技术支持，并且进行了开源共享。

最著名的是阿里公司的Dubbo和spring开源的spring cloud

1 微服务之spring cloud 和dubbo。

Dubbo是阿里巴巴公司基于自身业务构建并且开源的一个高性能的服务注册中心。

Dubbo只实现了微服务技术体系的注册中心。对于微服务的分布式问题，如脑裂，各服务运行可靠性问题等，Dubbo需配合zookeeper使用。Zookeeper是一个分布式协调框架。另外，对于微服务的配置中心，分布式跟踪，健康问题等都需要自己去集成。

Dubbo现状：一度暂停维护，目前移交给Apache组织

Spring Cloud是Spring 开源组织开源的一套完善的微服务解决方案。

Spring Cloud 提供了高可用的服务注册中心，注册中心支持集群；提供了服务网关、消息总线、服务治理、健康监控、分布式链路跟踪、熔断、配置中心等。

Spring cloud 和 Dubbo对比示意图如下：

|  |  |  |
| --- | --- | --- |
| **核心要素** | **Dubbo** | **Spring Cloud** |
| 服务注册 | Dubbo**+**Zookeeper | Eureka、Consul、Zookeeper |
| 服务调用方式 | RPC | REST、伪RPC |
| 服务网关 | 无 | Spring Cloud Netflix Zuul |
| 断路器 | 不完善 | Spring Cloud Netflix Hystrix |
| 分布式配置 | 无 | Spring Cloud Config |
| 分布式追踪 | 无 | Spring Cloud Sleuth |
| 消息总线 | 无 | Spring Cloud Bus |

注：RPC调用底层采用RPC协议，长连接，特点：建立连接时间长、代码不容易复用，侵入式强

Rest调用底层采用http协议。特点：无状态协议，轻量级通讯，代码无耦合

结论：spring cloud更有优势 。

2 spring cloud 之 三大组件服务注册发现支持。

Spring cloud 服务注册发现的实现目前有Eureka、Consul和Zookeeper。

Eureka是Netfix公司推出的服务注册发现组件、用于实现服务的发现和注册、它的特点是高可用、服务不可用或者协调通过熔断去处理。保证数据一致性需要通过分布式TCC事务或者可靠消息一致性方案

Consul 是 HashiCorp 公司推出的开源工具，用于实现分布式系统的服务发现与配置。

Consul 内置了服务注册与发现框架、分布一致性协议实现、健康检查、Key/Value 存储、多数据中心方案。Consul为了保证数据的一致性，需要使得保证节点上的服务是可用的，因此采用了选举机制、consul采用的选举算法是raft。但是consul由于选举时需要等待，牺牲了整体的性能。Consul从选举算法上来说也不能100%保证服务的可用性

Zookeeper和Consul类似，但是选举算法是poxas，并且zookeeper不内置服务注册与发现，需要集成三方或自己实现。

结论：Eureka更有优势

3 mq消息中间件

目前主流的mq消息中间件有如下几种：kafka、robitmq、activemq。

Kafka是高可用的消息中心，但是不保证消息是否丢失，kafaka持久化支持的是文件、官方给出的吞吐量为10.6w/s

RobitMQ在一致性上做了保证，保证消息的可靠性，RobitMQ持久化支持的是文件、官方给出的吞吐量为6.1w/s。并且spring cloud bus默认内置。

ActiveMQ与RobitMQ实现理念类似、持久化支持数据库、官方吞吐量数据不明。

针对此三种MQ团队内部做了实测，测试结论见附件

结论：robitMQ更可靠。

4 数据库连接池

目前主流的数据库连接池有如下几种：dbcp、c3p0、druid、tomcatjdbc、hikari

针对此五种数据库连接池团队内部做了实测，测试结论见附件

结论：容器如果选择tomcat时，tomcatjdbc更优；但是如果选择其他容器部署、hikari更优秀

5 工作流

目前主流的工作流框架有jbpm和activiti

BPM是目前市场上主流开源工作引擎之一，在创建者Tom Baeyens离开JBoss后，jBPM5完全放弃了jBPM4的基础代码重头来过，目前官网已经推出了jBPM6的beta版本；Tom Baeyens加入Alfresco后很快推出了新的基于jBPM4的开源工作流系统Activiti。由此可以推测JBoss内部对jBPM未来版本的架构实现产生了严重的意见分歧。

|  |  |  |
| --- | --- | --- |
| 技术组成 | Activiti | JBPM |
| 持久层 | Mybits | Hibernate |
| 事务管理 | Spring事务控制 | Jta事务管理 |
| 支持数据库 | Oracle、Mysql、Sqlserver | Oracle、Mysql、Sqlserver |
| 设计模式 | Command模式、观察者模式 |  |
| 内部服务通信 | API调用 | Mina |
| 集成接口 | soap、restful | 消息通讯 |
| 流程格式 | BPMN2、xPDL、JPDL | BPMN2 |
| 引擎核心 | 流程虚拟机 | Drools |

优劣对比

从技术组成来看，Activiti最大的优势是采用了PVM（流程虚拟机），支持除了BPMN2.0规范之外的流程格式，与外部服务有良好的集成能力，延续了jBPM3、jBPM4良好的社区支持，服务接口清晰，链式API更为优雅；劣势是持久化层没有遵循JPA规范。

jBPM最大的优势是采用了Apache Mina异步通信技术，采用JPA/JTA持久化方面的标准，以功能齐全的Guvnor作为流程仓库，有RedHat(jBoss.org被红帽收购)的专业化支持；但其劣势也很明显，对自身技术依赖过紧且目前仅支持BPMN2。

结论：选择使用jbpm。

6 WebOffice控件选型

目前常用的WebOffice控件有金格的iWebOffice、软航的NTKO OFFICE。对于这2个厂商的产品我们都已经做过集成。

金格的iWebOffice当时集成过来之后，在频繁的打开操作、退出之后，在不关闭浏览器的前提下，当前浏览器页面会越来越卡，打开的速度会越来越慢，有性能问题。和厂商沟通之后，在换版本之后，问题还是没有解决。

软航的NTKO OFFICE集成之后，没有和金格相同的问题，性能基本保持不变，另外客户现场目前使用的也是软航的NTKO OFFICE。

结论：ntko。

7 redis与memcached

Redis支持多数据类型、memchaed只支持key、value。

Redis支持事务、Memcached不支持事务。

Redis通过选举算法支持分布式、Memcached支持主从结构

结论： redis。

8 前后端分离及前端工程技术

前后端分离的必要性在前面已经提到过。

前端目前主流技术vue、elements以及angluar。由于公司客户的特殊性、目前决定采用angular+node+ng；

Vue和Elements不支持低版本的浏览器

结论：选择angular

## 总体架构



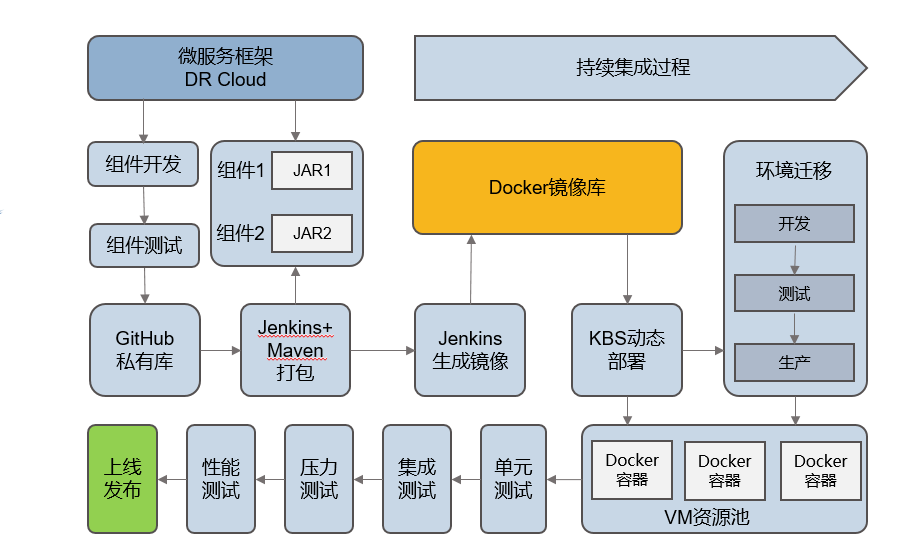
云平台

云技术是指在广域网或局域网内将硬件、软件、网络等系列资源统一起来，实现数据的计算、存储、处理和共享的一种托管技术

DR cloud提供redis缓存集群、mq消息中间件集群、日志服务、配置中心、工作流、weboffice插件、权限中心、服务注册与发现，分布式等基础能力，业务服务只需按规范约定开发，便可以托管到云中心中运行。

对于服务可靠性的问题，团队采用熔断，集群、分布式事务方案做了有效备案。

## 研发流程示意图



**六、结论**

以上是我们本次技术架构的整体情况和核心细节，请大家具体评判

注：权限模块处理模式