

【软考达人】

# 软考资料免费获取

- 1、最新软考题库
- 2、软考备考资料
- 3、考前压轴题



微信扫一扫，立马获取



6W+免费题库



免费备考资料

PC版题库：[ruankaodaren.com](http://ruankaodaren.com)

# 论微服务架构及其应用

## 摘要

2016年7月，我所在的公司为全国各级人民检察院开发了行贿犯罪档案互联网查询系统的产品，我担任系统架构师职务，主要负责软件架构和安全体系设计的工作，该项目是基于互联网，为单位、企业和个人等公众群体提供7\*24小时的查询申请服务，同时兼顾行贿犯罪预防宣传。本文结合作者的实践，以行贿犯罪档案互联网查询系统为例，论述微服务架构及其应用。首先概述我参与管理和开发，并采用微服务架构开发的工作，然后具体描述微服务架构的特点，最后结合项目描述软件的架构，说明该系统是如何采用微服务架构模式的，并说明采用微服务架构模式后，在软件开发过程中遇到的实际问题和解决方案。经过项目组近一年的努力，本产品已顺利开发完成，目前，已在浙江、云南等多省上线使用，取得客户和公司领导的一致好评。

## 正文

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等通信协议和轻量级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。

2015年7月，我所在的公司为全国各级人民检察院开发了行贿犯罪档案互联网查询系统的产品，我担任系统架构师职务，主要负责软件架构和安全体系设计的工作。本文结合作者的实践，论述微服务架构及其应用。首先概述我参与管理和开发，并采用微服务架构开发的工作，然后具体描述微服务架构的特点，最后结合项目描述软件的架构，说明该架构是如何采用微服务架构模式的，并说明采用微服务架构模式后，在软件开发过程中遇到的实际问题和解决方案。

## 项目概述

随着互联网的飞速发展，基于互联网平台建设行贿犯罪档案查询系统（Internet Bribery Crime Record Query IBCRQ），为单位、企业和个人等公众群体提供实时、高效、方便的申请查询服务，是互联网+智慧检察的深度融合，也是社会经济发展的必然趋势。IBCRQ 系统的建设，作为政府采购和招标审查的必经关口，将有行贿犯罪记录者拒之“门”外，大大降低了政府采购、工程建设等领域官商勾结、权钱交易的几率，为有效预防贿赂、震慑犯罪提供了很好的积极作用。IBCRQ 系统包括用户注册、个人查询申请、单位查询申请、集中查询申请、异议复核申请、排号管理、法律知识问答、数据交换等业务模块，实际使用时，用户可根据实际情况的需要选择是否需要用户注册、排号管理、法律知识问答等模块自由组合，限于篇幅，在此我们不再详细介绍各个模块的功能。

微服务的目的是充分地分解应用程序以促进敏捷开发和部署。在 IBCRQ 系统项目的管理和开发中，我们按功能需求将系统划分为用户中心、查询申请、数据交换、预约排号、法律知识问答 5 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。

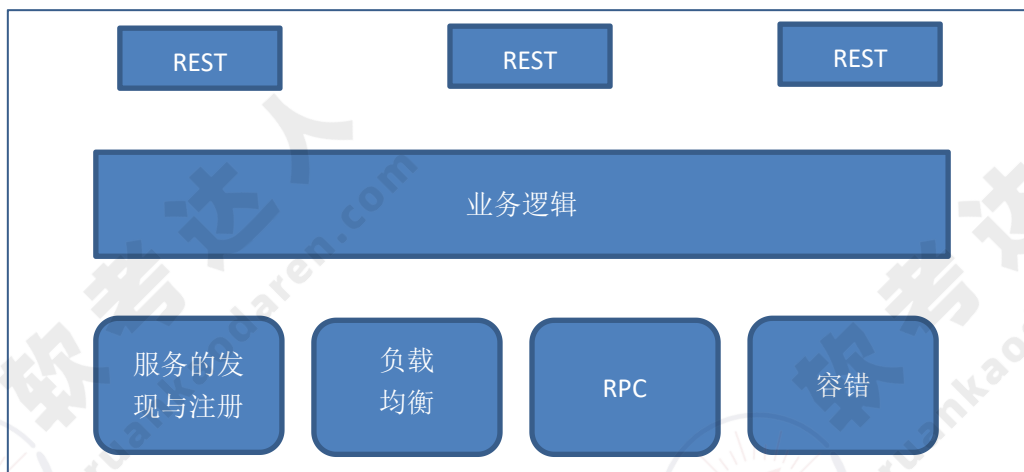
## 微服务架构的特点

传统的单块软件架构在构建部署和扩展伸缩方面有很大的局限性，传统的单块架构一般分为客户端用户界面、数据库、服务端应用程序三部分。系统中任何程序的改变都需要整个应用重新构建和部署新版本。另外传统的单块软件架构在进行水平扩展时也只能整个系统扩展，而不能针对某一个功能模块进行扩展。而微服务架构可以完美的解决统一风格架构所遇到的种种问题。微服务架构将系统以组件化的方式分解为多个服务，服务之间相对独立，单一功能的改变只需要重新构建部署相应的服务即可。与单块架构相比，微服务架构有如下的特点：

- 1) **通过服务实现应用的组件化 (Componentization Via Services)**，在应用架构设计中，通过将整体应用切分成可独立部署及升级的微服务方式，进行组件化设计。
- 2) **围绕业务能力组织服务 (Organized Around Business Capabilities)**，微服务架构采取以业务能力为出发点组织服务的策略，因此微服务团队的组织结构必须是跨功能的、强搭配的 DevOps 开发运维一体化团队，通常这些团队不会太大。
- 3) **基础设施自动化 (Infrastructure Automation)**，云化及自动化部署等技术极大地降低了微服务构建、部署和运维的难度，通过应用持续集成和持续交付等方法有助于达到加速推出市场的目的。
- 4) **故障处理设计 (Design For Failure)**，微服务架构所带来的一个后果是必须考虑每个服务的失败容错机制。因此微服务非常重视建立架构及业务相关指标的实时监控和日志机制。
- 5) **演进式的设计 (Evolutionary Design)**，微服务应用更注重快速更新，因此系统的计会随时间不断变化及演进。

## 微服务架构应用

在 IBCRQ 系统中，采用通讯层、业务逻辑层、基础服务层组成系统的微服务架构。如下图所示：



IBCRQ 系统微服务框架

在微服务架构下，我们选择 RESTful 的进行通讯。每个微服务都统一对外提供 REST 服务。无论前端调用后端服务还是后端之间的服务调用都统一走 RESTful，

这样就统一了协议栈。微服务架构可以支持各种异构系统服务间的交互。

服务的注册与发现, 服务之间需要创建一种服务发现机制, 用于帮助服务之间互相感知彼此的存在。服务启动时会将自身的服务信息注册到注册中心, 并订阅自己需要消费的服务。

负载均衡是服务高可用的保证手段, 为了保证高可用, 每一个微服务都需要部署多个服务实例来提供服务。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。

目前主流的 RPC 框架包括 dubbo, thrift, grpc 等, 我们采用 dubbo 为团队提供了一整套序列化, 反序列化, 网络框架, 连接池, 线程池超时处理等业务处理之外的能力。

服务容错采用快速失败, 失效切换的策略。如果连续失败多次则直接熔断, 不再发起调用。这样可以避免一个服务异常拖垮所有依赖于他的服务。

## 遇到的问题及解决方案

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; 二代部分码重复, 某些底层功能需要被多个服务所用; 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。因此, 首先服务划分应尽量合理, 不要划分太细太多, 其次采用公共模块的方式提供底层服务, 再次微服务可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。

通过项目实践证明, 实施微服务架构收益会大于成本。在拥抱微服务之前, 也需要认清它所带来的挑战。需要避免为了“微服务”而“微服务”。对传统企业而言, 开始时可以考虑引入部分合适的微服务架构原则, 对已有系统进行改造或新建微服务应用, 逐步探索及积累微服务架构经验, 而非全盘实施微服务架构。

张建刚 2017 年 10 月 03 日