[文章编号] 1003－4684(2026)02-0132-07

基于微服务的食品安全检测数据管理系统

杨孟范[1]，徐进霞[2]，刘凌云[1]，徐　博[1]，王　磊[1]
(1 湖北工业大学理学院，湖北 武汉 430068；2 襄阳湖北工业大学产业研究院，湖北 襄阳 441100)

[摘　要] 随着食品安全快速检测技术的发展，食品安全检测数据规模快速增长，传统的人工数据管理方式已不能满足现阶段数据快速汇总并形成有效反馈的数据管理需求。针对这一现状，使用微服务技术建立了一套高并发、高可用、可扩展的分布式检测数据管理系统，为用户提供一体化的检测数据收集、管理、汇总和实时数据反馈服务。同时，该系统采用了数据可视化技术建立了交互式的前端应用，辅助用户进行数据分析并提升用户体验。

[关键词] 微服务；分布式系统；数据管理系统；交互式；数据可视化
[中图分类号] TP274　　[文献标识码] A

作为社会关注的热点，食品安全问题关系着人们的健康和社会的稳定。近年来，随着食品检测技术的快速发展和食品安全检测的普遍开展产生了大量检测数据，食品安全检测数据的高效利用对防范重大食品安全风险、建立食品安全一体化监管机制至关重要[1]。目前针对食品中农药残留和重金属污染物检测所设计的基于分光光度法、阳极溶出伏安法的快速检测技术已十分成熟，且市场的快速检测终端应用已具备一定规模[2-3]。然而，相应的数据收集、管理、可视化等配套的数字化基础设施建设较为落后。以湖北省为例，大多数地区仍使用人工收集、上报、汇总、公示的数据管理模式。该模式下，检测信息的反馈周期长、数据分析难度大、数据可视化形式单一，导致消费者和相关从业者难以准确、及时、直观地获取到食品安全检测信息。为解决上述问题，本文基于 SpringCloud 框架建立了一套基于微服务的食品安全检测数据管理系统，提供从数据收集到数据可视化的一体化服务。为武汉、咸宁、鄂州等地区的消费者和相关从业者提供可靠、及时的食品安全信息获取渠道和便捷的检测数据管理服务。

1　微服务技术概述

1.1　微服务技术原理

微服务概念由 MartinFowler 在 2014 年提出并逐渐在软件开发领域兴起[4]，它是对在技术上和组织上扩大大型单体应用程序所面临的诸多挑战的直接回应。微服务架构通常具有以下特征：应用逻辑分解为具有明确的职责范围定义的细粒度组件并通过组件之间的协调解决问题；每个组件物理上完全独立；组件之间采用轻量级的通信协议；服务具有底层无关性，可以使用多种程序语言和技术实现；服务小型化、独立化、分布式化[4]。微服务技术允许将一个大型应用分解为具有严格职责定义的便于管理的组件。微服务通过将大规模复杂代码分解为小型精确定义的功能部分，解决了大型代码库中传统的复杂性问题[5]。小而独立的微服务应用程序与分布式技术契合，通过与分布式技术结合可以实现高可用、可伸缩的服务端应用程序。

1.2　SpringCloud 框架

SpringCloud 是 Spring 开发体系的微服务开源框架，提供了快速构建分布式系统中一些常见模式的工具[5]。SpringCloud 使用 SpringBoot 集成了配置管理、服务发现、断路器、智能路由、微代理、控制总线、一次性令牌、全局锁、领导选举、分布式会话、集群状态等工具和底层开发框架，其框架图如图 1 所示。开发人员可以快速建立基于这些框架的服务和应用程序[5]。基于上述优点，本文使用 SpringCloud 框架实现微服务系统的搭建，并选用框架中较成熟的分布式组件作为微服务系统的分布式基础设施的具体实现。

2　功能需求分析

2.1　数据收集微服务需求分析

系统的食品安全检测数据来源为一体化快检仪

[收稿日期] 2023－12－13
[基金项目] 湖北省重点研发计划项目(2022BAA022)
[第一作者] 杨孟范(1997－)，男，河南南阳人，湖北工业大学硕士研究生，研究方向为电子信息。

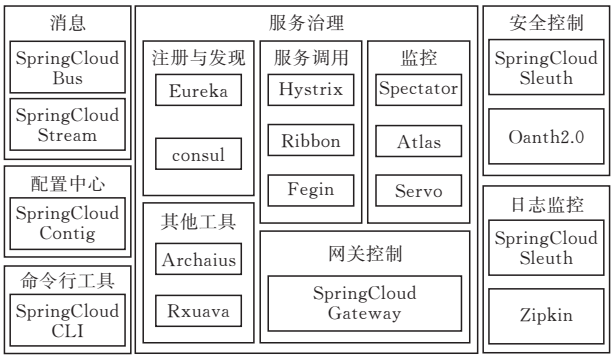

图 1　SpringCloud 框架

(下称数据收集终端)。检测员使用终端登录平台账号后可以开启将存储在终端本地的检测数据批量上传至平台的上传活动。针对上述需求，本系统建立了处理终端身份验证和上传请求的服务模型。服务模型中由管理员协助注册上传账号，再由检测员登录该账号并执行数据上传，其用例如图 2 所示。

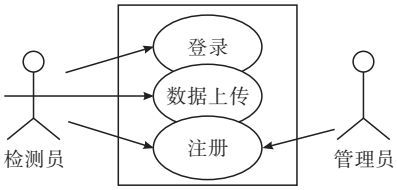

图 2　数据收集服务用例

2.2　数据管理微服务需求分析

在食品安全检测活动中，检测数据的拥有者(即经营实体、检测所等)需要对检测数据进行管理，食品安全监管方需要对检测数据进行汇总分析，消费者需要实时获取食品安全检测信息。针对三类用户的不同需求，系统建立了数据管理服务模型，用例图如图 3 所示。其中，经营实体用户可通过系统追踪其上传数据的处理流程并在流程初始阶段进行数据复查，监管者用户可以对数据进行管理、统计分析和查询，消费者用户具备有限的统计分析和数据查询权限。

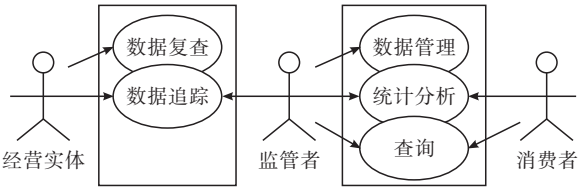

图 3　数据管理用例

2.3　数据推送微服务需求分析

食品安全监管部门和消费者对食品安全信息的实时性有更高的要求，因此需要系统收集到的实时食品安全检测信息快速、准确地反馈给相关用户。系统针对检测数据的上传和检测数据管理中产生的数据变更，建立了一套快速响应机制，其时序模型图如图 4 所示。系统基于观察者设计模式通过数据上传服务、变更检测服务、订阅服务以及客户端之间的异步通信，将系统中的数据变更信号发送给变更活动的订阅者。

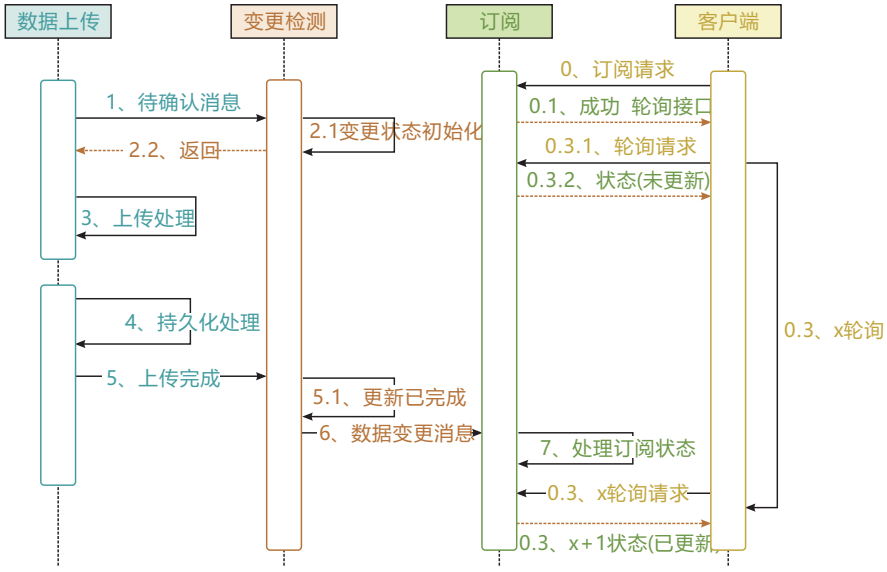

图 4　数据推送服务时序

3　系统总体设计

3.1　功能模块设计

基于微服务设计的单一职责、高内聚、低耦合、可扩展原则[2]和食品安全检测的实际业务需求，将系统服务端按功能模块划分为上传服务、数据订阅服务、交互服务和权限验证服务五种微服务，功能模块结构图如图 5 所示。

3.2　基础架构设计

平台采用了 MVC(Model View Controller)架构[6]搭建基于 web 的服务系统。系统整体架构可按传统分层模式[7]分为应用层、网关层、接口层、服

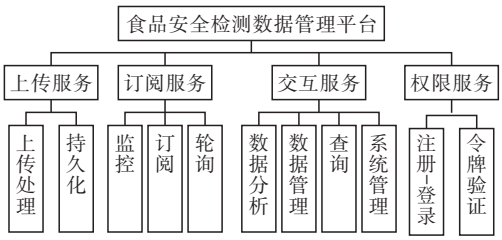

图 5 系统功能模块

务层、存储层，系统架构图如图 6 所示。

1)应用层 是系统为用户提供的交互界面。包括数据可视化前端、数据管理界面系统和数据采集终端。其中，数据采集终端为食品安全检测活动的执行载体，通过 HTTP 通讯协议向数据管理界面系统服务端发送食品安全检测数据，数据管理者和食品安全监管者通过数据管理界面系统对上传的检测数据进行管理，并借助数据可视化前端将食品安全检测活动中获取的食品安全信息反馈给消费者。

2)网关层 为对外提供统一路径的服务接口同时确保系统的安全性，系统在门户服务器上搭建了系统网关。用户向系统发送的服务请求将经由网关(Gateway)转发给微服务，避免外部 IP 直接访问系统核心业务服务器[8]，并支持根据请求源的 IP 地址进行请求过滤和传递令牌。由于后台服务以分布式微服务集群实现，网关通过服务发现组件(Discover Service)获得系统内微服务实例的路由信息，并交由服务发现组件(Ribbon)实现负载均衡[9]。

3)接口层 该层是用户访问和系统内部调用的 RESTful[10] 接口的公共抽象。按访问形式可分为内部访问接口和外部访问接口，按功能类别可分为上传处理接口、订阅接口、数据统计接口和数据管理接口。

4)服务层 服务层为接口层提供了具体功能的实现，分为上传服务、订阅服务和监控服务、数据交互服务。

食品安全检测活动通常由监管部门以临时抽查的形式开展，具有一定的突发性，因此需要数据上传服务具备良好的并发处理能力。针对上述需求，上传服务采用上传处理过程和持久化过程分离的设计，以提高上传接口的响应速度并减少单个请求对服务器资源的占用。上传处理微服务负责接收数据采集终端上传的数据，校验成功后缓存。数据持久化微服务以异步任务[11]的方式将缓存中的数据写入主数据库，同时生成一个数据上传完成的事件通知发送给数据变化监控微服务。

为使食品安全检测活动形成完整的信息反馈，发挥食品安全检测活动对市场的监督作用，需要将系统接收到的食品安全检测数据快速推送至数据可

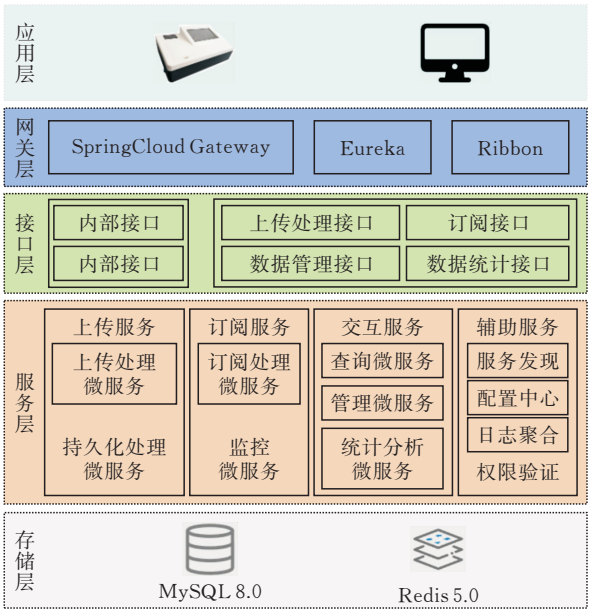

图 6 系统架构

视化前端，为实现上述目的系统建立了以订阅服务和数据监控服务为主体的订阅-推送子系统。订阅服务负责处理由可视化前端应用发起的实时数据频道订阅请求，并提供订阅对象的状态轮询接口和更新内容获取接口。监控微服务负责监控服务端的数据变化，即上传(新增)、修改、删除活动，并通过消息中间件[12]通知订阅服务修改相应的订阅状态。

为解决监管者和数据拥有者对食品安全检测数据的管理需求，服务端建立了交互式的数据管理服务，并根据用户角色进行了细粒度的管理权限划分。同时，以功能性为导向交互服务按读、写类型进行了子服务划分，并针对数据分析服务实现了查询优化，以提升数据分析接口的相应性能。

同时，服务层也包含了 SpringCloud 框架的分布式工具服务实现，包括基于 Eureka 的服务发现、Ribbon 负载均衡器，配置中心服务，日志聚合服务[13,14]和基于 JWT(JSON Web Token)和 Spring Security 技术的权限验证服务[15,16]。

5)存储层 存储层提供了基础的数据持久化能力。系统使用 MySQL 作为主数据库存储主要业务数据，同时使用 Redis 数据库作为临时数据存储，实现为缓存、异步通信等服务层逻辑提供支持。

3.3 分布式设计

系统基于 SpringCloud 框架实现后台微服务的分布式架构，以提高系统可用性，增强服务并发能力并为系统扩展提供支持[17]。分布式架构由一个网关节点和若干实例节点组成。网关节点负责接收、过滤、转发或代理客户端对后台系统的服务调用请求，实例节点运行微服务实例。系统使用 Eureka 工具实现自发式的服务发现，并使用 Ribbon 对服务

调用进行负载均衡。为确保系统的可靠运行，各功能模块为微服务设置了备灾节点，备灾节点的服务实例将在主实例集群宕机或性能严重下降时自动启用。针对并发性需求较高的微服务，除使用多个实例进行负载均衡外，网关节点使用基于断路器策略的弹性模式以避免单个高延时节点影响服务整体的响应性能[18]。此外，微服务实例以 Docker 镜像形式使用自动化构建引擎 TravisCI 和源代码控制库 GitHub 进行自动化部署并使用配置中心自动化构建 Java 应用，为系统性能扩展和灾后快速恢复提供支持。

3.4　数据库设计

以检测数据实体为核心，通过与区域、检测项目、经营实体等实体的对应关系，建立了反映食品安全检测活动的数据模型。数据库中主要数据实体之间的联系可以用 E-R 图描述(图 7)。依据关系图在数据库中建立的数据表包括：用户表、经营实体表、检测数据表、样品类别表、检测项目表。其中作为主数据表的检测数据表的存储字段如表 1 所示。

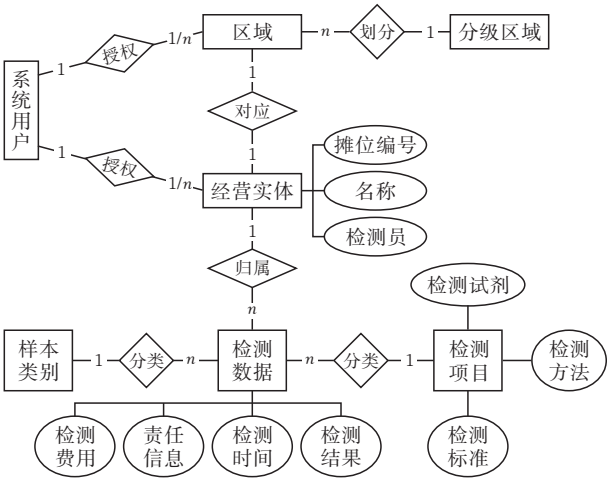

图 7　检测数据 E-R 图

表 1　检测数据

字段	备注	Key
Id	数据唯一码	主键
Code	检测编号	
CheckTime	检测时间	
UploadTime	上传时间	
BelongTo	归属实体	外键
SampleType	样品类别	外键
CheckItem	检测项目	外键
Result	检测结果	
Fee	检测费用	

此外，基于检测编号和检测数据之间存在一对一映射关系的特点，系统使用 Redis 在关系型主数据库的基础上实现了缓存层。检测编号-检测数据的缓存结构有助于减少上传接口的响应时间。同时，以查询参数为键值的缓存结构在提高接口响应性的同时减少了主数据库执行聚类查询、外连接等操作的次数[19]。

3.5　数据预处理

3.5.1　数据清理　由于检测环境污染、检测员错误操作、人为造假等多种因素，系统中的检测数据集中存在噪声数据。因此，在进行数据分析前系统将对检测数据集执行数据清理过程。基层检测活动很多以生产商自检的形式展开，不可避免地存在人为上传假数据的现象，造成数据集中存在大量如农残检测值为 0 的无效数据，数据清理过程需首先删除此类无效数据。另外，通过对检测值字段的聚类分析，剔除检测数据集中的离群点，以消除检测过程中的偶然性因素对数据集产生的影响[20]，如图 8 所示。

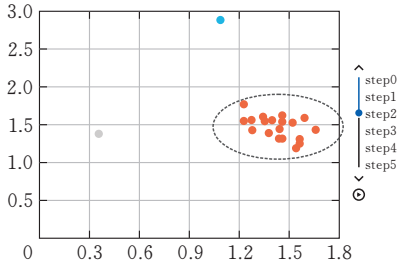

图 8　离群点分析

3.5.2　数据归约　为降低检测数据分析的复杂性，提高分析效率，对系统中的检测数据集进行归约处理。首先，选择检测数据的最小属性集(检测时间、检测值、检测项目、样品名称、产地)作为数据分析的数据元。其次，针对不同分析进行数据变换和数据离散化，如根据检测项目属性的固有分类进行数据汇总，根据检测时间或产地进行时域、地域的数据聚集和根据检测值的阈值区分合格、不合格、存在风险数据进行离散化。

3.6　数据可视化设计

系统针对各类用户的数据可视化需求设计并实现了基于数据分析的可视化前端。针对检测数据监管方的需求，在对检测数据进行地域属性聚类分析的基础上，结合在线地图 API 实现了可视化应用如图 9 所示。

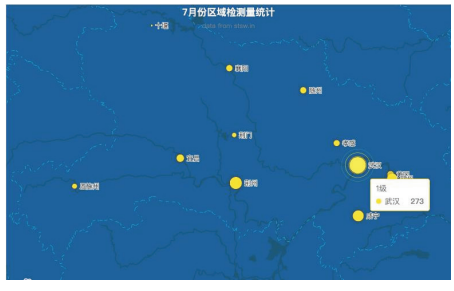

图 9　湖北地区检测量统计

针对商户和检测机构用户需求，在对检测数据

的检测值进行离散化分析的基础上，实现合格数统计可视化应用如图 10 所示。

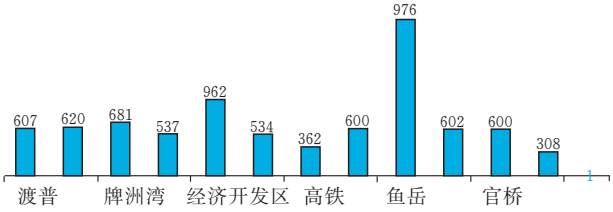

图 10 检测机构和商户检测合格数统计

基于检测数据量的时变性，实现的可视化应用示例如图 11 所示。

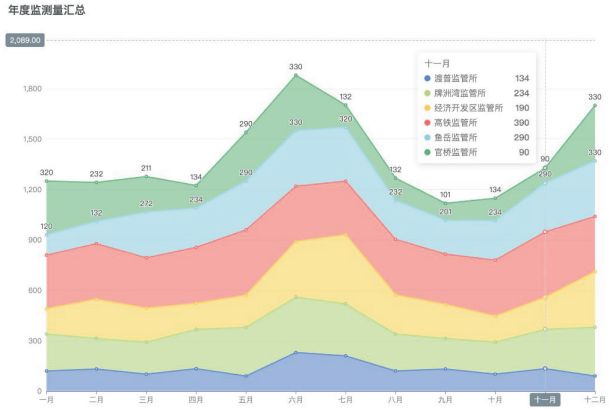

图 11 检测量的时间变化曲线

4 性能测试与系统评价

4.1 测试环境

在测试环节，对微服务进行集群部署。其中上传处理微服务实例 3 个，变更监控微服务实例 2 个，订阅微服务实例 4 个，数据管理微服务实例 1 个，数据查询微服务实例 2 个。部署中采用容器化的部署方式将微服务实例打包成 Docker 镜像部署在阿里云 ECI 的 3 个容器组上。ECI 容器组主机配置为 2 核 X86 指令架构 CPU 和 4GB 内存，MySQL 主数据库的物理环境配置为 2 核 CPU，4GB 内存并挂载 100GSSD 磁盘。

4.2 功能测试

4.2.1 权限验证微服务功能测试 权限验证微服务的测试用例如表 2 所示。

表 2 登录测试用例

测试用例编码	TEST_01
测试名称	登录测试
测试目的	验证各客户端的登录流程的执行结果和权限验证接口的正确性
测试流程	1)登录获取令牌；2)验证令牌，并获取用户信息
预期结果	输入正确口令可以获取到令牌，使用令牌访问验证接口可获取正确的用户信息
实际结果	与预期相符

该测试主要验证登录认证接口和权限校验接口的返回结果以及登录流程的正确性。

4.2.2 上传处理微服务功能测试 上传处理微服务的上传接口测试用例如表 3 所示。测试主要对接口响应和缓存状态进行了验证。

表 3 上传接口测试用例

测试用例编码	TEST_02
测试名称	上传接口测试
测试目的	验证上传接口的返回结果和后台缓存逻辑的正确性
测试流程	1)上传数据)；2 查看缓存；3)获取响应
预期结果	对校验无误的数据，服务端将建立缓存并返回上传成功的响应码，对校验有误的数据，返回上传失败对应的响应码和校验失败信息，对无效令牌请求直接拒绝访问
实际结果	符合预期结果

4.2.3 数据订阅微服务功能测试 对数据订阅微服务的功能测试用例如表 4 所示。

表 4 订阅系统测试用例

测试用例编码	TEST_03
测试名称	订阅相关接口测试
测试目的	验证轮询接口的返回结果和异步消息队列通信系统的正确性
测试流程	1)发起订阅请求，并检测返回结果；2)查看消息队列中的订阅状态和缓存中的状态变量；3)调用轮询接口，检查返回结果；4)模拟数据变更或上传完成事件；5)再次调用轮询接口，检查返回结果
预期结果	服务接受订阅后将在消息队列中新增订阅并初始化一个状态变量，第一次调用轮询接口将返回 false，第二次调用将返回 true
实际结果	与预期结果相符

4.3 性能测试

针对各微服务的需求和设计特点对其主要功能接口做以下性能指标的测试。上传处理微服务的数据上传处理接口在大量并发请求下响应时间是反映该服务性能的主要参考指标，接口响应时间的并发测试结果如表 5 所示。

表 5 上传接口响应时间测试结果 ms

并发量	接口平均响应时间	接口最大响应时间
100	23	29
1000	32	47
5000	37	92
10000	304	673

结果表明，并发量不超过 5000 时上传接口性能稳定，基本满足系统日常并发量和响应时间要求。并发量到达 10000 时响应性出现断崖式下滑，其原

因可能是 Redis 缓存结构的锁竞争严重导致平均响应时间大幅增加[21,22]。

订阅服务提供的轮询接口对数据变动做出响应的时间间隔和准确率,决定了用户获取信息的实时性,故针对这两个性能指标做性能测试。测试结果中,上传活动开始到轮询接口获得响应的时间间隔约为 3 min 14 s,减去上传服务等待用户完成全部数据上传的时间 180 s 和轮询造成的平均时延后,得到订阅系统的处理时间在 7 s 左右,与交互微服务产生的数据变更测试结果一致。对上传活动的响应正确率为 100%,对修改和删除的响应正确率为 81.3%。后者正确率较低的原因在于系统中数据修改和删除操作不具备原子性,导致判断数据变化时出现幻读和脏读。实际中的数据修改和删除频度远低于测试,因此不会产生严重问题。

5　结论

本文搭建了一套基于微服务的食品安全检测数据管理系统,为食品经营者、食品安全监管方和食品消费者提供了从数据采集到数据展示的一体化数据管理服务。系统使用微服务技术对服务端进行解耦,并采用分布式的部署方式增强了系统的可靠性和扩展性,同时通过多线程、异步通信、负载均衡等技术的应用提升了系统的实时响应能力和并发处理性能。并基于 Element UI 和 Vue 实现了功能丰富的网页端应用。系统在武汉、咸宁、鄂州等地区投入使用以来,持续为农副产品消费者提供具有时效性的安全检测信息,解决了农产品经营、管理者对检测数据的分析管理需求,获得了用户的一致好评。

[参 考 文 献]

[1] 行程. 基于海量数据分析的食品药品安全检测系统[D].西安:西安工业大学,2017.

[2] 赵昆山.快速检测技术在食品安全监管中的应用分析[J].中国食品工业,2022,361(23):61-62.

[3] 党锐,徐琳.食品安全检测系统设计研究:评《我国食品安全风险防控研究》[J].食品工业,2020,41(06):359.

[4] 方志朋. 深入理解 Spring Cloud 与微服务构建[M].北京:人民邮电出版社, 2019.

[5] GUTIERREZ F. Spring cloud data flow: native cloud orchestration services for microservice applications on modern runtimes [M]. Berkeley: Apress, 2021: 156-201.

[6] 杜成龙.基于 MVC 模式的三层架构研究[J].软件,2022,43(06):100-102.

[7] 钟志东,孟清.网站开发的五层构架模块化设计[J].计算机时代,2013(08):19-21.

[8] FEIY, WEI R, CHANG L, et al.Architecture, integrated gateway design, and performance evaluation for high concurrency access of power internet of things[J]. Mobile Information Systems,2022,2022: 9164338.

[9] 经成,谢军.基于微服务架构的负载均衡优化算法及实现[J].南昌大学学报(理科版),2022,46(04):399-404.

[10] 陈锐,何华军,王辰.基于 RESTful 接口的基础数据对接设计[J].电脑编程技巧与维护,2022(09):122-124.

[11] 魏瑾.Java 多线程技术在网络通信系统中的应用[J].山西电子技术,2022(01):66-68.

[12] 蒲云鹏.分布式消息中间件技术应用研究[J].信息技术与信息化,2022(07):136-139.

[13] 张之宣. 云计算环境下实时日志分析系统的设计与实现[D].杭州:浙江大学,2016.

[14] XINMING L, HAITAO W, JING Z. Research on a method of log aggregation[C]// 3rd International Conference on Traffic Engineering and Transportation System: ICTETS 2019 : Opportunities and Challenges for Transportation Development : Jiaozuo, China, 20-22 September 2019, Part 1 of 3.2020.

[15] 范鑫怡.Spring 架构下的安全机制研究与应用[D].南京:南京理工大学,2020.

[16] 郝恬,左黎明,陈艺琳.基于口令签名和 OAuth2.0 协议的强身份认证方案[J/OL].计算机系统应用:1-7[2023-03-14]. https: // doi. org/10. 15888/j. cnki. csa. 009011.

[17] LATHA V L P, REDDY N, BABU A S.Optimizing scalability and availability of cloud based software services using modified scale rate limiting algorithm [J].Theor. Comput. Sci. 2023, 943:230-240.

[18] JEYALAKSSHMI S, SMILES J A, AKILA D, et al. Energy-efficient load balancing technique to optimize average response time and data center processing time in cloud computing environment[J]. Journal of Physics: Conference Series, 2021, 1963(01):012145.

[19] CARLSON J L, SANFILIPPO S. Redis in action[M]. Shelter Island: Manning, 2013: 79-113.

[20] SONG X, WU M, JERMAINE M C, et al.Conditional anomaly detection[J].IEEE Trans. Knowl. Data Eng., 2007,19(05):631-645.

[21] 张蒙蒙,曹成茂.大数据时代 Redis 缓存的性能分析与优化[J].巢湖学院学报,2022,24(03):80-87,96.

[22] 王欢,李民,邓秀辉,等.基于 Redis 缓存数据库和 Nginx 负载均衡技术的购物网站性能优化[J].软件导刊,2022,21(08):114-119.

Food Safety Detection Data Management System Based on Micro-service Architecture

YANG Mengfan[1], XU Jinxia [2], LIU Lingyun[1], XU Bo[1], WANG Lei[1]
(1 *School of Science, Hubei Univ. of Tech., Wuhan* 430068, *China*;
2 *XIANGYANG Industrial Research Institute, Hubei Univ. of Tech., Xiangyang* 441100, *China*)

Abstract: With the development of food safety rapid detection technology, the traditional manual data management cannot meet the current data management needs of rapid data summary and effective feedback. However, the management system of food safety detection data is inefficient. In view of this situation, a scalable distributed detection data management system is established based on micro-service technology with high concurrency, high usability and scalability. The System provides an integrated data service to system users for data collection, management and analysis. Also, to improve the user experience and support data analysis activities, system provides an interactive web application with data visualization capability.

Keywords: micro-service; distributed system; data management system; interactive; data visualization

[责任编校: 闫 品]

(上接第 111 页)

Establishment of a *STING* Deficient HCT116 Cell Line and Its Impact on the Replication of Oncolytic HSV2

CHEN Mingxin, ZHONG Li, WANG Runyang, HU Han, WANG Yang, LIU Binlei
(*School of Food and Biological Engineering, Hubei Univ. of Tech., Wuhan* 430068, *China*)

Abstract: To investigate the regulatory role of the *STING* gene in the replication of oncolytic herpes simplex virus type 2 (oHSV2) in colon cancer cells, in this study, the CRISPR/Cas9 (Clustered Regularly Interspaced Short Palindromic Repeats/CRISPR-associated protein 9) gene-editing technique was employed. Two specific single-guide RNAs (sgRNAs) were designed targeting the exons of the human *STING* gene, constructed into the pX459 vector, and co-transfected into HCT116 colon cancer cells. Monoclonal cell lines were obtained via puromycin selection and limiting dilution. Successful establishment of the HCT116$^{STING-/-}$ knockout cell line was verified by Sanger sequencing and Western-blot analysis. CCK-8 assays demonstrated that *STING* knockout did not affect the basal viability of HCT116 cells. Wild-type and *STING*-knockout cells were infected with oHSV2, respectively, and the oncolytic efficacy and viral replication levels were evaluated using CCK-8, RT-qPCR (real-time quantitative PCR), and $CCID_{50}$ (50% cell culture infectious dose) assays. The results revealed that *STING* knockout markedly increased the oHSV2 gene copy number and viral titer, and enhanced the susceptibility of HCT116 cells to oncolytic virus. In conclusion, this study successfully constructed a *STING* knockout HCT116 cell line and confirmed that *STING* suppresses oHSV2 replication in colon cancer cells, providing an experimental basis for screening host restriction factors of oncolytic viruses and optimizing oncolytic virotherapy strategies.

Keywords: *STING*; CRISPR/Cas9; HCT116 cells; oHSV2; viral replication

[责任编校: 张 众]