ISSN 1009-3044
Computer Knowledge and Technology 电脑知识与技术
Vol.17, No.23 August 2021

E-mail: kfyj@dnzs.net.cn
http://www.dnzs.net.cn
Tel:+86-551-65690963 65690964

基于SpringBoot微服务架构下前后端分离的MVVM模型浅析

巢晟盛

(常州纺织服装职业技术学院 信息服务中心,江苏 常州 213164)

摘要:随着互联网时代不断的发展,软件应用于各行各业,随之带来了单体应用无法满足某些场景的业务需求、维护升级、耦合性、部署效率及扩展等问题,微服务架构逐步取代;SpringBoot致力于快速方便地开发单个微服务,解决ssm大量手工配置的问题,高效地处理复杂的业务逻辑,稳定地支撑SpringCloud微服务协调治理框架;前后端分离实现了前后端架构的解耦,有利于提高开发效率,有利于降低软件设计的耦合度,有利于提高处理复杂业务的能力。MVVM模型是MVC的优化增强,实现视图和模型的分离,应用于程序的分层开发,适用于数据操作频繁的场景。

关键词:SpringBoot技术;微服务;前后分离;MVVM模型;数据驱动

中图分类号:TP319 文献标识码:A

文章编号:1009-3044(2021)23-0128-02

开放科学(资源服务)标识码(OSID):

DOI:10.14004/j.cnki.ckt.2021.2412

Overview on MVVM Model Baesd on SpringBoot MicroServices Architecture with Front-end and Rear -end Separation Technology

CHAO Sheng-sheng

(Information Service Center , Changzhou Vocational Institute of Textile and Garment, Changzhou 203164, China)

Abstract: With the development of Internet Age, softwares are adopted in all walks of life. Monolith can't satisfy certain scenarios demand, maintenance upgrade, coupling, deployment efficiency, extension and etc so that Microservices gradually replace it. SpringBoot is devoted to developing a single microservice quickly, solving the problem of many manual configuration issues in SSM, handling complex business logic effectively and supporting the SpringCloud microservices coordination governance framework Stably. The front-end and rear-end separation technology realizes the decoupling of front and rear end architecture, which is beneficial to improve the development efficiency, reduce the coupling degree of software design, and improve the ability to deal with complex business. MVVM model is the optimization and enhancement of MVC, separate view and model, applied to the layered development of the program, used for data operation frequent scene.

Key words: SpringBoot; microservices; front and rear separation; MVVM model; data driven

1 背景

在移动互联网应用高速发展的推动下,软件设计规模随着需求逐渐庞大,业务场景更为复杂,对软件系统的性能、吞吐率、稳定性、扩展等特性提出了更高的要求,在此背景之下,微服务[1]架构逐渐取代单体架构,迎来了新的技术迭代更新。其中,SpringCloud[2]是Pivotal推出的基于SpringBoot的一套关注全局的微服务协调治理框架,致力于合并管理单体微服务,而SpringBoot专注于方便地开发单个个体微服务。在前后端分离的开发模式下,实现了前端和后端的并行开发,互不影响,一方面提高了开发效率,另一方面代码质量更为规范、专业。

MVVC模型是马丁·福勒的PM(Presentation Model)设计模式的变体,由微软架构师Ken Cooper和Ted Peters开发,通过利用WPF(微软.NET图形系统)和Silverlight(WPF的互联网应用派生品)的特性来简化用户界面的事件驱动程序设计。由于MVVM是MVC的改版,因此在研究MVVM模式时,有必要结合MVC一并进行讨论和比较。

2 基于目前互联网应用开发架构、框架相关对比

2.1 微服务架构与单体架构

1)系统更改部署。单体应用是按单体应用程序单元来构建的,对系统进行更改之后,必须构建和部署服务器端应用程序的更新版本;微服务通过标准化的业务API实现功能,采用松耦合的设计原则,允许服务的所有者自由实现并更改基于API的数据处理或者组合服务系统,从而消费者不会受到服务内部实现变化的影响。

2)软件开发流程。单体应用庞大复杂,整个应用理解难度

收稿日期:2021-04-16
基金项目:常纺院一页纸项目管理系统的开发与应用(项目编号:CFK201906)
作者简介:巢晟盛(1991-),男,江苏常州人,助理工程师,硕士,主要研究方向为信息服务、软件工程。

大,模块重用困难、扩展麻烦及重复部署更新版本缓慢;微服务将传统模式下的单体应用拆成独立的服务,从而实现单独开发、单独部署、单独维护。

3)市场价值。单体应用库使用耗时,适应性差,维护成本高,从而增加了技术债务;微服务减少了技术债务,压缩了开发时间,从而降低了成本提高利润。

4)特性。微服务具有复杂度可控、独立部署、容错性能好以及扩展性高等特性。

2.2 SpringBoot与SSM

1)SpringBoot与Spring[3]的区别。SpringBoot可以建立独立的Spring应用程序,内置服务端容器,不需要部署工作,简化了烦琐的xml文件配置,可自动配置Spring,简化Maven的配置。

2)SpringBoot与SpringMVC[4]的区别。SpringMVC是基于Spring的一个MVC框架,SpirngBoot是基于Spring的条件注册的一套快速开发整合包。

3)SpringBoot集成MyBatis[5]。MyBatis是一个简化和实现了Java数据持久化层(persistence layer)的开源框架,它抽象了大量的JDBC冗余代码,并提供了一个简单易用的API和数据库交互。在SpingBoot中,只需要POM文件设置Mybatis的依赖,即可实现数据库的持久化。

2.3 前后端分离与非前后端分离

1)开发效率。原模式软件开发过程每个环节都依赖进行,串行模式延长开发周期长,而前后端分离[6]是并行模式,每个环节独立进行。

2)耦合度。在原模式的MVC模式下,开发不规范严谨,控制层C重写了视图层V的业务代码,导致代码耦合度高,同时难以维护。然而前后端分离针对不同的端,实现工程化的设计开发。

3)复杂的业务场景。针对页面交互效果、数据处理,传统的模式很难支持该种业务场景。在前后端分离的模式下,前端负责对应的交互业务,后端负责数据的处理。

3 MVVC模型

3.1 模型原理

MVVC[7]模型简称是Mode-View-View Model。View与Model层之间没有直接关联,通过ViewModel进行双向交互,把Model的数据同步到View显示,View的更新同步Model,开发者只需要关注业务逻辑,实现真正的前后分离。

View层为视图展示层,由View和Controller剥离的部分逻辑组成,主要功能是显示前端页面、前端输入的校验等。

Model层为数据模型层,直接面向数据层,主要功能是处理业务逻辑和接受服务端返回的数据模型。

ViewModel层为视图模型层,View的每个UI元素设置对应的属性,实现Controller剥离的UI逻辑,实现View与Model双向绑定。

3.2 模型优势

在一般的MVC[8]应用中,随着业务越来越复杂,视图交互越发杂乱,导致Controller越发臃肿,其中包含了大量的表示逻辑。因此,MVVC模型取代了MVC模型。MVVC具有的优势:1)低耦合,View独立与Model变化和修改;2)增强应用模块的及时测试性能;3)可重用ViewModel中的视图逻辑、例数据格式的非空;(4)独立开发,前后端并行开发。

4 SpingBoot前后端的MVVC架构

4.1 前端实现

前端实现View层、ViewModel层。ViewModel层将视图和业务逻辑分开,获取Model的数据并且处理成为View层预期的内容形式,封装的数据模型包含视图的状态和行为,实现View层与Model层的解耦。前端一般采取如下的方式,着重介绍下前端MVVM框架:

1)非前端框架:使用h5开发页面,nginx作转发请求代理,解决跨域问题,ajax发送请求给后端。静态页面实现View层,js可视化业务逻辑实现ViewModel层,监听事件手动操作DOM,使用DomainObject实现数据的格式转换。

2)前端框架:Vue[9]、React[10]和Angular。三者都是响应式的前端框架,采用Node和webpack的生产环境、运行的模式,可高效集成功能组件,自带ViewModel层,使用双向绑定的机制,实现视图渲染,发送请求到后端。

4.2 后端实现

后端实现Model层,SpringBoot实现后台的接口,返回Json格式的数据给前端。主要的工作是实现数据的处理,对数据的增删查改;业务的逻辑处理和算法优化;环境网络层的接口调用;服务器硬件资源的调度等。

后端实现数据持久化,从数据库获取数据,返回封装的对象,根据场景需求,实现逻辑的处理,进而二次封装,返回数据到前端;对接其他平台接口,后端直接发送请求到服务器,获取数据信息。

4.3 整体流转

以数据驱动为核心,当数据发生变化时,主动推送数据给界面展示最新的数据;当界面操作时,直接请求处理数据,弱化了控件和控件事件。如图1所示,在SpringBoot下的MVVM模型中,DO层位于ViewModel中,DO层是ViewModel层View层的对应关系,负责前端的数据双向绑定,便于对数据做增删查改。当DO对象属性发生变化时候,通知View更新,当View上表单值发生变化时,通知DO更新,并异步通知队列同步到数据源。前端发送请求到后端采用原生或者封装好的ajax请求,后端采用DTO用来实现客户端和服务器端之间高效、安全的数据传输,适用于需要传输大量信息的数据场景。最后,后端连接数据库,实现业务逻辑,返回数据前端显示。

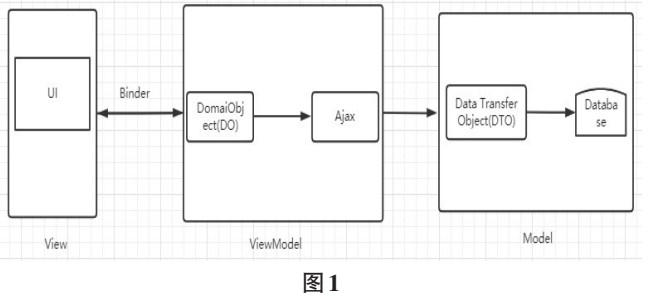

图1

5 结束语

在SpringBoot框架下的微服务架构下,基于MVVM模型的应用简化了控制层的逻辑,剥离了表示逻辑,前后端的开发模

(下转第141页)

叫作是倾侧水银开关,以一个连接著两个电极的玲珑圆形容器可以储蓄一小滴滴的水银,惰性化的气体或者是真空就会注入这个容器中。一旦发生歪斜,两个电极都会被水银接触,相当于闭合开关,发出信号。这样做可以省去数据分析的步骤,但是水银对人体及环境均有毒害,万一发生泄漏不利环保。所以,设计中使用了加速度传感器来测量角度。近年来,随着技术的不断发展,传感器的体积不断减小,成本不断降低,这也增加了方案的可行性。

从我们分析这个单片核心系统模块的基本设计以及应用情况可以不难看出,该模块设计的一个中心控制单元其实就是一个单片机,因此这个核心系统设计是必须围绕这个单片机的基本设计要求出发的。硬件、软件主要部分构成了一个单片式微机系统。其中应用硬件主要包括应用单片机、输入/输出设备、和外围软件应用集成电路,软件主要指各种应用工作处理程序。

从设计的要求来分析该设计须包含如下结构:加速度传感器、报警模块、单片机等;设计中采用了2个单片机,一个与传感器相连,作为下位机,另一个一个与 CDMA 模块相连,作为上位机,上下位机之间利用串口通信。平时单片机工作于休眠模式,两单片机分步控制,传感器工作于量测模式,CDMA 模块关机。

3.2 软件介绍

省电是该设计的重要目标,采用器件的型号选择也是出于这个目的。Mega88 正常情况下在 32KHZ 频率下工作,功耗仅为 1.8V:15uA,掉电模式电流只有 0.1uA;CMA3000 在 40/10HZ 采样率下,正常工作电流也只有 11uA。该设计中使用的 CDMA 模块是最耗电的,待机时电流为 3mA,而通话时平均 230mA,最高可达 630mA,所以不需要发送信息的时候应当关闭 CDMA 模块。发送一次短信需要 10 秒左右,一天的平均电流为 50uA。由于井盖沉重,窃贼不可能在 1 分钟内就完成盗取工作,所以设计中,下位机每 1 分钟唤醒一次,持续 5 秒,接收传感器的信息,并作初步判断,是否大于 20 度,然后再决定是否需要唤醒上位机,平时处于休眠模式。CDMA 模块每隔 5 分钟开机一次,如无信息需要发送,再进入关机状态。这种设计方式平均工作电流为 100uA 左右,比起单片机一直处于工作状态节省了大约 20% 的功耗,如果没有盗窃情况发生,使用 3.7 伏、1000 毫安时的电池,可工作 416 天,长达 1 年之后才需要更换电池。

Mega88 提供了多种休眠模式:

SM2.0 的指令为 010 时,SLEEP 就表示可以直接让 MCU 进入自动掉电关机模式。进入电源掉电保护模式后,停止快速晃动的放电晶体会使外部电源中断、两线串联并行输入地址进行匹配、看门狗继续正常工作。只有当外部进行复位、看门狗复位再中断后,BOD 复位、两线串联并行地址开始匹配、外部输入电压水平出现中断 INT0 或者 INT1 现象,还有引脚电平中断变化才真正能够让引脚 MCU 进入解脱或关闭掉电模式。当系统进入异步休眠控制模式时,时钟自动停止,异步模块继续工作。当我们使用外部电压水平中断的操作方式将 MC 中断唤醒时,只能继续使外部有源电压水平保持一定的时间。从系统施加一个掉电唤醒延迟条件开始到真正掉电唤醒,这个过程是用于时钟重新开启并稳定下来。其中熔丝叫醒停止时间与电机熔丝位置图所定义的熔丝复位叫醒时间要求是一致的。

在现实应用中,我们不仅需要知道井盖是否被盗,还要知道这套电子报警系统是否仍在正常运作以及电池电量是否充足,于是,在程序中添加了一段每隔 24 小时报告系统情况的程序,主要作用是报告系统是否正常工作。

4 总结

目前,由于环境因素,在“智能井盖”的市场上还没有一种适合所有情况、大规模的远程管理解决方案,而这没有解决方案的主要原因是安全性要求和成本方面。随着联网技术的飞速发展,该技术主要由卫星定位,移动通信和云计算的结合。从而诞生了智慧城市概念,沿着这个发展方向,对传统的市政基础设施进行改造,使其智能化、物联网化,提升城市治理水平,推进智慧城市的落地建设。防盗警报器将使公共设施管理部门能够快速有效地获取有关井盖的各种状况,可以以适当及时的方式保证公民的生命和财产安全,通过可靠,合理的整合和数字化的控制使市政管理的合理化。

通过建设城市已经改远程监控管理系统,还可大大提升地下管线施工、维护、安全防范管理水平,提高管理效率。

参考文献:

[1] 白云,杨宇帆.多功能井盖及包含多功能井盖的监测系统:CN210507540U[P].2020-05-12.

[2] 袁野.基于RFID技术的智慧海陵智能井盖系统的设计与实现[D].扬州:扬州大学,2018.

【通联编辑:闻翔军】

(上接第129页)

式提升了开发效率,缩短开发周期,更多地满足业务场景的需求。

参考文献:

[1] 赵然,朱小勇.微服务架构评述[J].网络新媒体技术,2019,8(1):58-61,65.

[2] 张雷,王悦.基于SpringBoot微服务架构下的MVC模型研究[J].安徽电子信息职业技术学院学报,2018,17(4):1-9.

[3] 朱运乔.基于SpringBoot+SSM框架的Web应用系统搭建与实现[J].电脑编程技巧与维护,2019(10):23-25.

[4] 张嘉豪,赵亮,翁铭隆,等.基于SSM+SpringBoot技术实现服务器监控的研究[J].科学技术创新,2020(33):101-102.

[5] 吴小伟,于加娟,陆莹洁.基于SSM框架民营科技企业备案系统的研究[J].中国科技信息,2021(6):91-93.

[6] 吴昌政.基于前后端分离技术的web开发框架设计[D].南京:南京邮电大学,2020.

[7] 朱海萍,丁西,刘链.Web前端中基于MVVM框架的技术应用研究[J].科技资讯,2020,18(30):8-10.

[8] 游俊慧.MVC、MVP、MVVM三种架构模式的对比[J].办公自动化,2020,25(22):11-12,27.

[9] 王志文.Vue+Elementui+Echarts在项目管理平台中的应用[J].山西科技,2020,35(6):45-47.

[10] 王叶加.基于JavaScript的React一站式智能开发工具的设计与实现[D].北京:北京邮电大学,2020.

【通联编辑:谢媛媛】