一种手工半智能的 Web 统一开发平台

胡飞菊 张少平
（江西农业大学计算机与信息工程学院 江西 南昌 330045）

摘 要 对当前 Web 应用开发及技术的现状进行概括分析，发现 Web 应用开发存在着很多共同的特性，导致有很多重复性工作。提出构建一种手工半智能的基于微服务的 Web 统一开发平台。介绍微服务架构的概念及原理、Web 统一开发平台架构设计、底层框架。提出“事务-流程-表单”的开发模式，高效地管理并实现项目中业务流程的开发。设计一种可视化表单设计器，进行手工半智能开发页面表单，从而规范开发流程，提高开发效率，缩减项目交付周期。

关键词 Web 应用开发 统一开发平台 微服务 事务-流程-表单 规范开发流程

中图分类号 TP311 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2019.01.003

A MANUAL SEMI INTELLIGENT WEB UNIFIED DEVELOPMENT PLATFORM

Hu Feiju Zhang Shaoping
(*School of Computer and Information Engineering, Jiangxi Agricultural University, Nanchang 330045, Jiangxi, China*)

Abstract We summarized and analyzed the current status of Web application development and technology. It was found that there are many common features in Web application and development, which led to a lot of repetitive work. We built a manual semi intelligent Web unified development platform based on microservice. We introduced the concept and principle of micro-service architecture, the design of Web unified development platform architecture, and the underlying framework. The development mode of "transaction-process-form" was proposed. We efficiently managed and implemented the development of business process in the project, and designed a visual form designer for manual semi intelligent development of page forms, so as to standardize the development process, improve the development efficiency and reduce the delivery cycle.

Keywords Web application development Unified development platform Microservice Transaction-process-form Standard development process

0 引 言

随着互联网的蓬勃发展，越来越多的企业摈弃了传统的手工管理信息业务的方式，而采用信息系统新型高效的管理方式。发展至今，基于 C/S 架构的 Web 应用系统在各大领域开枝散叶，如企业的各大门户网站、高校信息管理系统、电商系统、电子政务系统、医疗系统等。随着 Web 应用系统的规模和复杂度日益上升，Web 技术与架构也不断地改进，以适应新的需求。Web 应用架构[1]是早在 1989 年被英国人 Tim Berners-Lee 提出来，发展至今已有三四十年的历史。Web 开发技术也从原始的静态 Html 页面，历经 CGI[2]客户端浏览器交互程序、Servlet 和 Jsp[3]的组合，再到现在各种成熟的 MVC[4]框架。Web 应用开发方式可谓是越来越多元化、开发效率也有一定的提高。然而对于复杂的大型 Web 应用，如何快速开发，统一规范化管理开发流程，是当下 Web 应用开发领域值得研究的问题。

Java 的性能优越、功能强大，因此作为 Web 应用开发三大主流技术之一的 Java Web 倍受软件开发者的青睐。所谓 Java Web，是指用 Java 及其衍生出来的技术来解决 Web 网页开发领域的技术总和[5]。目前

收稿日期：2018－06－14。国家自然科学基金项目(71403112)。**胡飞菊**，硕士生，主研领域：计算机软件。**张少平**，教授。

Web 应用开发框架大部分是基于 MVC 的开发模式。如文献[4]使用 MVC 模式分离数据层和表现层,并使用 Servlet 和 Jsp 技术进行应用开发。文献[6-7]也是基于 MVC 的模式,引入 SSH 或 SSM 框架进行应用开发。但是随着软件行业的繁荣发展,像智慧城市这样复杂的新业务不断涌现。传统基于 MVC 的开发模式显现出功能越复杂,代码越复杂;灵活度低,局部功能修改,要重新构建整个项目;高并发性能低;交付周期长等不足之处,应对复杂应用力不从心。文献[8]提出 SOA-SSH 分层架构,将 Web Service 融合到 SSH 框架进行应用开发,有效地解决了复杂 Web 应用开发带来的问题。

当我们开启一个 Java web 项目时,首先要搭好框架,新建一个空的 Web 项目工程,融合各种前端和后端框架,包含核心框架、视图框架、持久层框架、缓存框架、前端 CSS 框架等。然后数据库配置基础的用户表、角色表、菜单表等,后台写好基本的用户登录、用户管理、角色管理等功能。最后调试项目运行,到此为此前期的工作准备就绪,之后在这基础上根据不同的业务功能,对单表或多表进行流程操作。由此我们可以看到对于每个 Web 项目都要搭建基础环境,配置基本的系统管理功能。程序员应对普遍的业务开发,都离不开对业务表的操作,工作量大且重复性很高。并且随着 WEB UI 框架、后台开发框架的逐渐成熟,代码生成+手工半智能开发将是新的趋势。

基于以上问题,本文提出一种手工半智能开发的 Web 统一开发平台。更优地组合目前的开发技术及框架,达到 Web 统一开发平台做到更好的规范、快速开发的效果。这个平台底层框架采用 spring boot 和 mybatis 框架[9],并引入微服务架构[10-12]的模式,将复杂性应用原子划分,实现并行开发、分布式管理。然后结合目前稳定的 WEB UI 框架,提出“事务-流程-表单”的开发模式,集成通用的报表插件、消息中间件等实用性插件。构建基于微服务架构的 Web 统一开发平台,帮助企业 Web 应用达到快速、统一开发的目的,提供一个良好的平台支撑。

1　微服务架构

传统的企业级应用是单体式应用,一般采用 MVC 模式开发,即分为模型层、表现层、控制层,用一种业务逻辑、数据、界面显示分离的方法组织代码。MVC 架构图如图 1 所示,应用系统被划分为多个模块化组件,打包放后运行在一个服务。用户通过 url 地址发送请求,访问系统中的服务。MVC 架构中模型层在应用程序中用于向数据库中存取数据;表现层依赖于模型层,主要渲染到页面中;控制层主要负责业务流程的处理。但对于复杂的应用,要采用传统的 MVC 模式开发,开发人员新增或修改某个功能要耗费很多时间去走查代码,构建部署和测试的时间也因代码量的增加而成倍的增长,可维护性低。

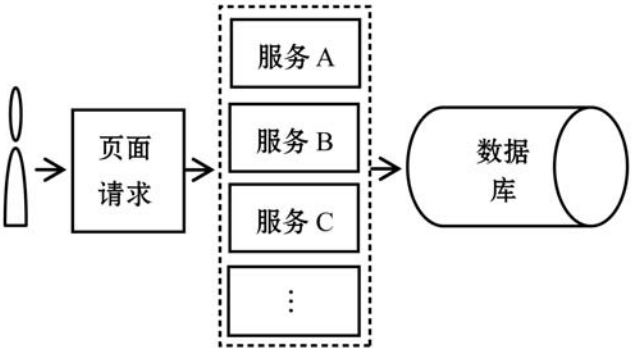

图 1　MVC 架构图

因此 Web 统一开发平台引入微服务架构模式。这种模式的开发方法,是以开发一组松耦合的小型服务方式来开发一个独立的应用系统。其中每个小型服务都运行在自己的进程中,并经常采用 HTTP 资源 API 轻量的机制来相互通信,如 REST。微服务架构模式的目的是实现传统业务功能应用的原子化拆分[13],不同于传统的 web 应用模块化开发的方式。传统的方式是按照系统业务功能的拆分,分成相互独立的模块,且各个模块只包含与其功能相关的内容,模块之间通过接口调用。而微服务的方式是让业务系统彻底的组件化和服务化,达到分离组件边界和责任,减少耦合,便于独立维护和升级。如图 2 所示是微服务架构图,庞大的单个应用系统划分为一套小且互相关联的服务,服务与服务之间互不影响,且可以相互调用,便于开发、理解和维护。

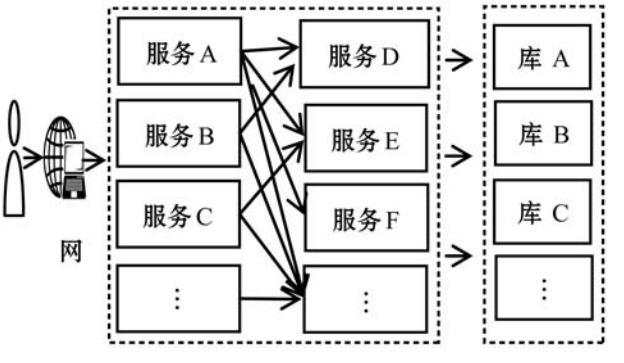

图 2　微服务架构图

微服务架构模式支持组件级的微服务功能的独立或集群部署[14]。一个服务只关注某个特定的功能,例如用户管理、采购管理等。且各服务的开发者可以独

立开发和部署,而不用考虑其他服务部署之后会对本服务产生影响。这种改变可以加快部署速度。UI 团队可以采用 AB 测试,快速的部署变化。微服务架构模式使得持续化部署成为可能。

微服务架构模式使得每个服务独立扩展[15]。在这种模式下技术选型是去中心化的,不同类型的业务组件可由不同的专业团队开发,且每个团队可以根据自身服务的需求和行业发展状况做出自己的判断,选择恰当的开发技术,提供 API 服务。

2 统一开发平台设计

2.1 平台架构设计

Web 统一开发平台采用微服务、微应用的架构设计,定位企业应用开发提供平台支撑。如图 3 所示是 Web 统一开发平台架构设计,从图中可以看到,平台提供基础的技术框架(微服务、API 网关),基础的功能组件,包括工作流、消息通知、报表开发等,帮助开发人员快速搭建基本的开发运行环境,提高开发效率。

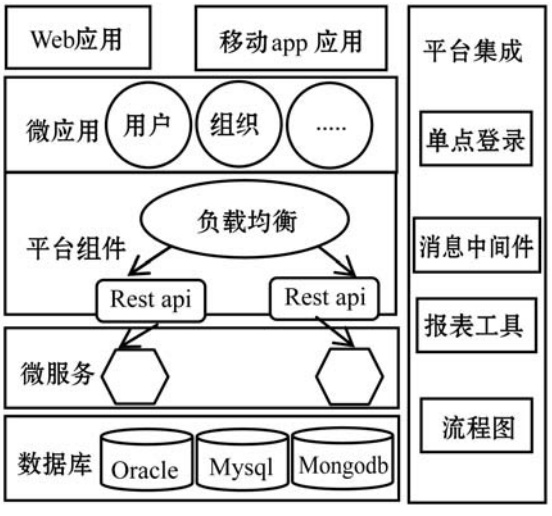

图 3 Web 统一开发平台架构图

2.2 Spring cloud + spring boot + mybatis

Spring cloud 是实现微服务架构的一套框架,包含服务注册、配置中心、消息通信总线、负载均衡等一系列插件。并且 Spring cloud 是一个基于 Spring boot 底层框架实现的应用框架,巧妙地利用 Spring boot 的优势简便地实现分布式系统基础设施的开发。如图 4 是 Spring cloud 原理图,系统中 A、B 服务都要在 Eureka 提供的服务端进行注册,向其提供服务、ip、端口,等其他附加信息。服务 A 要调用服务 B,首先会获取 Eureka 服务端所有服务的列表信息,然后找到服务 B 进行调用。

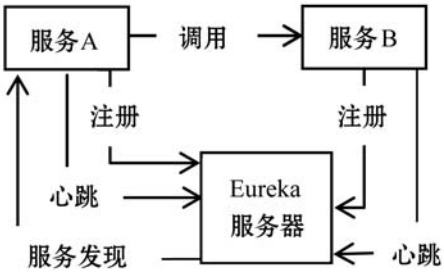

图 4 Spring cloud 原理图

Spring boot 是由 Pivotal 团队提供的框架,其设计目的是用来简化 Spring 应用的初始搭建以及开发过程。MyBatis 支持定制化 SQL、存储过程以及高级映射的优秀的持久层框架。Web 统一开发平台整合 Spring boot 和 mybatis 框架,并引入 Spring cloud 微服务架构的模式作为底层框架。

Web 统一开发平台采用 mvp 模式。如图 5 所示,mvp 与 mvc 模式不同的是使用 precenter 取代了 controller,切断了 View 与 Model 的通信[16]。首先用户发出一个页面请求,然后被 Spring 的 DispatcherServlet(前端控制器)拦截,映射直接进入 precenter 层处理,precenter 层调用 model 对数据进行存取操作,最后 precenter 层从 model 读取数据返回给页面。

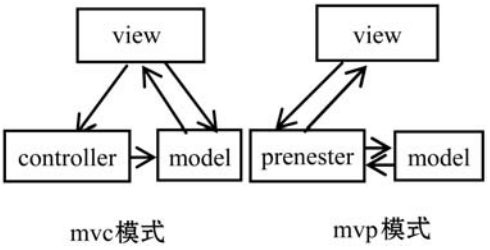

图 5 mvc 模式与 mvp 模式对比图

2.3 “事务-流程-表单”的开发模式

Web 统一开发平台融合 Spring cloud 微服务架构,将 Web 应用根据不同的业务功能划分成不同的单元组件(类似企业划分不同的事业部)。每一个单元组件都是以事务为核心,以流程为驱动,用表单承载数据。这些单元组件拥有独立而明确的业务边界,可以被单独运行和维护。不同的单元组件也可通过使用特定的调用协议和接口进行交互。且组件之间分布式部署和独立编码测试,可以有效地降低服务之间耦合性。

图 6 为事务功能实现的流程图,是整个统一开发平台设计的核心,所有的业务功能可按照这样的模式进行开发。单元组件的实现是以事务为入口,定义事务的基本属性、扩展属性以及相关基础数据。然后为事务定义服务流程,每个事务,可定义一个或多个流程。流程节点将关联多个表单以及任务句柄,任务句柄调用业务服务接口,业务服务实现具体事务的业务逻辑。

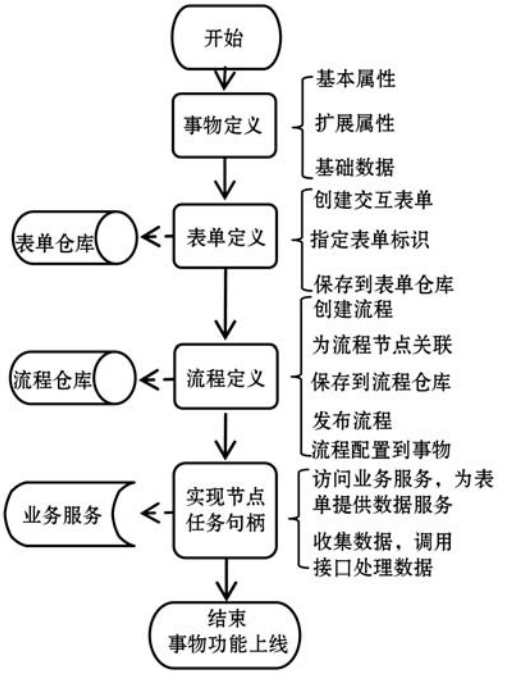

图 6　事务功能实现

2.4　关键技术

2.4.1　页面模板化设计

目前 Web 应用程序中页面设计主要采用的技术有:JSP、PHP、ASP 等。JSP 技术功能强大,可以直接在页面上编写业务逻辑代码,但是页面承载过多且复杂的逻辑代码后,会变得臃肿,可读性很差,且在第一次执行时还需要编译转换,耦合度也比较高,维护难度大。因此 Web 应用统一开发平台采用 FreeMarker 模板引擎[17],遵循模板 + 数据模型 = 输出的规范。模板负责动态页面的展示责任,不涉及业务逻辑代码,而所有的业务逻辑都是交由数据模型来处理,达到前后台完全分离,互不影响。类似功能页面可共享同一模板,提高代码的复用率。

2.4.2　表单可视化设计

目前 Web 应用开发中,很多公司的后台开发人员经常身兼多职,除了要实现功能以外,还要兼职页面、布局等的实现,但是程序员可能没有专业的前端人员效率快。因此考虑把页面的设计从码代码的形式转变为可视化设计,给页面量身定做一个表单设计器,提高页面开发的效率。如通用的表单样式布局提供多种样式选择,开发人员在可视化设计页面通过拖拽的形式一键生成代码。通过在 Web 应用统一开发平台中专门开发一个功能,实现页面的可视化设计。实现方式可引用市场上的 js 插件,也可以自定义可视化表单生成引擎[18]实现布局拖拽。表单设计器实现的效果如图 7 所示,页面样式设计可采用向导式开发。首先选择数据源进行表单初始化绑定,然后从控制面板中拖拽合适的控件进行布局,同时提供自定义属性进行控件的颜色,大小,位置配置再给每个控件绑定元数据。整体布局完成后可进行表单预览,设计完成后,表单引擎自动流转结合流程引擎,进行程序自动化生成。

图 7　表单可视化设计界面

简单表单可以通过表单设计器定义,使用流程引擎的表单服务,按不同数据类型定义表单的界面元素,关联统一的样式渲染后展示。对于复杂表单通过专有的 HTML 模板实现,可根据需要,以重定向或者流程引擎的外部表单方式关联到流程。

2.4.3　流程配置化

在复杂的业务系统中,业务流程是必要的,因此在平台中加入流程引擎[19]。同时平台基于流程即服务的理念,独立化运行准则,结合表单引擎,流转至流程任务句柄中。用户可根据业务场景进行流程节点及逻辑的设计,同样在用户流程任务句柄控制中,选择合适的句柄进行逻辑节点的绑定,通过流程引擎的自动化改造,将复杂的功能快速实现。

3　统一开发平台应用

基于微服务架构的 Web 统一开发平台已成功用于“企业项目管理系统”、“智慧校园管理系统”的开发。在此平台上进行开发工作,有效地缩减了平台搭建、基础管理功能的开发周期。平台中的“事务-流程-表单”的开发模式,统一规范了项目中的事物管理。且使项目适应力强,能高效地适应功能变更和增加的需求。

4　结　语

本文主要从平台整体架构设计理念,“事务-流程-表单”的开发模式设计,以及平台采用的关键技术这三个主要方面进行阐述,构画统一开发平台的蓝图。平台以 Spring boot + mybatis 做为底层框架,采用 Spring cloud 微服务架构模式能够将整个复杂的业务划分成多个组件,这些组件可以独立运行,分期上线;

提出“事务-流程-表单”的开发模式,可以集中规范化管理流程服务,降低功能模块之间的耦合性,同时可进行跨平台、跨项目移植,提高服务的复用性;平台融合 FreeMarker 模板引擎技术,可以彻底地分离表现层和业务逻辑,提高前后和后台开发人员的工作效率。融合表单可视化生成引擎和流程引擎,很好地实现了“事务-流程-表单”的开发模式,统一、规范业务开发流程,手工半智能开发页面表单,提高了开发效率。但平台有个不足之处,就是使用 FreeMarker 模板时要注意,开发人员更新模板后,一定要更新模板生成的 HTML 页面,不然页面展示的还是历史数据。平台整体框架还集成其他的成熟组件,如报表插件、单点登入等,这样一来开发人员只需实现业务功能,结合向导式进行设置开发,达到统一、快速开发的目的,给企业应用开发提供一个良好的平台支撑。

参 考 文 献

[1] 鲍捷. Web:为所有人——记图灵奖得主 Tim Berners-Lee 的伟大贡献 J/OL]. 中国计算机学会通讯,2017(6). http://www.sohu.com/a/150695221_634795.

[2] 黄怀亮. CGI 技术及其应用[J]. 计算机应用研究,1999(3):75-77.

[3] 杨黎明,董传良,董玮文. 服务器端动态网页技术——JSP+Servlet[J]. 计算机工程,2001(2):126-127.

[4] 陆荣幸,郁洲,阮永良,等. J2EE 平台上 MVC 设计模式的研究与实现[J]. 计算机应用研究,2003(3):144-146.

[5] 百度百科. java web [EB/OL]. https://baike.baidu.com/item/java%20web/1611944.

[6] 谌湘倩,狄文辉,孙冬. 基于 SSH 框架与 AJAX 技术的 JavaWeb 应用开发[J]. 计算机工程与设计, 2009, 30(10): 2590-2592, 2596.

[7] 李洋. SSM 框架在 Web 应用开发中的设计与实现[J]. 计算机技术与发展,2016,26(12):190-194.

[8] 付更丽,曹宝香. SOA-SSH 分层架构的设计与应用[J]. 计算机技术与发展,2010,20(1):74-77.

[9] Spring 整合 mybatis[EB/OL]. http://www.mybatis.org/spring-boot-starter/mybatis-spring-boot-autoconfigure/.

[10] 艺敏. “微服务”架构:更灵活、更可靠、更开放[EB/OL]. [2015-04]. http://www.infoq.com/cn/articles/pworld2015-interview-guwei/.

[11] 王磊. 解析微服务架构(一)单块架构系统以及其面临的挑战[EB/OL]. [2015-05]. http://www.infoq.com/cn/articles/analysis-the-architecture-of-microservice-part-01.

[12] Spring 官网[EB/OL]. http://projects.spring.io/spring-cloud/.

[13] Zimmermann O. Microservices tenets[J]. Computer Science-Research and Development, 2017, 32:1-10.

[14] Ortin F, O'Shea D. Towards an easily programmable IoT framework based on microservices[J]. Journal of Software, 2018,13(2): 90-102.

[15] Namiot D, Sneps-Sneppe M. On Micro-services Architecture [J]. International Journal of Open Information Technologise, 2014,2(9):24-27.

[16] 曾露. MVP 模式在 Android 中的应用研究[J]. 软件,2016,37(6):75-78.

[17] 孙聚. 基于 FreeMarker 引擎的代码生成工具的设计与实现[D]. 哈尔滨:哈尔滨工业大学,2015.

[18] 周晖,尹建伟,陈刚,等. 基于 Struts 框架的 Web 表单快速开发平台[J]. 计算机应用研究,2004(8):191-194.

[19] 吴国旭. 基于 SOA 技术的 XPDL 流程引擎的实现[D]. 长春:吉林大学,2009.

(上接第 11 页)

[12] 李锋,汤宝平,刘文艺. 遗传算法优化最小二乘支持向量机的故障诊断[J]. 重庆大学学报,2010, 33(12):14-20.

[13] 赵志国,司传胜. 基于多岛遗传算法的铰接车轮边减速器优化设计[J]. 机械设计与制造, 2010(12):213-215.

[14] Cui J, Wang Y. A novel approach of analog circuit fault diagnosis using support vector machines classifier[J]. Measurement, 2011, 44(1):281-289.

[15] Long B, Tian S, Wang H. Diagnostics of filtered analog circuits with tolerance based on LS-SVM using frequency features[J]. Journal of Electronic Testing, 2012,28 (3):291-300.

[16] Samanta B, Nataraj C. Prognostics of machine condition using soft computing[J]. Robotics and Computer-Integrated Manufacturing, 2008,24(6):816-823.

[17] Widodo A, Yang B S. Machine health prognostics using survival probability and support vector machine[J]. Expert Systems with Applications, 2011,38(7): 8430-8437.

[18] Cherkassky V, Ma Y. Practical selection of SVM parameter and noise estimation for SVM regression[J]. Neural Networks, 2004, 17(1):113-126.

[19] Huang C, Zhang R, Chen Z, et al. Predict potential drug targets from the ion channel proteins based on SVM[J]. Journal of Theoretical Biology, 2010, 262(4):750-756.

[20] Widodo A, Yang B S, Kim E Y, et al. Fault diagnosis of low speed bearing based on acoustic emission signal and multi-class relevance vector machine [J]. Nondestructive Testing & Evaluation, 2009, 24(4):313-328.