2 WEB 应用开发及持续集成

本论文的研究目的是对于开发的 WEB 应用进行应用、数据库和服务器层级进 行优化，通过对比不同优化策略对于系统性能提升的影响研究一套可行的系统优 化策略。为了实现优化的目的，首先需要创建一个 WEB 应用或者对现有的应用进 行优化，本人在研究生期间，参与了海信集团智能商用公司的小微商户平台系统 的开发，主要工作是后台开发和服务器运维工作，因此本论文将基于参与的小微 商户 WEB 平台进行系统优化策略的研究。

在优化之前，首先需要描述作为优化原型的小微商户WEB平台的平台的开发 框架以及应用部署流程。

2.1 Spring MVC 开发框架

Spring MVC 是一种基于 Web 开发的系统架构，目前主流的系统框架还有 Struts，但是 Spring MVC 架构引起优势依然处于不可替代的位置。在 Spring MVC 架构中，系统的终极目标被解聚、分块，并在不同的模块中表达出来，如此系统 的耦合度得到大幅降低。同时，该框架的整体运行也是在模块驱动的基础之上的， 用户首先发出请求，然后系统予以相应，即常规的请求-相应模型。

2.1.1 Spring MVC 架构的特点

Spring MVC 架构不同于传统的体系架构，在其发展的过程中吸收其他系统的 特点，如支持 Userful 风格，灵活的本地化等。这些特征保证了 Spring MVC 框架在 发展过程中，更加受到开发者的青睐。同时，吸收了前人的经验以后，Spring MVC 框架也具备专属的特点，比如用 Spring MVC 框架设计的 WEB 系统其耦合度较 低，系统总体表现出简洁明了的风格，这样用户就能更快捷的掌握数据库系统的 操作[2]。

Spring MVC 框架是在 Spring 框架的基础上发展起来的，因此 Spring MVC 框 架集成了 Spring 框架的所有功能，而且两者在相互继承方面表现出很好的融合性。 这一特点给系统开发带来了很多方便。除此之外，Spring MVC 模型在图形兼容性 和数据验证的灵活性上都有先天的优势。正式因为这些特点，使 Spring MVC 模型 在实际应用中发挥出巨大的优势，同时也在程序员中受到青睐。本文的 WEB 应用 就是建立在该框架之上的。

2.1.2 Spring MVC 框架处理流程

Spring MVC 架构是一个基因用户请求的系统架构，其过程也要求有 Web 驱动 的参加。具体过程现有用户发送请求到前端控制器，这时候前端控制器将请求委 托给处理器;于是处理器就开始处理请求，即调用业务对象进行分析，分析完毕后 将模型数据返回给处理器;然后处理器再将处理结果通过模型和视图返回给前端 控制器;此时前端控制器显示的数据用户还无法处理，还需要进行视图渲染，最 终在前端控制器显示出来，即用户发送的请求产生了响应[6]。整个过程参见图 2-1， 此图反映了用户请求-响应的过程。

2.1.2.1 体系的三层架构

MVC(Model View Controller), 即视图、模型和控制器三个英文单词的首字母缩 写，按照应用的流程本系统可以在 Spring MVC 架构下分为三层，具体的架构为显 示层、控制层和数据访问层。

1. 显示层。主要有 Spring 和 JSP 经过页面 REQUEST 请求，Spring 的 URL Mapping 对应到响应的控制器，通过对页面发送的请求，Spring 自动的去寻 找响应的控制类，并将结构展示在页面上。

2. 控制层。也就是业务层，是系统中最重要的一层，这层主要是接受URL发送 过来的业务请求，然后去访问 Service 层，把需要的数据装入缓存中，任何通 知 Action 进行跳转。

3. 数据访问层。这层最重要的作用是对数据库中数据以及 XML 文件中的数据 进行访问及操作，并把操作的结果反馈到业务逻辑层，将相应的结果交由下 一层去处理。

这个三层架构的体系的应用很好的解决了系统资源的分配问题，而且将系统的耦 合度降到了可以接受的最低水平。同时，关于异步处理问题上，本系统使用了 FDP 框架，FDP 框架是对 AJAX 的一种轻量级的封装，它的目标是使 AJAX 和 XMLHttp Request 更简单。通过 FDP 框架，我们只需要在相应的配置对应的 JAVA 方法 以及相对于的实体类，我们很容易的就可以实现许多异步操作，使一些操作不用 刷新页面即可达到相应的效果。可以认为 FDP 框架是对 MVC 三层框架的一种拓 展，其本质还是在 MVC 框架内的。由此可见，笔者在系统体系的选择上更倾向于 架构简单、可移植性好的系统框架模型。

2.2 应用开发工具

在开发基于 Spring MVC 的 WEB 应用过程中，需要用到的基础编程语言是 JAVA，系统的架构采用的 MVC 三层架构。这些都属于手段，但是想要达到最终目 的则需要用到一些基础的软件工具，这些工具的使用将加快工作效率。

2.2.1 IntelliJ IDEA

IntelliJ IDEA 是一款 Java 集中开发环境工具软件，由捷克软件公司 JetBrains 在 2001 年 1 月时推出最初版[7]。该软件被认为是当前 Java 开发效率最快的 IDE 工 具。它集成了开发过程中实用的众多功能，几乎可以不用鼠标可以方便的完成你 要做的任何事情，最大程度的加快开发的速度。简单而又功能强大。与其他的一 些繁冗而复杂的 IDE 工具有鲜明的对比[8]。

2.2.2 Maven

Apache Maven 是由 Apache 软件基金会所提供德一个项目管理及自动构建工 具[9]。它基于项目对象模型概念、能够实现一个 Java 项目的构建和依赖管理。本 论文所涉及的 WEB 应用就是使用 Maven 来构建 Java Web 项目。

2.2.3 Tomcat

WEB 应用的 Web 服务器采用的是有 Apache 软件基金会 (Apache Software Foundation) 下属的 Jakarta 项目开发的一个 Servlet 容器，由于其本身也包含一个 HTTP 服务器，因此也常常被用作一个单独的 Web 服务器[10]。Tomcat7 支持最新 的 Servlet 3.0 规范，而且技术先进、稳定性强一级免费的特点，深受 Java 爱好者 的喜爱并得到了部分软件开发商的认可，成为目前比较流行的 Web 应用服务器。

2.2.4 MySQL

MySQl 是个小型关系型数据库管理系统，之所以使用 MySQL 是因为 MySQL 是一款免费的数据库管理系统，而且其建议的操作以及其兼容性都是其优点[11]。

MySQL 的特点主要有:

• 为多种編程语言提供了API。这些編程语言包括C、C++、C♯、Java、Perl、PHP、

Python、Ruby 等。

• 支持多线程，充分利用 CPU 资源，支持多用户。

• 提供 TCP/IP、ODBC 和 JDBC 等多种数据库连接途径。

• 提供用于管理、检查、優化数据库操作的管理工具。

• 可以处理拥有上千万条记录的大型数据库。

2.2.5 Couchbase

在互联网时代，我们面对的是更多的客户端，更低的请求延迟，这就需要对数 据做大量的 Cache 以提高读写速度[12]。目前主流的 Cache 系统 memcached 和 redis 虽然有很熟的解决方案，但是也都有其局限性 [13] :

• Cluster 支持不够。在扩容、负载均衡、高可用等方面存在明显不足。

• 持久化支持不好，出现问题后恢复的代价大。memcached完全不支持持久化，

redis 的持久化会造成系统间歇性的负载很高。

Couchbase 是一个 NoSQL 数据库，它是世界各国的开发者在 2011 年推出的，由于 它有良好的 cluster 支持、异步持久化的支持得到了众多开发者的青睐，他的特点 主要有:

• Couchbase 的对等网设计，smart client 直接获取整的集群的信息，在客户端 实现负载均衡，整个集群没有单点失效，并且完全支持平行扩展。

• vBucket 的引入，完全实现了 auto sharding，可以方便灵活的把数据的子集在 不同节点上移动，以实现集群动态管理。

• Couchbase 有一个非常专业的 web 管理界面，并且支持通过 RESTful API 管 理，这也是 memcached, redis 不能企及的。

2.3 应用开发流程

WEB 应用在开发的时候设计为前后端分离，通过 FDP 平台实现前端到后端 的请求，所以本论文测试 WEB 应用的开发主要分为前端开发、后端开发以及数据 库搭建三个方面。

2.3.1 前端开发

应用的前端相当于 MVC 架构中的视图层，主要实现用户的交互，包括页面信

息的展示以及用户请求的转发和响应，在前端开发中，通过 Html 和 JavaScript 来设计实现应用的页面展示，通过 FDP 的 Ajax 请求将前端的用户请求转发到应用

的后端。

2.3.2 后端开发

应用的后端主要分成了 Controller、Service、Pst 三层，其中 Controller 层负责 处理用户的请求然后将业务转发给 Service 层，Service 层负责实现用户的请求，通 过设计不同的业务逻辑将用户需要的数据返回，涉及到数据的读取则通过 Pst 层， Pst 层主要负责对数据库的增删改查，将结果返回到 Service 层。

2.3.3 数据库搭建

应用的数据主要分为基础数据也业务数据，其中基础数据主要包括页面的功 能板块、系统的定时任务、数据的权限设置、页面的功能逻辑等数据，业务数据主 要包括用户的信息、用户操作记录、用户的商店、商品、销售信息等。

2.4 基于 Jenkins 的持续集成方案开发

在每一次 WEB 应用的开发、上线过程中，不可避免的要将本地环境打包上传 到生产环境或者是测试环境进行解压，每一次人工的干预无疑增加了时间成本和 错误率，通过 Jenkis 设计实现应用的持续集成，这在很大程度上能够帮助开发着 实现快速的应用部署和错误重现 [14] 。

Jenkins 是一个用 Java 编写的开源的持续集成工具，它提供了软件开发的持续 集成服务，运行在 Servlet 容器中，它可以执行基于 Apache Ant 和 Apache Maven 的项目，以及任意的 Shell 脚本和 Windows 批处理命令[15]。

由于小微平台是通过 Java 语言开发的，因此可以通过 Jenkins 的 Maven 工具 进行构建并进行语法检查，通过 SSH 插件上传构建后的 war 包以及前端页面到服 务器端实现部署 [16] 。具体的自动构建及部署流程如下:

2.4.1 Jenkins 软件安装配置

由于 Jenkins 运行在 Servlet 容器中，因此在安装配置 Jenkins 之前需要保证安 装 Jenkins 的服务器中已经安装好了对应版本的 JDK 和相匹配的 Tomcat 软件[17]。 1. 访问http://mirrors.jenkins.io/war-stable/2.7.1/jenkins.war下载2.7.1版本的jenk-

ins 安装包。

2. 将 jenkins.war 文件放到本地构建服务器(10.18.8.19)的 Tomcat 应用目录

(/opt/tomcat/apache-tomcat-7.0.68/webapps)中。

3. 配置并启动 Tomcat 即可完成安装，端又配置为 8080，启动命令配置为: systemctl start/stop/status tomcat\_jenkins.service

4. 访问 jenkins 地址 (http://10.18.8.19:8080/jenkins/) 初始化 Jenkins:

• 完成认证,按照页面提示，将对应文件内容中的密码输入到文本框中，点

击 continue，如图2-2所示

• 在插件选择处选择安装建议的插件

• 安装完成后在新的页面中配置用户名和密码完成安装，如图2-3

• 进入图2-4页面表示安装成功

2.4.2 自动构建及检查方案设计

由于小微平台通过 Maven 工具来解决代码使用过程中的函数依赖和本地项目 构建[18]，所以在进行自动构建的时候同样可以通过使用 Maven Integration plugin 插 件实现项目的构建和代码检查，另外由于小微平台在开发过程中使用 SVN 来进行 代码的版本控制，因此也需要在 Jenkins 中配置 SVN 插件来进行代码的同步和版 本管理，除此之外，需要在安装 Jenkins 的服务器中提前安装好 Maven 软件。

小微项目在开发过程中通过前后端分离进行的开发，因此在构建过程中需要 新建前端项目和后端项目，考虑到前端项目的构建相对后端项目比较简单，本论 文只介绍后端项目的创建和构建过程。

• 新建 Jenkins 项目

1. 在 Jenkins 主页面新建项目后，首先填写项目名称(以 ServerDeployfor-

PROD1.2 为例)

2. 源码管理配置为 SVN，具体如图2-5所示:

在对应的框中填写代码库的地址、通过 Add 按钮实现 SVN 的权限认证，

其它选项选择默认即可。

3. 在 Pre Steps 处增加 shell 命令为: rm -rf target/MiniShop

确保每次构建时清除前一次构建信息。

4. 在build标签页下配置构建时的操作，主要包括配置POM文件(pom.xml)

的位置,pom.xml 文件主要描述本 Maven 工程的整个生命周期所需要 执行的功能和特性[19]，考虑到本项目的实际开发过程在这里选择项目 pom2.xml

5. 在 Post Steps 配置构建完成后的操作，首先需要在"Run only if build suc- ceeds or is unstable" 出打勾，保证在只有构建成功后才执行构建完成后 的操作，其次需要配置构建完成对文件的操作:

通过上述命令将每次构建完成后的文件进行备份保存和修改响应权限。

2.4.3

所有项目信息配置完成后保存即可。 构建Jenkins项目进入项目主页，在页面左侧点击“立即构建”按钮即可进行构 建，构建完成后将构建完成的 war 包上传到服务器即可，如果构建失败则表 示在代码的书写过程中存在错误或者项目的库存在异常，需要在项目构建页 面的构建信息页面中查看错误信息，并且根据错误信息来解决问题。

自动部署方案设计

Jenkins 自动部署的方案是在每次构建完成后，让 Jenkins 可以自动的通过 SSH 协议访问远程服务器并将构建完成后的文件上传到服务器，并且执行服务器中的 相应脚本来实现自动备份旧项目和部署新项目的目的。

1. 在具体配置自动部署之前，需要先安装 Publish Over SSH 插件，通过这个插 件，Jenkins 可以实现通过 SSH 协议对远程服务器的访问。

2. 在插件安装完成之后，需要配置插件来配置访问SSH的密钥和密码，在Jenk- ins“系统管理-系统设置”页面中会出现如图2-6配置:  
按照不同项目配置 SSH Server 信息和登录验证信息等。

3. 插件配置完成后需要配置自动部署，插件安装完成后在“构建后操作”的选项

中会出现“Send build artifacts over SSh”的选项，点击之后会出现配置:

配置上传的文件名，上传后的路径，上传后需要执行的脚本和参数等，脚本

可以参考附录A。

4. 配置完成后再进行构建，Jenkins会自动的在构建后将构建生成的文件上传到

服务器端指定路径，通过执行指定的脚本和参数将文件部署到 Tomcat 中，实 现自动部署。

2.5 本章总结

本章主要介绍了本人参与的小微项目的 WEB 平台所使用的开发框架、开发工 具、开发过程以及系统上线的持续集成方案，通过持续集成方案增强了代码上线 的稳定性，在本文后面的三章将会对本章开发的平台进行不通层级的优化，来实 现本项目的安全性和高性能。