A Seminar Report

on

Spring Boot REST API

submitted in partial fulfillment of requirements for the award of degree of

Bachelor of Technology

in

Computer Science and Engineering

submitted by

M Nikhil Pavan Sai (16211A05D6)

Under the Guidance of

Mr. P. BHASKAR RAO, MCA, M. Tech
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B V Raju Institute of Technology

(UGC Autonomous ,NBA and NAAC Accredited)
Vishnupur, Narsapur, Medak District - 502313.

2016 – 2020

CERTIFICATE

This is to certify that the Seminar entitled **"Spring Boot REST API"** has been submitted by **M Nikhil Pavan Sai (Roll No)** under my guidance in partial fulfillment of the requirements of Under Graduate Degree **Bachelor of Technology** in **Computer Science and Engineering** at **B.V.Raju Institute of Technology**, Narsapur.

Date:

Place: Narsapur

Signature of Guide

Mr. P. Bhaskar Rao, Assistant Professor

Signature of HOD

Dr. Ch. Madhu Babu

CONTENTS

ACKNOWLEDGMENT

The success and final outcome of this presentation required a lot of guidance and assistance from many people and I am extremely fortunate to have got this all along the completion. Whatever I have done is only due to such guidance and assistance and I would not forget to thank them.

I respect and thank **Mr. P. Bhaskar Rao** for giving me an opportunity to give a presentation and providing all support and guidance which made me complete the work on time. I am extremely grateful to **Mr. Karthik Kovuri**, Projects & Seminars Coordinator for providing such a nice support and guidance. Another person who has my gratitude is **Dr. Ch. Madhu Babu,** Professor and Head, Department of CSE.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staff of Department of Computer Science which helped me in successfully completing the work. Also, I would like to extend our sincere regards to all the non-teaching staff of department of Computer Science for their timely support.

Signature of the student

M Nikhil Pavan Sai
16211A05D6

Abstract

Now, it's a popular technology of Web programming development based on J2EE platform Framework, which Struts framework that based on MVC design pattern divided the system into the controlling part, business logic part and view one, and achieved the decoupling between layers, Hibernate framework offered a method of packaging a lightweight object for JDBC, and reduced the difficulty of developing business model part. This paper, under the system development background, analyzed the merit and shortcoming of Struts and Hibernate framework, integrated Struts and Hibernate framework technology into the system by Spring framework and implemented the college student information management platform. Through framework integration, the efficiency of system development increases greatly and workload of coding reduces observably. Thus, the system has higher reliability and maintainability. For the development and operating efficiency of Web applications based on the Model-View-Controller (MVC) framework, and, according to the actual business environment and needs in the project practice, the framework of Web application system is studied in this paper. Through the research of Spring MVC framework framework as well as some related core techniques, combined with JSP and JSTL technology, this paper realizes the design of a lightweight Web application framework based on Spring MVC.

Guide Signature :

Mr. P. Bhaskar Rao

Assistant Professor

Student Signature:

M Nikhil Pavan Sai

16211A05D6

CHAPTER 1
INTRODUCTION

1. Introduction

Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications. This chapter will give you an introduction to Spring Boot and familiarizes you with its basic concepts.

What is Micro Service?
Micro Service is an architecture that allows the developers to develop and deploy services independently. Each service running has its own process and this achieves the lightweight model to support business applications.

Advantages
Micro services offers the following advantages to its developers −

- Easy deployment
- Simple scalability
- Compatible with Containers
- Minimum configuration
- Lesser production time

What is Spring Boot?
Spring Boot provides a good platform for Java developers to develop a stand-alone and production-grade spring application that you can just run. You can get started with minimum configurations without the need for an entire Spring configuration setup.

Advantages
Spring Boot offers the following advantages to its developers −

- Easy to understand and develop spring applications
- Increases productivity
- Reduces the development time

Goals
Spring Boot is designed with the following goals −

- To avoid complex XML configuration in Spring
- To develop a production ready Spring applications in an easier way
- To reduce the development time and run the application independently
- Offer an easier way of getting started with the application

Why Spring Boot?
You can choose Spring Boot because of the features and benefits it offers as given here –

- It provides a flexible way to configure Java Beans, XML configurations, and Database Transactions.
- It provides a powerful batch processing and manages REST endpoints.
- In Spring Boot, everything is auto configured; no manual configurations are needed.
- It offers annotation-based spring application
- Eases dependency management
- It includes Embedded Servlet Container

How does it work?
Spring Boot automatically configures your application based on the dependencies you have added to the project by using *@EnableAutoConfiguration* annotation. For example, if MySQL database is on your classpath, but you have not configured any database connection, then Spring Boot auto-configures an in-memory database.

The entry point of the spring boot application is the class contains *@SpringBootApplication* annotation and the main method.

Spring Boot automatically scans all the components included in the project by using *@ComponentScan* annotation.

Spring Boot Starters
Handling dependency management is a difficult task for big projects. Spring Boot resolves this problem by providing a set of dependencies for developers convenience.

For example, if you want to use Spring and JPA for database access, it is sufficient if you include spring-boot-starter-data-jpa dependency in your project.

Note that all Spring Boot starters follow the same naming pattern spring-boot-starter-*, where * indicates that it is a type of the application.

CHAPTER 2
HISTORY AND DETAILS

2. History and Details

The Beginnings

In October 2002, Rod Johnson wrote a book titled Expert One-on-One J2EE Design and Development. Published by Wrox, this book covered the state of Java enterprise application development at the time and pointed out a number of major deficiencies with Java EE and EJB component framework. In the book he proposed a simpler solution based on ordinary java classes (POJO – plain old java objects) and dependency injection. Following is an excerpt from the book,

In the book, he showed how a high quality, scalable online seat reservation application can be built without using EJB. For building the application, he wrote over 30,000 lines of infrastructure code! It included a number of reusable java interfaces and classes such as ApplicationContext and BeanFactory. Since java interfaces were the basic building blocks of dependency injection, he named the root package of the classes as com.interface21. As Rod himself explained later, 21 in the name is a reference to 21st century!

One-on-One J2EE Design and Development was an instant hit. Much of the infrastructure code freely provided as part of the book was highly reusable and soon a number of developers started using it in their projects. Wrox had a webpage for the book with source code and errata. They also provided an online forum for the book. Interestingly even after 15 years, this book and its principles are still relevant in building high quality java web applications. I highly recommend that you get a copy for your collection!

Spring is Born

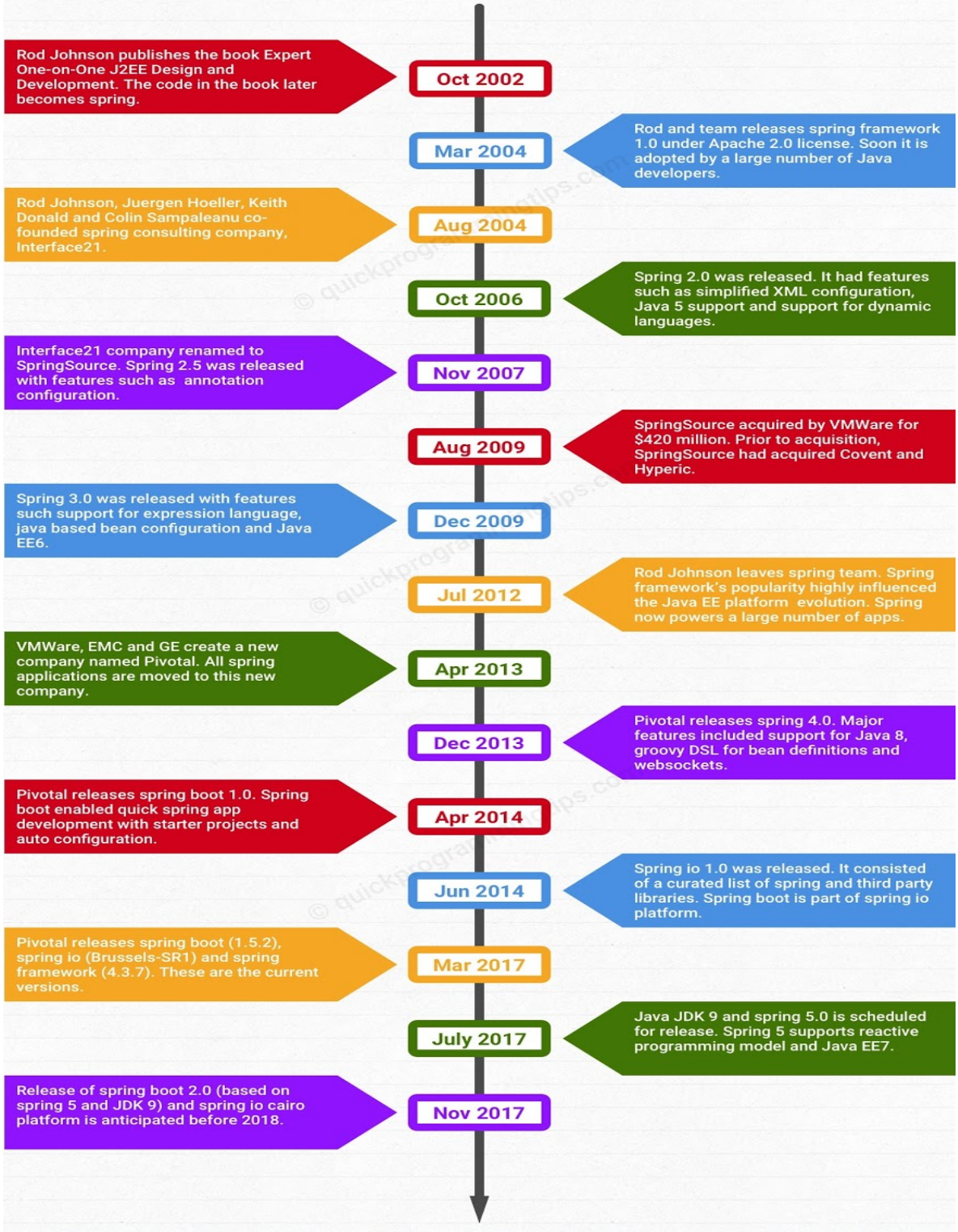
Shortly after the release of the book, developers Juergen Hoeller and Yann Caroff persuaded Rod Johnson to create an open source project based on the infrastructure code. Rod, Juergen and Yann started collaborating on the project around February 2003. It was Yann who coined the name "spring" for the new framework. According to Rod, spring represented a fresh start after the "winter" of traditional J2EE! Here is an excerpt from Yann Caroff's review of Rod's book in January 2003!,

Rod Johnson's book covers the world of J2EE best practices in an amazingly exhaustive, informative and pragmatic way. From coding standards, idioms, through a fair criticism of entity beans, unit testing, design decisions, persistence, caching, EJBs, model-2 presentation tier, views, validation techniques, to performance, the reader takes a trip to the wonderland of project development reality, constraints, risk and again, best practices. Each chapter of the book brings its share of added value. This is not a book, this is truly a knowledge base.

In June 2003, spring 0.9 was released under Apache 2.0 license. In March 2004, spring 1.0 was released. Interestingly, even before 1.0 release, spring was widely adopted by developers. In August 2004, Rod Johnson, Juergen Hoeller, Keith Donald and Colin Sampaleanu co-founded interface21, a company focused on spring consulting, training and support.

Yann Caroff left the team in the early days. Rod Johnson left spring team in 2012. Juergen Hoeller is still an active member of spring development team.

Story of Spring Framework



History of Spring Boot

In October 2012, Mike Youngstrom created a feature request in spring jira asking for support for containerless web application architectures in spring framework. He talked about configuring web container services within a spring container bootstrapped from the main method! Here is an excerpt from the jira request,

I think that Spring's web application architecture can be significantly simplified if it were to provided tools and a reference architecture that leveraged the Spring component and configuration model from top to bottom. Embedding and unifying the configuration of those common web container services within a Spring Container bootstrapped from a simple main() method.

This request lead to the development of spring boot project starting sometime in early 2013. In April 2014, spring boot 1.0.0 was released. Since then a number of spring boot minor versions came out,

Spring boot 1.1 (June 2014) – improved templating support, gemfire support, auto configuration for elasticsearch and apache solr.

Spring boot 1.2 (March 2015) – upgrade to servlet 3.1/tomcat 8/jetty 9, spring 4.1 upgrade, support for banner/jms/SpringBootApplication annotation.

Spring boot 1.3 (December 2016) – spring 4.2 upgrade, new spring-boot-devtools, auto configuration for caching technologies(ehcache, hazelcast, redis, guava and infinispan) and fully executable jar support.

Spring boot 1.4 (January 2017) – spring 4.3 upgrade, couchbase/neo4j support, analysis of startup failures and RestTemplateBuilder.

Spring boot 1.5 (February 2017) – support for kafka/ldap, third party library upgrades, deprecation of CRaSH support and actuator loggers endpoint to modify application log levels on the fly.

CHAPTER 3
FEATURES

3. Features

Spring Boot Features:

- Web Development
- SpringApplication
- Application events and listeners
- Admin features
- Externalized Configuration
- Properties Files
- YAML Support
- Type-safe Configuration
- Logging
- Security

1. **Web Development**

It is well suited Spring module for web application development. We can easily create a self-contained HTTP server using embedded Tomcat, Jetty or Undertow. We can use the spring-boot- starter-web module to start and running application quickly.

2. **SpringApplication**

It is a class which provides the convenient way to bootstrap a spring application which can be started from main method. You can call start your application just by calling a static run() method.

```
public static void main(String[] args){
    SpringApplication.run(className.class, args);
}
```

3. **Application Events and Listeners**

Spring Boot uses events to handle variety of tasks. It allows us to create factories file that are used to add listeners. we can refer it by using ApplicationListener key.

Always create factories file in META-INF folder like: META-INF/spring.factories

4. **Admin Support**

Spring Boot provides the facility to enable admin related features for the application. It is used to access and manage application remotely. We can enable it by simply using spring.application.admin.enabled property.

5. **Externalized Configuration**

Spring Boot allows us to externalize our configuration so that we can work with the same application in different environments. Application use YAML files to externalize configuration.

6. **Properties Files**

Spring Boot provides rich set of Application Properties. So, we can use that in properties file of our project. Properties file is used to set properties like: server-port = 8082 and many others. It helps to organize application properties.

7. **YAML Support**

It provides convenient way for specifying hierarchical configuration. It is a superset of JSON. The SpringApplication class automatically support YAML. It is successful alternative of properties.

8. **Type-safe Configuration**

Strong type-safe configuration is provided to govern and validate the configuration of application. Application configuration is always a crucial task which should be type-safe. We can also use annotation provided by this library.

9. **Logging**

Spring Boot uses Common logging for all internal logging. Logging dependencies are managed by default. We should not change logging dependencies, if there is no required customization is needed.

10. **Security**

Spring Boot applications are spring bases web applications. So, it is secure by default with basic authentication on all HTTP endpoints. A rich set of Endpoints are available for develop a secure Spring Boot application.

CHAPTER 4
FRAMEWORK AND ARCHITECTURE

4. FRAMEWORK AND ARCHITECTURE

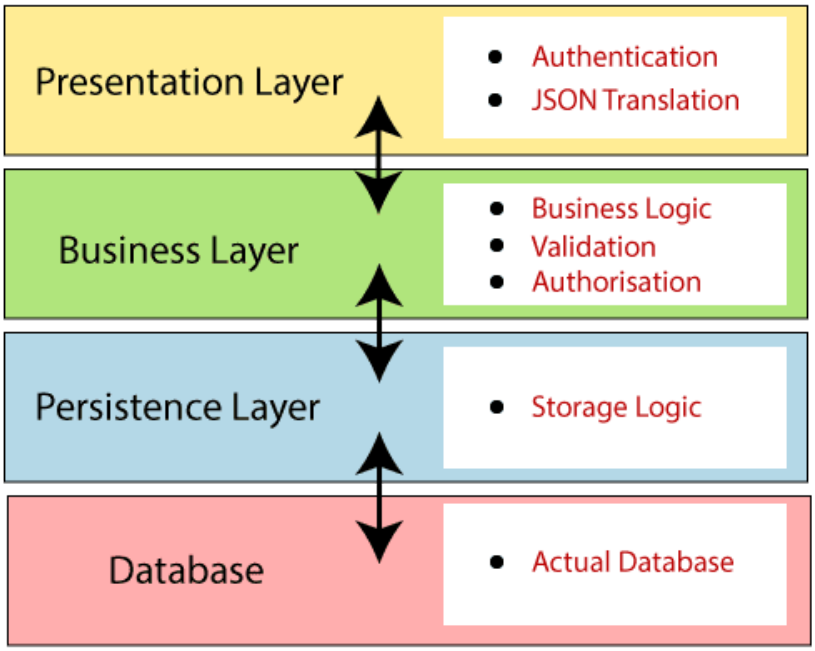
Spring Boot Architecture

Spring Boot is a module of the Spring Framework. It is used to create stand-alone, production-grade Spring Based Applications with minimum efforts. It is developed on top of the core Spring Framework.

Spring Boot follows a layered architecture in which each layer communicates with the layer directly below or above (hierarchical structure) it.

Before understanding the Spring Boot Architecture, we must know the different layers and classes present in it. There are four layers in Spring Boot are as follows:

- Presentation Layer
- Business Layer
- Persistence Layer
- Database Layer



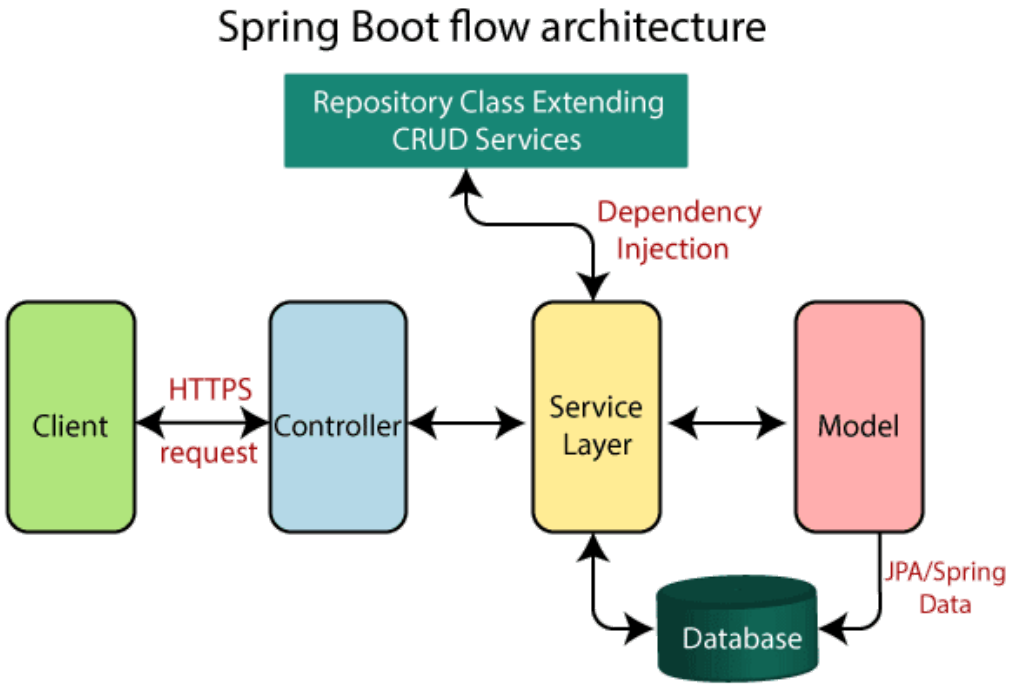
Presentation Layer: The presentation layer handles the HTTP requests, translates the JSON parameter to object, and authenticates the request and transfer it to the business layer. In short, it consists of views i.e., frontend part.

Business Layer: The business layer handles all the business logic. It consists of service classes and uses services provided by data access layers. It also performs authorization and validation.

Persistence Layer: The persistence layer contains all the storage logic and translates business objects from and to database rows.

Database Layer: In the database layer, CRUD (create, retrieve, update, delete) operations are performed.

Spring Boot Flow Architecture



- Now we have validator classes, view classes, and utility classes.
- Spring Boot uses all the modules of Spring-like Spring MVC, Spring Data, etc. The architecture of Spring Boot is the same as the architecture of Spring MVC, except one thing: there is no need for DAO and DAOImpl classes in Spring boot.
- Creates a data access layer and performs CRUD operation.
- The client makes the HTTP requests (PUT or GET).
- The request goes to the controller, and the controller maps that request and handles it. After that, it calls the service logic if required.
- In the service layer, all the business logic performs. It performs the logic on the data that is mapped to JPA with model classes.
- A JSP page is returned to the user if no error occurred.

CHAPTER 5
ADVANTAGES AND DISADVANTAGES

5. Advantages and Disadvantages

Spring Boot Advantages

- Simplified & version conflict free dependency management through the starter POMs.

- We can quickly setup and run standalone, web applications and micro services at very less time.

- You can just assemble the jar artifact which comes with an embedded Tomact, Jetty or Undertow application server and you are ready to go.

- Spring Boot provides HTTP endpoints to access application internals like detailed metrics, application inner working, health status, etc.

- No XML based configurations at all. Very much simplified properties. The beans are initialized, configured and wired automatically.

- The Spring Initializer provides a project generator to make you productive with the certain technology stack from the beginning. You can create a skeleton project with web, data access (relational and NoSQL datastores), cloud, or messaging support.

Spring Boot Disadvantages

- Spring boot may unnecessarily increase the deployment binary size with unused dependencies.

- If you are a control freak, I doubt Spring Boot would fit your needs.

- Spring Boot sticks good with micro services. The Spring Boot artifacts can be deployed directly into Docker containers. In a large and monolithic based applications, I would not encourage you to use Spring Boot.

CHAPTER 6
CONCLUSION

6. Conclusion

The Spring boot software development framework is being widely accepted within the software development community. Spring boot provides multiple benefits over the previously used waterfall framework. Spring boot attempts to simplify the software planning and estimation process by decomposing large requirements into small individual tasks. Analyzing small tasks allow the software development team to more accurately predict the level of effort required in order to implement the change. This allows the project manager to accurately depict the percentage complete of the software which allows them to continually track overall project progress against the originally planned progress. The Spring boot process also is designed to help train developer in their schedule estimating skills throughout the lifecycle. For each task the developer should be required to make an estimation of how long they believe they will need to complete the task, after the task is completed they should enter in the actual time spent on the task. This will show the developer the delta between their estimation and their actual time spent.

CHAPTER 7
REFERENCES

7. References

- L. GuangChun, W. Lu, and X. Hanhong, *"A novel web application frame developed by MVC, "* ACM SIGSOFT Software Engineering Notes, vol. 28, no. 2, p. 7, 2003.

- J. LIU and J. DAI, *"Research of lightweight Web application based on Spring MVC and iBATIS frameworks [J], "* Journal of Computer Applications, vol. 4, p. 26, 2006.

- M. Fisher, J. Ellis, and J. Bruce, JDBC API tutorial and reference. Addison-Wesley Professional, 2003.

- D. Brown, C. M. Davis, and S. Stanlick, Struts 2 in action. Dreamtech Press, 2008.

- R. Johnson, J. Hoeller, A. Arendsen, C. Sampaleanu, R. Harrop, T. Risberg, D. Davison, D. Kopylenko, M. Pollack, T. Templier, and others, *"The spring framework-reference documentation, "* Interface21. (accessed 30. 04. 07), 2008.

- B. LIU and Z. WANG, *"Application of Office Automation Based on SSH Framework [J], "* Computer Technology and Development, vol. 1, p. 39, 2010.