



pointing the way

# White Papers

Spring

Java/JEE Application Framework



## Introduction

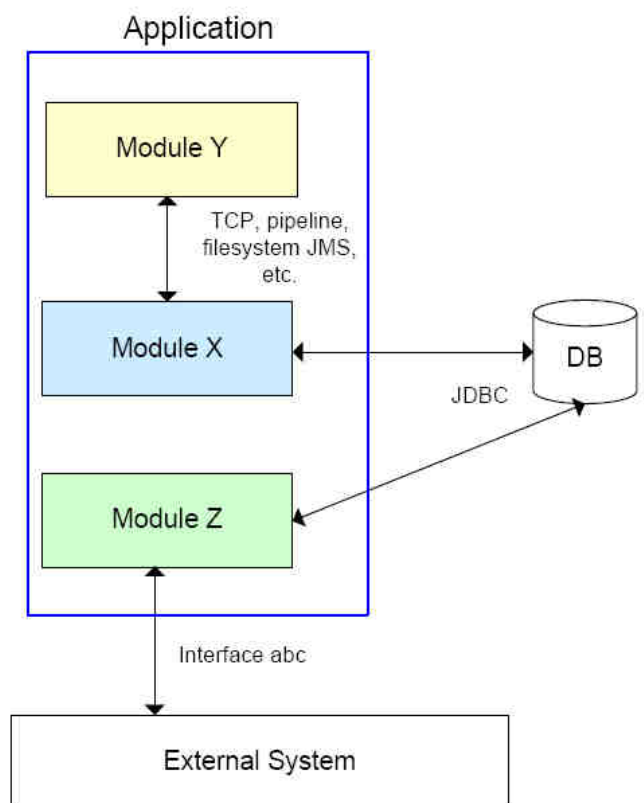
This paper describes basic concepts of Spring Framework. It shows main features and common use of Spring mechanisms.

## Background

Java Language released in 1995 introduced new application development concept WORA (Write Once, Run Anywhere). Java based applications were truly platform independent but unfortunately in its simplest form they did not have an application container or what we can call “application framework”. Experience shows that there are still business systems working with this not so well managed internal architecture.

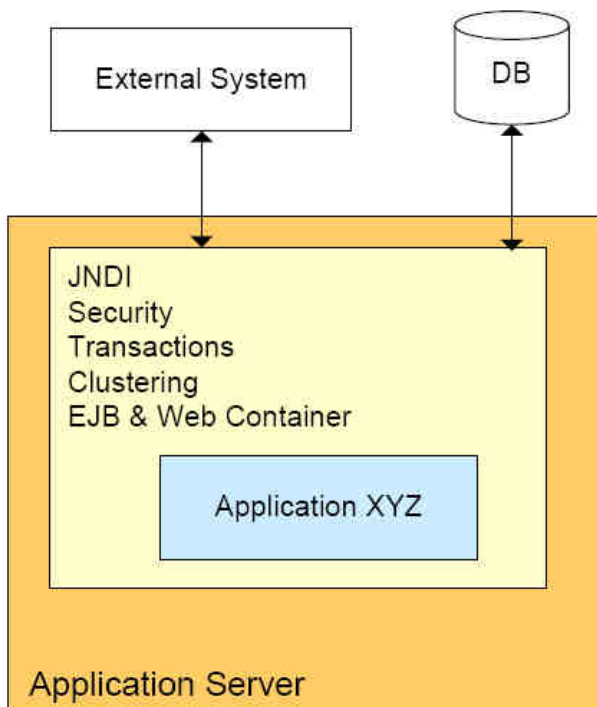
Picture shows that within one business application exist numbers of modules that are started separately (or one main module manages children threads providing a middle layer for internal services). This kind of architecture leads to use outdoor mechanisms in inefficient way.

Lack of integrated application environment (that we call “application container”) was filled with Application Server.



**Many application's modules interact using outdoor mechanisms like database, file system, different protocols.**

# Spring Framework



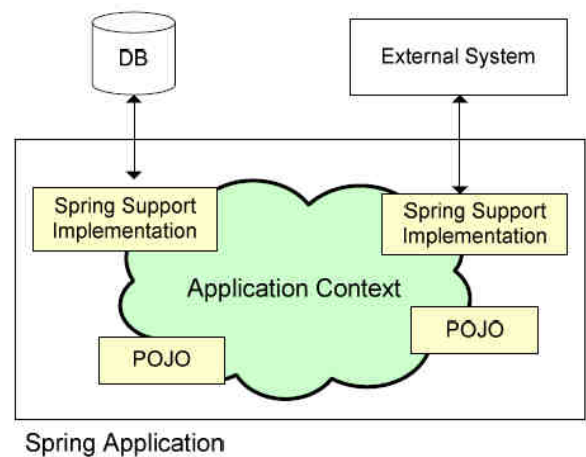
**Application server provides an enterprise application container with out of the box mechanisms.**

Application Server is a software engine that handles most of business logic and data access of the application. Typically it provides a middleware for dependent application intercommunication, centralized configuration, security, transactions, etc.

Application Server term sometimes refers to a J2EE or Java EE 5 application server due to success and popularity of Java platform.

Application Server provides an EJB (Enterprise JavaBeans) Container where application runs. Using JNDI (Java Naming and Directory Interface) it is possible to easy find and call other application's services.

EJB was quickly adopted by large companies but during development many problems appear. EJB specification (versions till 2.1) was very complicated and not intuitive for developers. Then was the time for lightweight application concept to evolve.



Spring is a lightweight application framework that in past few years becomes very popular. It is still growing and extended with new functionalities (i.e. Acegi is now a "Spring Security"). For Web applications Spring can run on servlet container such as Apache Tomcat.

## Spring Framework

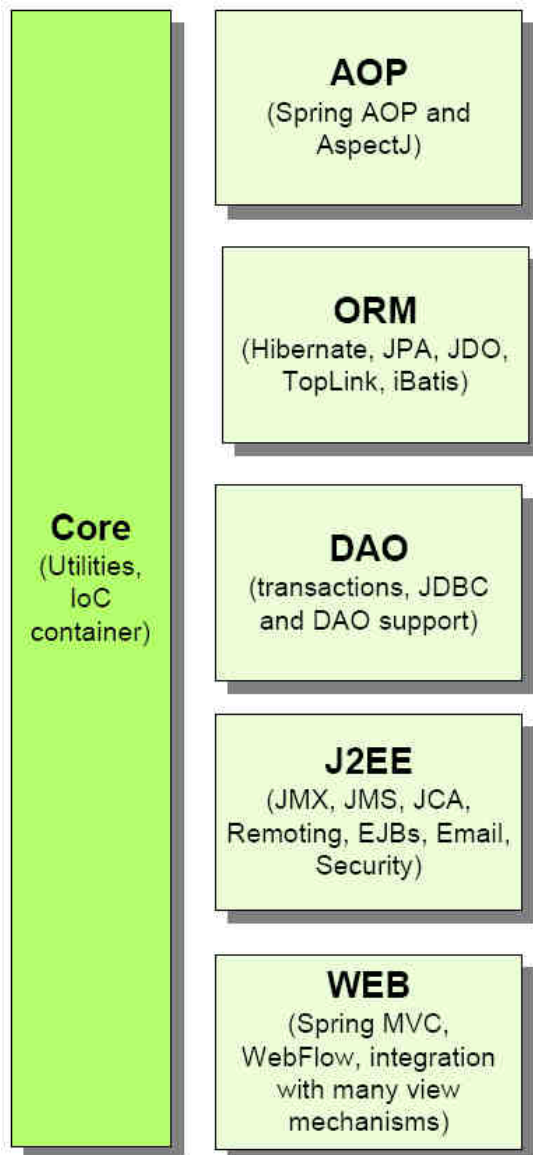
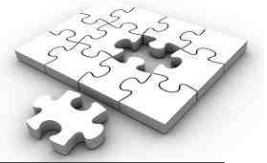
Spring Framework is much simpler than others J2EE solutions. Spring allows building applications from POJO (Plain Old Java Object) and in the same time it gives possibility to use more sophisticated services like transaction support, database access, security. Spring core feature like Dependency Injection makes application's modules more independent from each other. This approach significantly simplifies a unit testing. Also, by using Spring test support classes it is possible to made an integration test with ease.

## Components

Spring applications can use many different technologies and mechanism thanks to number of flexible Spring components.

## Core

Core module is a fundamental part of Spring Framework that provides the IoC and Dependency Injection features.



**Spring components are sufficient to build an advanced enterprise applications.**

## **AOP**

Aspect-Oriented Programming module allows defining method-interceptor and point cuts to decouple code implementation functionalities that logically should be separated.

## **ORM**

Object/Relational Mapping provides integration layer for popular APIs like JPA, JDO, Hibernate. It is possible to use those APIs with other Spring features like transaction management.

## **DAO**

Data Access Object module gives a JDBC abstraction layer that (from developer point of view) is database vendor independent.

## **WEB**

WEB module provides basic web-oriented integration features like application context or servlet listeners implementations. It integrates with many view options seamlessly: JSP/JSTL, Tiles, Velocity, FreeMarker, Excel, XSL, PDF, JSF, Tapestry, WebWork, Struts. Spring MVC (Model View Controller) provides a clean separation between domain model code and web forms.

## **J2EE**

J2EE component allows using Java Enterprise technologies such as JMX, JMS, JCA, EJB inside a Spring application.

## **Features**

Spring Framework is using existing and innovative solutions. It has a high quality code and it combines different ideas in one, agile framework.

## **IoC**

Inversion of Control Container is a core Spring element that allows to manage POJO and application context configuration. In IoC concept control flow is defined as desired responses linked to particular events or data requests. It improves code reusability.

## **Dependency Injection**

It is a type of IoC concept that manage beans dependencies inside a XML file. Java Class's setters and constructors are used to inject dependency by Spring Container. No other interfaces or middle-layer is needed to define Spring application context configuration.

## **AOP**

Aspect-Oriented Programming is used to implement additional behaviors or functionalities to existing classes without changes in code. AOP



mechanism is used implicate to manage transactions.

## Transactions

There are two kinds of transactions supported by Spring: global (JTA, managed by server) and local (used by JDBC interface, Hibernate, JDO). Spring creates an abstraction providing a coherent and declarative transaction model.

## Simplify use of J2EE interfaces

Using JNDI, JTA or EJB is simplified by eliminating code duplications (code that do not do any business logic) like handling exceptions, managing services context. By that Spring let developers to focus on essential application code.

## Remote Object Call

Spring allows to remote call POJOs using protocols like RMI, IIOP, Hessian or Burlap.

## JMX Integration

Java Management Extensions is a technology that contains tools for managing and monitoring applications. Spring transparently integrates into JMX infrastructure providing an automatic registration of any spring bean as a JMX Management Bean, flexible mechanism for controlling management interface of your beans, exposure MBeans over remote.

## Abstraction for data access

Spring Framework offers a coherent and very functional abstraction for data access. It allows creating framework-independent interfaces for persistence and later implementing them with chosen mechanism.

## MVC

Spring supports a Model-View-Controller internet architecture that is very well integrated with IoC Container. All Spring functionalities can be used in combination with other mostly common MVC frameworks: Struts, WebWork, Tapestry, JSF, etc. The newest Spring WebFlow implementation can be used to manage a page flow including advanced mechanism like AJAX or JSF (Spring Faces) integration.

## Testing

Support for broad testing strategies used by software developers including unit and integration tests.

## Examples

### Basic use of Dependency Injection:

1. "MyService" is a POJO service providing some behaviour.
2. "MyBean" is a POJO that uses a "MyService"
3. Application context defined in XML file contains a myService and myBean beans definitions:

```
<bean id="myService"
class="com.insono.MyService"/>
<bean id="myBean"
class="com.insono.MyBean">
<property name="service"
ref="myService" />
</bean>
```

MyService is injected into myBean bean. No special interfaces have to be created to define a beans dependency.

### AOP with BeanNameAutoProxyCreator:

```
<bean id="myAPC"
class="org.springframework.aop.framework
rk.autoproxy.BeanNameAutoProxyCreator"
>
```

BeanNameAutoProxyCreator is an Spring implementation used with AOP that identifies beans via list of names.

```
<property name="beanNames">
<list>
<value>bean1</value>
<value>bean2</value>
</list>
</property>
```

Those beans will have an associated interceptor myInterceptor.



```
<property name="interceptorNames">
<list>
<idref bean="myInterceptor" />
</list>
</property>
```

Every method of bean1 and bean2 will be intercepted and logic from myInterceptor will be executed before and/or after method invocation.

## Community

Spring Framework because of its popularity has a number of users and contributors. It is easy to find solutions and answers over Internet describing Spring features with some examples. Big community makes a development process even more efficient.

## Conclusion

Spring Framework provides a support for all business applications layers additionally it integrates with other single-layer frameworks like Struts or Hibernate. Spring let developers to take benefits from J2EE key solutions (JMS, JPA) at the same time Spring minimize complexity of applications. Spring Framework can be used to build not only an enterprise applications but rich client applications too (see Spring Rich Client Project). We would say that there is no such framework that Spring could be compared with. It

introduced a brand new approach to build business systems where term “enterprise application” does not have to mean “heavy” (when using Application Server) - Spring shows that “enterprise application” can mean: lightweight, simple and smart. In Insono, Spring Framework is a standard for creating advanced, business applications beside the newest EJB specification in version 3.0.

## Links & References

Rod Johnson, Juergen Hoeller, Alef Arendsen, Thomas Risberg, Colin Sampaleanu – “Spring Framework”, Helion 2006

Bruce Tate, Justin Gethland – “Spring. Zapiski Programisty”, Helion 2006

Comparing Web Frameworks

<https://equinox.dev.java.net/frameworkcomparison/WebFrameworks.pdf>

Common used frameworks - mailing list analysis

[http://raibledesigns.com/rd/entry/spring\\_mvc\\_the\\_most\\_popular](http://raibledesigns.com/rd/entry/spring_mvc_the_most_popular)

Spring Framework Introduction

<http://static.springframework.org/spring/docs/2.0.x/reference/introduction.html>