



The Spring Framework

Julien Dubois
France Regional Director
SpringSource

Julien Dubois

- France Regional Director, SpringSource
- Book author : « Spring par la pratique » (Eyrolles, 2006)
- 10 years of experience in Java development
- Member of OSSGTP



Agenda

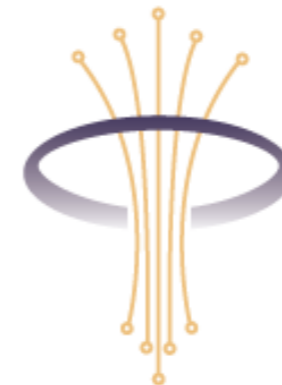
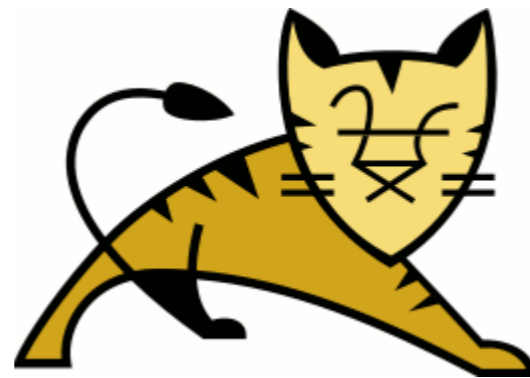
- SpringSource dm Server
- The Spring portfolio
- Demo

Agenda

- **SpringSource dm Server**
- The Spring portfolio
- Demo

SpringSource dm Server

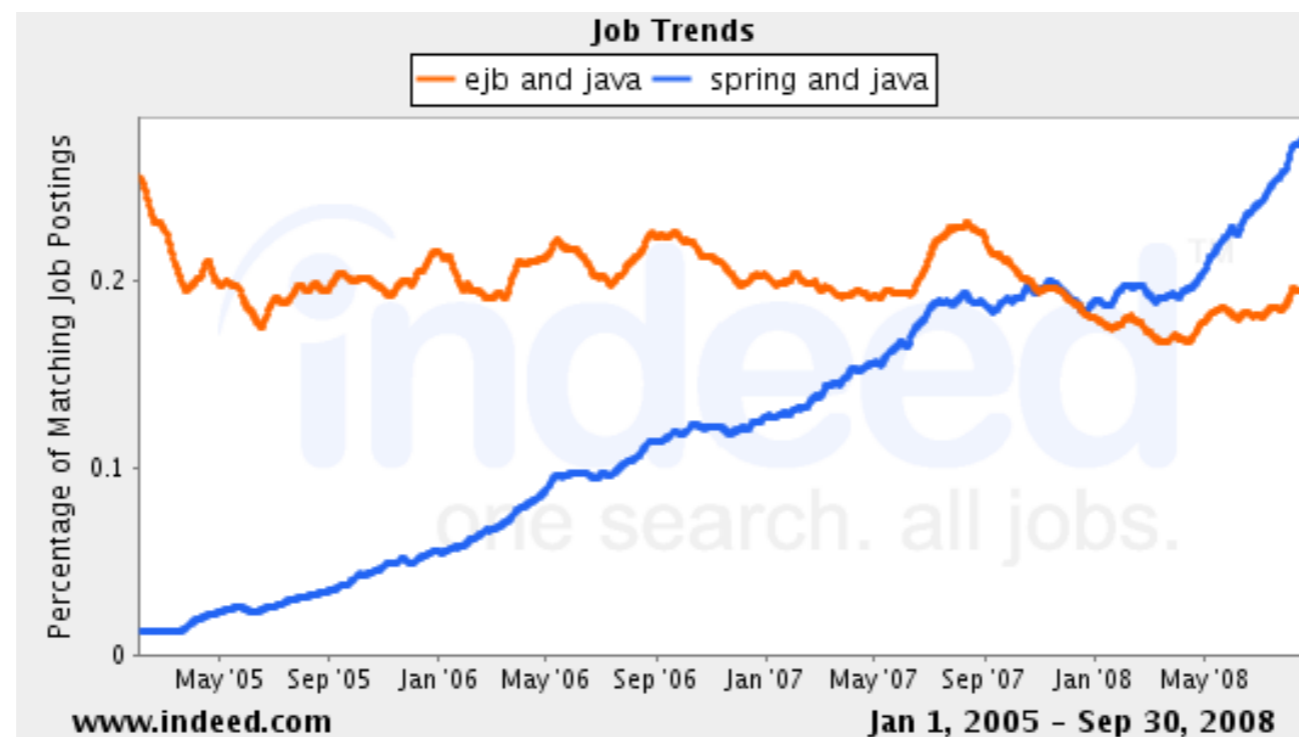
- The short version : it's Spring, Tomcat and OSGi working all together
 - Allows powerful and modular applications



OSGi™
Alliance
Member

What is Spring?

- Spring is the most popular Java enterprise framework.
 - More than 600,000 unique visitors on our website every month
 - More job offers than EJBs, and growing!



- Used in all industries, all over the world

What is Tomcat?

- Tomcat is the more popular Java application server
 - Used by many of the world's leading companies for running their Java applications
 - Open Source and lightweight alternative to complete J2EE and JEE servers
 - Led by the Apache Software Foundation



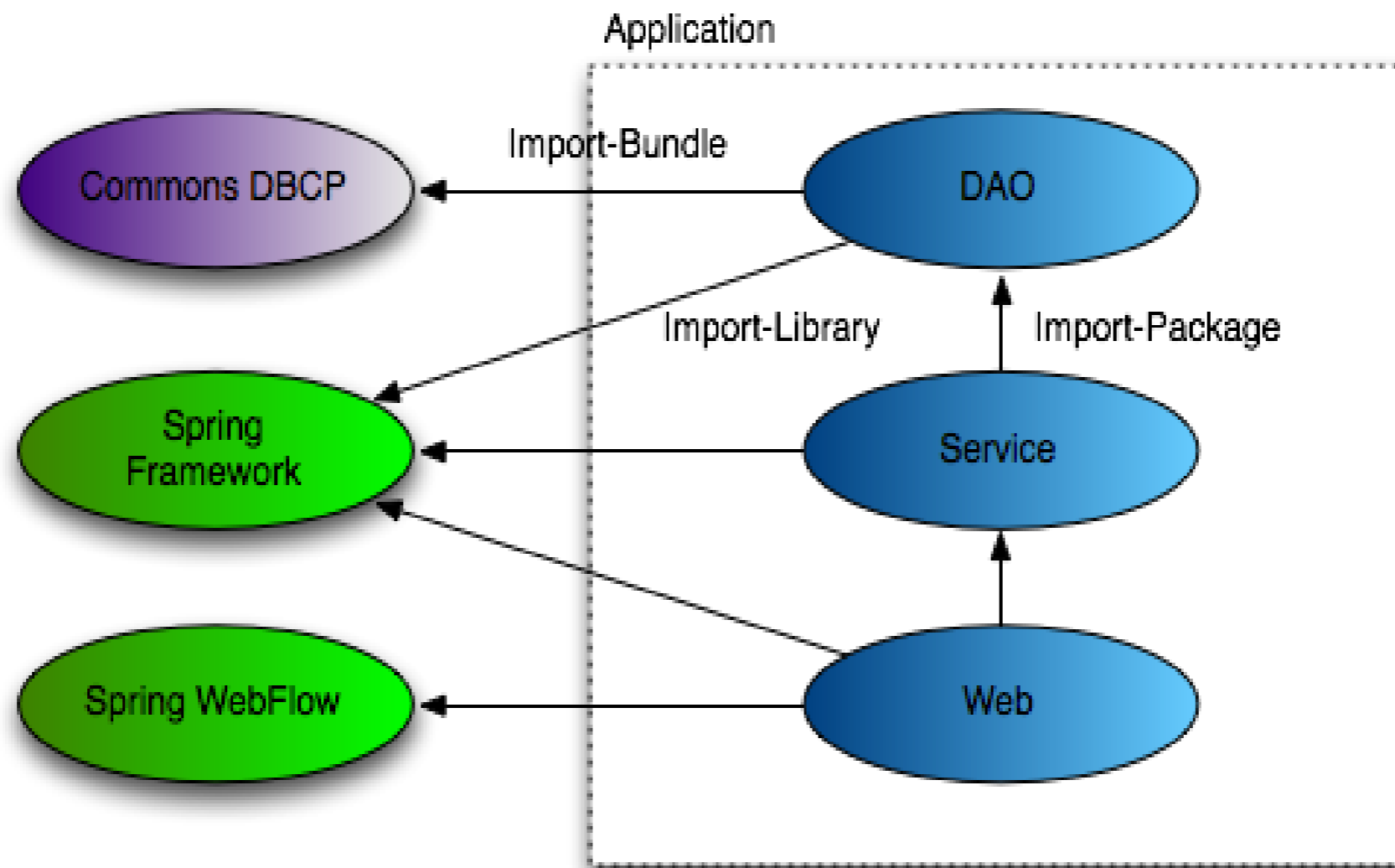
What is OSGi?

- OSGi is a specification
 - OSGi defines cleanly modules and their lifecycles
 - Used since many years in the telecom industry
 - With several implementations in Java : for instance Equinox, which is the basis of the Eclipse IDE



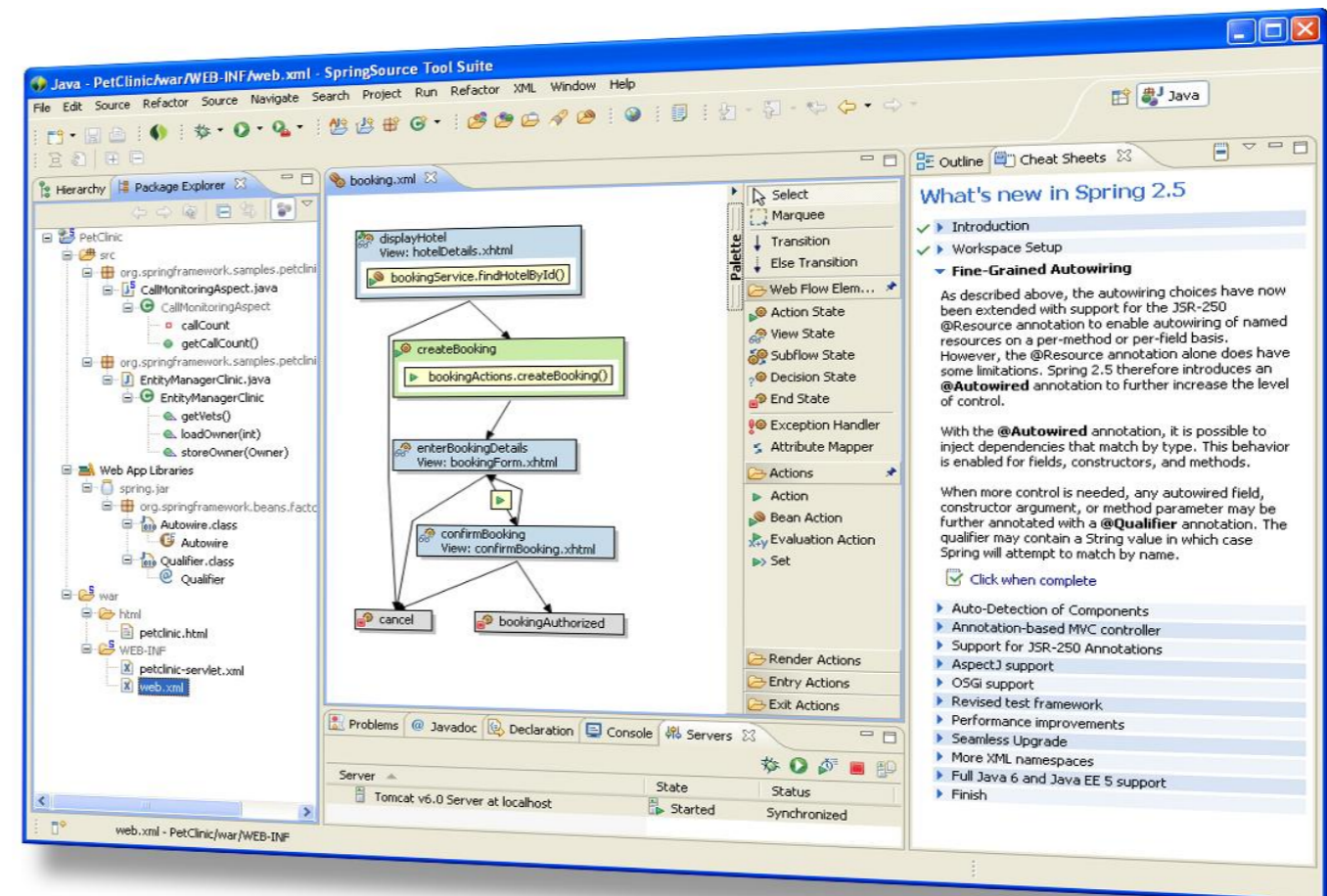
So, what does dm Server do?

- dm Server solves the “Jar hell” problem in your JEE applications :



Features for developers

- For developers :
 - A truly modular architecture : secured modules, with a well-defined lifecycle
 - Hot-deployment of modules during development time : you only work on your module, not on the whole application
 - The complexity of OSGi is hidden by dm Server : doing a modular application is easy, knowledge of the OSGi APIs and specificities is not a requirement



Features for production

- For the production team :
 - Hot deployment of modules of the application at runtime : lower downtime
 - Several versions of the same module can be deployed in parallel
 - Better usage of the server resources : dm Server only loads modules that are actually needed!



Summary

- dm Server is the only application server that gives the power of OSGi to development and production teams
- dm Server is built on standard, widely used, Open Source components : Spring, Tomcat, Equinox
- Of course, dm Server is free software (GPL v3), so why don't you start writing applications with it?



Agenda

- SpringSource dm Server
- **The Spring portfolio**
- Demo

The Spring Portfolio – Value across the application lifecycle



The Spring Portfolio

Groovy

Grails

Spring Web Services

Spring MVC

Spring Web Flow

Spring Dynamic Modules

Spring Framework

Spring Faces

Spring IDE

Spring Rich Client

Spring for .NET

Spring Security

AspectJ

Spring Integration

Spring Batch

Spring LDAP

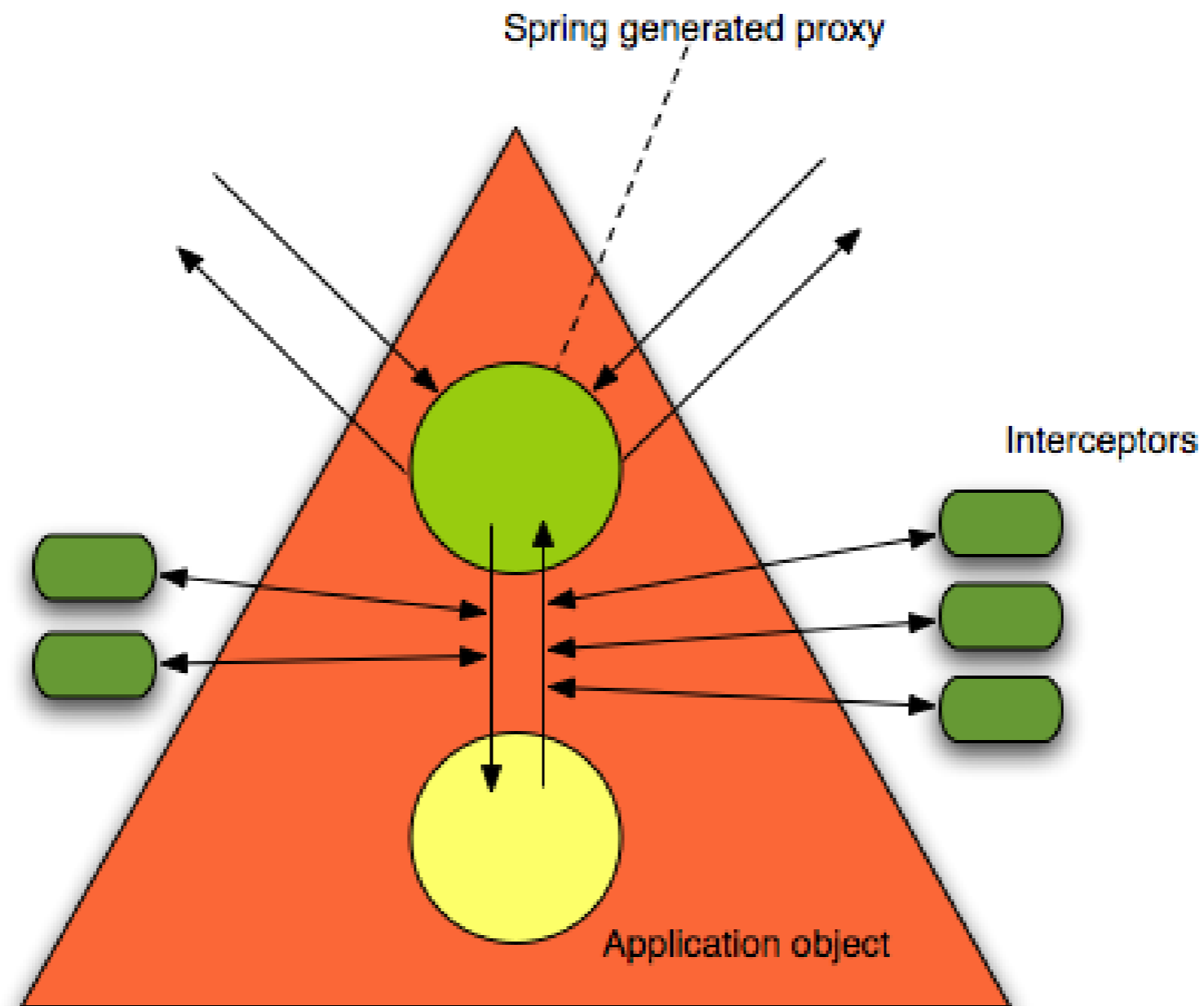
Spring as a platform

- Allows applications to be built from “Plain Old Java Objects” (POJOs)
- Allows enterprise services to be *applied* to POJOs in a non-invasive way

Examples of Spring as a platform

- Make a Java method execute in a database transaction
 - Without the implementer dealing with transaction APIs
- Make a local Java method a remote-procedure
 - Without the implementer dealing with remoting APIs
- Make a local Java method a management operation
 - Without the implementer dealing with JMX APIs
- Make a local Java method a message handler
 - Without the implementer dealing with JMS APIs

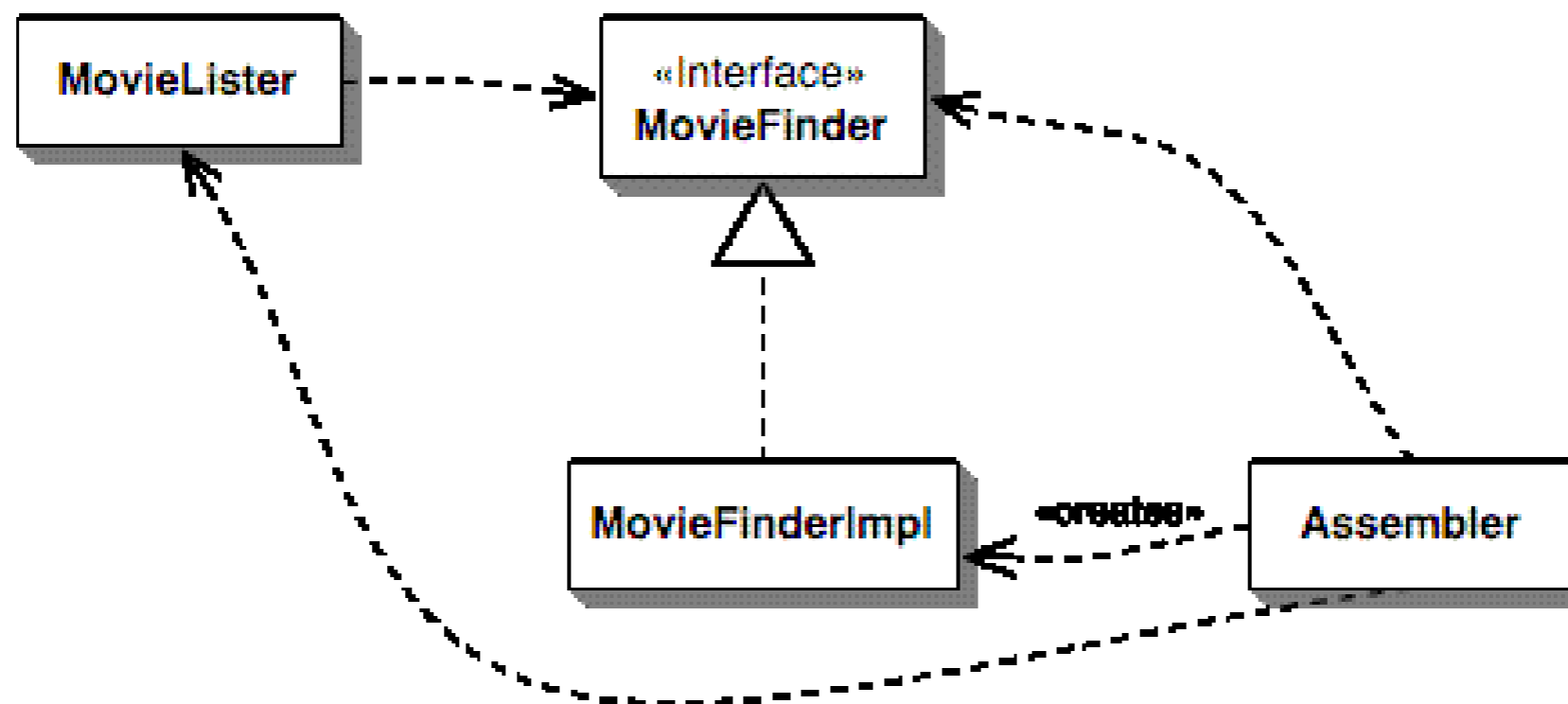
POJO Decoration



Spring managed component

Dependency Injection & Inversion of Control

- Allows a software component to express what it depends on to work
 - Without hard-coding
- A third party, called an assembler or container, is responsible for “plugging in” an implementation



It's All Metadata – Whether XML...

```
<beans>
  <tx:annotation-driven/>

  <bean id="rewardService" autowire="byType"
    class="example.SimpleRewardService"/>

  <bean id="accountRepository"
    class="example.JdbcAccountRepository"/>
    <property name="dataSource"
      ref="dataSource"/>
  </bean>
</beans>
```

It's All Metadata – ... or Annotations

```
public class MyService implements MyServiceInterface {  
    @Autowired  
    private DataSource dataSource;  
  
    @Autowired  
    public void setServiceA(ServiceA a) { ... }  
  
    @Transactional  
    @Secured("ROLE MANAGER")  
    public void businessMethod(String b) { ... }  
  
}
```

It's All Metadata – ... or JSR-250



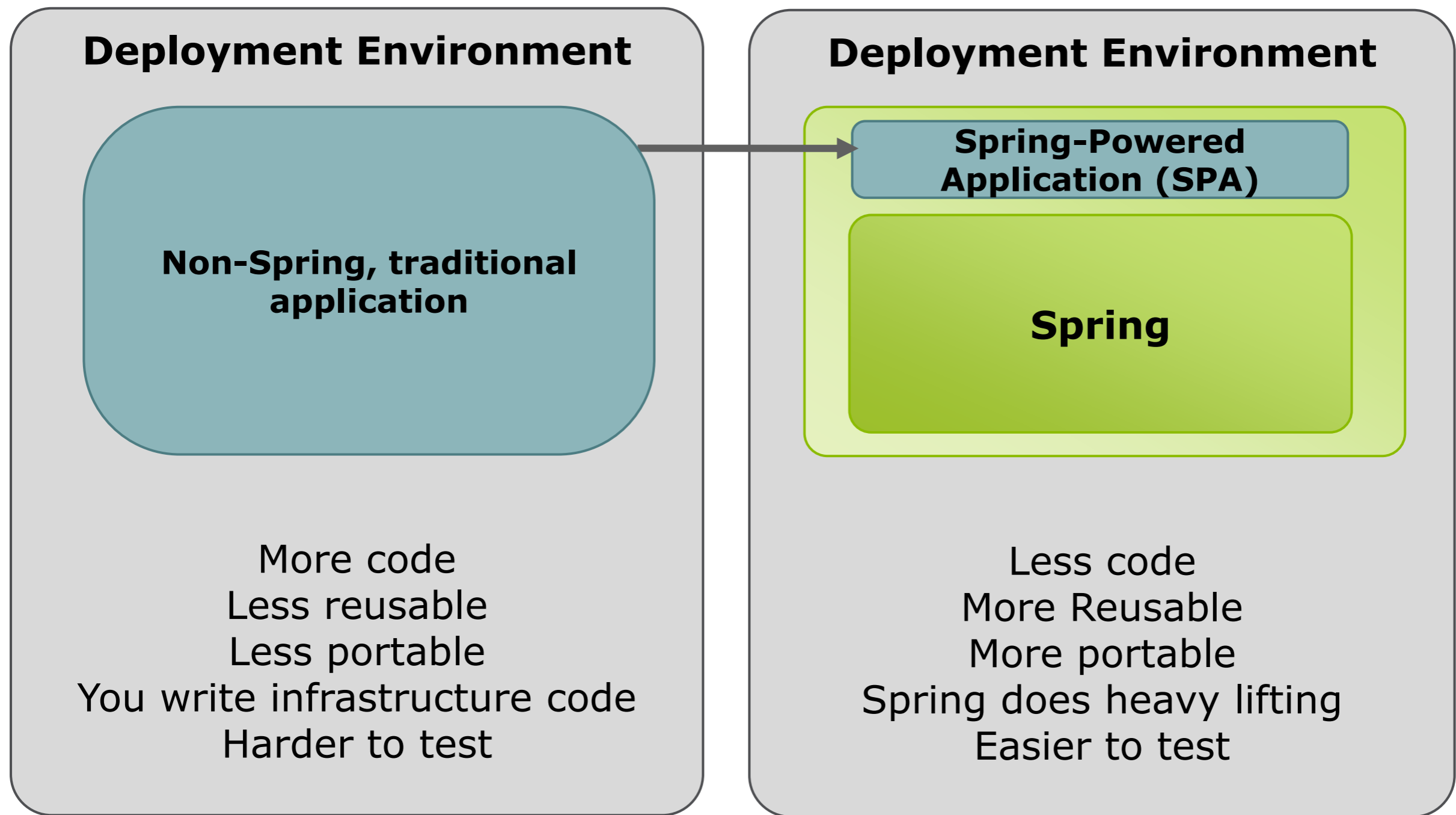
Annotations

```
public class MyService implements MyServiceInterface {  
    @Resource  
    private DataSource dataSource;  
  
    private Processor processor;  
  
    @Resource (name="myProcessor")  
    public void setProcessor (Processor processor) { ... }  
  
    @PostConstruct  
    public void initialize() { ... }  
  
    @PreDestroy  
    public void shutdown() { ... }  
}
```

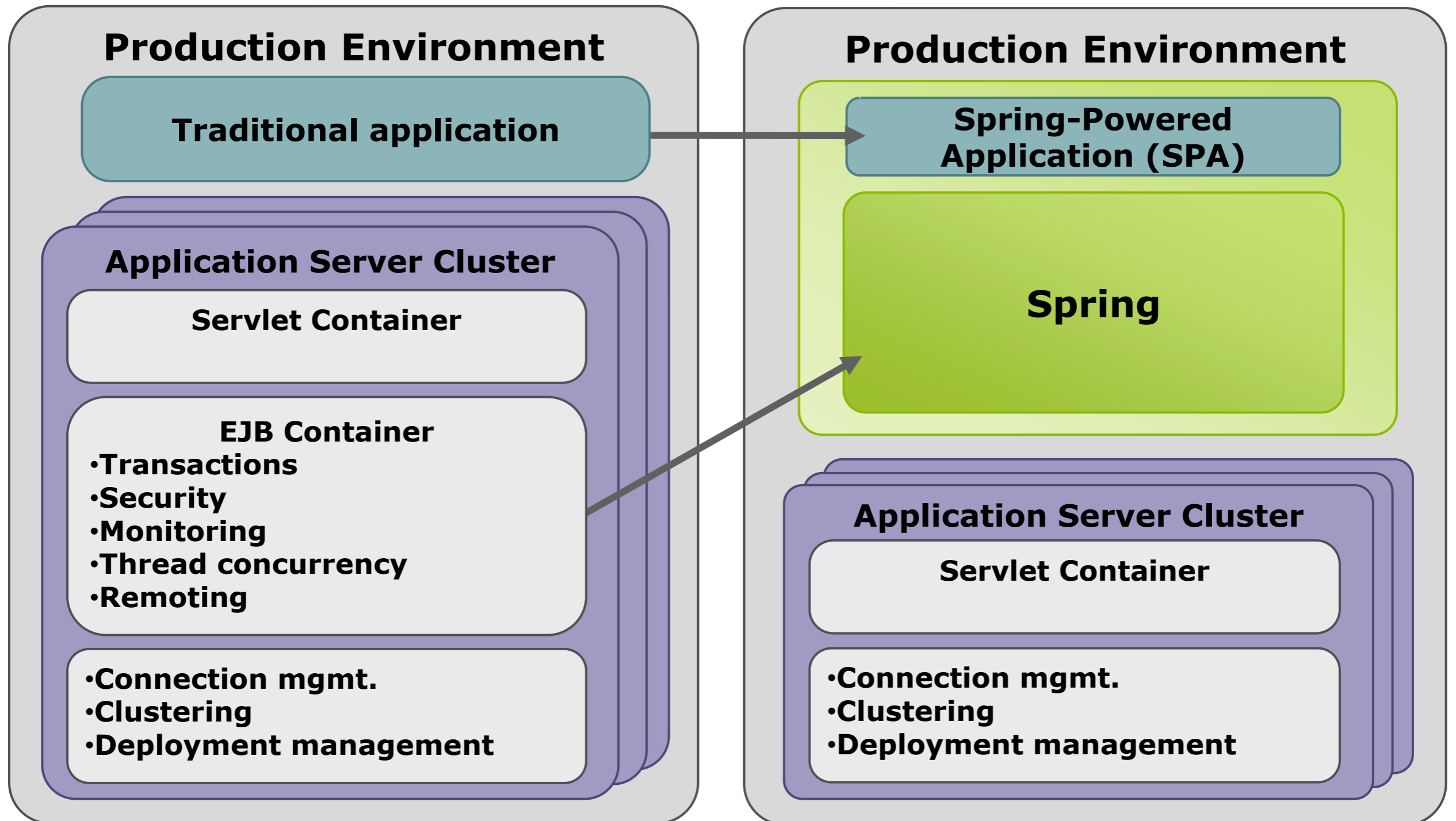
Major benefits of the POJO approach

- Leverage
 - Because you are not tied to an environment, your application code can run in *any* environment
- Isolation from volatile infrastructure
 - No longer at the mercy of app server release schedules
- Clarity
 - Application code stays focused on its core mission
 - Not bogged down with infrastructural concerns
- Testability
 - Simple objects with few dependencies are easy to test

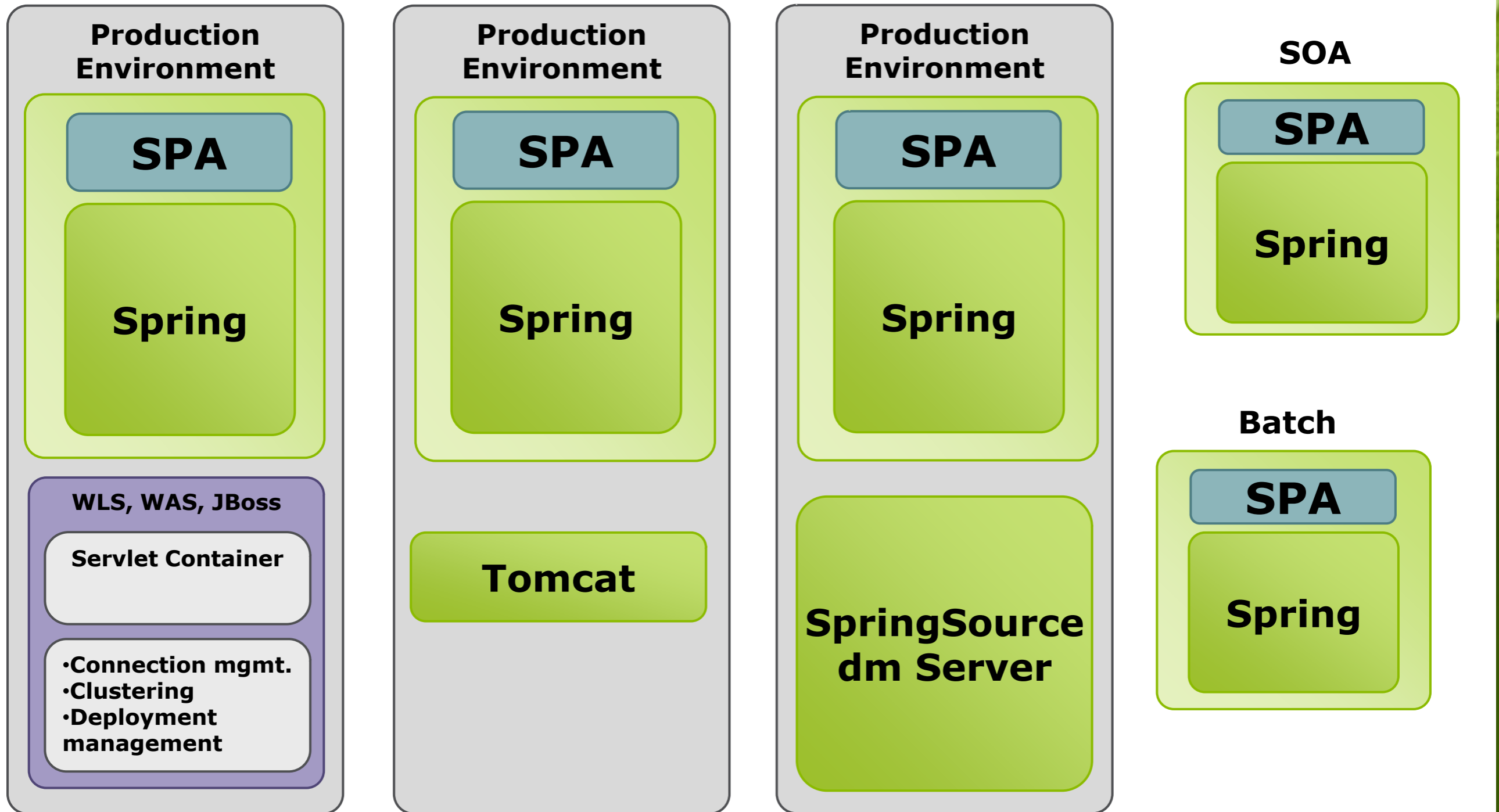
Spring Reduces Cost & Risk



Spring can replace major app. server run-time functionality



Portable across all deployment models



Demo

Questions?

Julien Dubois

julien.dubois@springsource.com

<http://www.springsource.com/fr>

