Spring Introduction

CS544: Enterprise Architecture

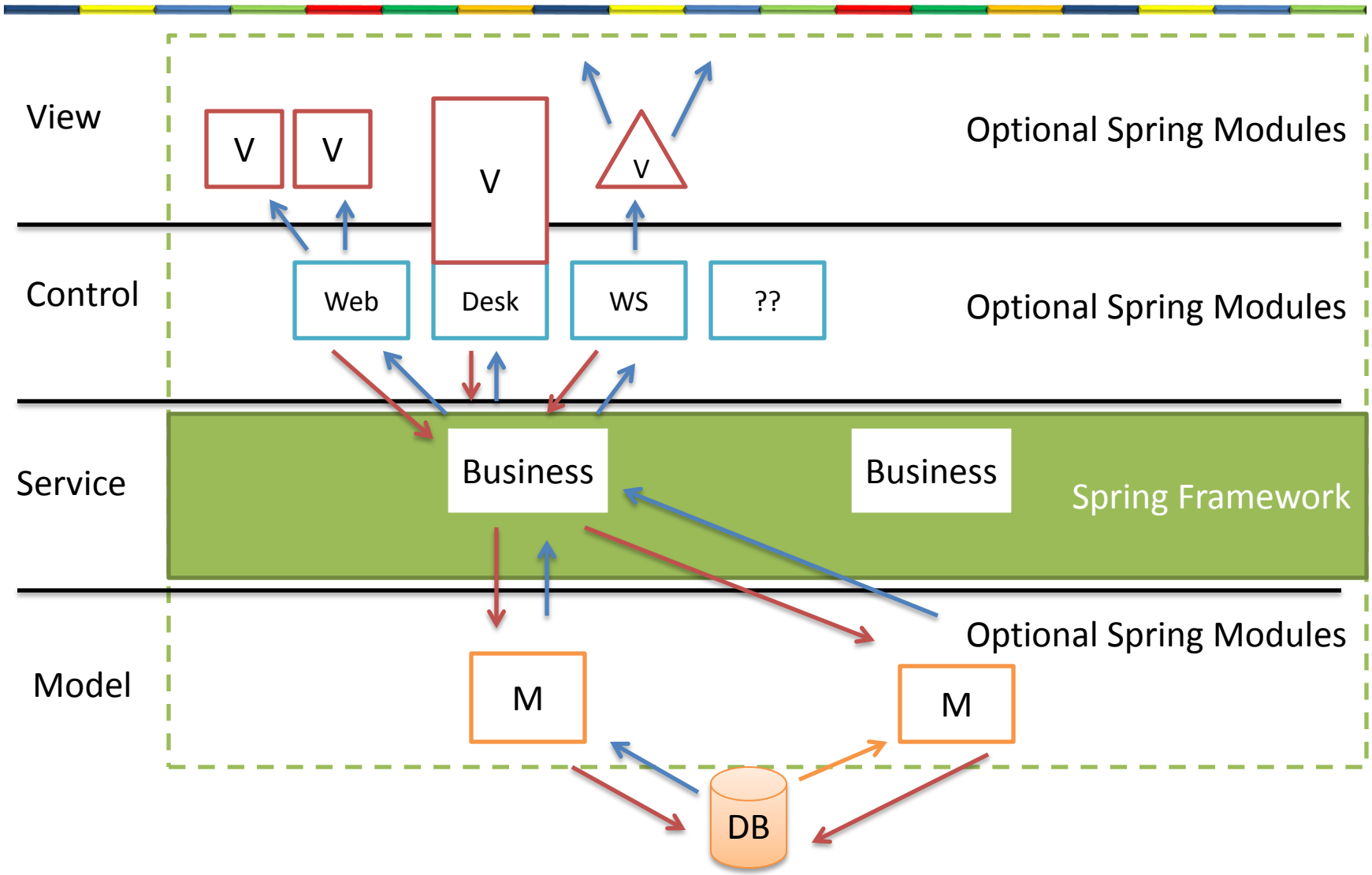
Introduction

- We will look at what Spring is and how it relates to the different application layers
- We will take a brief look at the history of Spring
- We will look at what the core elements of Spring are, and how they relate

Course Introduction:

SPRING

Framework for the Service Layer



History of Spring

- Started as alternative to EJB 2.1
  - Rod Johnson book: Expert One-on-one J2EE Design And Development
- EJB 3 Is like Spring / Hibernate
  - Spring moved ahead / not tied down by legacy
  - Spring community expanded beyond EJB
- Spring becomes another JEE implementation?

Aim of the Spring framework

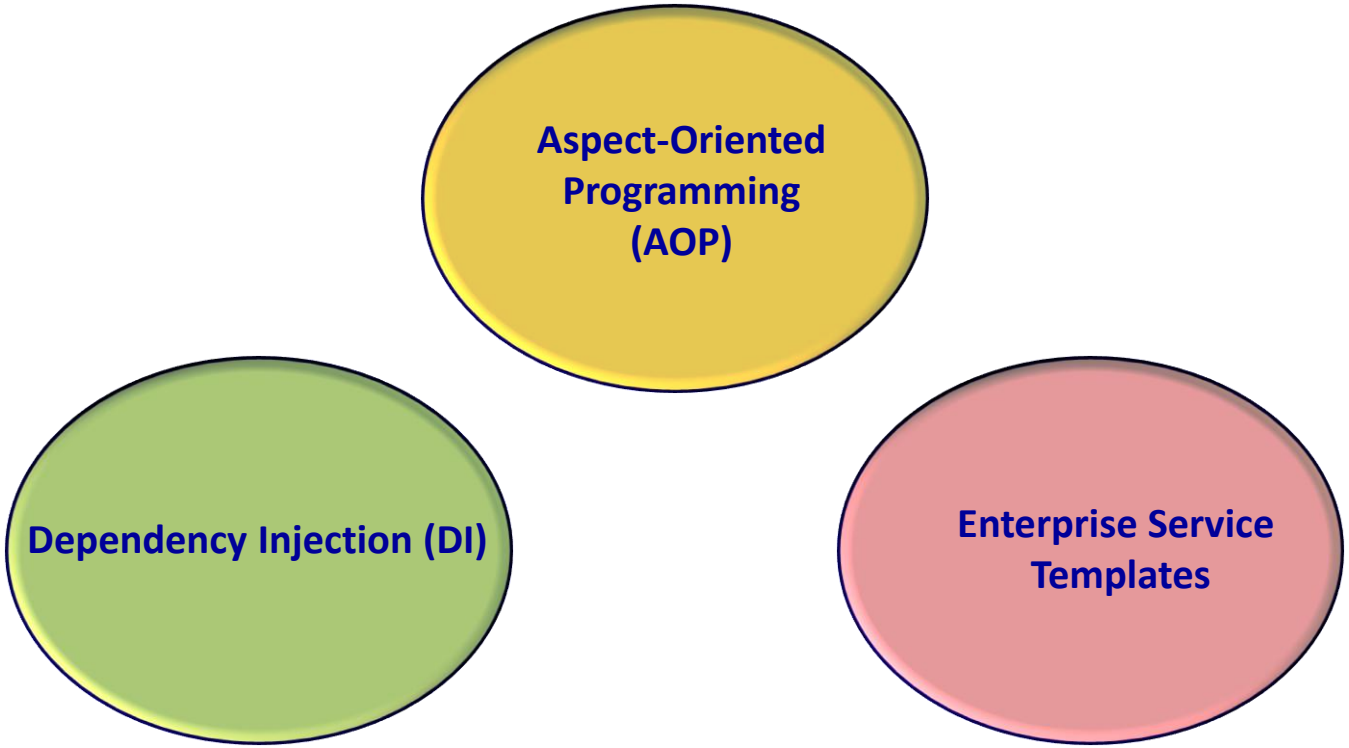
- Make enterprise Java application development as easy as possible, following good programming practices
  - POJO-based programming
  - Separation of concerns
  - Flexibility

6

POJO based programming

- All code is written in java objects
  - No EJB's
- Promotes Object-Oriented principles
- Simple to understand
- Simple to refactor
- Simple to unit test

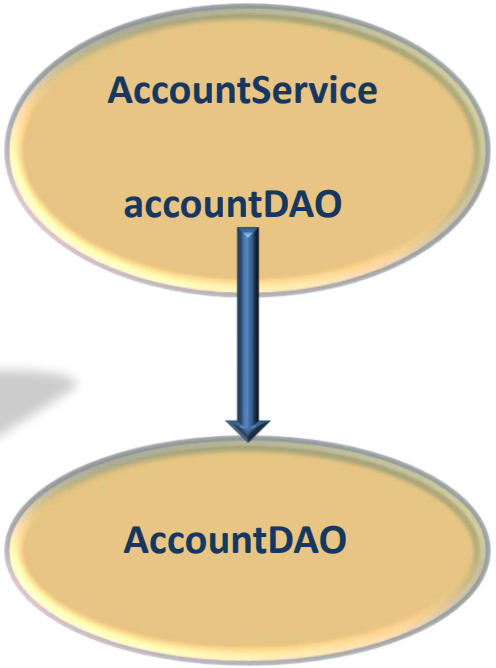
Core of Spring



Dependency Injection

- Spring instantiates objects and wires them together

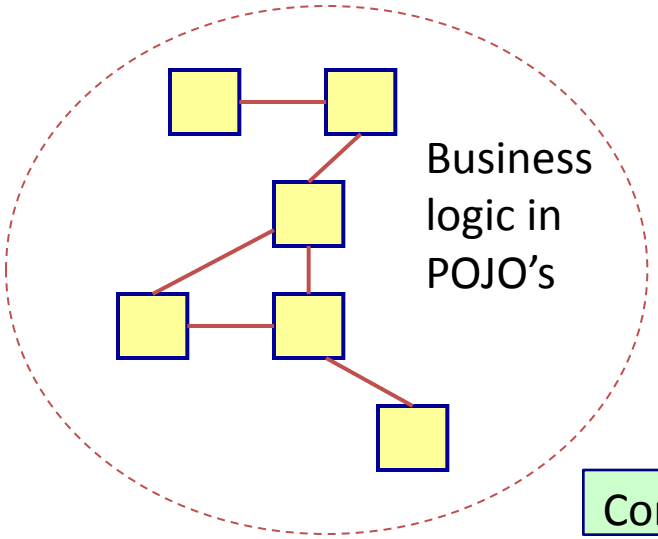
```
public class AccountService {
   private AccountDAO accountDAO;

    public void setAccountDAO(AccountDAO accountDAO) {
        this.accountDAO = accountDAO;
    }

    public Account getAccount(int accountNumber) {
        return accountDAO.loadAccount(accountNumber);
    }
}
```




```
<bean id="accountService" class="bank.AccountService">
  <property name="accountDAO" ref="accountDAO" />
</bean>
<bean id="accountDAO" class="bank.dao.AccountDAO" />
```

Aspect-Oriented Programming (AOP)

- Separate the crosscutting concerns (plumbing code) from the business logic code
- AOP development
  1. **Write the business logic without worrying about the enterprise services (security, transactions, logging, etc)**
  2. **Write the enterprise services**
  3. **Weave them together**



Configuration file

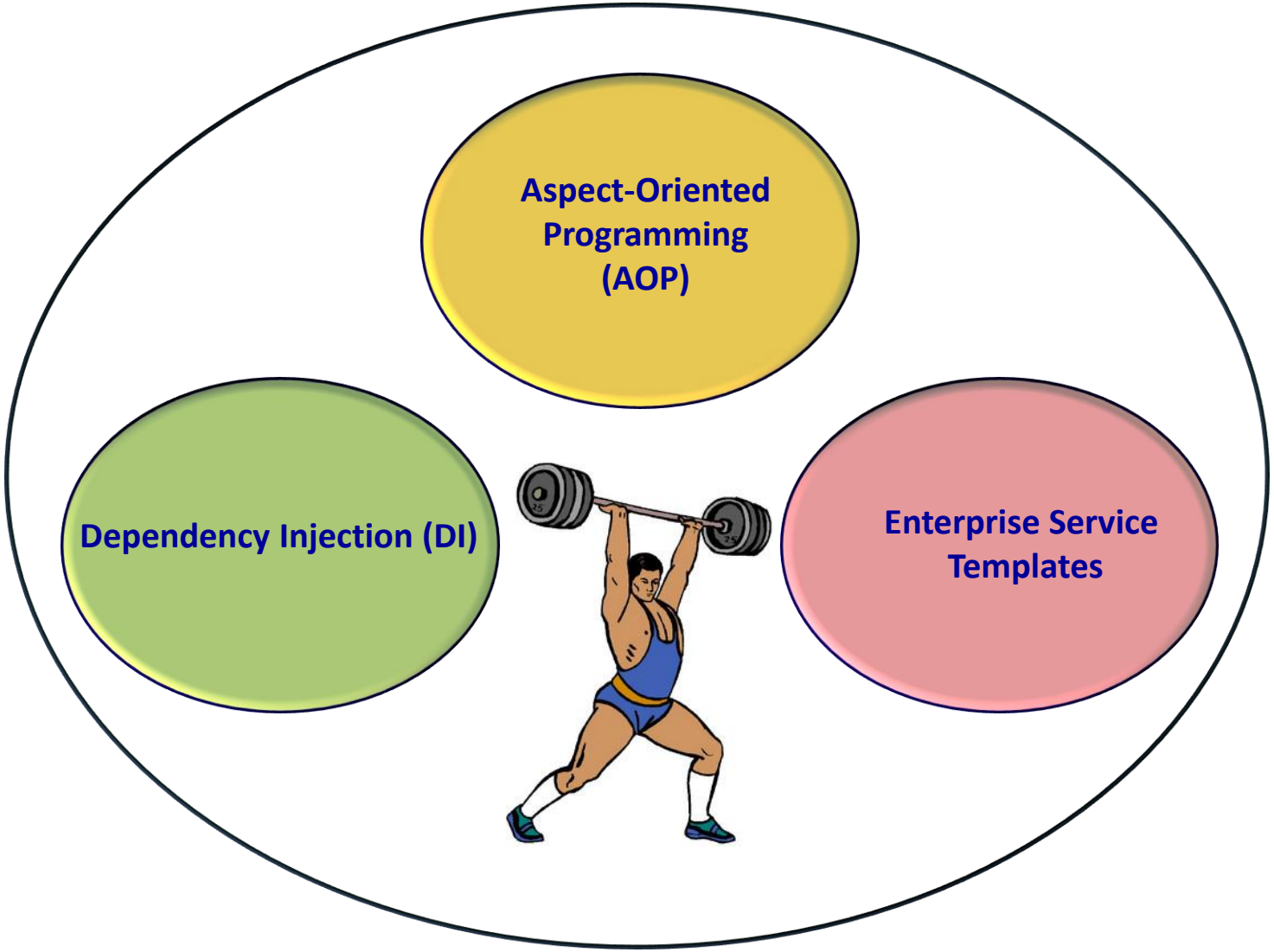
Enterprise services
- security
- transactions
- persistency
- messaging
- email

Enterprise Service Templates

- Makes programming the different enterprise service API's simpler.
  - JDBC template
  - JMS template
  - JavaMail template
  - Hibernate template
- Let the programmer focus on what needs to happen instead of complexity of the specific API
  - Resource management
  - Exception handling
  - Try-catch-finally-try-catch blocks

11

The power of Spring



12

Spring Portfolio

Spring Core

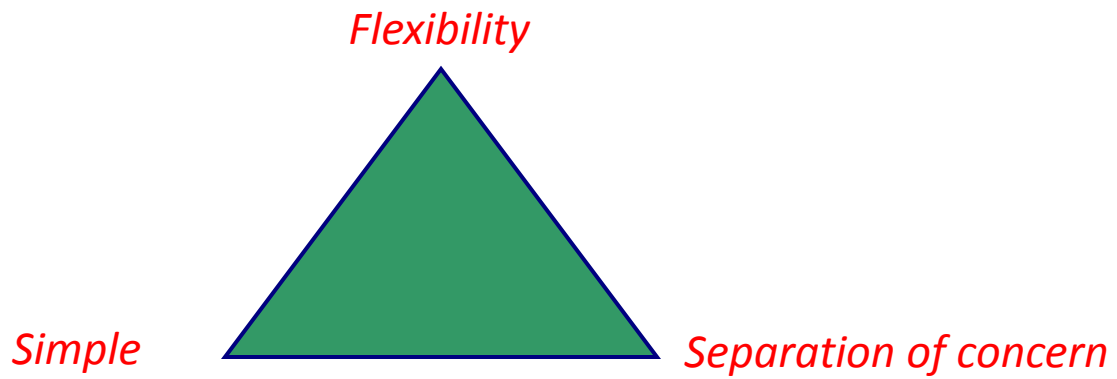
- **DI**
- **AOP**
- **Enterprise Service Templates**

- **Spring XD**
- **Spring Cloud**
- **Spring Data**
- **Spring Integration**
- **Spring Batch**
- **Spring Security**
- **Spring Social**
- **Spring Mobile**
- **Spring for Android**
- **Spring Web Flow**
- **Spring Web Services**
- **Spring LDAP**

http://spring.io/projects

Advantages of Spring

- Spring makes application development simple
  - POJO based programming
  - Simple coding of enterprise java API's
- Dependency injection gives flexibility in bean wiring
- AOP separates the business logic from the enterprise service (plumbing) code



Disadvantages of Spring

- Spring is another framework to learn
  - But, if you use another technique or another framework you have the same problem
- Spring is not a Java EE standard
  - But, the value of a standard is not that important anymore
    - Spring is more powerful than EJB (gives more options)
    - Spring has become an enterprise Java standard
- The XML file of Spring can become very complex
  - But: …
    - Spring also supports annotations
    - The Spring XML file is not that complex once you get used to it
    - The Spring XML file can be separated into multiple XML files
    - Spring also supports Java configuration

Active Learning

- What are the 3 main components of Spring?

- Why would you want to use Spring over standardized JavaEE EJBs?

Summary

- Spring makes developing enterprise Java applications simpler.
- Spring started as a replacement for EJB's, but has evolved to a framework that supports all different application layers
- The core of Spring consists of DI, AOP and enterprise service templates
- There are many additional projects in the Spring eco-system that easily integrate with Spring in a modular fasion.

Main Point

- Spring started as a framework for the service layer, but now has many other useful modules for other layers as well. The core of Spring's functionality consists of DI, AOP, and Templates
- *Science of Consciousness*: This is a 3 in 1 structure, similar to Rishi, Devata, Channdas