# 1 JAVA项目

## 1.1主流架构:

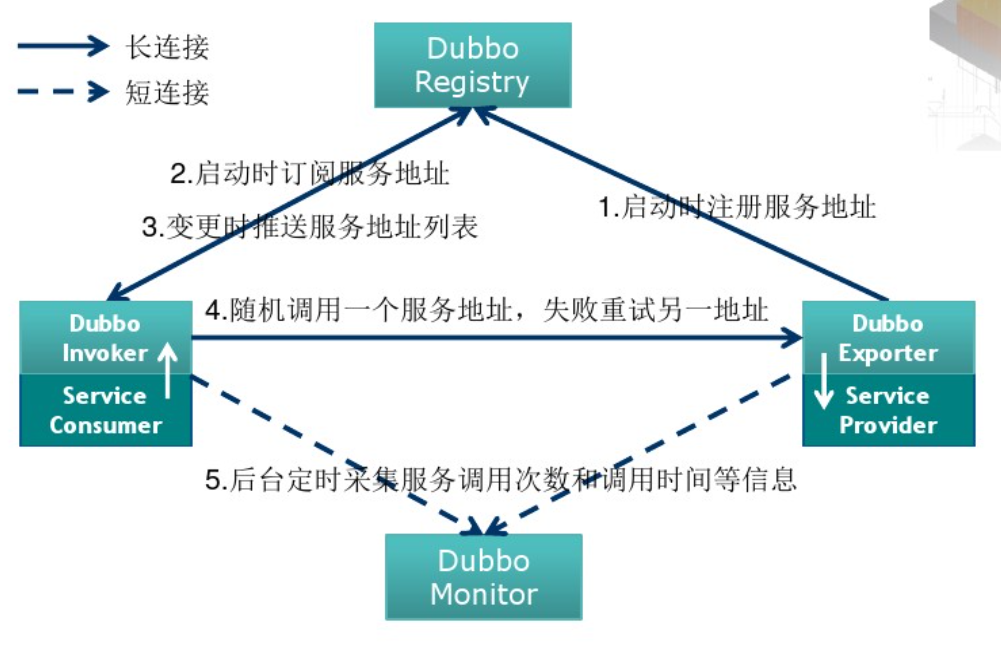
--spring boot 微服务的架构

--分布式+SOA面向服务的架构

本项目采用:分布式+SOA

SOA:Service Oriented Architecture面向服务的模式,就是将一个工程拆分为服务层和表现层,结合dubbo分布式构建

Dubbo结构:



本项目架构:



## 1.2项目结构

工具eclipse+maven

Maven:项目管理工具

功能: 1管理项目之间的依赖关系

2 统一管理jar包

项目结构:

Gtms 父工程，管理jar包的版本号。打包方式POM

|

|---api api接口父工程。打包方式POM

| |---sms-api api接口子工程。打包方式jar

| |---translator-api api接口子工程。打包方式jar

|

|---cbb 通用的类父工程。打包方式POM

| |---cbb 通用的工具子工程。打包方式jar

| |---entity 通用的实体子工程。打包方式jar

|

|---service 服务父工程。打包方式POM

| |---sms-service 服务子工程。打包方式war

| |---translator-service 服务子工程。打包方式war

|

|---war 表现层父工程。打包方式POM

|---gtms-service 表现层子工程。打包方式POM

maven管理的项目结构:

例:sms-sesrvice

|

|---java(code)

|---resources(配置文件)

|---test(测试用例)

|---test/resource

|

|

|---src

| |---main

| | |

| | |---webapp

| | | |--WEB-INF

| | | | |---web.xml

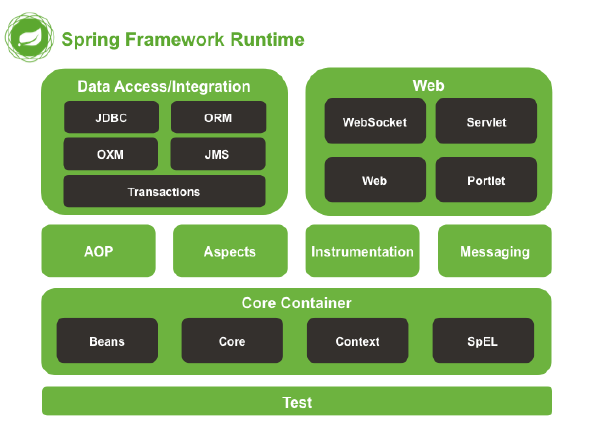
注意jar项目的更改需要install后update

## 1.3 spring

### Spring介绍

Spring它是一个一站式的分层轻量级框架。

### Spring体系结构



1. core container
   1. beans与core 它们提供spring框架最基本功能，包含ioc与di
   2. context 上下文对象，基于beans与cores
   3. spel它是sprng提供的一个表达式语言
2. Data access/integration
   1. 数据访问
   2. 集成
3. Web
   1. Spring本身提供spring mvc
   2. 也可以与其它的web层进行集成
4. AOP

AOP大部分情况下是使用动态代理来实现的。

1. Test

使用spring可以方便的进行测试

### Spring框架优点

* 方便解耦，简化开发

Spring就是一个大工厂，可以将所有对象创建和依赖关系维护，交给Spring管理

* AOP编程的支持

Spring提供面向切面编程，可以方便的实现对程序进行权限拦截、运行监控等功能

* 声明式事务的支持

只需要通过配置就可以完成对事务的管理，而无需手动编程

* 方便程序的测试

Spring对Junit4支持，可以通过注解方便的测试Spring程序

* 方便集成各种优秀框架

Spring不排斥各种优秀的开源框架，其内部提供了对各种优秀框架（如：Struts、Hibernate、MyBatis、Quartz等）的直接支持

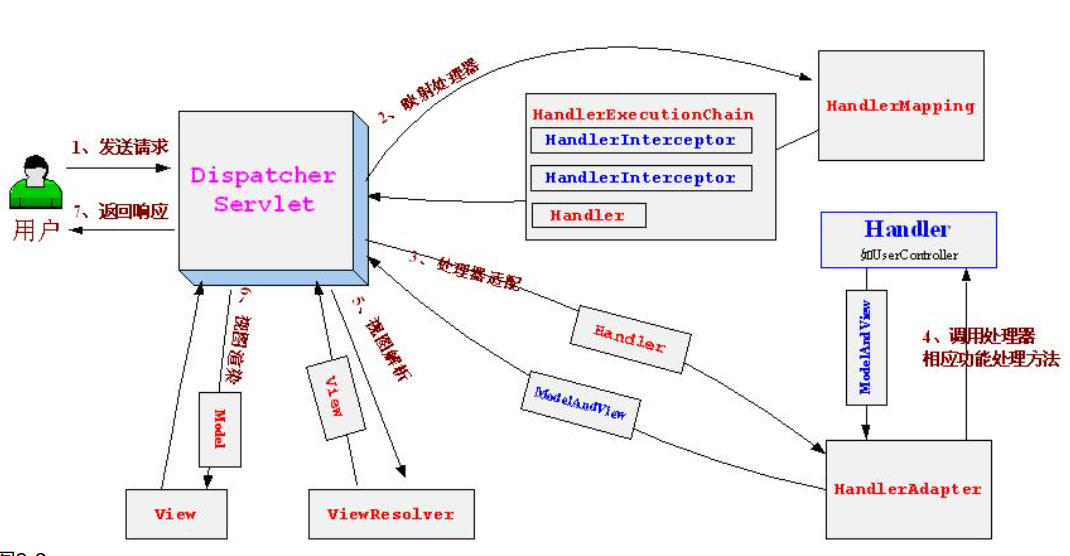
* 降低JavaEE API的使用难度

Spring 对JavaEE开发中非常难用的一些API（JDBC、JavaMail、远程调用等），都提供了封装，使这些API应用难度大大降低

## 1.4 spring MVC

Spring Web MVC是一种基于Java的实现了Web MVC设计模式的请求驱动类型的轻量级Web框架，即使用了MVC架构模式的思想，将web层进行职责解耦，基于请求驱动指的就是使用请求-响应模型，框架的目的就是帮助我们简化开发，Spring Web MVC也是要简化我们日常Web开发的。

Spring Web MVC也是服务到工作者模式的实现，但进行可优化。前端控制器是DispatcherServlet；应用控制器其实拆为处理器映射器(Handler Mapping)进行处理器管理和视图解析器(View Resolver)进行视图管理；页面控制器/动作/处理器为Controller接口（仅包含ModelAndView handleRequest(request, response) 方法）的实现（也可以是任何的POJO类）；支持本地化（Locale）解析、主题（Theme）解析及文件上传等；提供了非常灵活的数据验证、格式化和数据绑定机制；提供了强大的约定大于配置（惯例优先原则）的契约式编程支持。



具体执行步骤如下：

1、  首先用户发送请求————>前端控制器，前端控制器根据请求信息（如URL）来决定选择哪一个页面控制器进行处理并把请求委托给它，即以前的控制器的控制逻辑部分；图2-1中的1、2步骤；

2、  页面控制器接收到请求后，进行功能处理，首先需要收集和绑定请求参数到一个对象，这个对象在Spring Web MVC中叫命令对象，并进行验证，然后将命令对象委托给业务对象进行处理；处理完毕后返回一个ModelAndView（模型数据和逻辑视图名）；图2-1中的3、4、5步骤；

3、  前端控制器收回控制权，然后根据返回的逻辑视图名，选择相应的视图进行渲染，并把模型数据传入以便视图渲染；图2-1中的步骤6、7；

4、  前端控制器再次收回控制权，将响应返回给用户，图2-1中的步骤8；至此整个结束。

对于get请求中文参数出现乱码解决方法有两个：

修改tomcat配置文件添加编码与工程编码一致，如下：

修改tomcat目录 的conf/server.xml文件，添加URIEncoding="utf-8"

<Connector URIEncoding="utf-8" connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>

另外一种方法对参数进行重新编码：

String userName = new

String(request.getParamter("userName").getBytes("ISO8859-1"),"utf-8");

ISO8859-1是tomcat默认编码，需要将tomcat编码后的内容按utf-8编码

## 1.5 mybatis

作用：封装了JDBC操作，简化数据库访问代码。  
封装的功能：  
1.获取连接，执行SQL，释放连接  
2.SQL参数设置（可以直接传入对象，Mybtis会将对象的属性传入SQL语句）  
　　#{属性值}取代JDBC的？占位符  
3.执行结果映射成实体对象。JDBC中需要开发者自己转换。  
实体类的属性名与查询结果集ResultSet的列名保持一致，结果集有别名的话以别名为属性名。  
开发者工作：写SQL语句和实体类，然后使用SqlSession对象执行SQL操作。