# 前言

Dubbo是阿里巴巴开源的一款分布式服务框架，致力于提供高性能和透明化的RPC远程服务调用方案，以及SOA服务治理方案(主要是这两个功能)。Dubbo采用Spring配置方式集成，这样避免了代码侵入

# 概述

## 1.1什么是RPC

我们先来看看什么是RPC，RPC全称为Remote Procedure Call，翻译过来为“远程过程调用”， 也就是说假设有两台服务器A，B，一个应用部署在A服务器上，想要调用B服务器上应用提供的函数**/**方法，由于两个应用不在同一个内存空间内运行，因此不能直接调用，需要通过网络来表达调用的语义和传达调用的数据，这就是所谓的RPC

## 1.2什么是SOA

SOA是面向服务的架构，即把系统分离成不同的服务，服务和服务之间使用接口来进行数据交互，最终达到整合系统的目的。

举一个简单的例子：

程序员A负责考勤模块的开发，程序员B负责薪资模块的开发

B：“A，我要用考勤的数据，你考勤模块做好了吗，数据录了吗？”

A：“早就做好了，考勤表叫\*\*\*，\*\*字段代表员工id，\*\*字段代表员工出勤天数，\*\*字段代表..”

这是我们传统开发中日常出现的场景，模块之间的数据交互通过数据库进行，A模块存入考勤数据，B模块读取考勤数据并处理，模块与模块之间耦合。

那么SOA是怎么处理的呢？

考勤作为单独模块，成为一个考勤服务，发布了一个考勤数据接口（WebServices），向外提供查询考勤数据的接口

B需要使用考勤数据，只需要调用考勤服务的接口即可，这样就实现了模块分离，模块间要进行数据交互，通过接口来完成，这就是SOA

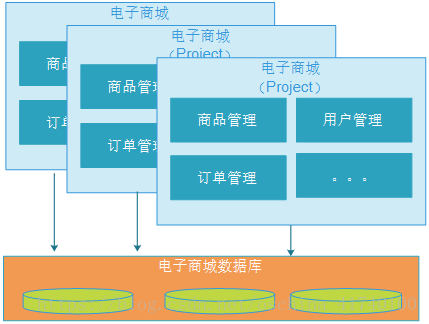
那么SOA有哪些好处呢？

1. 模块间解耦，B只需要调用A的接口就能得到想要的数据，并不关心A是如何获取数据的，这样一来即使更改数据库的表结构，只需要重写A的接口即可(试想一下如果多个模块需要A的接口，如果没有SOA，那么多个模块都得重写接口)
2. 拆分系统更为方便，在用户量越来越大的情况下，数据库服务器和应用服务器的压力也越来越大，解决方法是拆分系统为几个子模块，子模块分别使用不同的数据库服务器和应用服务器(分布式)
3. 可扩展性强，当功能被严格划分后，新的功能有时可以通过原有功能的拆分重组实现，就像搭积木一样，把积木拆了就可能重新搭出新的积木模型

## 1.3 为什么需要dubbo

网站的架构一般要经历如下的发展过程

* **单一应用架构**



**特点：**

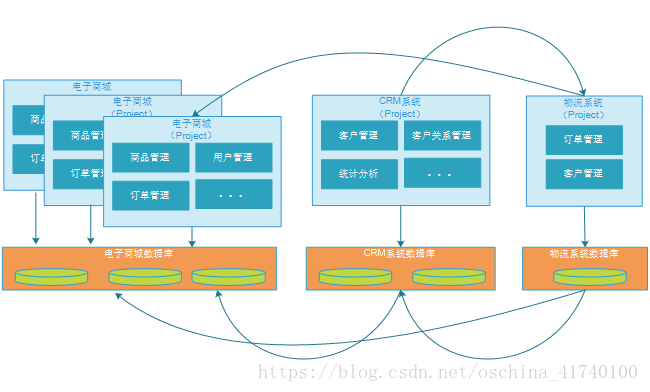
* + 当网站流量很小时，只需一个应用，将所有功能都集成在一个项目中，打成一个war包部署到服务器(可能是多个服务器通过负载均衡技术进行集群部署)，以减少部署节点和成本。
  + 应用和数据库分开部署到应用服务器和数据库服务器上
  + 此时，用于简化增删改查工作量的**数据访问ORM框架(如Mybaits)**是关键。
  + 可以通过部署应用集群和数据库集群来提高系统的性能。

**优点：**

* + 项目架构简单，前期开发成本低，周期短，小型项目的首选。

**缺点：**

* + 功能集成在一个项目，耦合度高，不利于开发和扩展。
  + 系统性能扩展只能通过增加集群结点，成本高、有瓶颈。
  + 必须采用单一技术栈进行开发
* **垂直应用架构**



**特点：**

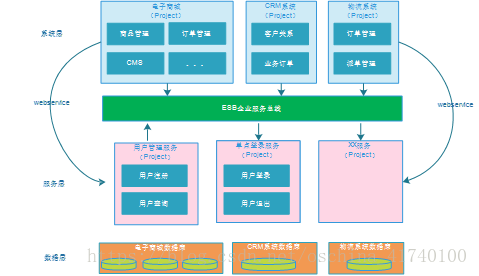
* + 当访问量逐渐增大，单一应用增加机器带来的加速度越来越小(一般来说流量洪峰过去后，大量部署的服务器就会出现资源浪费)，将大应用垂直拆成互不相干的几个单体应用，以提升效率。如上图将商城项目中的用户关系管理(CRM)和物流系统拆分出来
  + 此时，用于加速前端页面开发的**Web框架(MVC)**是关键
  + 项目与项目之间的存在数据冗余，耦合性较大，比如上图中三个项目都存在客户信息。

**优点：**

* + 项目架构简单，前期开发成本低，周期短，小型项目的首选。
  + 通过垂直拆分，对原项目进行了一定程度上的解耦
  + 不同项目可以采用不同技术栈开发

**缺点：**

* + 耦合度仍然过高，不利于开发和扩展。
  + 系统性能扩展只能通过增加集群结点，成本高、有瓶颈。
* **分布式服务架构**



**特点：**

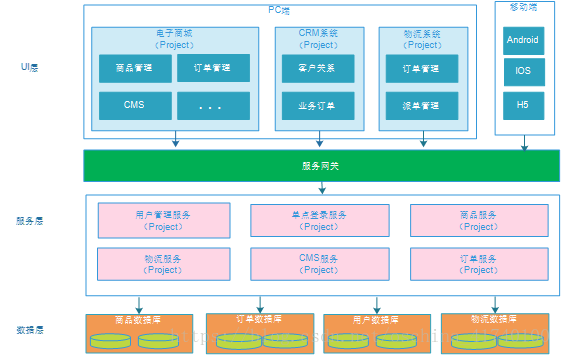
* + 当垂直拆分出来的应用越来越多，应用之间交互不可避免，这个时候将多个应用中的核心业务功能抽取出来，形成一个个公用的服务，逐渐组成稳定的服务中心，给上层应用提供服务，使上层应用能更快速的响应多变的市场需求，上层应用直接调用公共服务，提高复用率。
  + 各项目（系统）与服务之间采用webservice、rpc等方式进行通信。
  + ESB企业服务总线(绿色)作为项目与服务之间通信的桥梁。
  + 此时，用于提高业务复用及整合的  **分布式服务框架(RPC)**  是关键。

**优点：**

* + 将重复的功能抽取为服务，提高开发效率，提高系统的可重用性、可维护性。
  + 可以针对不同服务的特点制定集群及优化方案。
  + 采用ESB企业服务总线减少接口耦合

**缺点：**

* + 系统与服务的界限模糊，不利于开发及维护。
  + 虽然使用了ESB，但是服务的接口协议不固定，种类繁多，不利于系统维护。
  + 抽取的服务的粒度过大，系统与服务之间耦合性高。
* **微服务架构**



* + 当服务越来越多，各个服务占有的资源越来越难以把控，比如一些压力小的服务，可能占据了大量硬件资源，我们需要基于访问流量实时监控和管理各个服务
  + 将系统服务层完全独立出来，并将服务层抽取为一个一个的微服务。这些服务的创建仅限于一个特定的业务功能，如用户管理、用户角色、电子商务车、搜索引擎、社交媒体登录等。此外，它们是完全独立的，也就是说它们可以写入不同的编程语言并使用不同的数据库。
  + 微服务遵循单一原则，即单个服务功能完整，职责单一。
  + 微服务之间采用RESTful等轻量协议传输。

<http://www.cnblogs.com/pflee/p/4507579.html>

## 1.4 dubbo的运行流程

在了解如何使用Dubbo之前，我们首先了解一下Dubbo中的几个角色：

**运行容器（Container）：**负责启动，加载，运行服务提供者(provider)，可以是tomcat等

**服务器注册中心（Register）：**服务注册的中心，服务提供者需要向注册中心注册自己提供的服务

**服务提供方（Provider）：**服务提供方暴露服务，将服务提供给消费方使用

**服务消费方（Consumer）：**调用远程服务的服务消费方

**统计监控中心（Monitor）：**统计服务的调用次调和调用时间的监控中心



0、服务容器负责启动，加载，运行服务提供者。

1、服务提供者在启动时，向注册中心注册自己提供的服务。

2、服务消费者在启动时，向注册中心订阅自己所需的服务。

3、注册中心返回服务提供者地址列表给消费者，如果有变更，注册中心将基于长连接推送变更数据给消费者。

4、服务消费者，从提供者地址列表中，基于软负载均衡算法，选一台提供者进行调用(因为该服务可能是集群部署)，如果调用失败，再选另一台调用。

5、服务消费者和提供者，在内存中累计调用次数和调用时间，定时每分钟发送一次统计数据到监控中心。

# Dubbo的使用

## 2.1 HelloWorld

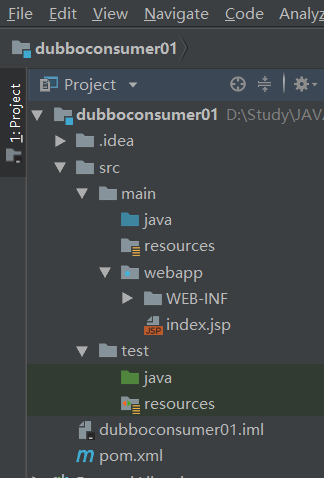
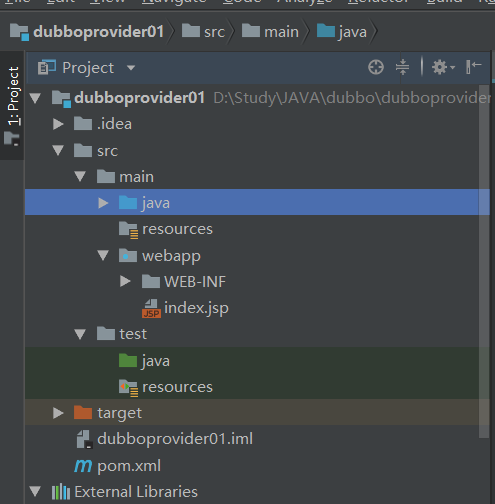
### 2.1.1 不使用注册中心

在开发和测试环境下，为了方便起见，我们常常绕过注册中心(生产环境中不允许)，直接连接到服务提供者，这就是无注册中心的点对点直连，我们先从这里开始

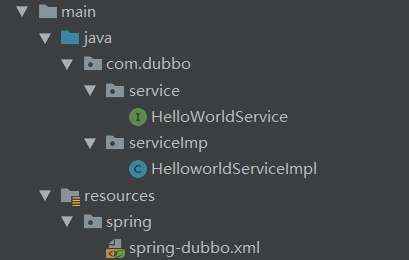
首先我们需要加载dubbo的jar包，由于dubbo和spring无缝集成(也可以和spring boot集成)，我们首先需要导入spring(或者spring boot)的jar包，然后导入dubbo的jar包

<dependency>  
 <groupId>com.alibaba</groupId>  
 <artifactId>dubbo</artifactId>  
 <version>${dubboVersion}</version>  
 <!--dubbo默认会再加载一个2.5版本的spring进来，这里阻止该行为防止spring版本冲突-->  
 <exclusions>  
 <exclusion>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring</artifactId>  
 </exclusion>  
 </exclusions>  
</dependency>

接下来，我们创建两个工程——服务提供者项目和服务消费者项目(简单的demo，各创建一个就好)，如图创建了一个provider01和consumer01，这里我们假设provider01为consumer01提供服务：



接下来，我们在服务提供者provider01中写一个服务类



HelloWorldService.java

public interface HelloWorldService {  
 String say(String name);  
}

HelloworldServiceImpl.java

public class HelloworldServiceImpl implements HelloWorldService {  
 public String say(String name) {  
 return name + ",hello world!";  
 }  
}

spring-dubbo.xml

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"  
 xmlns:tx="http://www.springframework.org/schema/tx"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd  
 http://code.alibabatech.com/schema/dubbo  
 http://code.alibabatech.com/schema/dubbo/dubbo.xsd">  
 <!--指定服务提供者的名称-->  
 <dubbo:application name="dubbo-provider01"></dubbo:application>  
 <!--扫描并将com.dubbo.serviceImpl类注册为bean-->  
 <bean id="helloWorldService" class="com.dubbo.serviceImp.HelloworldServiceImpl"></bean>  
 <!--配置服务提供者的服务，register="N/A"是绕过注册中心的配置-->  
 <dubbo:service interface="com.dubbo.service.HelloWorldService" ref="helloWorldService" register="false"></dubbo:service>  
</beans>

然后我们给web.xml添加监听器，保证项目一启动就加载spring配置文件

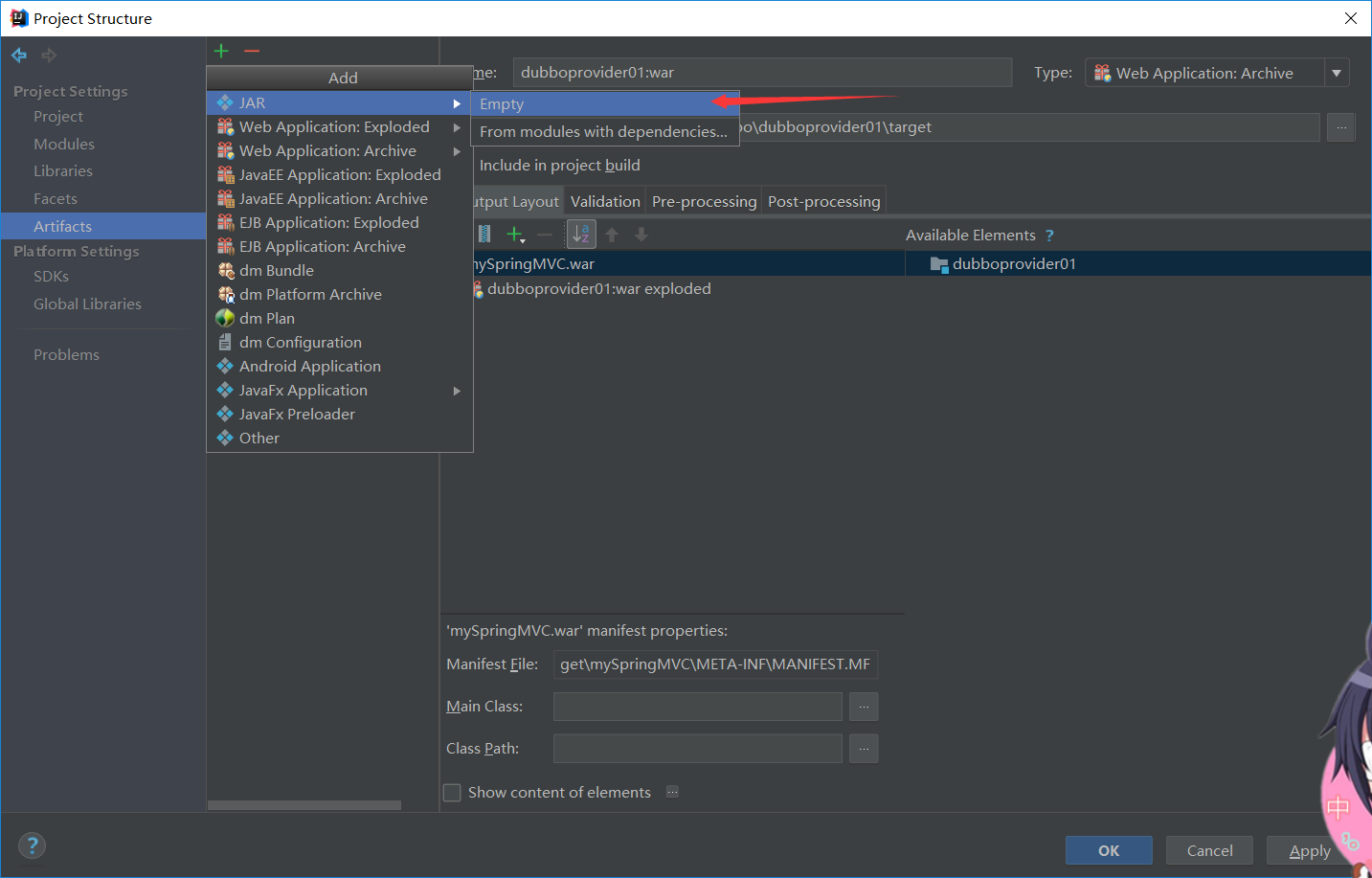
<!--加载spring配置文件-->  
<display-name>Archetype Created Web Application</display-name>  
<context-param>  
 <param-name>contextConfigLocation</param-name>  
 <param-value>classpath\*:spring/spring-\*.xml</param-value>  
</context-param>  
<listener>  
 <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  
</listener>

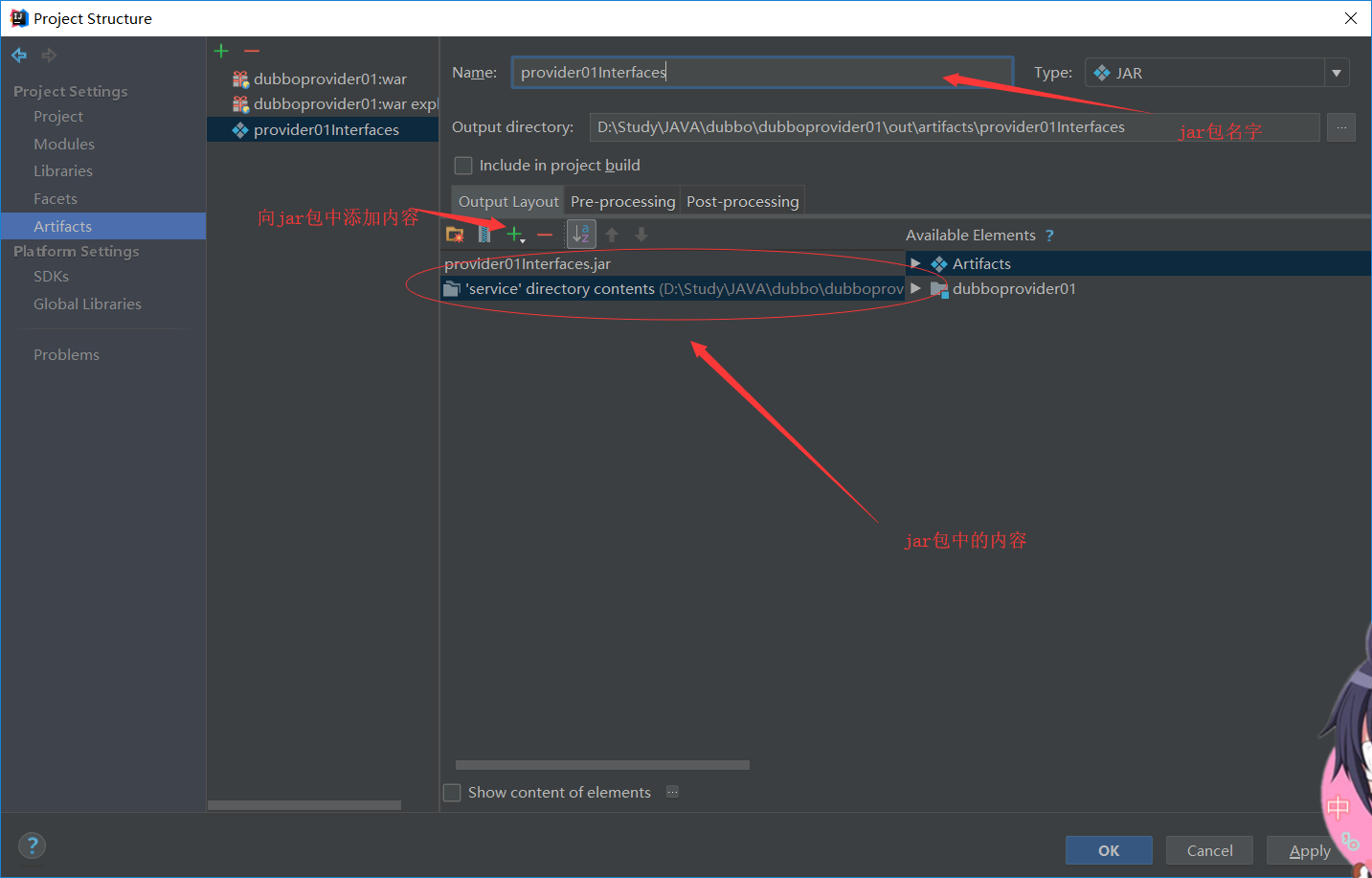
到这里，服务提供者就完成了，接下来是服务消费者

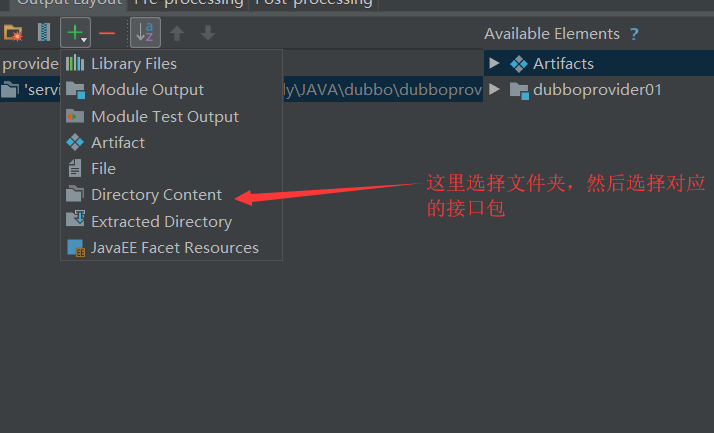
首先和上面一样，我们需要导入duboo的jar包到消费者项目中去，同时给web.xml设置好spring配置文件自动加载的监听器

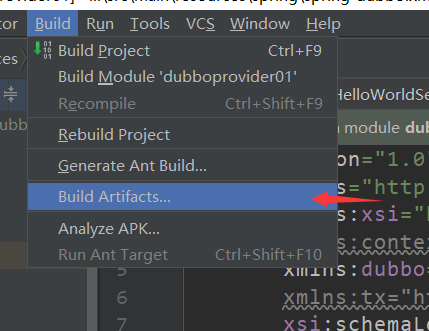
然后，我们需要把服务提供者的服务接口包打成jar包给服务消费者使用。注意，不需要服务实现类，只需要interface即可，也就是com.dubbo.service里面的接口

接下来演示idea打包过程，首先选择File-project Structure-Artifacts









打包之后的jar包存放的地址在第二张图中红字”jar包名字”那一行

mvn install:install-file -Dfile=D:\Study\JAVA\dubbo\dubboconsumer01\out\artifacts\provider01Interface\provider01Interface.jar -DgroupId=com.dubbo -DartifactId=dubbo-provider01 -Dversion=1.0-SNAPSHOT -Dpackaging=jar

将打好的包注册到本地库中通过maven加载到消费者项目中去