

Maven 将dubbo工程打包成可执行jar maven打包方式比较讲究

|  |
| --- |
| <build>  <finalName>edu-service-user</finalName>  <resources>  <resource>  <targetPath>${project.build.directory}/classes</targetPath>  <directory>src/main/resources</directory>  <filtering>true</filtering>  <includes>  <include>\*\*/\*.xml</include>  <include>\*\*/\*.properties</include>  </includes>  </resource>  <!-- 结合com.alibaba.dubbo.container.Main -->  <resource>  <targetPath>${project.build.directory}/classes/META-INF/spring</targetPath>  <directory>src/main/resources/spring</directory>  <filtering>true</filtering>  <includes>  <include>spring-context.xml</include>  </includes>  </resource>  </resources>    <pluginManagement>  <plugins>  <!-- 解决Maven插件在Eclipse内执行了一系列的生命周期引起冲突 -->  <plugin>  <groupId>org.eclipse.m2e</groupId>  <artifactId>lifecycle-mapping</artifactId>  <version>1.0.0</version>  <configuration>  <lifecycleMappingMetadata>  <pluginExecutions>  <pluginExecution>  <pluginExecutionFilter>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-dependency-plugin</artifactId>  <versionRange>[2.0,)</versionRange>  <goals>  <goal>copy-dependencies</goal>  </goals>  </pluginExecutionFilter>  <action>  <ignore />  </action>  </pluginExecution>  </pluginExecutions>  </lifecycleMappingMetadata>  </configuration>  </plugin>  </plugins>  </pluginManagement>  <plugins>  <!-- 打包jar文件时，配置manifest文件，加入lib包的jar依赖 -->  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-jar-plugin</artifactId>  <configuration>  <classesDirectory>target/classes/</classesDirectory>  <archive>  <manifest>  <mainClass>com.alibaba.dubbo.container.Main</mainClass>  <!-- 打包时 MANIFEST.MF文件不记录的时间戳版本 -->  <useUniqueVersions>false</useUniqueVersions>  <addClasspath>true</addClasspath>  <classpathPrefix>lib/</classpathPrefix>  </manifest>  <manifestEntries>  <Class-Path>.</Class-Path>  </manifestEntries>  </archive>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-dependency-plugin</artifactId>  <executions>  <execution>  <id>copy-dependencies</id>  <phase>package</phase>  <goals>  <goal>copy-dependencies</goal>  </goals>  <configuration>  <type>jar</type>  <includeTypes>jar</includeTypes>  <useUniqueVersions>false</useUniqueVersions>  <outputDirectory>  ${project.build.directory}/lib  </outputDirectory>  </configuration>  </execution>  </executions>  </plugin>  </plugins>  </build> |

Kill pid 申请停止进程，等到业务逻辑执行完

Kill -9 pid 立刻停止进程

Linux服务维护脚本service-user.sh

|  |
| --- |
| #!/bin/sh  ## java env  export JAVA\_HOME=/usr/local/java/jdk1.7.0\_72  export JRE\_HOME=$JAVA\_HOME/jre  ## service name  APP\_NAME=user  SERVICE\_DIR=/home/wusc/edu/service/$APP\_NAME  SERVICE\_NAME=edu-service-$APP\_NAME  JAR\_NAME=$SERVICE\_NAME\.jar  PID=$SERVICE\_NAME\.pid  cd $SERVICE\_DIR  case "$1" in  start)  nohup $JRE\_HOME/bin/java -Xms256m -Xmx512m -jar $JAR\_NAME >/dev/null 2>&1 &  echo $! > $SERVICE\_DIR/$PID  echo "=== start $SERVICE\_NAME"  ;;  stop)  kill `cat $SERVICE\_DIR/$PID`  rm -rf $SERVICE\_DIR/$PID  echo "=== stop $SERVICE\_NAME"  sleep 5  ##  ## edu-service-aa.jar  ## edu-service-aa-bb.jar  P\_ID=`ps -ef | grep -w "$SERVICE\_NAME" | grep -v "grep" | awk '{print $2}'`  if [ "$P\_ID" == "" ]; then  echo "=== $SERVICE\_NAME process not exists or stop success"  else  echo "=== $SERVICE\_NAME process pid is:$P\_ID"  echo "=== begin kill $SERVICE\_NAME process, pid is:$P\_ID"  kill -9 $P\_ID  fi  ;;  restart)  $0 stop  sleep 2  $0 start  echo "=== restart $SERVICE\_NAME"  ;;  \*)  ## restart  $0 stop  sleep 2  $0 start  ;;  esac  exit 0 |

将dubbo服务端打成可执行jar，然后用java命令启动来启动服务。主类：org.apache.dubbo.container.Main

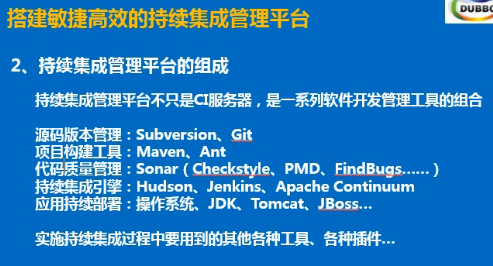
Dubbo 监控中心安装

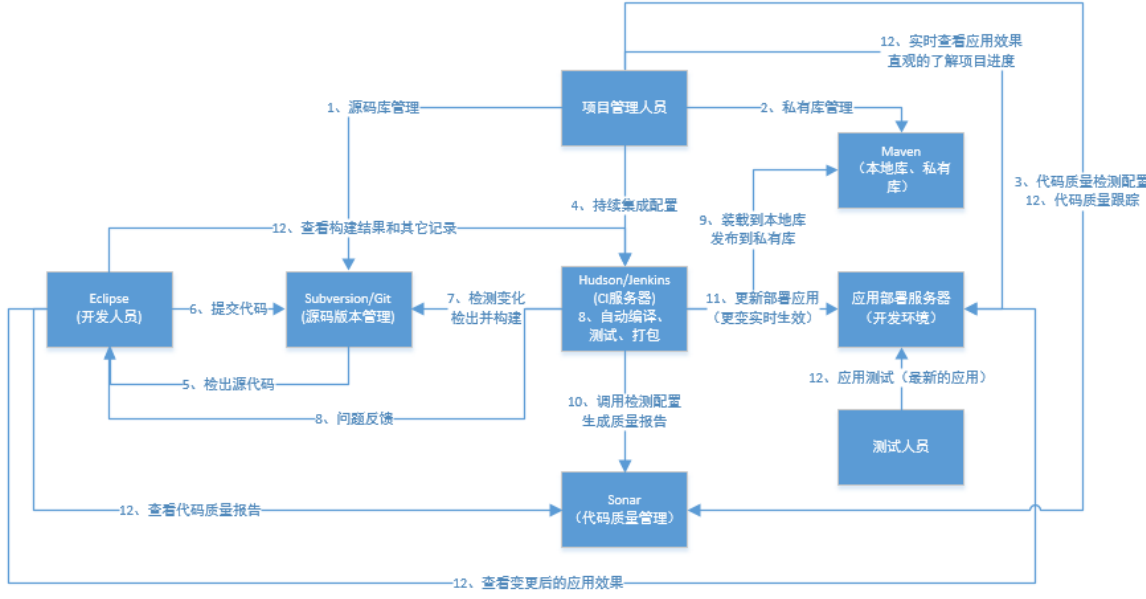
Dobbo admin 安装

最新版本dubbo 监控中心直接通过一个配置

<dubbo:monitor protocol="registry" />

不知道在什么地方可以查看监控结果？





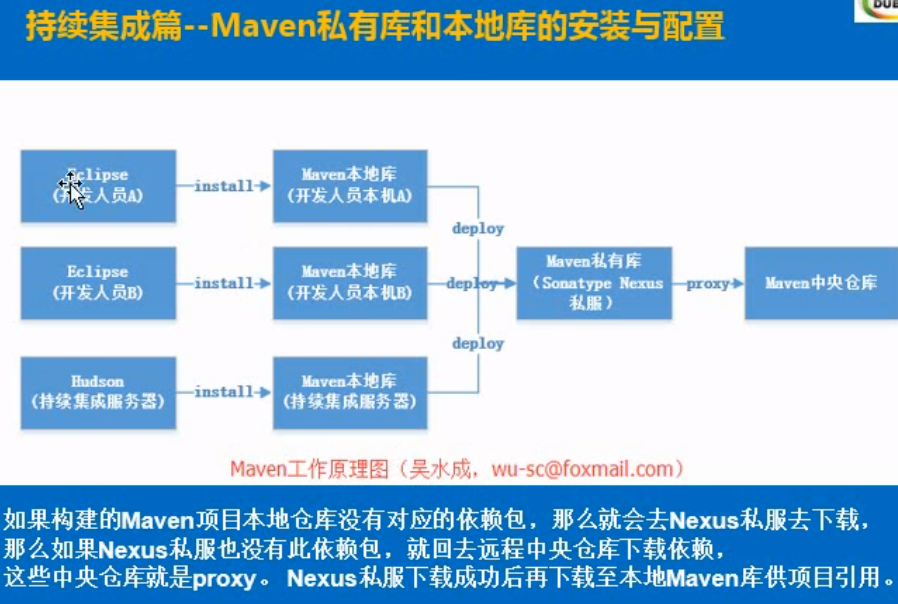
Svn服务端安装，管理，新建仓库，一些平时版本管理的操作。

Svn客户端：TortoiseSVN

Svn服务端：subversion

Svn服务端管理界面：

Svn服务器和Apache http整合，一个配置文件管理多个库。Svn-admin 需要数据库。不过svn-admin登录的界面用的账户admin的密码是在哪里配置的？



Nexus安装，管理

Maven私有库 releases snapshot，proxy，3th 库，还有一个总库group

版本是snaphot的会发布到snapshot里面，版本是releases 的会发布到releases 里面，上传jar需要账号密码，在maven的settings文件中配置的，还有私服地址等等。上传第三方jar，直接在私服的管理界面，通过gav参数上传就好了。