**多普勒(Dopper)3.0部署文档**

[1. 多普勒简介 2](#_Toc17679)

[1.1 部署简介 2](#_Toc20770)

[1.2 包含组件的说明 2](#_Toc24693)

[1.3部署要求 3](#_Toc11017)

[1.3.1 硬件要求 3](#_Toc26641)

[1.3.2 软件要求 3](#_Toc1886)

[2 业务系统对接多普勒 4](#_Toc24632)

[2.1 调用链数据统计配置 4](#_Toc17309)

[2.2 打点日志配置 5](#_Toc22788)

[3 多普勒后台程序安装 6](#_Toc32191)

[3.1 源码获取 6](#_Toc13740)

[3.2 源码打包 6](#_Toc23736)

[3.3 DB脚本执行 7](#_Toc30600)

[3.4 源码部署 7](#_Toc16879)

# 1. 多普勒简介

多普勒3.0是在开源软件jd-hydra的基础上二次开发形成的日志监控系统，主要功能有，调用链跟踪，服务监控，异常监控，业务打点日志记录及查询等；

## 1.1 部署简介

部署主要是针对dubbo项目的监控及统计,非dubbo项目不能统计及监控；

Dubbo项目要符合规范《4.Dubbo开发指南\_V1.1》

多普勒系统本身也是dubbo服务，所以能很容易和dubbo项目对接；

## 1.2 包含组件的说明

* Service

业务系统，一整套基于dubbo的分布式服务；项目中需要引入日志系统的接口（server项目中引入hydra-client，web项目中引入hydra-client和hydra-client-web）；

业务系统由开发人员部署；

* Hydra-client

打包的为Jar包，和具体的某个应用（server项目或web项目）部署在一起，共享应用JVM，为 多普勒系统 的接入层

* Hydra-client-web

打包的为Jar包，和具体的某个应用（web项目）部署在一起，共享应用JVM，为 多普勒系统 的接入层

* Hydra-manager

打包为tar.gz/zip包格式，负责所有业务跟踪点的协调和管理

* Hydra-collector

打包为tar.gz/zip包格式，负责跟踪数据的搜集，并落地到数据库中

* Doppler-server

多普勒系统展示模块服务端

* Doppler-web

多普勒系统展示模块web端

* Doppler-facade

Doppler-server和doppler-web的接口包，并且都已经引入；也是打点日志需要引入的包。

* Mysql-DB

用于存储多普勒日志系统数据

* Tomcat

Doppler-web项目的web容器

## 1.3部署要求

部署的组件：Hydra-manager，Hydra-Collector，Doppler-server，Doppler-web；

另外Mysql需要安装

### 1.3.1 硬件要求

|  |  |  |
| --- | --- | --- |
| **组件** | **硬件要求** | **说明** |
| Hydra-manager | 1台 | 对机器硬件性能要求不高 |
| Hydra-Collector | 对机器硬件要求跟日志请求量相关，需要根据业务请求量确定实例数量，部署在一起 |
| Doppler-server | 1台 | 监控一般由开发人员使用，访问量不大，部署在一起 |
| Doppler-web |
| 数据库服务器 | 1台 | 日志系统属于读少写多的应用，需要DBA按读少写多来优化配置 |

上述机器可以根据实际情况，做部署上合理规划。

### 1.3.2 软件要求

|  |  |  |
| --- | --- | --- |
| **软件** | **版本** | **说明** |
| JDK | 1.7 | 1.7 |
| dubbo | 2.8.1 | 基于当当的dubbox 2.8.1开发 |
| zookeeper | 3.4+ | 使用业务系统zookeeper |
| Mysql | 5.5+ | 支持存储过程，EVENT |
| Tomcat | 7.0+ | 部署doppler-web项目 |

注意：dubbox和dubbo有兼容性问题.应用支持dubbox 2.8.1及以上

(https://github.com/dangdangdotcom/dubbox/issues/110)

# 2 业务系统对接多普勒

## 2.1 调用链数据统计配置

业务系统对接多普勒，只需涉及三个组件hydra-client，hydra-client-web，hydra-interface

；这三个组件已上传到公司私服（私服地址：<http://192.168.0.109:8081/nexus/content/groups/public/）>

对接步骤如下

业务系统是web项目：

* **pom文件中引入**：

<**dependency**>  
 <**groupId**>com.jd.bdp</**groupId**>  
 <**artifactId**>hydra-client</**artifactId**>  
 <**version**>1.0</**version**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>com.sinoservices</**groupId**>  
 <**artifactId**>hydra-client-web</**artifactId**>  
 <**version**>1.0</**version**>  
</**dependency**>

<**dependency**>  
 <**groupId**>com.jd.bdp</**groupId**>  
 <**artifactId**>hydra-interface</**artifactId**>

<**version**>1.0</**version**>  
</**dependency**>

* **Web.xml中加入过滤器：**

<**filter**>  
 <**filter-name**>recordLog</**filter-name**>  
 <**filter-class**>com.sinoservices.trace.filter.MonitorFilter</**filter-class**>  
 <**init-param**>  
 <**param-name**>urlPattern</**param-name**>  
 <**param-value**>.\*\.[s]{0,1}html.\*</**param-value**>  
 </**init-param**>  
</**filter**>  
<**filter-mapping**>  
 <**filter-name**>recordLog</**filter-name**>  
 <**url-pattern**>/\*</**url-pattern**>  
</**filter-mapping**>

urlPattern的value：正则形式；如果URL匹配这个正则，系统会统计，否则，不予统计；

这个参数主要是为了过滤无效URL，比如 \*.js , \*.css;

上述例子表示只统计以shtml或html为后缀的URL

如果为空，系统默认会统计以shtml或html为后缀的URL

业务系统是Server项目：

* **pom文件中引入**：

<**dependency**>  
 <**groupId**>com.jd.bdp</**groupId**>  
 <**artifactId**>hydra-client</**artifactId**>  
 <**version**>1.0</**version**>  
</**dependency**>

<**dependency**>  
 <**groupId**>com.jd.bdp</**groupId**>  
 <**artifactId**>hydra-interface</**artifactId**>

<**version**>1.0</**version**>  
</**dependency**>

## 2.2 打点日志配置

多普勒系统界面菜单“异常监控”中的数据来自打点日志和调用链的异常日志两部分。其中打点日志需要手动打点生成，打点日志主要作用是监控内部方法（项目接口方法中调用的子方法或更深层次方法）。

打点日志不用的情况下，可以忽略本节配置。

配置步骤：

A. 项目中引入pom文件

<**dependency**>  
 <**groupId**>com.sinoservices</**groupId**>  
 <**artifactId**>doppler-facade</**artifactId**>  
 <**version**>1.0</**version**>  
</**dependency**>

B.项目中配置traceFacade接口

在dubbo的xml启动文件（一般在META-INF/spring/目录下）中加入（消费者）：

<dubbo:reference id="traceFacade" interface="com.sinoservices.doppler.facade.TraceFacade"/>

C. 使用方式跟调用普通服务方式一样，注入对象，然后调用：

@Autowired

TraceFacade traceFacade;

Public void test(){

/\*\* 业务代码 \*\*/

TraceUtil.sendLog(traceFacade, TraceContants.ERROR\_TYPE\_ERROR,"my log info");

/\*\* 业务代码 \*\*/

}

TraceUtil.sendLog接口参数说明：traceFacade：服务对象

errorType：错误级别，

str ： 日志信息

# 3 多普勒后台程序安装

多普勒后台系统包括：Hydra-manager，Hydra-Collector，Doppler-server和Doppler-web，这些都需要安装部署；

另外，zookeeper，mysql，tomcat需要额外安装

## 3.1 源码获取

SVN地址：<https://192.168.0.204/svn/GeT_ARCH_Src/POC/Doppler>

如果没有权限，联系SS获取；

## 3.2 源码打包

打包前每个项目需要修改的内容：

**Hydra-manager**

目录：doppler\hydra-master\modules\hydra-manager\src\main\resources\

文件：mysql.properties

修改mysql地址，用户名，密码

文件：hydra-manager.properties

修改文件中dubbo.registry.address的值为zookeeper地址

**Hydra-collector**

目录：doppler\hydra-master\modules\hydra-collector\src\main\resources

文件：mysql.properties

修改mysql地址，用户名，密码

文件：dubbo-collector.properties

修改文件中dubbo.registry.address的值为zookeeper地址

**Doppler-server**

目录：doppler\doppler-server\src\main\resources

文件：mysql.properties

修改mysql地址，用户名，密码

目录：doppler\doppler-server\src\main\assembly\conf

文件：dubbo.properties

修改文件中dubbo.registry.address的值为zookeeper地址

**Doppler-web**

目录：doppler\doppler-web\src\main\resources

文件：dubbo.properties

修改文件中dubbo.registry.address的值为zookeeper地址

之后用mvn命令打包即可；

## 3.3 DB脚本执行

Mysql脚本包括，多普勒（doppler）数据库的表结构

表结构：doppler\hydra-master\docs\table-mysql\initTable.sql

## 3.4 源码部署

doppler-web的war包直接传到目标tomcat，启动，暂停tomcat即可；

对于Hydra-manager，hydra-collector，doppler-server三个项目

部署步骤：

A 解压

Tar -zxvf xxx.tar.gz

B 启动

Hydra-manager: Hydra-manager/bin/manager.sh start

Hydra-collector: Hydra-collector/bin/collector-mysql.sh start

doppler-server: doppler-server-1.0/bin/start.sh

C 停止

Hydra-manager: Hydra-manager/bin/manager.sh stop

Hydra-collector: Hydra-collector/bin/collector-mysql.sh stop

doppler-server: doppler-server-1.0/bin/stop.sh

D 日志查看

业务日志：/mnt/logs/{$project\_name}.yyyy-MM-dd.log

dubbo日志：/mnt/logs/dubbo/{$project\_name}\_dubbo.yyyy-MM-dd.log