DataFlowKit

DataFlowKit是一个基于 Flume 1.6.0 框架的数据处理平台，修改部分Bug，并对File，Kafka，JMS，HDFS，Redis采集进行了封装，可集中管理数据源的配置，简化配置文件冗余数据。本文档就扩展部分进行说明，并兼容原有 Flume 1.6.0 的功能，更加详细的文档请参考Flume 1.6.0 User Guide。

功能

- 文件采集（粒度文件）
- 滚动文件采集（持续文件写入log4j）
- Kafka采集
- JMS采集
- Redis采集
- Http采集
- 文件输出
- HDFS输出
- Kafka输出
- JMS输出
- Redis输出
- Syslog输出

文档

系统需求

1. JDK1.7+
2. 内存4G＋，大吞吐量的数据处理需要更大的内存
3. 磁盘100G＋，文件型Channel会需要更大的磁盘空间
4. 目录权限，安装目录要具备读／写／执行权限

参数配置

使用vi命令修改dataflowkit执行文件，修改相关配置：

```
vi dataflowkit
```

JDK配置，如果系统默认安装了JDK1.7那么请忽略此步骤，否则就需要人工设置JDK的安装路径，找到文件中JAVA_HOME_DIR，删除前边的注释，设置JDK的安装路径

```
JAVA_HOME_DIR=/usr/local/jdk1.7
```

采集配置文件设置，找到文件中FLUME*CONF一行，指定需要执行的采集规则文件路径（dataflowkit/conf/demo目录下有一些配置文件的例子），AGENT*NAME与配置文件中的agent名称一致

```
FLUME_CONF=./demo/sample.properties
AGENT_NAME=agent
```

权限修改

使用chmod命令修改bin、dataflowkit读写权限

```
chmod +x dataflowkit
chmod +x bin/*
```

启动

请先确认dataflowkit具有可执行权限，确认dataflowkit文件中的相关配置,

使用dataflowkit命令启动、停止、重新启动DataFlowKit采集服务

```
dataflowkit start
dataflowkit stop
dataflowkit restart
```

示例

在dataflowkit/conf/demo/sample.properties文件中定义了一个简单的数据流采集流程，捕获44444端口的数据并输出到日志中，通过dataflowkit start直接启动这个采集流程，然后通过以下命令模拟发送数据:

```
$ telnet localhost 44444
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
Hello world! <ENTER>
OK
```

采集配置

数据源配置

通过集中配置Kafka、JMS的地址端口，简化Flume配置文件中的设置，避免相同的配置冗余

- Kafka配置

  在dataflowkit/conf/env_root/common/kafka目录下可以手动增加多个配置文件，每个配置文件设置一个kafka集群，在flume的配置文件中只需要制定配置文件的名称就可以直接使用这个配置文件中的kafka地址，

属性名	举例	描述
brokerList	localhost:9092	Kafka的地址
zookeeper	localhost:2181	Zookeeper的地址
requestRequiredAcks	1	请求应答
batchSize	100	批量处理数量
partitionerClass	kafka.producer.DefaultPartitioner	Kafka分区实现类

例如:
dataflowkit/conf/env_root/common/kafka/itosi.xml

```
<xsd:BeanMap xmlns:xsd="http://www.xsd.webservice.env.iplatform.org" xmlns:xsi="http
    <xsd:Desc>集团Kafka集群</xsd:Desc>
    <xsd:Property>
        <xsd:Key>brokerList</xsd:Key>
        <xsd:Value>localhost:9092</xsd:Value>
        <xsd:Type>String</xsd:Type>
    </xsd:Property>
    <xsd:Property>
        <xsd:Key>zookeeper</xsd:Key>
        <xsd:Value>localhost:2181</xsd:Value>
        <xsd:Type>String</xsd:Type>
    </xsd:Property>
    <xsd:Property>
        <xsd:Key>requestRequiredAcks</xsd:Key>
        <xsd:Value>1</xsd:Value>
        <xsd:Type>Integer</xsd:Type>
    </xsd:Property>
    <xsd:Property>
        <xsd:Key>batchSize</xsd:Key>
        <xsd:Value>100</xsd:Value>
        <xsd:Type>Integer</xsd:Type>
    </xsd:Property>
    <xsd:Property>
        <xsd:Key>partitionerClass</xsd:Key>
        <xsd:Value>kafka.producer.DefaultPartitioner</xsd:Value>
        <xsd:Type>String</xsd:Type>
    </xsd:Property>
</xsd:BeanMap>
```

在flume配置文件中可通过以下方式制定使用哪个kafka配置文件

```
发送
agent.sinks.k.kafka_name = itosi
```

详细说明

```
接收
agent.sources.s.topics = itosi.topic名称
```

详细说明

- JMS配置 在dataflowkit/conf/env_root/evn/common/jms目录下可以手动增加多个配置文件，每个配置文件设置一个jms服务，在flume的配置文件中只需要制定配置文件的名称就可以直接使用这个配置文件中的jms地址

属性名	举例	描述
providerURL	tcp://192.168.100.56:61616	JMS地址
username	-	用户名
password	-	密码

例如:
dataflowkit/conf/env_root/env/common/jms/group.xml

```
<xsd:BeanMap xmlns:xsd="http://www.xsd.webservice.env.iplatform.org" xmlns:xsi="http
  <xsd:Desc>集团MQ</xsd:Desc>
  <xsd:Property>
    <xsd:Key>providerURL</xsd:Key>
    <xsd:Value>tcp://192.168.100.56:61616</xsd:Value>
    <xsd:Type>String</xsd:Type>
    <xsd:Desc></xsd:Desc>
  </xsd:Property>
  <xsd:Property>
    <xsd:Key>username</xsd:Key>
    <xsd:Value></xsd:Value>
    <xsd:Type>String</xsd:Type>
  </xsd:Property>
  <xsd:Property>
    <xsd:Key>password</xsd:Key>
    <xsd:Value></xsd:Value>
    <xsd:Type>String</xsd:Type>
  </xsd:Property>
</xsd:BeanMap>
```

在flume配置文件中可通过以下方式制定使用哪个jms配置文件

```
发送
agent.sinks.k.topics = group.topic名称
```

详细说明

```
采集
agent.sources.s.topics = group.topic名称
```

详细说明

输入采集

JMS Source

读取配置文件的信息，以Topic（发布/订阅）的方式进行消息处理。有如下特点：

- 每个消息可以有多个消费者
- 发布者和订阅者之间有时间上的依赖性。
- 针对某个主题（Topic）的订阅者，它必须创建一个订阅者之后，才能消费发布者的消息，而且为了消费消息，订阅者必须保持运行的状态。

属性名	默认值	描述
spoolDir	-	监控目录路径
batchSize	100	批量处理条数
topics	-	消息组名称
deletePolicy	never	读取完毕后的文件删除策略，never修改扩展名，immediate删除
fileSuffix	.COMPLETED	文件读取完毕后增加的后缀名
fileHeader	false	是否增加一个消息头在文件中
bufferMaxlineLength	8192	单行最大长度，超过这个长度的数据忽略

示例:

```
agent.sources.s.channels = c
agent.sources.s.type =  com.cmos.bomc.dataflowkit.cmcconline.flume.source.jms.JMSSou
agent.sources.s.initialContextFactory = org.apache.activemq.jndi.ActiveMQInitialCont
agent.sources.s.batchSize = 1000
agent.sources.s.topics = group.zhangleitest
```

Kafka Source

此消息源类似于JMS消息机制，与之不同的是，Kafka消息被消费后，消息仍然不会被立即删除.日志文件将会根据broker中的配置要求,保留一定的时间之后删除;比如log文件保留2天,那么两天后,文件会被清除,无论其中的消息是否被消费.

属性名	默认值	描述
spoolDir	-	监控目录路径
topics	-	消息组名称
batchSize	1000	批量处理条数
batchDurationMillis	-	批处理持续毫秒数
bufferMaxLineLength	8192	单行最大长度，超过这个长度的数据忽略
auto.commit.enable	true	设为true，consumer会定时向ZooKeeper发送已经获取到的消息的offse
zookeeper.connection.timeout.ms	6000	client连接到ZK server的超时时间
zookeeper.session.timeout.ms	6000	ZooKeeper的session的超时时间

示例:

```
agent.sources.s.channels = c
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.source.kafka.Kafka
agent.sources.s.topics = itosi.zhangleitest
agent.sources.s.batchSize = 1000
agent.sources.s.batchDurationMillis = 500
agent.sources.s.kafka.auto.commit.enable = true
agent.sources.s.kafka.zookeeper.connection.timeout.ms = 30000
agent.sources.s.kafka.zookeeper.session.timeout.ms = 30000
```

File Source

这个数据源是spooldir的扩展，用来监控一个目录下的文件，当目录下有新文件产生后将被读取到channel中，然后删除这个文件或者重命名这个文件，此数据源读取数据有如下约束:

- 文件将被一次性读取，不支持持续写入的文件读取
- 判断文件已经生成完毕的条件是，每隔2秒读取一次文件修改时间，如果两次读取文件修改时间一致，则认为文件已经写完
- 读取过的文件名将被记录，目录下每次放入的文件不能和以前放入的文件名重复

属性名	默认值	描述
spoolDir	-	监控目录路径
batchSize	500	批量处理条数
bufferMaxLineLength	8192	单行最大长度，超过这个长度的数据忽略
fileHeader	false	是否在消息头中自动添加文件路径
deletePolicy	never	读取完毕后的文件删除策略，never修改扩展名，immediate删除
targetPattern	file-*(\\d+).dat	采集目标文件名匹配，只采集匹配的文件
ignorePattern	^$	忽略的文件名
deserializer.multiline	^\\d{4}-\\d{2}-\\d{2} \\d{2}:\\d{2}:\\d{2},\\d{3}	多行文件合并一行，定义行开始正则表达式，例如以“2016-05-19 14:55:43,593”开始字符判断是一行的开始
deserializer.lineEndWith	\\n	当文件不是采用\n进行换行，而是使用“\n”字符标识换行时需要增加此选项

示例:

```
agent.sources.s.channels = c
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.source.spooldir.Sp
agent.sources.s.spoolDir = /tmp/src
agent.sources.s.batchSize = 500
agent.sources.s.fileHeader = true
agent.sources.s.bufferMaxLineLength = 8192
```

Roll File Source

这个数据源是File Source的扩展，主要用于处理一个目录下的文件持续写入的情况，支持两种情况的数据持续写入情况:

1. 滚动写入新文件：文件在周期内是持续写入的，按周期（每天/每小时/每100M等等）产生一个新文件，并在新文件中持续写入。例如: system-20160401.log、system-20160402.log、system-20160403.log，注意：system-20160403.log是最新文件

2. 滚动写入一个文件：文件始终是持续写入的，按周期（每天/每小时/每100M等等）产生一个新文件，同时清空此文件后，仍然在此文件中持续写入。例如: system.log、system-20160401.log、system-20160402.log，注意：system.log永远是最新的文件

文件滚动条件判断原理

1. 滚动写入新文件：系统自动判断是否有新文件产生，如果有新文件产生，则认为当前读取的文件已经写入完毕，待系统读取完此文件后自动读取新文件

2. 滚动写入一个文件：系统自动判断是否有新文件产生，如果有新文件产生，则认为当前读取的文件已经发生了滚动，此文件内容被清空，系统重新读取此文件

属性名	默认值	描述
spoolDir	-	监控目录路径
batchSize	100	批量处理条数
bufferMaxLineLength	8192	单行最大长度，超过这个长度的数据忽略
rollModule	–	滚动策略，next滚动产生新文件，self滚动写一个文件
deletePolicy	never	读取完毕后的文件删除策略，never修改扩展名，immediate删除
ignorePattern	^$	忽略的文件名丨
targetPattern	-	读取目标文件，rollModule=next的时候，这里应该设置成符合滚动规则的正则表达式，rollModule=self的时候，这应该设置成一个固定的文件名
rollPattern	-	当rollModule=self的时候生效，这里设置一个正则表达式用来判断新生成的文件名是是否是滚动产生
deserializer.multiline	-	当数据不是单行作为一个消息题发送的时候，设置行首匹配正则，例如：java的错误日志就是多行作为一个消息题，此类日志要设置行首的判断规则
deserializer.maxLineLength	-	多行数据处理时单行的最大长度

滚动写入新文件示例:

在/tmp/logs目录下每小时产生一个文件名符合正则 file-(\d)*.dat 的文件

```
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.source.spooldirect
agent.sources.s.spoolDir = /tmp/logs
agent.sources.s.channels = c
agent.sources.s.ignorePattern = ^$
agent.sources.s.targetPattern = file-(\\d)*.dat
agent.sources.s.rollModule = next
agent.sources.s.batchSize = 100
```

滚动写入一个文件示例:

在/tmp/logs目录下持续写入file.log文件，每到整点时file.log文件被复制成一个file.log.yyyyMMddHH的文件，然后清空file.log后重新写入

```
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.source.spooldirect
agent.sources.s.spoolDir = /tmp/logs
agent.sources.s.channels = c
agent.sources.s.ignorePattern = ^$
agent.sources.s.targetPattern = file.log
agent.sources.s.rollModule = self
agent.sources.s.rollPattern = file.log.(\\d){4}(\\d){2}(\\d){2}(\\d){2}
agent.sources.s.deserializer.multiline = ^\\d{4}-\\d{2}-\\d{2} \\d{2}:\\d{2}:\\d{2},
agent.sources.s.deserializer.maxLineLength = 3000
agent.sources.s.batchSize = 1
```

Redis Source(pop)

redisSource 采集输入配置。

属性名	默认值	描述
type	-	redis source处理类
spoolDir	-	redis目标文件所在夹
targetPattern	-	redis 监控文件的名称的特定格式，允许正则表达式

示例

```
agent.sources.s.channels = c
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.source.spooldir.Sp
agent.sources.s.spoolDir = ./tmp/src
agent.sources.s.batchSize = 500
agent.sources.s.fileHeader = true
agent.sources.s.bufferMaxLineLength = 8192
agent.sources.s.selector.type = replicating
agent.sources.s.targetPattern = file-*(\\d+).dat
```

Redis Source(pop)

redisSource 采集输入配置。

属性名	默认值	描述
type	-	redis source处理类
spoolDir	-	redis目标文件所在夹
targetPattern	-	redis 监控文件的名称的特定格式，允许正则表达式

示例

```
agent.sources.s.channels = c
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.source.spooldir.Sp
agent.sources.s.spoolDir = ./tmp/src
agent.sources.s.batchSize = 500
agent.sources.s.fileHeader = true
agent.sources.s.bufferMaxLineLength = 8192
agent.sources.s.selector.type = replicating
agent.sources.s.targetPattern = file-*(\\d+).dat
```

Http Source

HttpSource 采集输入配置。

属性名	默认值	描述
type	http	source类型
bind	0.0.0.0	IP地址
port	50000	端口号
paramName	data	http传输数据名称
handler	-	httpSource处理类

示例

```
agent.sources.s.type=http
agent.sources.s.bind=0.0.0.0
agent.sources.s.port=50000
agent.sources.s.channels=c
agent.sources.s.handler.paramName=data
agent.sources.s.handler=com.cmos.bomc.dataflowkit.cmcconlne.webwatch.ParamDataDecode
```

采集输出

JMS Sink

以JMS消息主题分类，配置JMS消息输出路径。

属性名	默认值	描述
batchSize	100	批处理条数
rollInterval	30	循环滚动文件间隔秒数
directory	-	目标路径

示例

```
agent.sinks.k.channel = c
agent.sinks.k.type = com.cmos.bomc.dataflowkit.cmcconline.flume.sink.jms.JMSSink
agent.sinks.k.initialContextFactory = org.apache.activemq.jndi.ActiveMQInitialConte
agent.sinks.k.batchSize = 1000
agent.sinks.k.compress = true
```

Kafka Sink

以Kafka消息主题分类，配置Kafka消息输出路径。

属性名	默认值	描述
brokerList	-	代理列表，以逗号分隔，形式为hostname:port
toppic	-	消息的主题
batchSize	100	批处理条数
required.acks	1	确认消息类型。0-不需要确认，1-等待领导者确认，-1等待所有副本确认

示例

```
agent.sinks.k.type = com.cmos.bomc.dataflowkit.cmcconline.flume.sink.kafka.KafkaSink
agent.sinks.k.topic = zhangleitest
agent.sinks.k.kafka_name = itosi
agent.sinks.k.batchSize = 1000
agent.sinks.k.acks = 0
agent.sinks.k.kafka.request.required.acks = 0
agent.sinks.k.kafka.compression.codec = gzip
agent.sinks.k.max.request.size = 2097152
agent.sinks.k.kafka.request.timeout.ms = 60000
```

File Sink

配置File类，输出路径。

属性名	默认值	描述
spoolDir	-	监控目录路径
sinks.directory	-	文件存储目录
sink.rollInterval	30	每隔30秒滚动文件
sink.serializer	TEXT	文件序列化方式
batchSize	100	批量处理条数

示例

```
agent.sinks.channels = c1
agent.sinks = k1
agent.sinks.k1.type = file_roll
agent.sinks.k1.channel = c1
agent.sinks.k1.sink.directory = /var/log/flume
```

HDFS Sink

配置HDFS消息输出路径。

属性名	默认值	描述
spoolDir	-	监控目录路径
batchSize	500	批量处理条数
path	-	写入hdfs的路径，需要包含文件系统标识，可以使用flume提供的日期及%{host}表达式
filePrefix	FlumeData	写入hdfs的文件名前缀
fileSuffix	-	写入hdfs的文件名后缀
useLocalTimeStamp	false	是否使用当地时间
round	false	是否启用时间上的”舍弃”，这里的”舍弃”，类似于”四舍五入”
minBlockReplicas	1	写入HDFS文件块的最小副本数
idleTimeout	0	当目前被打开的临时文件在该参数指定的时间（秒）内，没有任何数据写入，则将该临时文件关闭并重命名成目标文件；
threadsPoolSize	10	启动的操作HDFS的线程数
callTimeout	10000	执行HDFS操作的超时时间（单位：毫秒）
fileType	SequenceFile	文件格式，包括：SequenceFile, DataStream,CompressedStream。当使用DataStream时候，文件不会被压缩，不需要设置hdfs.codeC;当使用CompressedStream时候，必须设置一个正确的hdfs.codeC值；
codeC	-	文件压缩格式，包括：gzip, bzip2, lzo, lzop, snappy
rollCount	10	当events数据达到该数量时候，将临时文件滚动成目标文件,如果设置成0，则表示不根据events数据来滚动文件；
rollSize	1024	当临时文件达到该大小（单位：bytes）时，滚动成目标文件,如果设置成0，则表示不根据临时文件大小来滚动文件；
rollInterval	30	间隔多长将临时文件滚动成最终目标文件(单位：秒),如果设置成0，则表示不根据时间来滚动文件；

示例

```
agent.sinks.k.type = com.cmos.bomc.dataflowkit.cmcconline.flume.sink.hdfs.HDFSEventS
agent.sinks.k.hdfs.path = file:///tmp/10085/%y%m%d
agent.sinks.k.hdfs.filePrefix = file_%y%m%d_%H
agent.sinks.k.hdfs.fileSuffix = .dat
agent.sinks.k.hdfs.useLocalTimeStamp = true
agent.sinks.k.hdfs.round = true
agent.sinks.k.hdfs.minBlockReplicas = 1
agent.sinks.k.hdfs.idleTimeout = 0
agent.sinks.k.hdfs.threadsPoolSize = 10
agent.sinks.k.hdfs.batchSize = 1
```

不压缩

```
agent.sinks.k.hdfs.fileType = DataStream
```

压缩

```
agent.sinks.k.hdfs.codeC = gzip
agent.sinks.k.hdfs.fileType = CompressedStream
```

每1万行滚动一个文件

```
agent.sinks.k.hdfs.rollCount = 10000
agent.sinks.k.hdfs.rollSize = 0
agent.sinks.k.hdfs.rollInterval = 0
```

每分钟滚动一个文件

```
agent.sinks.k.hdfs.rollCount = 0
agent.sinks.k.hdfs.rollSize = 0
agent.sinks.k.hdfs.rollInterval = 60
```

每10M滚动一个文件

```
agent.sinks.k.hdfs.rollCount = 0
agent.sinks.k.hdfs.rollSize = 10240000
agent.sinks.k.hdfs.rollInterval = 0
```

Redis Sink(push)

redis从source中读取数据，以${0}-${1} 的形式获取redisKey，redisVlaue，保存到redis中。

属性名	默认值	描述
host	localhost	redis服务器ip
port	6379	redis服务器端口
keyFormart	-	redis key格式约束: ${0}-${1}
valueFormart	-	redis value格式约束: ${0}-${1}-{2}，如果不设置此属性，那么将不对value进行格式化，直接存入
split	|	source文件默认分隔符
keyExpire	0	redis key过期时间。单位（秒）
redisCmd	set	此属性不设置默认就是使用set命令，还支持 push，lpush，rpush命令

示例

```
agent.sinks.k.type = com.cmos.bomc.dataflowkit.cmcconline.flume.sink.redis.RedisSpli
agent.sinks.k.host = 127.0.0.1
agent.sinks.k.port = 6379
agent.sinks.k.database = 0
agent.sinks.k.batch_size = 100
agent.sinks.k.split = |
agent.sinks.k.keyFormat = ${0}~${1}
agent.sinks.k.valueFormat = ${0}~${1}~${2}
agent.sinks.k.keyExpire = 1000
agent.sinks.k.redisCmd = lpush
```

Syslog Sink

SyslogSink 采集输入配置。

属性名	默认值	描述
type	-	Syslog sink处理类
host	-	IP地址
port	1468	端口号
instance	tcp	传输类型，包括tcp、udp
serverity	0	级别

示例

```
agent.sources.s.channels = c
agent.sources.s.type = com.cmos.bomc.dataflowkit.cmcconline.flume.sink.syslog.Systlo
agent.sinks.k.host = 127.0.0.1
agent.sinks.k.port = 1468
agent.sinks.k.instance = tcp
agent.sinks.k.serverity =  3
```

拦截器

拦截器可以作用在source或者sink上，用于对数据进行拦截处理。

Regex Filtering

正则拦截器通过正则表达式进行数据过滤，支持匹配有效和匹配无效两种方式，对于复杂的字符串也支持通过特殊字符分段后再进行正则匹配。

属性名	默认值	描述
type	com.cmos.bomc.dataflowkit. cmcconline.flume.interceptor. RegexSplitFilteringInterceptor$Builder	
regex	[0l2]	正则表达式,例如匹配0或者2
excludeEvents	true/false	false的时候符合正则的数据将被处理， true的时候符合正则的数据将被丢弃
split	\\l	数据分割符,例如竖线分割
split_format	${19}	分割后的字段，如果定义了这个属性，那么正则将匹配这个字段的值，例如分割后的第19个字段，注意第一个字段的序号是0

桥接

Avro Source 2 Avro Sink

示例

```
agent.sources = s
agent.channels = c
agent.sinks = k

agent.channels.c.type = memory
agent.channels.c.keep-alive = 60
agent.channels.c.capacity = 1000000000
agent.channels.c.transactionCapacity = 100000000

#监听本级51515端口
agent.sources.s.channels = c
agent.sources.s.type = avro
agent.sources.s.bind = 0.0.0.0
agent.sources.s.port = 51515

#消息发往192.168.100.100的51515端口
agent.sinks.k.channel = c
agent.sinks.k.type = avro
agent.sinks.k.type.hostname = 192.168.100.100
agent.sinks.k.type.port = 51515
```

安装

编译

使用ant编译DataFlowKit 命令:

```
cd dataflowkit/module-common
ant release
```

解压

编译后在eclipseWorkspace/release/目录下生成dataflowkit-common.zip，
解压dataflowkit-common.zip

```
cd dataflowkit/release
tar -xvf dataflowkit-common.zip
```

配置

设置dataflowkit中FLUME_CONF

```
cd dataflowkit-common
vi dataflowkit
```

示例

```
FLUME_CONF=./demo/redis_sink_agent.properties.demo
```

启动

启动dataflowkit

```
cd dataflowkit-common
./dataflowkit start
```

关闭

关闭dataflowkit

```
./dataflowkit stop
```

重启

重启dataflowkit

```
./dataflowkit restart
```

日志

查看日志文件

```
tail -f logs/flume.log
或
more logs/flume.log
```

删除日志文件

```
rm logs/*
```

多实例

本系统默认单实例，若需要多实例，须增加dataflowkit。

复制dataflowkit为dataflowkit1，修改PIDFILE的值。

```
cd dataflowkit-common
cp dataflowkit dataflowkit1
vi dataflowkit1
```

示例

```
PIDFILE = dataflowkit1.pid
```

关联项目

- 命令执行代理
- 数据采集代理
- 策略下发代理
- MQ消息分发代理
- Esper数据处理代理