企业级分布式应用服务EDAS

快速开始

阿里云
aliyun.com

快速开始

EDAS控制台使用指南

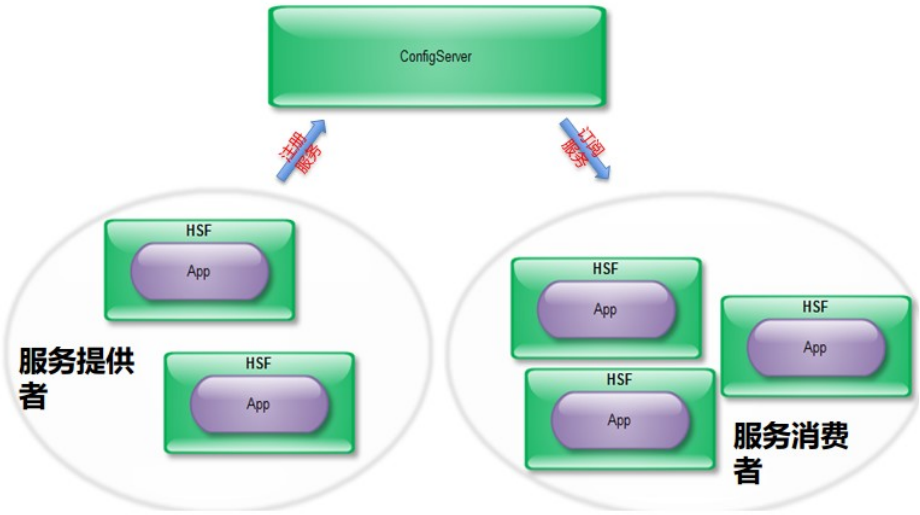
EDAS控制台使用指南

应用管理

生命周期管理EDAS产品体系中非常重要的一部分，以应用为中心的中间件PaaS平台，提供一键应用部署与扩容，简化用户操作。用户基于该功能能够完成对一个应用在发布与运行过程的全面管理，包括：应用的创建、部署、启动、回滚、扩容和停止下线等。

服务化架构

依托于淘宝十年互联网架构经验，帮助用户构建高性能易扩展的Web应用。在EDAS产品中，服务化框架的整体结构如下所示：



EDAS的服务框架在具备基本的远程服务调用之外，还具有几下几个产品特性：

- 同机房优先

用于对跨机房间的HSF调用流量进行规划控制，能够保证HSF服务消费者在请求HSF服务时，优先选择与服务消费者同机房的服务提供者。

- 动态归组

将机器发布的HSF服务自动归入特定的分组。这样，不同分组中的HSF服务实例就组成了以组为单位的集群，针对某些特定的机器提供服务。

- 服务限流

作用于 HSF服务提供端 ，允许应用提供方指定某个接口的TPS，当单位时间内的TPS达到设定值时，该接口将停止对外提供服务，所有的请求都会被拦截，直到下一个刷新时间点。

- 三种调用方式

同步调用（默认）、异步调用、异步回调调用。

运维管控

内置多种运维与管控工具，用户能够实时获取应用全方位的运行信息，快速诊断问题并修复。

实时日志

在EDAS里查看一个应用的基本详情，左侧的菜单栏里有实时日志的功能，这个功能可以帮助开发者查看这个应用集群的任意一台机器上的日志文件的最新日志，帮助开发者快速诊断问题。

容器诊断

客户的应用是部署，并运行在我们的Ali-Tomcat容器中，我们的应用容器提供几个诊断功能：

- 内存变化

如下图所示，可以查看任何一台应用服务器堆内存的变化的历史曲线，方便诊断java内存问题

- 类加载

Java应用会依赖很多的jar包，这些jar包，哪些被应用容器加载，通过此功能，会展示出来。这个功能将帮助用户诊断类冲突问题

- 线程

列举容器里面的线程，各个线程的状态，以及线程名

- 连接器

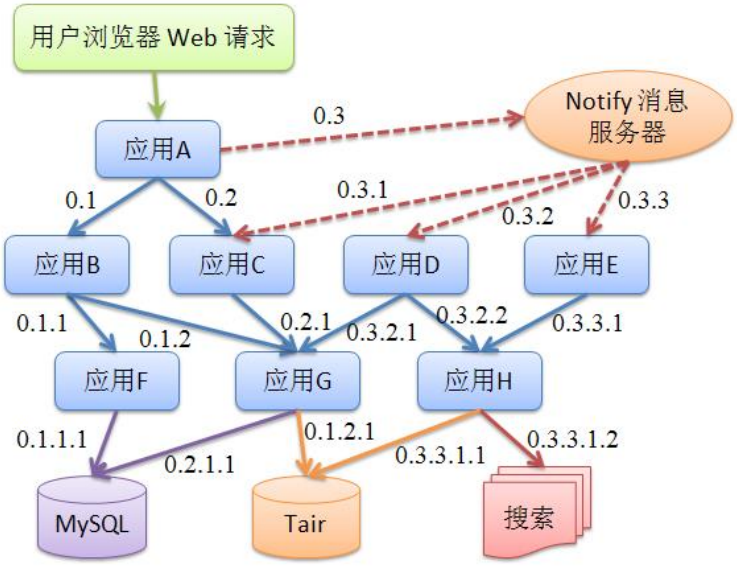
每个连接器当前处理的请求的一些数据

服务列表

每个应用发布了哪些HSF服务，消费了哪些HSF服务，可以通过此功能可以查看。此应用发布的服务，可以对其中一个比较重要的服务，不需要修改应用代码，在EDAS上进行动态分组配置，比如服务组A，服务组B，这样就可以控制不同的consumer连接不同服务组中的服务，方便隔离。

链路分析

在于服务化的架构体系中，对于任何一次业务上的请求，底层都会有很多远程的HSF服务化调用，访问数据库，收发消息，或者其它的操作，通过链路分析的功能，可以准确的描绘出上层一次用户请求，所经历的所有系统，服务，调用所花的时间，是否有错误。



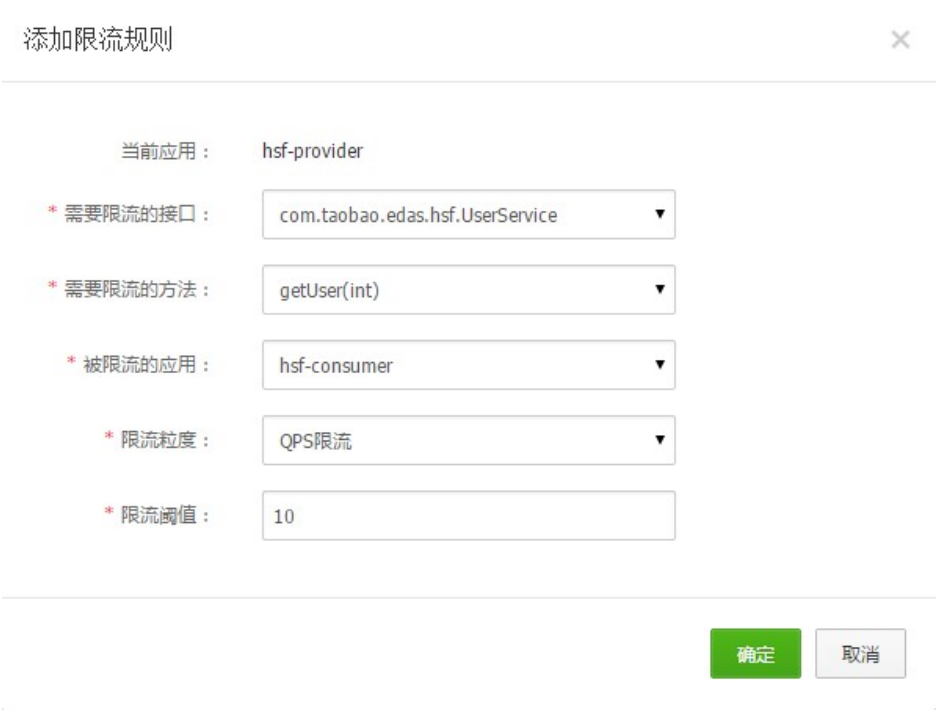
容量规划

一个应用系统，所提供的HSF服务，在当前的应用机器集群下，最大可以支撑多少的QPS调用，通过EDAS容量规划的自动压测功能，即可以测算出当前系统的容量。这个将方便我们对未来流量增长情况下，我们对应用所需机器数的一个预估，这将变得更加科学准确。

限流降级

一个应用系统，会提供很多的HSF服务，对于这些 HSF服务，可以配置限流降级规则，限制其它应用，对此服务的调用，可以从QPS，以及线程的两个维度来进行设置。这个功能，将可以帮助用户，在应对流量高峰时，系统能以最大的支撑能力平稳运行。

- 1.添加限流规则:



- 2.在应用当中添加如下的依赖:

```
<dependency>
  <groupId>cglib</groupId>
  <artifactId>cglib-nodep</artifactId>
  <version>2.2(或者更新)</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>org.springframework.aop</artifactId>
  <version>2.5.6(或者更新)</version>
</dependency>
```

3.添加接入方式:

给hsf provider限流，请引入以下Bean

```
<bean id="customEditorConfigurer"
        class="org.springframework.beans.factory.config.CustomEditorConfigurer">
<property name="propertyEditorRegistrars">
  <list>
    <bean class="com.taobao.csp.sentinel.entrypoint.entrance.HSFSpringProviderBeanRegistrar"/>
  </list>
</property>
</bean>
```

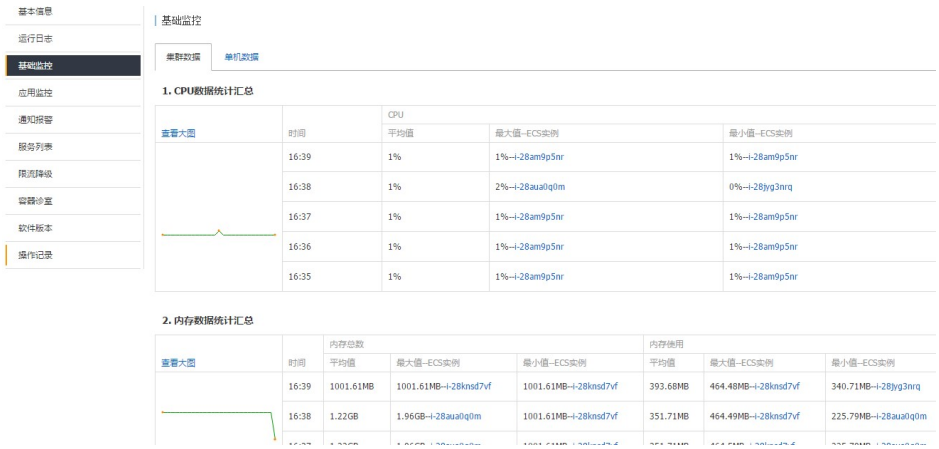
为web限流，请在web.xml里添加下列代码(对URL的限流,当前在EDAS里面暂不支持):

```
<filter>
  <filter-name>CommonFilter</filter-name>
```

```
  <filter-class>com.taobao.csp.sentinel.entrypoint.entrance.CommonFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>CommonFilter</filter-name>
  <url-pattern>*.htm</url-pattern>
</filter-mapping>
```

基础监控

应用的基础监控模块，将会帮助用户，以应用的维度，查看集群机器的平均load,平均网络读写，平均磁盘读写等指标，以及发现每一个指标，哪台机器出现了最高值。我们也提供单机的监控指标视图，方便客户了解单机各指标的历史发展变化。



应用监控

应用监控模块，会提供应用全方位的监控信息，主要包括的核心功能点如下：

- 系统概要

以下几个数据概要

- Http入口

http url的访问数据,您无须做任何配置,将自动采集。

需要在web.xml中作如下的配置:

```
  <filter>
        <filter-name>EagleEyeFilter</filter-name>
        <filter-class>com.taobao.eagleeye.EagleEyeFilter</filter-class>
        <init-param>
            <param-name>useLocalIp</param-name>
            <param-value>true</param-value>
        </init-param>
</filter>

  <filter-mapping>
    <filter-name>EagleEyeFilter</filter-name>
```

```
        <url-pattern>/*</url-pattern>
    </filter-mapping>
```

- 提供的HSF服务(我提供的HSF服务)

Hsf数据,您无须做任何配置,将自动采集

- HSF服务调用来源(谁调了我?)

Hsf数据,您无须做任何配置,将自动采集

- HSF服务调用依赖(我调了谁?)

Hsf服务调用依赖数据,您无须做任何配置,将自动采集

- ONS消息发送量

您只需要依赖ons-client,我们自动采集应用的消息发送量.

- DRDS数据库访问量

EDAS监控DRDS的访问量，只需要在原有使用方式下替换原来的数据库Driver为DRDS的com.taobao.tddl.driver.Driver。这里我们以连接池的方式演示完整的配置过程。 首先，添加连接池的maven以来（这里以alibaba的Druid连接池为例演示）

```
<dependency>
<groupId>com.alibaba</groupId>
    <artifactId>druid</artifactId>
    <version>0.2.8</version>
</dependency>
```

然后，在Spring的配置文件中配置连接池

```
<bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource"
init-method="init" destroy-method="close">
      <!-- 数据源驱动类可不写，Druid默认会自动根据URL识别DriverClass -->
      <property name="driverClassName" value="com.taobao.tddl.driver.Driver" />
      <!-- 基本属性 url、user、password -->
      <property name="url" value="${DB_URL}/>
      <property name="username" value="${DB_USERNAME}" />
      <property name="password" value="${DB_PWD}" />
      <!-- 配置初始化大小、最小、最大 -->
      <property name="initialSize" value="3" />
      <property name="minIdle" value="3" />
      <property name="maxActive" value="20" />
      <!-- 配置获取连接等待超时的时间 -->
      <property name="maxWait" value="60000" />
      <!-- 配置间隔多久才进行一次检测，检测需要关闭的空闲连接，单位是毫秒 -->
      <property name="timeBetweenEvictionRunsMillis" value="60000" />
      <!-- 配置一个连接在池中最小生存的时间，单位是毫秒 -->
      <property name="minEvictableIdleTimeMillis" value="300000" />
      <property name="validationQuery" value="SELECT 'x'" />
      <property name="testWhileIdle" value="true" />
      <property name="testOnBorrow" value="false" />
```

```
    <property name="testOnReturn" value="false" />
</bean>
```

注意：这里的Driver一定要使用我们DRDS的com.taobao.tddl.driver.Driver

EDAS服务框架使用指南

环境准备

服务框架环境准备

HSF安装

Ali-Tomcat 安装

下载并解压 Ali-Tomcat 即可。下载地址：http://edasteam.oss-cn-hangzhou.aliyuncs.com/public%2Ftaobao-tomcat-7.0.56.tar.gz

Pandora 安装

下载并解压 Pandora 到 Ali-Tomcat 的 deploy 目录即可。下载地址：http://edasteam.oss-cn-hangzhou.aliyuncs.com/public%2Ftaobao-hsf.tgz

环境配置

绑定 jmenv.tbsite.net 域名到对应的地址服务器（没有域名解析的情况下）

可以安装 Ali-tomcat 的eclipse插件，这样可以在 eclipse 中直接调试程序无需额外的打包

到此，HSF 的运行环境就安装完毕。

发布服务

发布服务

创建Web项目

以eclipse为例创建一个 maven web 项目。File -> New -> Project -> Maven Project -> maven-archetype-webapp ->

输入groupId、artifactId 连续Next。

添加Maven依赖

在项目 pom.xml 中添加如下依赖：

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-web</artifactId>
  <version>3.1.1.RELEASE</version>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>javax.servlet-api</artifactId>
  <version>3.0.1</version>
  <scope>provided</scope>
</dependency>
```

编写需要发布的服务

创建需要发布的服务接口，com.taobao.edas.test.SampleService

SampleService服务提供了一个 echo 的方法调用。编写实现类：com.taobao.edas.test.impl.SampleServiceImpl

```
public interface SampleService {
 String echo(String str);
}
public class SampleServiceImpl implements SampleService {
 @Override public String echo(String str) {
 return str;
 }
}
```

配置 Spring

在 web.xml 中配置 spring 的监听器：

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>classpath:config/applicationContext.xml</param-value>
</context-param>
<listener>
  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

在 resources 目录下面添加 spring 配置文件：config/applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN 3.0//EN"
        "http://www.springframework.org/dtd/spring-beans-2.0.dtd">
<beans>
   <import resource="classpath:config/providers-spring.xml"/>
</beans>
```

这里配置文件中包含了发布者的配置文件：config/providers-spring.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN 3.0//EN"
            "http://www.springframework.org/dtd/spring-beans-2.0.dtd">
<beans>
   <bean id="target" class="com.taobao.edas.test.impl.SampleServiceImpl"/>
   <bean id="sampleServiceProvider"
      class="com.taobao.hsf.app.spring.util.HSFSpringProviderBean" init-method="init">
      <property name="serviceInterface" value="com.taobao.edas.test.SampleService"/>
      <property name="serviceVersion" value="1.0.0"/>
      <property name="serviceGroup" value="HSF"/>
      <property name="target" ref="target"/>
   </bean>
</beans>
```

到此发布者就编写好了，运行 Maven 打包，生成项目 war 包。部署到 ali-tomcat 的 deploy 目录下。

运行 ali-tomcat/bin/startup.bat，就可以在 hsf 服务治理上查询到发布的服务了。

消费服务

消费服务

配置 Spring

在配置文件 config/applicationContext.xml 添加消费者配置：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN 3.0//EN" "http://www.springframework.org/dtd/spring-beans-2.0.dtd">
<beans>
   <import resource="classpath:config/providers-spring.xml"/>
   <import resource="classpath:config/consumers-spring.xml"/>
</beans>
```

在consumers-spring.xml配置：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN 3.0//EN" "http://www.springframework.org/dtd/spring-beans-2.0.dtd">
<beans>
   <bean id="sampleService" class="com.taobao.hsf.app.spring.util.HSFSpringConsumerBean" init-method="init">
      <property name="interfaceName" value="com.taobao.edas.test.SampleService"/>
      <property name="version" value="1.0.0"/>
```

```
    <property name="group" value="HSF"/>
  </bean>
</beans>
```

编写测试代码

已经完成了消费者的定义，下面创建 servlet 来调用测试代码进行测试：com.taobao.edas.test. HsfServlet

```
public class HsfServlet extends HttpServlet {
        @Override
        protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
            ApplicationContext ctx = WebApplicationContextUtils.getWebApplicationContext(
            req.getServletContext());
            SampleService sampleService = (SampleService) ctx.getBean("sampleService");
            resp.getWriter().println(Long.toString(System.currentTimeMillis()));
        }
    }
```

在 web.xml 中添加

```
<servlet>
  <servlet-name>hsf</servlet-name>
  <servlet-class>com.taobao.edas.test.HsfServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>hsf</servlet-name>
  <url-pattern>/hsf.htm</url-pattern>
</servlet-mapping>
```

打包测试

Maven 打包，部署，启动 ali-tomcat，打开浏览器访问：localhost:8080/hsf.htm.

注意：这里虽然消费成功，但是默认没有走远程调用。就是说，如果相同的 jvm 中提供了服务，默认是不会走网络远程调用了。就是说，即使其他机器提供了相同的服务也永远不会调用。

编辑 ali-tomcat/bin/catalina.bat 添加如下参数：

```
set JAVA_OPTS=%JAVA_OPTS% -Dhsf.client.localcall=false
```

这个参数禁用掉了本机优先调用策略。

如果这里是使用 Ali-tomcat 插件，直接用插件运行，无需打包。并且，-Dhsf.client.localcall=false

把这个参数在 eclipse 中加到 JVM 参数中。