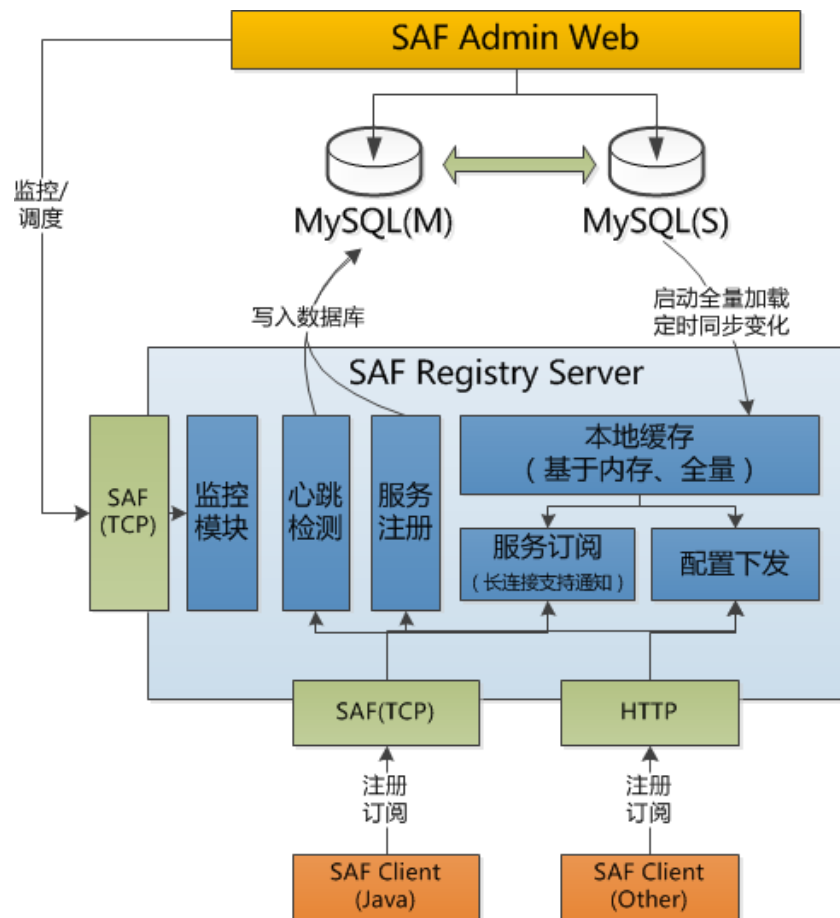


SAF2.1注册中心设计

被章耿添加，被杨志伟最后更新于十月 08, 2019

- [1.总体架构图](#)
- [2.接口设计](#)
 - [注册订阅服务](#)
 - [接口定义](#)
 - [参数描述](#)
 - [监控调度服务](#)
- [3.注册中心模块拆解](#)
 - [服务注册](#)
 - [服务订阅](#)
 - [配置下发](#)
 - [配置订阅](#)
 - [心跳检测](#)
 - [本地缓存](#)
 - [监控模块](#)
 - [黑白名单](#)
- [4.注册中心数据库设计](#)

1.总体架构图



2.接口设计

注册中心对外暴露2个服务，

1. 针对客户端的注册订阅服务
2. 针对管理端的监控调度服务

注册订阅服务

注：服务端限流设置为，每个 **appid** 10秒调用1次；请注意限流，可以本地做缓存；

接口定义

主要是针对客户端调用，

提供服务的注册，服务的订阅（长连接支持事件通知），心跳发送，jsf 配置下发等服务。

将发布两种协议的服务：

- 1. jsf协议，基于tcp的长连接，主要针对java客户端支持服务端发现服务列表变化后的主动通知（callback）。
- 2. http协议，基于http的短连接，主要针对jsf未实现的语言的客户端。服务订阅只能客户端定时去取，无法做到服务端主动通知。
- 3. com.jd.jsf.service.RegistryService 的JSF分组： reg

接口参考：

com.jd.jsf.service.RegistryService

```
1  package com.jd.jsf.service;
2
3  import java.util.List;
4
5  import com.jd.jsf.gd.error.RpcException;
6  import com.jd.jsf.gd.transport.Callback;
7  import com.jd.jsf.vo.HbResult;
8  import com.jd.jsf.vo.Heartbeat;
9  import com.jd.jsf.vo.JsfUrl;
10 import com.jd.jsf.vo.SubscribeUrl;
11
12 /**
13  * 注册中心对外发布服务接口
14  * jd.com JSF
15  * @author baoningtian
16  * @email baoningtian@jd.com
17  * @Date 2014-5-28 上午10:25:21
18  * RegistryService
19  */
20 public interface RegistryService {
21
22     /**
23      * provider/consumer注册<br>
24      * 处理契约: <br>
25      * 1. JsfUrl的ip,port,pid,iface,alias,protocol(参考ProtocolType枚举),timeout,stTime为必填
26 项, 这些是provider/consumer端的配置信息<br>
27      * 2. JsfUrl中的attrs应该设置appath=客户端的应用路径, safVersion=210, jsfVersion=1000,
28      serialization=msgpack(参考CodecType枚举)<br>
29      * 3. 如果为provider时, 当JsfUrl中的attrs设置了check=false 且是第一次上线(即 re-reg不为true时),
30      则将状态设置为下线, 用做灰度上线<br>
31      * 4. 如果为provider时, JsfUrl中的attrs设置weight=provider权重值<br>
32      * 5. 如果为consumer时, JsfUrl中的attrs应该设置consumer=1<br>
33      * 6. 如果为consumerGroup时, JsfUrl中attrs应该设置consumerGroup=consumer端的consumerGroup的值
34      <br>
35      * 7. 返回结果中, 会在JsfUrl中给出insKey值<br>
36      * @param jsfUrl
37      * @return
38      * @throws Exception
39      */
40     JsfUrl doRegister(JsfUrl jsfUrl) throws RpcException;
41
42     /**
43      * provider/consumer批量注册<br>
44      * 处理契约: <br>
45      * 1. JsfUrl的ip,port,pid,iface,alias,protocol(参考ProtocolType枚举),timeout,stTime为必填
46 项, 这些是provider/consumer端的配置信息<br>
47      * 2. JsfUrl中的attrs应该设置appath=客户端的应用路径, safVersion=210, jsfVersion=1000,
48      serialization=msgpack(参考CodecType枚举)<br>
49      * 3. 如果为provider时, 当JsfUrl中的attrs设置了check=false 且是第一次上线(即 re-reg不为true时),
50      则将状态设置为下线, 用做灰度上线<br>
51      * 4. 如果为provider时, JsfUrl中的attrs设置weight=provider权重值<br>
52      * 5. 如果为consumer时, JsfUrl中的attrs应该设置consumer=1<br>
53      */
54 }
```

```

54     * 6. 如果为consumerGroup时, Jsful中attrs应该设置consumerGroup=consumer端的consumerGroup的值
55     <br>
56     * 7. 返回结果中, 会在Jsful中给出insKey值<br>
57     *
58     * 异常说明: 只要jsfulList中有一个provider或consumer出现问题就取消这次的所有注册
59     * @param jsfulList
60     * @return
61     * @throws Exception
62     */
63     List<Jsful> doRegisterList(List<Jsful> jsfulList) throws RpcException;
64
65     /**
66     * 检查注册列表的方法
67     * 检查通过返回true, 不通过, 直接抛异常
68     * @param jsful
69     * @return
70     * @throws RpcException
71     */
72     boolean doCheckRegister(Jsful jsful) throws RpcException;
73
74     /**
75     * provider/consumer取消注册<br>
76     * 处理契约: <br>
77     * 1. Jsful的ip,port,pid,iface,alias,protocol(参考ProtocolType枚举), insKey为必填项, 这些是
78     provider/consumer端的配置信息<br>
79     * 2. 如果为consumer时, Jsful中的attrs应该设置consumer=1<br>
80     * 3. 返回为true成功, false失败<br>
81     * @param jsful
82     * @return
83     * @throws Exception
84     */
85     boolean doUnRegister(Jsful jsful) throws RpcException;
86
87     /**
88     * 检查反注册列表的方法
89     * 检查通过返回true, 不通过, 直接抛异常
90     * @param jsful
91     * @return
92     * @throws RpcException
93     */
94     boolean doCheckUnRegister(Jsful jsful) throws RpcException;
95
96     /**
97     * provider/consumer批量取消注册<br>
98     * 处理契约: <br>
99     * 1. Jsful的ip,port,pid,iface,alias,protocol(参考ProtocolType枚举), insKey为必填项, 这些是
100    provider/consumer端的配置信息<br>
101    * 2. 如果为consumer时, Jsful中的attrs应该设置consumer=1<br>
102    * 3. 返回为true成功, false失败<br>
103    * 异常说明: 只要jsfulList中有一个provider或consumer出现问题就取消这次的所有注册
104    * @param jsfulList
105    * @return
106    * @throws Exception
107    */
108    boolean doUnRegisterList(List<Jsful> jsfulList) throws RpcException;
109

```

```

110  /**
111   * consumer订阅服务provider<br>
112   * 处理契约: <br>
113   * 1. JsfUrl的iface,insKey,alias,ip,protocol为必填项<br>
114   * 2. subscribeData是callback对象,用于推送, java客户端使用<br>
115   * 3. 返回SubscribeUrl中sourceUrl中, 会给出dataVersion的值, 以便客户端进行订阅版本比对<br>
116   * @param jsfUrl
117   * @param subscribeData
118   * @return
119   * @throws RpcException
120   */
121   SubscribeUrl doSubscribe(JsfUrl jsfUrl, Callback<SubscribeUrl, String> subscribeData)
122   throws RpcException;
123
124  /**
125   * 取消订阅, 删除callback<br>
126   * 处理契约: <br>
127   * 1. JsfUrl的iface,insKey为必填项<br>
128   * 2. 返回为true成功, false失败<br>
129   * @param jsfUrl
130   * @return
131   * @throws RpcException
132   */
133   boolean doUnSubscribe(JsfUrl jsfUrl) throws RpcException;
134
135  /**
136   * 读取provider信息<br>
137   * 处理契约: <br>
138   * 1. JsfUrl的iface,insKey,alias,ip,protocol,dataVersion为必填项<br>
139   * 2. 返回SubscribeUrl中sourceUrl中, 会给出dataVersion的值, 以便客户端进行订阅版本比对. 如果
140   dataVersion没有变化, 就不返回provider列表了<br>
141   * @param jsfUrl
142   * @return
143   * @throws RpcException
144   */
145   SubscribeUrl lookup(JsfUrl jsfUrl) throws RpcException;
146
147  /**
148   * 读取provider信息<br>
149   * 处理契约: <br>
150   * 1. JsfUrl的iface,insKey,alias,ip,protocol,dataVersion为必填项<br>
151   * 2. 返回SubscribeUrl中sourceUrl中, 会给出dataVersion的值, 以便客户端进行订阅版本比对. 如果
152   dataVersion没有变化, 就不返回provider列表了<br>
153   * @param list
154   * @return
155   * @throws RpcException
156   */
157   List<SubscribeUrl> lookupList(List<JsfUrl> list) throws RpcException;
158
159  /**
160   * 实例心跳<br>
161   * 处理契约: <br>
162   * 1. heartbeat的insKey为必填项<br>
163   * 2. 返回结果HbResult中, config包含recover, 说明需要客户端重新注册服务; config包含callback, 说明需
164   要重新注册callback<br>
165   * @param heartbeat

```

```

166     * @return
167     * @throws Exception
168     */
169     HbResult doHeartbeat(Heartbeat heartbeat) throws RpcException;
170
171     /**
172     * 批量实例心跳<br>
173     * @param heartbeatList
174     * @return
175     * @throws RpcException
176     */
177     List<HbResult> doHeartbeatList(List<Heartbeat> heartbeatList) throws RpcException;
178
179     /**
180     * 订阅配置信息，分为客户端配置和接口配置
181     * subscribeData是callback对象，用于推送，java客户端使用
182     * 订阅客户端配置处理契约：<br>
183     * 1. JsfUrl中的ip,pid,stTime为必填项
184     * 2. 返回的配置信息，请参看管理端中的参数配置。此外返回结果中的dataVersion的值，以便客户端进行订阅版本
185 比对
186     * 订阅接口配置处理契约：<br>
187     * 1. JsfUrl中的ip,pid,stTime,iface为必填项
188     * 2. 返回的配置信息，请参看管理端中的服务管理的属性配置。此外返回结果中的dataVersion的值，以便客户端进
189 行订阅版本比对
190     * @param jsfUrl
191     * @param subscribeData
192     * @return
193     * @throws RpcException
194     */
195     JsfUrl subscribeConfig(JsfUrl jsfUrl, Callback<SubscribeUrl, String> subscribeData)
196     throws RpcException;
197
198     /**
199     * 获取配置信息
200     * 获取配置处理契约：<br>
201     * 1. JsfUrl中的ip,pid,stTime为必填项
202     * 2. 返回的配置信息，请参看管理端中的参数配置。此外返回结果中的dataVersion的值，以便客户端进行订阅版本
203 比对。如果dataVersion没有变化，就不返回配置信息了
204     * 订阅接口配置处理契约：<br>
205     * 1. JsfUrl中的ip,pid,stTime,iface为必填项
206     * 2. 返回的配置信息，请参看管理端中的服务管理的属性配置。此外返回结果中的dataVersion的值，以便客户端进
207 行订阅版本比对。如果dataVersion没有变化，就不返回配置信息了
208     * @param jsfUrl
209     * @return
210     * @throws RpcException
211     */
212     JsfUrl getConfig(JsfUrl jsfUrl) throws RpcException;
213
214     /**
215     * 获取配置信息列表
216     * 获取配置处理契约：<br>
217     * 1. JsfUrl中的ip,pid,stTime为必填项
218     * 2. 返回的配置信息，请参看管理端中的参数配置。此外返回结果中的dataVersion的值，以便客户端进行订阅版本
219 比对。如果dataVersion没有变化，就不返回配置信息了
220     * 订阅接口配置处理契约：<br>
221     * 1. JsfUrl中的ip,pid,stTime,iface为必填项

```

```
222      * 2. 返回的配置信息，请参看管理端中的服务管理的属性配置。此外返回结果中的dataVersion的值，以便客户端进
223      行订阅版本比对。如果dataVersion没有变化，就不返回配置信息了
      * @param list
      * @return
      * @throws RpcException
      */
      List<JsfUrl> getConfigList(List<JsfUrl> list) throws RpcException;

      /**
      * 获取服务实例（即：provider或consumer属性）
      * @param jsfUrl
      * @return
      * @throws RpcException
      */
      JsfUrl lookupServiceInsAttrs(JsfUrl jsfUrl) throws RpcException;

      /**
      * 批量获取服务实例（即：provider或consumer属性）
      * @param jsfUrlList
      * @return
      * @throws RpcException
      */
      List<JsfUrl> lookupServiceInsAttrsList(List<JsfUrl> jsfUrlList) throws RpcException;
    }
}
```

参数描述

bean: com.jd.jsf.vo.JsfUrl

| 成员变量 | 是否必填 | 说明 |
|----------------------|------|----|
| private String ip | 是 | |
| private int port | 是 | |
| private int pid | 是 | |
| private String iface | 是 | |
| private String alias | 是 | |

| 成员变量 | 是否必填 | 说明 |
|--|------|---|
| <code>private int protocol</code> | 是 | 枚举值： <code>consumer(0), @Deprecated saf(1), jsf(1), rest(2), dubbo(3), webservice(4), jaxws(5), @Deprecated jaxrs(6), @Deprecated hessian(7), @Deprecated thrift(8), http(9), grpc(11);</code> |
| <code>private Map<String, String> attrs</code> | 是 | 扩展属性；如果没有属性填写： "attrs":{}，见下面【示例】 |
| <code>private int timeout</code> | 是 | 默认请使用：5000，不要使用0 |
| <code>private boolean random</code> | 是 | 填：true |
| <code>private long stTime</code> | 是 | 提供服务的进程启动时间：精确到毫秒；注意：服务进程没有重启的话，这个时间不能变； |
| <code>private String insKey</code> | 否 | 进程key（请求参数不要填写；） |
| <code>private long dataVersion</code> | 否 | 如果调用lookup，需要传递上次lookup返回值中的版本，版本： SubscribeUrl.sourceUrl.dataVersion |
| 示例 | 注册示例 | <code>curl -H "Content-Type: application/json" H "token: \${tokenvalue}" -X POST --data '[{"ip":"192.168.141.13","port":22000,"pid":13110,"iface":"com.jd.testjsf.HelloService","protocol":11,"timeout":5000,"stTime":1565339617458,"alias":"gscr","attrs":{}}]' http://192.168.32.228:40660/com.jd.jsf.service.RegistryService/reg/doRegister</code> |
| | | |

bean: com.jd.jsf.vo.SubscribeUrl

| 成员变量 | 是否必填 | 说明 |
|---------------------------------------|------|-----|
| <code>private JsfUrl sourceUrl</code> | 是 | 调用者 |

| 成员变量 | 是否必填 | 说明 |
|--|------|---|
| <code>private List<JsfUrl> providerList</code> | 否 | 获取 provider列表使用； 当没有provider列表或者provider列表没变化时，返回null或空列表；需要调用端兼容； |
| <code>private Map<String, String> config;</code> | 否 | 获取配置信息使用 当没有配置或者配置没变化时，返回null或空列表；需要调用端兼容； |
| <code>private int type</code> | 是 | 类型 <code>public static final int CHECK_RUOK = 0;</code> // 检查是否正常的空事件，正常就返回imok <code>public static final int PROVIDER_ADD = 1;</code> // 服务列表增加，增量变化 <code>public static final int PROVIDER_DEL = 2;</code> // 服务列表删除，增量变化（为了安全，删光服务列表的操作将被忽略） <code>public static final int PROVIDER_UPDATE_ALL = 3;</code> // 服务列表变化，全量变化（为了安全，删光服务列表的操作将被忽略） <code>public static final int PROVIDER_CLEAR = 4;</code> // 黑白名单时使用，清空服务列表 <code>public static final int PROVIDER_FORCE_DEL = 5;</code> // 服务列表删除，增量变化（不会判断是否被删光） <code>public static final int PROVIDER_FORCE_UPDATE_ALL = 6;</code> // 服务列表删除，增量变化（不会判断是否被删光） |

Bean: com.jd.jsf.vo.Heartbeat

| 成员变量 | 是否必填 | 说明 |
|---|------|-------------------|
| <code>private String insKey</code> | 是 | 使用注册接口返回值中的insKey |
| <code>private Map<String, String> config</code> | 否 | 不填，即：使用null |

Bean: com.jd.jsf.vo.HbResult

| 成员变量 | 是否必填 | 说明 |
|--|------|------------|
| <code>private String insKey</code> | 是 | |
| <code>private List<String> config</code> | 否 | http调用忽略此值 |

主要针对管理端，

提供注册中心服务状态的监控，注册中心的调度，配置中心一些参数的下发（也可以注册中心从db定时加载）等其它功能。

由于管理端为Java的，所以只要发布saf协议接口。

接口参考：

```
public interface ControlService {
    /**
     * 查询状态，包括连接数，请求数，数据文件大小等信息
     *
     * @param safUrl
     */
    public StatusInfo stat(SafUrl safUrl);
    /**
     * 暂停<br>
     * 已有请求不中断，不再接受新的请求，新请求返回暂停服务标识
     *
     * @param safUrl
     * @return
     */
    public StatusInfo pause(SafUrl safUrl);
    /**
     * 下线<br>
     * 已有请求中断，客户端将重连，不再接受新的请求，新请求返回下线服务标识
     *
     * @param safUrl
     * @return
     */
    public StatusInfo offline(SafUrl safUrl);

    /**
     * 配置下发，在saf注册中心运行时，可以改变一些配置
     *
     * @param safUrls
     *          查询的信息
     * @return 带更新的配置信息
     */
    public List<SafUrl> pushConfig(List<SafUrl> safUrls);
}
```

3.注册中心模块拆解

数据库结构大致如下：

saf_interface 接口表

saf_interface_group 接口分组（服务别名）表，一行数据表示唯一的一组服务

saf_interface_node 接口节点表，区分服务端和客户端

saf_serviceinfo 接口表

saf_server_alias 接口分组表，表示用户

saf_server 服务端节点表

saf_client 客户端节点表

saf_routerule 路由规则表

saf_ipwb ip黑白名单表

服务注册

Provider客户端调用此方法完成provider节点注册：

- 1. 写入saf_server
- 2. 更新saf_server_alias，表示有数据变更，等待同步
- 3. 是否按照实例的方式进行注册？？

Consumer客户端调用此方法完成consumer节点注册：

- 1. 写入saf_client

服务订阅

Consumer客户端调用此方法完成服务节点订阅：

- 1. 返回内存中对应的provider列表（路由后的provider列表。并检查当前订阅者是否在黑白名单中）。如何做到增量订阅？？
- 2. 保留订阅到服务别名的listener（等变化的时候通知客户端）

配置下发

简单的配置查询接口。

提供定时检查配置，此配置为一些全局的运行时配置，例如心跳间隔，监控数据发送间隔等数据。

可以和心跳包结合返回。

配置订阅

subscribeconfig/getconfig返回的属性包括

- 1、接口本身属性；
- 2、param路由；
- 3、alias路由；
- 4、黑白名单；
- 5、mock；
- 6、conusmer限流配置信息

心跳检测

客户端调用此方法发送心跳信息。

如果是长连接，可以只发少量数据。

如果是短连接，则需要带上客户端标识（ip、端口、pid等）。

步骤如下：

- 1. 得到客户端标识
- 2. 更新服务节点的最后心跳时间
- 3. 同时有一个worker 检查最后心跳时间，如果长时间未心跳，则置为死亡状态，发报警。

本地缓存

注册中心实例内存中缓存全部的服务列表和配置数据。

- 1. 在系统启动时候一次性加载全部数据到内存中
- 2. 数据更新采取一个定时服务每隔指定时间去数据库查询最新数据，同步最新数据。
- 3. 如果有变化，则需要触发回调机制。

监控模块

注册中心的统计功能，包括连接数，心跳数，注册数等。

同时有一些开关，包括状态开发，是否提供服务等。

黑白名单

如果没有黑名单，即，黑名单默认 `invoke.blacklist=""`;

如果没有白名单，即，白名单默认 `invoke.whitelist=""`

设置黑白名单后， 黑白名单应该是推送到provider端的， provider端校验consumer是否能访问。这个是sdk逻辑

设置黑白名单后，注册中心会在1分钟内，下发黑白名单到consumer和provider端。但是consumer应该是不检查黑白名单。

设置黑白名单后，注册中心不会自动触发服务发现逻辑，即：不会立即根据黑白名单重新推送provider给consumer，但是consumer重新订阅或者因为其他原因触发推送provider的时候，注册中心会根据黑白名单，检查provider列表。

4.注册中心数据库设计

registry.sql-注册中心表结构

```
CREATE TABLE saf_serviceinfo (
  serviceinfo_id int(10) NOT NULL AUTO_INCREMENT,
  interface_id varchar(200) NOT NULL UNIQUE,
  application varchar(64) NOT NULL,
  is_important tinyint(1) NOT NULL,
  owner_group varchar(32) NOT NULL,
  data_version varchar(32) NOT NULL,
  subscribe_sttype tinyint(2) DEFAULT 0 NOT NULL comment '0-订阅online和offline状态, 1-只能订阅online',
  serviceinfo_desc varchar(512),
  create_date datetime NOT NULL,
  PRIMARY KEY (serviceinfo_id)) comment='接口表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_serviceinfo_depend (
  id int(10) NOT NULL AUTO_INCREMENT,
  serviceinfo_id int(10) NOT NULL,
  serviceinfo_childid int(10) NOT NULL,
  PRIMARY KEY (id)) comment='接口关联表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_group (
  group_id int(10) NOT NULL AUTO_INCREMENT,
  group_name varchar(64) NOT NULL UNIQUE comment '该字段做唯一值约束，并且字段的数据需要做大小写区分。该字段的BINARY需要手动添加，不能通过VP工具生成sql',
  group_desc varchar(512),
  creator_id int(10) NOT NULL,
  create_date datetime NOT NULL,
  update_date datetime NOT NULL,
  PRIMARY KEY (group_id)) comment='group表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_server_group (
  id int(10) NOT NULL AUTO_INCREMENT,
  server_id int(10) NOT NULL,
  serviceinfo_id int(10) NOT NULL,
  group_id int(10) NOT NULL,
  src_type tinyint(2) NOT NULL comment '添加类型: 1-proxy, 2-manual',
  PRIMARY KEY (id)) comment='server和group关联表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_server (
  server_id int(10) NOT NULL AUTO_INCREMENT,
  server_uniquekey varchar(128) NOT NULL comment 'ip_port_group_version_protocol_serviceInfoId，将这6个字段作为provider的唯一判断和索引，在此不做唯一约束，通过程序进行判断',
  server_group varchar(64) NOT NULL comment '默认group',
  server_version varchar(16) NOT NULL comment 'server的版本号',
  server_ip varchar(32) NOT NULL comment 'server的IP地址',
  server_port int(6) NOT NULL comment 'server的端口号',
  server_pid int(8) comment 'server的PID',
  server_token varchar(32),
  saf_version varchar(16) NOT NULL,
  server_status tinyint(2) NOT NULL,
  server_room tinyint(2) NOT NULL,
  server_timeout int(8),
  server_subnet tinyint(2),
  server_weight int(6),
```

```

server_monitor    tinyint(2),
server_apppath    varchar(128) NOT NULL,
protocol          varchar(16),
protocol_context  varchar(32),
serviceinfo_id    int(10) NOT NULL,
is_random         tinyint(1) NOT NULL,
src_type          tinyint(2) NOT NULL comment 'data type: 1-proxy, 2-manual, 3-zookeeper 用于数据同步',
attr_url          varchar(255) comment '保存safurl中的attr参数, 过滤掉一些在表中已经存在的参数',
safurl_desc       varchar(1023),
create_date       datetime NOT NULL,
update_date       datetime NOT NULL,
PRIMARY KEY (server_id) comment='服务提供者表' CHARACTER SET UTF8 engine innodb;

```

```

CREATE TABLE saf_client (
  client_id        int(10) NOT NULL AUTO_INCREMENT,
  client_uniquekey varchar(128) NOT NULL comment 'ip_pid_group_version_serviceInfoId , 将这5个字段作为consumer的唯一判断和索引。为了降低唯一约束对在此不做唯一约束, 通过程序进行唯一判断',
  client_ip        varchar(32) NOT NULL,
  client_apppath    varchar(128),
  client_pid       int(8),
  client_group      varchar(64) NOT NULL,
  client_token      varchar(32),
  client_version    varchar(16) NOT NULL,
  saf_version       varchar(16) NOT NULL,
  client_status     tinyint(2) NOT NULL,
  serviceinfo_id    int(10) NOT NULL,
  group_id         int(10) NOT NULL,
  src_type          tinyint(2) NOT NULL comment '数据来源: 1-proxy, 2-manual, 3-zookeeper',
  safurl_desc       varchar(1023),
  start_time        varchar(31) NOT NULL comment '节点启动时间',
  create_date       datetime NOT NULL,
  update_date       datetime NOT NULL,
  PRIMARY KEY (client_id) comment='服务调用者表' CHARACTER SET UTF8 engine innodb;

```

```

CREATE TABLE saf_routerule (
  routerule_id     int(10) NOT NULL AUTO_INCREMENT,
  interface_id      varchar(200) NOT NULL,
  source_ip         varchar(32),
  source_version    varchar(32),
  source_group      varchar(32),
  source_room       varchar(32),
  source_subnet     varchar(32),
  dest_ip           varchar(32),
  dest_port         varchar(8),
  dest_version      varchar(16),
  dest_group        varchar(32),
  dest_room         varchar(32),
  dest_subnet       varchar(32),
  expression        varchar(128),
  routerule_type    tinyint(2) NOT NULL,
  is_valid          tinyint(1) NOT NULL,
  PRIMARY KEY (routerule_id) comment='路由表' CHARACTER SET UTF8 engine innodb;

```

```

CREATE TABLE saf_room (
  id               int(10) NOT NULL AUTO_INCREMENT,

```



```
computer_room tinyint(2) NOT NULL,  
ip_section    varchar(32) NOT NULL,  
ip_regular    varchar(255),  
PRIMARY KEY (id)) comment='机房表' CHARACTER SET UTF8 engine innodb;
```

```
CREATE INDEX query_index_server_uniquekey  
ON saf_server (server_uniquekey);  
CREATE INDEX query_index_client_uniquekey  
ON saf_client (client_uniquekey);
```

admin.sql 管理端表结构

```
CREATE TABLE saf_upload_jar_info (
  id          int(10) NOT NULL AUTO_INCREMENT comment '主键',
  erp_id      varchar(63) NOT NULL,
  upload_user_name varchar(63) NOT NULL comment '上传用户名, 冗余字段',
  create_date datetime NOT NULL,
  update_date datetime NOT NULL,
  jar_version_name varchar(63) NOT NULL comment '上传jar包名称',
  save_path   varchar(127) NOT NULL comment '保存路径',
  md_five_value varchar(127) NOT NULL comment '文件上传时md5值',
  PRIMARY KEY (id)) comment='上传jar包信息表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_serviceinfo_jar (
  id          int(10) NOT NULL AUTO_INCREMENT comment '主键',
  ip_regular  varchar(63) NOT NULL comment 'ip表达式',
  create_date datetime NOT NULL,
  update_date datetime NOT NULL,
  jar_version_name varchar(255) NOT NULL comment 'jar包名称',
  app_path    varchar(255) NOT NULL comment '实例路径, 对应一个jvm',
  PRIMARY KEY (id)) comment='下载jar规则表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_monitor_property (
  id          int(10) NOT NULL AUTO_INCREMENT comment '自增id',
  interface_id varchar(200) NOT NULL comment '接口名称',
  monitor_type varchar(20) NOT NULL comment '监控类型',
  monitor_value varchar(511) NOT NULL comment '值',
  PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_monitor_proxy (
  id          int(10) NOT NULL AUTO_INCREMENT,
  proxy_ip    varchar(32) NOT NULL,
  port        int(6) NOT NULL,
  monitor_type int(10) NOT NULL comment '1-每分钟注册数, 2-每分钟心跳数, 3-当前连接数',
  monitor_time datetime NOT NULL,
  m_value     int(10) NOT NULL,
  create_date datetime NOT NULL,
  PRIMARY KEY (id)) comment='监控proxy实例' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_service_tracelog (
  id          int(10) NOT NULL AUTO_INCREMENT,
  interfaceID varchar(200) comment '接口名字',
  group_name  varchar(64) comment '组',
  version     varchar(16) comment '版本',
  ip_addr     varchar(32) comment 'ip地址',
  port        int(6) comment '端口',
  protocol    varchar(16) comment '协议',
  pid         int(6) comment '进程id',
  saf_version varchar(16),
  saf_action  tinyint(1) comment 'saf上下线 0下线 1上线',
  isprovider  tinyint(1) comment '是否服务端 1服务端0客户端',
  url         varchar(1024),
  create_time datetime comment '创建时间',
  update_time datetime comment '更新时间',
  params      varchar(120) comment '拓展字段, 用于功能拓展使用, 保存格式为key:value;',
  msg_src     varchar(20) comment '日志来源, 1 状态扫描, 2 服务注册',
  PRIMARY KEY (id)) comment='接口上下线记录表' CHARACTER SET UTF8 engine innodb;

CREATE TABLE saf_trace_tps (
  id          int(10) NOT NULL AUTO_INCREMENT comment 'id',
```

```

interface_id varchar(200) NOT NULL comment '接口名',
method       varchar(80) NOT NULL comment '方法名',
avg_tps      double NOT NULL comment '平均tps',
total_tps    double NOT NULL comment '总tps',
create_time  timestamp DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP NOT NULL comment '发送
时间',
PRIMARY KEY (id)) CHARACTER SET UTF8;
CREATE TABLE saf_trace_perform (
id          int(10) NOT NULL AUTO_INCREMENT comment '自增id',
host       varchar(80) NOT NULL comment 'ip',
port       int(6) NOT NULL comment '端口',
interface_id varchar(200) NOT NULL comment '接口名',
method     varchar(80) NOT NULL comment '方法名',
elapsed    int(5) NOT NULL comment '耗时',
send_time  timestamp DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP NOT NULL comment '发送
时间',
PRIMARY KEY (id)) comment='单个接口trace性能' CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_monitor_error (
id          int(11) NOT NULL AUTO_INCREMENT,
interface_id varchar(200) NOT NULL comment '接口名',
method     varchar(100) NOT NULL comment '方法名',
`group`    varchar(60) comment '组名',
version    varchar(60) comment '版本',
error_name varchar(100) comment '错误类型',
times      int(10) comment '异常次数',
provider   varchar(100) NOT NULL comment '提供者',
is_provider tinyint(1) NOT NULL comment '是否提供者',
port       int(5) comment '端口',
error_message varchar(5000) comment '异常信息',
error_stack varchar(10000) comment '异常堆栈',
create_time timestamp DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP NOT NULL comment '发
送时间',
consumer   varchar(100) NOT NULL comment '调用者',
protocol   varchar(40) comment '协议',
PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_statistics (
id          int(10) NOT NULL AUTO_INCREMENT,
weekend     tinyint(2) NOT NULL comment '第几周',
selection   tinyint(2) NOT NULL comment '统计标准 实例: ip',
sum         int(4) NOT NULL comment '数量',
create_date timestamp DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP NOT NULL,
PRIMARY KEY (id)) CHARACTER SET UTF8;
CREATE TABLE saf_invocation_monitor (
id          int(10) NOT NULL AUTO_INCREMENT comment '自增id',
client_ip   varchar(40),
host       varchar(40),
`group`    varchar(60),
interface_id varchar(200),
method     varchar(80),
version    varchar(40),
record_time datetime comment '发送时间',
protocol   varchar(20),
application varchar(80),
port       int(6) comment '服务端口',
success_num int(10) comment '调用成功数量',
fail_num   int(10) comment '调用失败数量',

```

```

concurrent    int(11) comment '当前并发数',
elapsed       int(11) comment '调用服务平均耗时',
max_elapsed   int(11) comment '最长耗时',
record_by     tinyint(1) comment '记录着 0 provider 1 consumer',
save_time     datetime,
PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_dept (
  id           int(6) NOT NULL AUTO_INCREMENT comment '主键',
  ref_id       int(6) NOT NULL comment '所属一级部门',
  dept_name    varchar(200) comment '部门名称',
  remark       varchar(400) comment '备注',
  create_time  datetime comment '创建时间',
  update_time  datetime comment '更改时间',
  PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_user_action (
  id           int(10) NOT NULL AUTO_INCREMENT,
  user_name    varchar(32) NOT NULL comment '用户名',
  user_erpid   varchar(32) NOT NULL comment 'erp帐户',
  action_type  varchar(32) NOT NULL comment '操作类型',
  create_time  datetime NOT NULL,
  detail       varchar(5000),
  PRIMARY KEY (id)) comment='用户操作日志' CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_alarmrule_user (
  id           int(10) NOT NULL AUTO INCREMENT comment '记录ID',
  rule_id      int(11) comment '所触发的规则(外键)',
  user_id      int(11) comment '规关联用户(外键)',
  PRIMARY KEY (id)) comment='报警规则用户关联表' CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_alarmrule (
  id           int(10) NOT NULL AUTO_INCREMENT comment '规则ID',
  name         varchar(128) NOT NULL comment '规则名称',
  object_type  varchar(32) comment '对象类型(service-p-count,service-c-count,service-up-down)',
  object_id    varchar(200) comment '对象ID',
  threshold    varchar(256) comment '阈值',
  durable_time int(11) comment '最长允许连续异常时间',
  alarm_min_interval int(11) comment '警报最短间隔时间,用于防止频繁发警报',
  create_date  datetime comment '规则创建时间',
  created_user varchar(32) comment '规则创建人',
  modified     datetime comment '规则修改时间',
  modified_user varchar(30) comment '规则修改人',
  status       tinyint(3) comment '状态(1:启用;2:停用;3:删除)',
  supplement   varchar(1024) comment '规则补充字段,用于拓展(key:value形似存储)',
  ip_list      varchar(1024) comment '关联ip,多个逗号分隔',
  common       tinyint(1) comment '标记是否是共有规则',
  PRIMARY KEY (id)) comment='报警规则表' CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_alarmevent (
  id           int(10) NOT NULL AUTO_INCREMENT comment '报警记录ID',
  object_type  varchar(32) NOT NULL comment '对象类型(service-p-count,service-c-count,service-up-down)',
  object_id    varchar(200) NOT NULL comment '对象ID',
  rule_id      int(11) NOT NULL comment '所触发的规则(外键)',
  detail       varchar(2048) NOT NULL comment '警告内容',
  status       tinyint(3) NOT NULL comment '状态',
  created      datetime NOT NULL comment '创建时间',
  sent_time    datetime comment '发送时间',
  try_times    tinyint(10) NOT NULL comment '尝试发送报警次数',
  task_group   int(11) comment '任务组号',


```

```

PRIMARY KEY (id)) comment='报警任务,status为0没有处理,1处理但不报警,2处理且报警' CHARACTER SET UTF8
engine innodb;
CREATE TABLE saf_suggestion (
  id          int(10) NOT NULL AUTO_INCREMENT comment '主键ID',
  title       varchar(128) comment '建议名称',
  context     varchar(2048) comment '内容',
  create_time datetime comment '创建时间',
  create_user_id int(11) comment '创建人',
  update_time datetime comment '修改时间',
  update_user_id int(11) comment '修改人',
  parent_id   int(11) comment '父id',
  PRIMARY KEY (id)) comment='建议问题表' CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_param (
  id          int(10) NOT NULL AUTO_INCREMENT,
  param_key   varchar(32) NOT NULL,
  param_name  varchar(32) NOT NULL,
  param_value varchar(128),
  param_type  tinyint(2) NOT NULL,
  note       varchar(512),
  PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_notice (
  id          int(10) NOT NULL AUTO_INCREMENT comment '自增id',
  content     text NOT NULL comment '公告内容',
  title       varchar(128) NOT NULL comment '标题',
  is_top      tinyint(1) comment '是否置顶',
  default_show tinyint(1) comment '是否默认显示',
  create_date datetime NOT NULL comment '创建时间',
  update_date datetime NOT NULL comment '更新时间',
  PRIMARY KEY (id)) comment='公告通知表' CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_user_resource (
  user_id int(10) NOT NULL,
  resource varchar(64) NOT NULL,
  PRIMARY KEY (user_id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_user_ip (
  id          int(10) NOT NULL AUTO_INCREMENT,
  user_id int(10) NOT NULL,
  ip         varchar(32) NOT NULL,
  PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_provider_log (
  id          int(10) NOT NULL AUTO_INCREMENT comment '主键ID',
  provider    varchar(512) NOT NULL comment '操作日志:变化的节点',
  log_type    varchar(16) NOT NULL comment '日志类型:zk_add;zk_del;redisAdd;redisDel;db_add;db_del;',
  creator     varchar(32) NOT NULL comment '日志的来源:例如redisSynZkWorker',
  create_date datetime NOT NULL comment '创建时间',
  log_desc    varchar(256) comment '备用字段',
  PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_user (
  user_id      int(10) NOT NULL AUTO_INCREMENT,
  user_erpid   varchar(32) NOT NULL UNIQUE,
  user_name    varchar(64) NOT NULL,
  user_department varchar(256) NOT NULL,
  user_email   varchar(32) NOT NULL,
  user_cellphone varchar(16) NOT NULL,
  PRIMARY KEY (user_id)) CHARACTER SET UTF8 engine innodb;
CREATE TABLE saf_serviceinfo_user (
  id          int(10) NOT NULL AUTO_INCREMENT,

```

```
serviceinfo_id int(10) NOT NULL,  
user_id        int(10) NOT NULL,  
user_type      tinyint(1) NOT NULL comment '1- watch user; 2- owner user',  
PRIMARY KEY (id)) CHARACTER SET UTF8 engine innodb;  
CREATE INDEX query_index  
  ON saf_invocation_monitor (interface_id, host, client_ip, record_time);  
CREATE INDEX query_index_group  
  ON saf_invocation_monitor (`group`);  
CREATE INDEX query_index_version  
  ON saf_invocation_monitor (version);  
CREATE INDEX query_index_method  
  ON saf_invocation_monitor (method);  
CREATE INDEX query_index_interface  
  ON saf_invocation_monitor (interface_id);
```

 赞同 成为第一个赞同者

无
