

11 | 服务发布和引用的实践

2018-09-15 胡忠想

从0开始学微服务

[进入课程 >](#)



讲述：胡忠想

时长 08:40 大小 3.98M



在[专栏第 4 期](#)，我给你讲解了服务发布和引用常见的三种方式：Restful API、XML 配置以及 IDL 文件。今天我将以 XML 配置方式为例，给你讲解[服务发布和引用的具体实践以及可能会遇到的问题](#)。


首先我们一起来看下 XML 配置方式，服务发布和引用的具体流程是什么样的。

XML 配置方式的服务发布和引用流程

1. 服务提供者定义接口

服务提供者发布服务之前首先要定义接口，声明接口名、传递参数以及返回值类型，然后把接口打包成 JAR 包发布出去。

比如下面这段代码，声明了接口 `UserLastStatusService`，包含两个方法 `getLastStatusId` 和 `getLastStatusIds`，传递参数一个是 `long` 值、一个是 `long` 数组，返回值一个是 `long` 值、一个是 `map`。

 复制代码

```
1 package com.weibo.api.common.status.service;
2
3 public interface UserLastStatusService {
4     * @param uids
5     * @return
6     */
7     public long getLastStatusId(long uid);
8
9     /**
10     *
11     * @param uids
12     * @return
13     */
14     public Map<Long, Long> getLastStatusIds(long[] uids);
15 }
```

2. 服务提供者发布接口

服务提供者发布的接口是通过在服务发布配置文件中定义接口来实现的。

下面我以一个具体的服务发布配置文件 `user-last-status.xml` 来给你讲解，它定义了要发布的接口 `userLastStatusLocalService`，对外暴露的协议是 `Motan` 协议，端口是 `8882`。并且针对两个方法 `getLastStatusId` 和 `getLastStatusIds`，通过 `requestTimeout="300"` 单独定义了超时时间是 `300ms`，通过 `retries="0"` 单独定义了调用失败后重试次数为 `0`，也就是不重试。

 复制代码

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:aop="http://www.springframework.org/schema/aop"
6       xsi:schemaLocation="http://www.springframework.org/schema/context
7                           http://www.springframework.org/schema/context/spring-context-2.5.xsd
8 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans,
9 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spr:
```

```

10 ">
11
12 <motan:service ref="userLastStatusLocalService"
13     requestTimeout="50" retries="2"    interface="com.weibo.api.common.status.s
14     basicService="serviceBasicConfig" export="motan:8882">
15 <motan:method name="getLastStatusId" requestTimeout="300"
16     retries="0" />
17 <motan:method name="getLastStatusIds" requestTimeout="300"
18     retries="0" />
19 </motan:service>
20
21 </beans>


```

然后服务发布者在进程启动的时候，会加载配置文件 user-last-status.xml，把接口对外暴露出去。

3. 服务消费者引用接口

服务消费者引用接口是通过在服务引用配置文件中定义要引用的接口，并把包含接口定义的 JAR 包引入到代码依赖中。

下面我再以一个具体的服务引用配置文件 user-last-status-client.xml 来给你讲解，它定义服务消费者引用了接口 commonUserLastStatusService，接口通信协议是 Motan。

 复制代码

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xmlns:context="http://www.springframework.org/schema/context"
5     xmlns:aop="http://www.springframework.org/schema/aop"
6     xsi:schemaLocation="http://www.springframework.org/schema/context
7         http://www.springframework.org/schema/context/spring-context-2.5.xsd
8 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans,
9 http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spr:
10 ">
11 <motan:protocol name="motan" default="true" loadbalance="${service.loadbalance.name}"
12 <motan:basicReferer id="userLastStatusServiceClientBasicConfig"
13     protocol="motan" />
14
15 <!-- 导出接口 -->
16 <motan:referer id="commonUserLastStatusService" interface="com.weibo.api.common.status.
17     basicReferer="userLastStatusServiceClientBasicConfig" />
18

```



然后服务消费者在进程启动时，会加载配置文件 `user-last-status-client.xml` 来完成服务引用。

上面所讲的服务发布和引用流程看似比较简单，但在实际使用过程中，还是有很多坑的，比如在实际项目中经常会遇到这个问题：一个服务包含了多个接口，可能有上行接口也可能有下行接口，每个接口都有超时控制以及是否重试等配置，如果有多个服务消费者引用这个服务，是不是每个服务消费者都必须在服务引用配置文件中定义？

你可以先思考一下这个问题，联系自己的实践经验，是否有理想的解决方案呢？

服务发布和引用的那些坑

根据我的项目经验，在一个服务被多个服务消费者引用的情况下，由于业务经验的参差不齐，可能不同的服务消费者对服务的认知水平不一，比如某个服务可能调用超时了，最好可以重试来提供调用成功率。但可能有的服务消费者会忽视这一点，并没有在服务引用配置文件中配置接口调用超时重试的次数，**因此最好是在服务发布的配置文件中预定义好类似超时重试次数**，即使服务消费者没有在服务引用配置文件中定义，也能继承服务提供者的定义。这就是下面要讲的服务发布预定义配置。


1. 服务发布预定义配置

以下面的服务发布配置文件 `server.xml` 为例，它提供了一个服务 `contentSliceRPCService`，并且明确了其中三个方法的调用超时时间为 500ms 以及超时重试次数为 3。

 复制代码

```
1 <motan:service ref="contentSliceRPCService"          interface="cn.sina.api.data.service.Co
2             basicService="serviceBasicConfig" export="motan:8882" >
3   <motan:method name="saveContent" requestTimeout="500"
4             retries="3" />
5   <motan:method name="deleteContent" requestTimeout="500"
6             retries="3" />
7   <motan:method name="updateContent" requestTimeout="500"
8             retries="3" />
9 </motan:service>
```

假设服务引用的配置文件 client.xml 的内容如下，那么服务消费者就会默认继承服务发布配置文件中设置的方法调用的超时时间以及超时重试次数。

 复制代码

```
1 <motan:referer id="contentSliceRPCService" interface="cn.sina.api.data.service.ContentS.  
2 </motan:referer>
```

通过服务发布预定义配置可以解决多个服务消费者引用服务可能带来的配置复杂的问题，这样是不是最优的解决方案呢？


实际上我还遇到过另外一种极端情况，一个服务提供者发布的服务有上百个方法，并且每个方法都有各自的超时时间、重试次数等信息。服务消费者引用服务时，完全继承了服务发布预定义的各项配置。这种情况下，服务提供者所发布服务的详细配置信息都需要存储在注册中心中，这样服务消费者才能在实际引用时从服务发布预定义配置中继承各种配置。

这里就存在一种风险，当服务提供者发生节点变更，尤其是在网络频繁抖动的情况下，所有的服务消费者都会从注册中心拉取最新的服务节点信息，就包括了服务发布配置中预定的各项接口信息，这个信息不加限制的话可能达到 1M 以上，如果同时有上百个服务消费者从注册中心拉取服务节点信息，在注册中心机器部署为百兆带宽的情况下，很有可能会导致网络带宽打满的情况发生。

面对这种情况，**最好的办法是把服务发布端的详细服务配置信息转移到服务引用端**，这样的话注册中心中就不需要存储服务提供者发布的详细服务配置信息了。这就是下面要讲的服务引用定义配置。

2. 服务引用定义配置

以下面的服务发布配置文件为例，它详细定义了服务 userInfoService 的各个方法的配置信息，比如超时时间和重试次数等。

 复制代码

```
1 <motan:service ref="userInfoService" requestTimeout="50" retries="2"
```

```

2 <motan:method name="addUserInfo" requestTimeout="300" retries="0"/>
3   <motan:method name="updateUserPortrait" requestTimeout="300" retries="0"/>
4   <motan:method name="modifyUserInfo" requestTimeout="300" retries="0"/>
5   <motan:method name="addUserTags" requestTimeout="300" retries="0"/>
6   <motan:method name="delUserTags" requestTimeout="300" retries="0"/>
7   <motan:method name="processUserCacheByNewMyTriggerQ" requestTimeout="300" retries="{
8   <motan:method name="modifyObjectUserInfo" requestTimeout="300" retries="0"/>
9   <motan:method name="addObjectUserInfo" requestTimeout="300" retries="0"/>
10  <motan:method name="updateObjectUserPortrait" requestTimeout="300" retries="0"/>
11  <motan:method name="updateObjectManager" requestTimeout="300" retries="0"/>
12  <motan:method name="add" requestTimeout="300" retries="0"/>
13  <motan:method name="deleteObjectManager" requestTimeout="300" retries="0"/>
14  <motan:method name="getUserAttr" requestTimeout="300" retries="1" />
15  <motan:method name="getUserAttrList" requestTimeout="300" retries="1" />
16  <motan:method name="getAllUserAttr" requestTimeout="300" retries="1" />
17  <motan:method name="getUserAttr2" requestTimeout="300" retries="1" />
18
19 </motan:service>

```

可以像下面一样，把服务 userInfoService 的详细配置信息转移到服务引用配置文件中。

复制代码

```

1 <motan:referer id="userInfoService" interface="cn.sina.api.user.service.UserInfoService
2   <motan:method name="addUserInfo" requestTimeout="300" retries="0"/>
3   <motan:method name="updateUserPortrait" requestTimeout="300" retries="0"/>
4   <motan:method name="modifyUserInfo" requestTimeout="300" retries="0"/>
5   <motan:method name="addUserTags" requestTimeout="300" retries="0"/>
6   <motan:method name="delUserTags" requestTimeout="300" retries="0"/>
7   <motan:method name="processUserCacheByNewMyTriggerQ" requestTimeout="300" retries="{
8   <motan:method name="modifyObjectUserInfo" requestTimeout="300" retries="0"/>
9   <motan:method name="addObjectUserInfo" requestTimeout="300" retries="0"/>
10  <motan:method name="updateObjectUserPortrait" requestTimeout="300" retries="0"/>
11  <motan:method name="updateObjectManager" requestTimeout="300" retries="0"/>
12  <motan:method name="add" requestTimeout="300" retries="0"/>
13  <motan:method name="deleteObjectManager" requestTimeout="300" retries="0"/>
14  <motan:method name="getUserAttr" requestTimeout="300" retries="1" />
15  <motan:method name="getUserAttrList" requestTimeout="300" retries="1" />
16  <motan:method name="getAllUserAttr" requestTimeout="300" retries="1" />
17  <motan:method name="getUserAttr2" requestTimeout="300" retries="1" />
18 </motan:referer>

```

这样的话，服务发布配置文件可以简化为下面这段代码，是不是信息精简了许多。


```
1 <motan:service ref="userInfoService" requestTimeout="50" retries="2"  
2     </motan:service>
```

在进行类似的服务详细信息配置，由服务发布配置文件迁移到服务引用配置文件的过程时，尤其要注意迁移步骤问题，这就是接下来我要给你讲的服务配置升级问题。

3. 服务配置升级

实际项目中，我就经历过一次服务配置升级的过程。由于引用服务的服务消费者众多，并且涉及多个部门，升级步骤就显得异常重要，通常可以按照下面步骤操作。

各个服务消费者在服务引用配置文件中添加服务详细信息。

服务提供者升级两台服务器，在服务发布配置文件中删除服务详细信息，并观察是否所有的服务消费者引用时都包含服务详细信息。

如果都包含，说明所有服务消费者均完成升级，那么服务提供者就可以删除服务发布配置中的服务详细信息。

如果有不包含服务详细信息的服务消费者，排查出相应的业务方进行升级，直至所有业务方完成升级。

总结

今天我给你介绍了 XML 配置方式的服务发布和引用的具体流程，简单来说就是服务提供者定义好接口，并且在服务发布配置文件中配置要发布的接口名，在进程启动时加载服务发布配置文件就可以对外提供服务了。而服务消费者通过在服务引用配置文件中定义相同的接口名，并且在服务引用配置文件中配置要引用的接口名，在进程启动时加载服务引用配置文件就可以引用服务了。

在业务具体实践过程中可能会遇到引用服务的服务消费者众多，对业务的敏感度参差不齐的问题，所以在服务发布的时候，最好预定义好接口的各种配置。在服务规模不大，业务比较简单的时候，这样做比较合适。但是对于复杂业务，虽然服务发布时预定义好接口的各种配置，但在引用的服务消费者众多且同时访问的时候，可能会引起网络风暴。这种情况下，比较保险的方式是，把接口的各种配置放在服务引用配置文件里。

在进行服务配置升级过程时，要考虑好步骤，在所有服务消费者完成升级之前，服务提供者还不能把服务的详细信息去掉，否则可能会导致没有升级的服务消费者引用异常。

思考题

如果你在实际项目中采用过 XML 配置的服务发布和应用方式，是否还遇到过其他问题？你是如何解决的呢？

欢迎你在留言区写下自己的思考，与我一起讨论。

 极客时间

从 0 开始学微服务

微博服务化专家的一线实战经验



胡忠想 微博技术专家

新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 10 | Dubbo框架里的微服务组件

下一篇 12 | 如何将注册中心落地？

精选留言 (20)

写留言



沙漠之鹰

2018-09-23

12

一个接口上百个方法，设计上是否不合理

展开 ∨



echo_陈

2018-09-15

👍 4

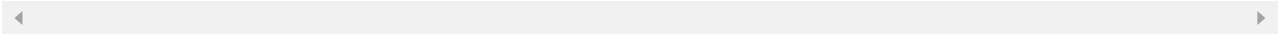
遇到过版本变更时序列化兼容问题

我们用的dubbo，经常会出现，某个dubbo接口的API升级：包含新增方法，或者某个方法的入参或者返回值新增字段。

我们的服务提供者更新消费者并不是一定要更新，如果我的api改动没有改动某个消费者调用的方法或者那个消费者可以兼容提供者的改动，那么消费者是可以不升级的。也就是...

展开 ∨

作者回复: 是的，兼容性很重要



智哥哥

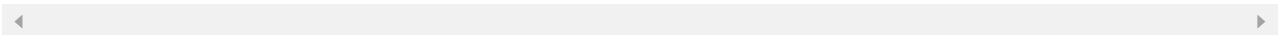
2018-10-24

👍 3

```
<motan:service ref="userLastStatusLocalService"
    requestTimeout="50" retries="2"
interface="com.weibo.api.common.status.service.UserLastStatusService"
    basicService="serviceBasicConfig" export="motan:8882">
    <motan:method name="getLastStatusId" requestTimeout="300"...
```

展开 ∨

作者回复: 极客目前没有这个功能，这里是抽取的代码片段，可以看下开源motan的实现



Mrsong721

2018-09-20

👍 3

多个服务消费者调用了服务提供者A，如果服务提供者A的接口参数发生变化，那所有消费者都需要变更，是否有好的解决方案呢？

展开 ∨



张小小的席...

2018-09-17

👍 2

目前只会spring cloud，dubbo没有细研究过 只是看dubbo 官网会使用，觉得体系都差不多 没有太多机会去实战

展开 ▾



雍

2018-12-26

👍 1

服务发布预定义配置如果遇到提供者接口超级多的极端情况的话建议把配置信息转移到服务信用配制中。请问老师如果这么做的话那不就每一个客户端都要配置一份了吗？这样的话客户端配制参差不齐的问题又出现了，总觉得这里跟您前面讲的矛盾了

展开 ▾



张迪

2019-02-22

👍

客户端是怎么知道服务端在哪里，ip地址或者域名没看到在配置文件中配置呀



shine

2019-02-16

👍

课程中提到的服务引用配置文件：user-last-status-client.xml和client.xml是同一个文件吗？

那些坑里，1. 服务发布预定义配置 和 2. 服务引用定义配置 是不是只有这二种配置方式，而且是相互对立的，要根据实际情况选择？

展开 ▾



平头哥

2019-01-08

👍

为什么要每个方法都声明出来呢？直接到类呢

展开 ▾



英宁

2018-11-15

👍

“服务提供者所发布服务的详细配置信息都需要存储在注册中心中，这样服务消费者才能在实际引用时从服务发布预定义配置中继承各种配置。”，没看懂这里为啥，是有些详细信息服务端配置放不了嘛？比如说呢？



Jerry

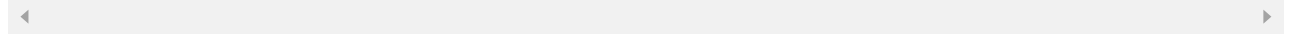
2018-11-02



这个服务引用配置文件放在哪里

展开 ▾

作者回复: 放在客户端本地



王晓军

2018-10-15



胡老师，如何合理的规划mianx前端的服务和面向服务的服务，您有什么建议吗？还是说，任何一个服务都可以由前端调用也可以由其他的服务调用

展开 ▾



有渔@蔡

2018-10-14



这里的mota协议跟dubbo是什么关系？

展开 ▾



波波安

2018-10-13



并观察是否所有的服务消费者引用时都包含服务详细信息。
服务端怎么看到消费者的引用配置？

展开 ▾



波波安

2018-10-13



使用dubbo遇到的一些坑。

之前没有形成规范，很多发布的服务都没有配置retry-time和和timeout导致经常出一些莫名其妙的问题。有些业务办理接口没有做幂等，接口超时导致重试，产生脏数据等。

...

展开 ▾





吴浩

2018-10-09



服务配置升级哪里没大看懂

服务提供者升级两台服务器，在服务发布配置文件中删除服务详细信息，并观察所有的服务消费者引用时都包含服务详细信息。

...

展开 ▾



wuhulala

2018-10-08



老师您好 在接口定义好之后 如何减轻消费者对api的强依赖性 比如A服务接口变动之后 增删参数 导致了B消费者调用失败 比如序列化问题 如何提高这种容错性呢

展开 ▾



莲花

2018-09-19



dubbo本身就会有provider和consumer的xml定义配置。这个和引用文件什么关系？没怎么明白



张振华

2018-09-19



Spring. cloud.呢

展开 ▾



少帅

2018-09-15



注解貌似没有找到方法级别的配置

展开 ▾