关于SAF升级JSF中方法重载的说明

工具

被lixininfo添加，被lixininfo最后更新于十一月 23, 2015

出于以下多种考虑JSF不再支持方法重载特性：

- 支持C++等跨语言调用；
- 方便方法级别的配置的保存与推送；很多参数都可以设置到方法级别，比如并发线程数等，目前这些都是根据方法名来区别保存的；
- 方便性能数据收集及统计；目前性能数据收集上报都是根据方法名来区别的；
- 减少数据传输过程中无谓的数据传输，仅传输方法名，参数列表不必再传输；

对于有些从SAF升级过来的团队，如果已使用了重载特性，可以采用如下方案来进行过渡及迁移：

1）.对于已有SAF接口Hello；

```
public interface Hello {

    String echo();
    String echo(Integer myInt);
    String echo(String str);
    String echo(Float fl);
}
```

2）.新建一个adapter接口HelloB，B接口中方法与A接口中方法一一对应区别仅仅是没有了方法重载；

```
public interface HelloB {

    String echo();
    String echoInt(Integer mInt);
    String echoStr(String str);
    String echoFloat(Float fl);

}
```

HelloB接口的实现如下：

```
public class HelloBIns implements HelloB{


    private Hello hello;

    @Override
    public String echo() {
        return hello.echo();
    }

    @Override
    public String echoInt(Integer mInt) {
        return hello.echo(mInt);
    }

    @Override
    public String echoStr(String str) {
        return hello.echo(str);
    }

    @Override
    public String echoFloat(Float fl) {
        return hello.echo(fl);
    }

    public void setHello(Hello hello) {
        this.hello = hello;
    }


}
```

3）.将接口HelloB发布到JSF中去，并敦促接口Hello的调用者，转而使用接口HelloB来进行服务调用；

4）.待所有调用者都已使用接口HelloB之后，从SAF上将接口Hello的服务实例做下线处理即可；

无