**赫千电子**

**<FogOS服务框架/J6>**

**CM接口的封装设计**

版本号：0.1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* CONFIDENTIAL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

本文件属于公司秘密信息，请您恪守保密义务，勿向第三人透露。请遵守研发文档管理办法。

This file is confidential. Recipient(s) is(are) obligated to maintain secrecy and is(are) not permitted to disclose the contents of this file to others.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**版本记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **版本** | **日期** | **操作** | **作者** | **审核** | **备注** |
| 0.1 | 2018.2.8 | 创建 | 朱光育 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[1. 目的 4](#_Toc505933723)

[2. 功能概述 4](#_Toc505933724)

[3. 定义与缩写 4](#_Toc505933725)

[4. 新增文件规则 4](#_Toc505933726)

[5. Event的封装规则 4](#_Toc505933727)

[6. Method的封装规则 5](#_Toc505933728)

[7. 样例 5](#_Toc505933729)

# 目的

本设计是为了给应用开发者与服务实现者可以更方便的使用AUTOSAR CM的API，进行AUTOSAR CM的API再封装，提供更简洁形式的接口给应用开发者与服务实现者。

# 功能概述

为了使CM的接口以更简洁的形式提供给应用开发者与服务实现者，进行接口再封装，隐藏Method调用的

Promise/Future机制，隐藏Event的订阅与更新细节。

# 定义与缩写

ARXML：AUTOSAR XML

CM：Communication Management

# 新增文件规则

对于Service的proxy，增加<name>\_proxy\_api.h头文件和<name>\_proxy\_api.cpp源文件，相应增加一个类名为<Name>ProxyAPI的类，这个类继承<Name>Proxy类。

对于Service的skeleton，增加<name>\_skeletion\_api.h头文件和<name>\_skeleton\_api.cpp源文件，相应增加一个类名为<Name>SkeletonAPI的类，这个类继承<Name>Skeleton类。

其中<name>是**ServiceInterface.shortName**的小写，<Name>是**ServiceInterface.shortName**。

# Event的封装规则

对于Service的proxy的封装 ，类<Name>ProxyAPI提供一个注册回调函数的接口，接口名为RegisteEventCallback，参数是一个结构体，结构体名为<Name>EventCallbackGroup。这个结构体的每个成员都是一个Event对应函数指针,函数指针名为< EventName >。每个Event对应指针函数名为<EventName>Callback，参数只有一个，参数类型与Event的数据类型是一致的。

对于Service的skeleton的封装，类<Name>SkeletonAPI提供每个Event对应的接口，接口名为< EventName >Send，只有一个参数，参数类型与Event的数据类型是一致的。

其中<Name>是**ServiceInterface.shortName** ，<EventName>是**VariableDataPrototype.shortName**。

# Method的封装规则

对于Service的proxy的封装，类<Name>ProxyAPI提供每个Method对应的接口，接口名为<MethodName>，参数与Method的参数一致，返回值为int类型，表示调用是否成功，0表示成功，-1表示失败。

对于Service的skeleton的封装，类<Name>SkeletonAPI提供每个Method对应的纯虚接口，接口名为do< MethodName >，参数与Method的参数一致，返回值为int类型，表示调用是否成功，0表示成功，-1表示失败。

其中其中<Name>是**ServiceInterface.shortName** ，<MethodName>是**ClientServerOperation.shortName**。

# 样例

伪ARXML格式

*CalculateService {*

*event TipEvent {*

*type: uint32*

*}*

*method Add {*

*param num1 {*

*type: uint32*

*direction:in*

*}*

*param num2 {*

*type: uint32*

*direction:in*

*}*

*param result {*

*type: uint32*

*direction:out*

*}*

*}*

*}*

头文件calculate\_proxy\_api.h：

*typedef void (\*TipEventCallback)(uint32 data);*

*struct CalculateEventCallbackGroup*

*{*

*TipEventCallback TipEvent;*

*};*

*class CalculateProxyAPI : private CalculateProxy*

*{*

*public：*

*CalculateProxyAPI(char \*serverIp, unsigned short ServerPort, unsigned short ClientPort, int protocol);*

*~CalculateProxyAPI();*

*void RegisteEventCallback(CalculateEventCallbackGroup\* eventCallbackGroup);*

*int Add(uint32 p1, uint32 p2, uint32 result);*

*private:*

*CalculateEventCallbackGroup m\_eventCallbackGroup;*

*};*

头文件calculate\_skeleton\_api.h

*class CalculateSkeletonAPI : private CalculateSkeleton*

*{*

*public:*

*CalculateSkeletonAPI(char \*ip, unsigned short port);*

*~CalculateSkeletonAPI();*

*private:*

*virtual ara::com::Future<AddOutput> Add(uint32 p1, uint32 p2, uint32 result);*

*public:*

*void TipEventSend(uint32 data);*

*virtual int doAdd(uint32 p1, uint32 p2, uint32 result) = 0;*

*};*