# 基础知识介绍

Yang模型是netconf协议的一种数据建模语言。

Yang模型的节点主要分为四种类型：set/get/rpc/notification（配置节点，状态节点，操作节点，通知节点）

交换机和控制器进行交互，首先控制器会下发基于yang模型生成的报文，naas收到以后进行处理，感知到有变化的节点时，会调用我们注册的对应节点的回调函数进行处理，从而实现功能（如果是通知节点，是交换机组装通知报文，通过naas提供的接口发送到控制器，再由控制器解析）

# 交换机侧代码结构

交换机yang模型归档都在extern\opensource\yang\yang\_module\_repository\Uniform-model\model中，如果是该归档文件夹中对应的yang模型里已有的，那么直接写实现代码即可，如果没有，需要在设计新增以后，在omsys上下载最新yang模型，把自己涉及的节点合入。

交换机侧业务代码主要有两个部分组成，一个是云管理组负责维护的整体的框架，另一部分就是各个业务对具体的节点进行的功能实现，本文主要关注第二部分

节点的实现在代码中主要集中在两处

C码：code\product\switch\source\openmng\cmng\netconf\cmng\_cfg\source\cmng\_cfg\_pub.c

Python：code\product\common\openmng\netconf\_plugin\plugins

C码需要先在3个全局变量中注册回调

cmng\_cfg\_config\_subs 配置节点（对某个属性进行配置）

cmng\_cfg\_status\_pubs 状态节点（获取某个属性的值）

cmng\_cfg\_oper\_subs 操作节点（进行某个操作以实现某种功能）

notification节点，在各个业务需要处直接组装发送即可

python以yang模型为单位，例yang模型为huawei-stack.yang，对应的python处理文件为yang\_huawei\_stack.py

python同样区分三个类型

@config\_cb 配置节点

@status\_cb 状态节点

@operate\_cb 操作节点

具体实现不赘述，但值得一提的是，有些python中还会调用c码，之后说明

# C码代码实现

首先需要明确节点的属性，在对应全局变量里添加回调函数。全局变量有三项，依次是yang节点路径，优先级，回调函数

以{.path = "/huawei-device:device/name", .priority = 100, .callback = (int(\*)(int,int,int,int,const char \*))CMNG\_SetSysName\_Callback},为例

.path表示需要进行处理的节点的路径，huawei-device是模块名，该路径说明是huawei-device.yang这个模型里device容器里的name节点，

Priority是处理该节点的优先级（只有配置节点有此优先级），默认是100，一般如果有配置依赖，或者增加/删除有不同的处理，需要修改此优先级，优先级低的先执行

优先级限制在1-1010区间，参加如下

回调函数是naas根据下发的报文与本地数据库里存储的相比较，如果发现该节点的值存在变化，则调用注册的回调函数

## 3.1配置节点

配置节点相对来说比较复杂，因为是配置节点支持配置后的属性查询（和状态节点配合）等，所以有回滚/增加/删除/配置恢复等的区别，所以在解析传入的报文时需要优先根据oper来判断当前处于什么阶段，以进行不同的处理

因为是配置，只有配置成功和配置失败两种结果，所以不需要组装返回报文，直接返回ok/err即可，偏重于解析报文。

常用函数

CMNG\_CFG\_Get\_Database\_Value获取naas的db数据，naas有两个数据库，一个是存储之前值的，一个是存储当前修改的，naas收到报文后首先把报文中涉及的节点存到当前修改的数据库里，再和之前值比较，如果有变化则调用回调函数。如果一个报文的回调结果都是ok，那么把当前修改的替换掉之前修改的，如果有一个返回结果是err，那么整体要进行回滚，回滚到之前的状态，所以对于一些需要取值进行配置的属性，需要取之前的值进行回滚。

## 3.2状态节点

状态节点（通常容器后面有-state标识）

状态节点一般都是get，通过报文知道要获取哪个节点的值，然后组装报文进行回复。所以状态节点偏重于组装回复报文，组装完成后，通过出参返回。

注意组装报文时，节点名称一定要和yang模型中对应节点名称相同，节点属性类型也要合yang模型中声明的类型一致，不然可能无法解析

常用函数

set\_content\_and\_add\_child因为yang模型是有层级概念的，所以需要用提供的函数完成层级的构造，第一个参数是父节点，第二个参数是本节点名称，第三个参数是本节点值

xmlNewNs给传入节点增加命名空间声明，注意第一个参数不能为空

xmlSetNs设置节点命名空间（通常xmlNewNs需要在xmlSetNs之前使用）

## 3.3操作节点

操作节点（yang模型中的标识是rpc）的处理既可能需要解析（通过入参）又可能需要组装（通过出参）

解析基本和配置节点的处理一致，但因为操作节点不会对属性值有影响，也不会存储，只是实现了某种功能，只有成功和失败，不需要进行回滚和配置恢复的区分。相对来说比较简单。

组装和状态节点基本一致

## 3.4通知节点

通知节点（yang模型中的标识是notification）需要直接用提供的接口

cmng\_cfg\_conf\_notify\_plugin，第一个参数是通知的名称（yang模型中notification后的声明名称），第二个参数是组装的报文，同样注意名称和类型要一致

# python实现

python多为外包实现，不多加赘述，但因为python也会调用c码来进行处理，所以重点描述这部分内容。

Python中一般会处理一些比较简单的逻辑，比如执行命令行这种可以全用python完成。但有些需要获取设备的信息，无法通过命令行完成，所以可以通过调用c码来完成

调用产品c码的时候，一般会先调用plugin.vrp.中的函数

（\code\product\common\openmng\netconf\_plugin\vrp\api，也是按照模块划分）例如

然后根据要调用的函数名vrp\api中找对应的处理，名称以api\_开头后面是函数名称

然后这里先组装要调用的c码需要的参数，然后调用c码。

C码先从入参中解析需要的信息，然后再组装报文在出参中返回（因为是约定的格式不是yang模型格式，所以直接添加节点即可，不需要增加命名空间之类）

约定的返回报文主要有两个主节点，name填写函数标识，result中填写需要带回的值（可以

在里面增加层级），有的时候还有reason节点，这边一般会在result为err时带失败原因

PS：

C码使用Openmng\_RegApi进行注册，注册有三个参数，通常用前两个，第一个参数是标识名，api通过此调用，第二个参数是对应的处理函数，进行真正的业务处理。下为示例

# 其他

## 5.1云管理&netconf模式

云管理和netconf模式都是基于netconf协议来实现的，两个区别是配置的方式不同，上线的方式也不同，这块主要是云管理组（维护框架）感知，对业务实现来说基本相同

## 5.2调试方法

调试可以通过控制器和模拟器两种，一般会在调试前期用模拟器例netconf，保证代码没问题后与控制器进行联调

Netconf使用和与设备配置连接，可参考

<http://3ms.huawei.com/hi/group/2025947/thread_6172765.html>  
<http://3ms.huawei.com/km/groups/1502303/blogs/details/2033501?l=zh-cn>

## 5.3代码注意点

因为云管和netconf在框架侧有区别，所以有些DFX手段是有区别的

------------------------------------TO BE CONTINUE--------------------------------------