

题 目 一种基于微服务架构的PAAS平台

\_\_\_\_\_\_软件学院\_\_\_\_\_\_\_\_院（系）\_\_\_\_软件工程\_\_专业

学 号\_\_\_\_\_\_\_\_\_\_71113333\_\_\_\_\_\_\_\_\_\_\_\_

学生姓名\_\_\_\_\_\_\_\_\_\_\_陈浩远\_\_\_\_\_\_\_\_\_\_\_\_\_\_

校内导师\_\_\_\_\_\_\_\_\_\_\_刘奇其\_\_\_\_\_\_\_\_\_\_\_\_\_\_

企业导师\_\_\_\_\_\_\_\_\_\_\_薄宏剑\_\_\_\_\_\_\_\_\_\_\_\_\_\_

起止日期\_\_\_\_\_2017-01-20至2017-05-20\_\_\_\_

设计地点\_\_\_\_\_\_\_\_\_\_\_南京\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 

**一种基于微服务架构的PAAS平台**

**摘要**

伴随电信行业3G/4G等业务的增加，数据量出现了爆炸式的增长，数据的规模已经达到了PB级别。海量数据带来了大量的信息，同时也迫使电信行业为了处理这些数据而不得不使用hadoop/spark等分布式大数据技术。由于电信运营商自身结构复杂，产品和业务繁多，电信用户多，每天能产生大量的数据，因此产生了很多不同的hadoop/spark需求，对于如何管理和调度众多的大数据任务也成为了运行商头疼的一件事。

华为大数据部门决定开发一个PAAS平台用来统一管理和调度hadoop/spark任务（以后称之为统一运行平台/Runtime-Platform/RP）。本文结合我所参与的华为大数据统一管理平台的经验，详细介绍了华为这个PAAS平台的设计。该项目采用了分布式的开发理念，选择了较为流行的微服务架构，并且使用轻量级的docker容器单独部署服务，因此，该项目具有极高的可用性和鲁棒性。

**关键词**：微服务；分布式；PAAS；大数据；docker

**Abstract**

In recent years,

**Keywords**: Micro service; distributed; PAAS; big data; docker

目 录

[一种基于微服务架构的PAAS平台 2](#_Toc8719)

[摘要 2](#_Toc28984)

[Abstract 2](#_Toc9722)

[第一章 绪论 4](#_Toc18414)

[1.1引言 4](#_Toc27800)

[1.2目前现状 4](#_Toc15740)

[1.3本文的研究目的和主要研究内容 5](#_Toc13018)

[第二章 研究设计 5](#_Toc27790)

[2.1在线采集 5](#_Toc15419)

[2.1.1概述 5](#_Toc14406)

[2.1.2在线换单一致性稽核 5](#_Toc7836)

[2.2解码模块 7](#_Toc18727)

[2.2.1解码模块概述 7](#_Toc10769)

[2.2.2解码模块功能定义 7](#_Toc14882)

[2.2.3BCD编码方式简介 7](#_Toc13358)

[2.2.4ASCII编码方式简介 8](#_Toc16195)

[2.2.5ASN.1编码方式简介 8](#_Toc2697)

[2.2.6解码模块内外部结构关系图 9](#_Toc6803)

[2.2.7解码模块功能实现说明 10](#_Toc11925)

[2.2.8规则引擎 11](#_Toc6432)

[2.2.9协议组件ASN.1 13](#_Toc19709)

[2.2.10协议组件－BCD 和ASCII 15](#_Toc24920)

[2.2.11Decode函数 15](#_Toc13606)

[2.2.12Deal函数 15](#_Toc25871)

[2.2.13check函数 16](#_Toc31113)

[2.2.14filter函数 16](#_Toc7405)

[2.2.15解码特殊逻辑功能设计说明 17](#_Toc26617)

[2.2.16错单处理 19](#_Toc8218)

[2.3漫游上发模块 21](#_Toc16668)

[2.3.1漫游上发模块概述 21](#_Toc597)

[2.3.2功能定义 22](#_Toc4251)

[2.3.3漫游上发内外部结构关系图 23](#_Toc29673)

[2.3.4漫游上发内部结构关系图 24](#_Toc21455)

[2.3.5漫游上发功能模块简要说明 25](#_Toc14871)

[2.3.6功能实现说明 29](#_Toc19203)

[第三章 相关测试 39](#_Toc3047)

[3.1测试内容 39](#_Toc29046)

[3.2测试相关配置数据 39](#_Toc32391)

[3.3测试用例 40](#_Toc15047)

[3.3.1贵州-花溪中心 40](#_Toc17824)

[3.3.2贵州-贵安中心 42](#_Toc26741)

[3.4测试结果评价 43](#_Toc28486)

[致 谢 44](#_Toc8336)

[参考文献 44](#_Toc16493)

第一章 绪论

1.1引言

随着上网用户带宽和流量的增加，运营商对用户产生数据分析和处理的需求也越来越大，本文旨在解决电信行业存在的大量的hadoop和spark任务的调度问题，以分布式的结构，提供一个易于部署、高可用性、高负荷能力的PAAS平台。

之所以采用微服务，是为了避免传统的分层架构引发的很多问题，微服务也整合了过去十年来的新概念和技术，因此可以避开许多面向服务的架构中的陷阱。

在多年前，很多组织就开始尝试使用更细粒度的架构来实现更快的交付，结果发现其带来了更好的可扩展性，增强了团度的自治。微服务就是一个这样的快速发展的主题，尽管它不是一个新的想法，但它受了多年来的各种各样的架构的影响，因此，它在当前阶段具有很好的前瞻性和挑战性。

1.2目前现状

全球信息数据量的迅猛增长成为大数据产业发展的基础。市场调研机构IDC预计，未来全球数据总量年增长率将维持在50%左右，到2020年，全球数据总量将达到40ZB。其中，我国数据量将达到8.6ZB，占全球的21%左右。另外，截至去年12月，我国网民规模达6.88亿人。其中，手机网民规模达6.20亿人，移动互联网流量呈现爆发式增长。工信部统计显示，去年全年移动互联网接入的流量超过400万TB，同比增长了103%。随着全球数据总量以及移动流量的不断提升，运营商的大数据任务不断增加，因此有必要开发一个统一运行平台来对这些任务进行

1.3本文的研究目的和主要研究内容

本文主要研究亚信在线计费系统中的两个模块，解码模块以及漫游上发模块。解码模块主要研究的是如何实现不同服务不同格式字段的转换，漫游上发模块主要研究的是不同格式文件的不同处理方式。

1. 研究设计

2.1在线采集

2.1.1**概述**

一开始，在线采集读数据接口被调用，模块启动后，根据不同的协议，与网元取得通讯联系。从网元尝试读取消息包文，成功获取数据后，填充给接口，使TS框架能够往后传送数据，TS通过TB与后续计费TS进行交互，等待计费模块返回后，发送数据接口被调用，在线采集根据SESSION中保存的通讯SOCK信息，把处理后的消息包发送到对应的网元回去。一个正常的采集，处理，发送过程就结束了。



2.1.2**在线换单一致性稽核**

在线计费系统的大规模商用，对于OCS系统作为计费点的落地落单需要确保其准确性； 因此需要提供基于在线话单、离线话单的的一致性稽核系统； 在线话单一致性稽核系统主要包含实时稽核汇总、定时稽核、稽核规则、稽核导出等功能模块，完成对差异话单的选择，并提供之运维管理系统进行话单回捞处理。

在线话单一致性稽核系统包含实时稽核汇总、定时稽核、稽核规则、稽核导出等功能模块， 其中定时稽核分为：用户级稽核、RG级稽核、话单级稽核功能模块。

2.1.2.1**功能简要说明**

实时稽核是基于会话的实时处理流程，从而减轻系统定时稽核的波动系统压力；实时稽核是将在线计费后话单、业务分析后离线错单进行实时流量汇总处理，并存储在分布式Forcebase文件\内存系统中。

2.1.2.2**定时稽核**

定时稽核分为三个稽核过程，包含对用户的稽核，一旦存在用户稽核差异时，再发起对业务级的稽核规则，随后对存在业务稽核差异时，发起话单级稽核处理。

2.1.2.3**用户级稽核**

用户级稽核完成对用户一天中产生的话单总流量进行稽核， 主要完成对在线计费话单，以及离线话单两者之前每个用户当前话单的总流量的稽核功能，并生成差异用户。

2.1.2.4**RG(SERVICE\_CODE)级稽核**

针对用户级稽核出现的差异用户，进行RG级的稽核处理； 即用来进行差异用户每个SERVICE\_CODE业务上使用流量的稽核处理；并生成差异用户对应的业务的描述。

2.1.2.5**话单级稽核**

针对基于RG级稽核出现的业务差异的用户进行话单级稽核，即完成用户下存在差异业务的话单稽核功能，明确挑选出存在差异的话单唯一key值信息。

2.1.2.6**稽核导出**

针对完成了用户级稽核、RG稽核、话单级稽核之后的差异进行导出至数据库中，供SO进行话单实时回捞处理。

稽核导出分为两种方案：

1.稽核明细中，只是挑选的保存了部分关键稽核信息字段，稽核导出完成将关键信息导入指定的数据库表中，SO运维管理系统依据导出的表，作为话单回捞处理的依据。

2.稽核明细中，直接保存了话单信息，稽核导出直接按照指定的话单格式导出，并按照需要回收的业务要求，对相应字段进行重置处理，直接进行回收处理。

针对以上两种方案，如果系统系统配备额主机存储、内存足够，可以考虑直接采用方案二，由稽核系统完成稽核汇总、稽核差异分析、稽核导出的同时，直接做话单回收的处理；如果主机资源不足，建议按照方案一方式，由SO配合改造支持根据导出表进行回捞处理。

2.2解码模块

2.2.1**解码模块概述**

解码处理是计费系统预处理的第一步，要将不同网元设备的服务使用记录处理为统一格式的服务使用记录。

解码子系统处理由采集子系统获取到的各种移动服务使用记录文件，对这些文件进行解码、格式转化、检错、过滤，生成OPENBILLING系统内部使用的标准计费服务记录XDR，交付给系统的下一级子系统处理。

解码模块拥有强大的解码功能和优秀的性能，并努力使配置人性化，努力使升级平滑，合理

解码模块的优点：支持多线程处理，拥有强大性能；强大的对ASN.1标准服务使用记录的解码功能：针对原有解码处理时平面化处理服务使用记录导致的兼容性能差，处理烦琐冗余等缺点做了很大改进，现已完全根据ASN.1标准解码服务使用记录，使用非递归算法；更强大的异常处理功能；更人性化的模板配置：针对模板配置过于专业，配置人员不易接受这个缺点，改进配置项，使配置信息，更容易被理解。

2.2.2**解码模块功能定义**

解码：原始服务记录的编码方式是多种多样的，根据不同的编码方式，调用相应的解码器进行解码，统一转化为ASC码的字符串。目前已知的编码方式有ASCⅡ编码、二进制BCD编码、二进制ASN1.0编码。

格式转化：将解码的结果根据原始话单定义的各字段含义，转化为内部定义数据类型和含义。比如原始字段枚举值与内部定义数据枚举值的转换，格式化原始字段的前缀，后缀等操作。

过滤：根据单个字段的特征值进行整条话单的过滤操作，话单做无效话单不输出

检错：针对单个字段的特征值对话单记录做检错操作，话单给出错误信息，输出到错单目录。

Convert：上述操作都是针对单个字段进行的处理，针对话单关联字段的处理讲通过convert函数进行，convert函数的处理是针对整条话单记录，根据相应业务逻辑进行操作，可做复杂的字段输出格式化(如号码规整，特殊话单判断)，打错单，过滤等功能。

2.2.3**BCD编码方式简介**

BCD编码是一种数字压缩存储编码，大家都知道一个字节有8个位，而数字0到9最多只需要使用4个位，如果用一个字节来存储一个数字相对就会有一定的浪费，尤其是在传输过程中，由此人们就想出了压缩的BCD编码。

BCD编码将一个字节的8个位拆分成高4位和低4位两个部分，也就是说一个字节能存储两个数字。所以BCD的编码过程就是将数字压缩的过程，将两个字节的数字压缩成一个字节。反之，解码就是把一个字节的数字拆分为两个数字单独存放（大部分的处理都是按字节处理的）。

示例：

编码过程：将数字69进行BCD编码（注：BCD编码低位在前，后面将不再注释）。

1.将6，9分别转换成二进制表示：6（00000110）9（00001001），大家可以看到，最大的数字9也只要4个位，在传输过程中白白浪费了4个位；

2.将69合并为一个字节，分别取6，9二进制编码的低4位，按照低位在前的原则，将9的低四位放前面6的低四位放后面得出新的字节二进制编码是10010110；

3.完成编码过程，69的BCD编码结果为10010110。

解码过程：将69的BCD码10010110进行解码。

1.将10010110的高4位与低4位拆分开，得到两个二进制数1001和0110；

2.分别将1001和0110的前面补充4位0000得到两个8位的二进制数00001001，00000110；

3.因为编码时低位在前，所以我们将两个二进制数编排顺序为00000110 000010001；

4.将二进制数转换为十进制得出解码结果为69（正确解码）。

2.2.4**ASCII编码方式简介**

ASCII编码就是由ASCII码编码的文件，多是定长或分隔符方式编码。

2.2.5**ASN.1编码方式简介**

按照BER（Basic Encoding Rule）约定，任何一个ASN.1对象是一个Type、Length、Value的三元组：

例如，整数100会被编码为“02 01 64”，Type域为0x02（UNIVERSAL + Primitive + Integer），Length域为0x01，Value域为0x64

标志域（type）可用若干字节来表示的，该字节被划分为三段，分别称为Class、P/C（Primitive or Constructed）和Tag。Class段用于指明该对象是由谁来定义的，P/C段用于指明Value域是否还可以再拆分，Tag段指明在当前的解码深度上，ASN.1对象的ID值。P/C段表明该ASN.1对象是primitive或constructed类型，primitive类型用0表示，constructed类型用1表示

Tag段根据class的不同有不同含义， class为UNIVERSAL的Tag有特定含义，值域为0到31，分别代表不同数据类型的数据，最常用到的是0x30，表示Sequence type and SequenceOf type ，class为CONTEXT-SPECIFIC 和PRIVATE 则仅代表不同数据

长度域（length）话单中各数据类型的长度不同，Length值可能很大，也可能很小；因此Length域是变长编码的。编码的规则视Length的值而定，值在0-127用一个字节存储，此时，Length域的最高位为零。对于大于等于128的Length值，BER规定，要用第一个字节存储“length of length”，该字节的最高位必须是1。

例如，要对长度值161编码，尽管一个字节0xa1可以表示161，但它的最高位是1，需要扩展存储，才能使数据表达无二义性。故此，需使用两个字节“81 a1”，其中第一个字节0x81存储了length of length，表明后续一个字节是真正的length值

对于话单中的Value域，只要不是Primitive形的Value，可以递归地按照TLV的定义不断解码，直至找到一个Primitive形的Value primtive类型的value存储的数据信息的实际值

2.2.6**解码模块内外部结构关系图**



解码节点：给框架调用。

ASN.1协议组件：根据ASN.1协议形成的数据结构和算法集，以组件的方式给被调用者使用，跟其他协议组件使用相同的调用接口，方便灵活的添加，调用。

BCD协议组件：根据BCD协议形成的数据结构和算法集合

ASCII协议组件：以现有ASCII的基本编码形式（分隔符，定长）形成的组件。

规则引擎：对需要解码的话单规则，我们可以进行详细定义说明，形成规则描述文件（dcd文件），规则引擎负责将规则描述文件形成解码算法，通过调用相应的协议组件进行解码。

检错：由检错函数被规则引擎调用执行，将字段级错误话单进行错单输出。

过滤：由过滤函数被规则引擎调用执行，根据字段判断需要过滤话单。

格式转换：由格式转换函数被规则引擎调用执行，负责字段的格式化转换

Convert函数：针对整条话单做逻辑处理，根据话单中相关联字段做出检错，过滤格式转换的操作。

2.2.7**解码模块功能实现说明**

解码流程图：



整个解码子系统从功能模块上讲，可以分成两部分，一是表示原始话单数据结构的协议组件，二是描述话单组成规则的模板（dcd文件）。前者是数据层的；后者是业务层的。

总体思路是，先建立底层数据结构，然后解码。其中，当原始记录采取BCD编码规则时，将按照按记录长度取数据，然后解码。其中，当原始记录采取ASCII编码规则时，将按行取数据，然后解码。其中，当原始记录采取ASN.1编码规则时，首先将根据ASN.1编码规则，通过协议组件生成树的结构，然后根据模板的配置定位、查询树，获得字段数据。

流程逻辑说明：

初始化：初始化主要做对规则描述文件（dcd）的初始化工作，通过LEX/YACC将dcd文件形成规则引擎，以备后续处理使用。

运行：话单文件传入以后，系统根据话单编码类型调用相应协议组件，结合规则引擎形成解码的话单结果，解码后的话单，再经过格式转换做格式化处理。而后针对其中需要做过滤，或错单处理的话单进行相应处理，最终由convert函数进行话单级的业务逻辑处理，话单分拣后被输出到不同目录。

2.2.8**规则引擎**

功能定义：

目前框架支持ASCII、BCD、和ASN1.0编码的原始文件的解码，但只要遵守一些规则，就可以扩展其他类型编码的解码。

DCD模板具有可维护性，易配置。框架分两步流程解码，先是原始数据的解析，建立某种数据结构，然后业务逻辑的处理，强调地是字段级的解码。例如ASN1.0编码格式，就是采用第三方工具解析数据，见下一章节的说明。

处理流程说明：

1. 系统初始化（格式框架图、实例框架图、引擎框架图的建立）

2.等待统一框架调用运行接口，系统按文件匹配规则获得模板实例，然后是记录级解码、字段解码，这是主流程，统一接口是

CRcdRecordAlgorithm::process\_record()。该接口是类模板方法，定义了一些算法骨架，而其具体实现延迟到实现者的子类，采用策略模式。它可以分为三部分:BILL RECORD的解码流程；CONTROL RECORD的解码流程； 通用栈的解码流程，其前提是CONTROL RECORD。

3.输出解码正确文件、错单文件、统计文件、并记录日志。

解码主流程的三个序列图，如下，注释见下表述。



其中，接口**build\_dataFrame**（）建立底层数据结构，就是前面所述的先分析原始数据。子类，仅当以底层数据结构返回的记录长度为准时覆盖，否则以模板流程FIELD返回的记录长度为准。

首先获得首字段，接着字段级解码，然后取下一个字段，判是否NULL而循环，最后结束。

因建立底层数据结构后将直接返回记录的长度，所以可能直接返回底层数据结构计算的记录长度。



其中，成功解析一个字段后，需要调用**check\_nextRecord**（）接口，分析是否RECORD嵌套，成立则递归调用，否则接下来判断是否需要合并、以及取下一个字段，最后判是否NULL而循环或结束。



其中，获得栈顶元素替代**get\_firstField**（）接口的调用，接着成功解析字段后将更新通用栈。同CONTROL流程，先判断是否有记录嵌套，如果存在，则计算嵌套记录的长度并压栈，然后递归调用，递归退出后，最后将更新通用栈和判断是否合并；如果不存在，则获取下一个字段，并判断是否NULL而循环或结束。

可见，对于记录引擎的实现者，必须关心的接口有**process\_field**、以及**build\_dataFrame**和get\_length；对于字段引擎的实现者，必须关心的接口有**decode\_field**、**check\_nextField**获取模板配置的下一个Field。

业务规则：

不同业务需要配置不同的DCD模板，从框架角度看是配置不同的模板实例。不同业务因不同的编码方式，所以记录引擎或字段引擎的实现者必须支持各种类型的编码。目前框架支持ASCII、BDC、ASN1.0，若是其他编码格式则必须扩展子类的实现。

异常处理：

由于在处理BILL RECORD流程中，首先建立了底层数据结构，所以框架还具备一定的异常处理能力。发生异常时，ASCII仅牺牲一条记录；BCD可以忽略配置（当配置的记录长度正确即可）；ASN1.0更可以配置仅关心的字段。

对于某些编码方式诸如BCD，具有数据块的概念，所以框架也支持块的处理。每处理一条话单后，将核对是否超越块的填充字符，并检测剩余填充字符个数是否正确。

异常处理，除了能跳过一个数据块而继续解码外，程序内部保留了3种解码系统级的过滤码。一是F9997，当底层数据结构建立失败时，目前仅当ASN1.0可能会出现此情况；二是F9998，当对一个字段处理失败而正常退出时，例如模板配置的字段在原始数据中未匹配的情况；三是F9999，当对一个字段处理失败而发生异常时，例如调用decode函数失败。以上三种异常情况仅牺牲一条计费话单!

2.2.9**协议组件ASN.1**

功能定义：

对ASN.1标准服务使用记录建立树状数据结构,由规则引擎调用此数据结构进行解码处理.由于ASN.1是不同于BCD和ASCII码编码的一种较复杂的数据编码规则(参考ASN.1编码方式简

介),有一定的自描述性,因此可以将其直接生成二叉树数据结构进行处理.

ASN.1协议组件提供了数据结构以及生成树算法和相应的查询算法。

ASN.1协议组件生成树算法流程图：



流程图说明：

1.算法实现主要依靠2个堆栈，一个用来存放父节点value长度，一个存放子节点累加长。

2.针对建树过程中要频繁使用树节点（CTLVNode），因此建立节点池以方便取用和销毁。

3.算法采用非递归算法，以保证效率和性能上的要求。

4.原始数据要缓冲到ASNBuffer中，ASNBuffer带有对ASN.1的描述属性，包括块长，填充字符，原始数据缓冲区指针，当前处理位置指针，这里要注意，原始数据缓冲只存在一份，ASNBuffer中采用指针指向缓冲区，因此，缓冲区在算法实现前和过程中不得改动和删除。

2.2.10**协议组件－BCD 和ASCII**

根据BCD码和ASCII码特性进行数据结构和算法准备，由于这两种编码较简单，因此程序依靠规则引擎进行直接处理，这里不再描述，请参考规则引擎设计。

2.2.11**Decode函数**

功能定义：

对原始话单的字段进行字符转换，原始话单每个字段都有自己的编码规则和方式，比如BCD码的字段有以0xFF填充的，ASCII码有以空格填充，等等，Decode函数是一套统一的字段级的编码处理函数集，由规则引擎调用来处理话单字段。

接口定义：

int32 (\*FUNC\_DECODE)( void\* pIn,

char\* pOut,

int32 nInLen,

char cSplit,

int32 nLen,

int32 nIncrement,

int32 nStart,

int32 nOrder);

接口使用方法：

解码引擎调用函数对输入的字段内容进行处理

pIn为输入的字段内容

pOut为输出内容

nInLen为解码缓冲区长度

cSplit 为ASCII码编码的分隔符

nLen 为需要解码的长度

nIncrement 标志缓冲区指针是否下移

nStart 解码开始位置

nOrder 对BCD码解码时，按顺序还是反序解码

2.2.12**Deal函数**

功能定义：

针对经过decode函数处理过的字段再做进一步处理，比如对枚举值的转换，对字段前补数据，后补数据等，由解码规则引擎调用。

接口定义：

int32 (\*FUNC\_DEAL)( const char\* pIn,

char\* pOut,

const char\* szReserve1,

const char\* szReserve2,

const char\* szReserve3 );

接口使用方法：

由解码引擎对decode函数执行后的字段进行处理

pIn为输入的字段内容

pOut为输出内容

szReserve1，szReserve2，szReserve3根据逻辑使用，比如需要在字段前加前缀，则前缀内容放入szReserve1

2.2.13**check函数**

功能定义：

针对经过decode函数处理过的字段做错单检查，如果根据字段可以判断话单是错单，则对话单打错单过滤。

接口定义：

int32 (\*FUNC\_CHECK)( const char\* pIn,

const char\* szReserve1,

const char\* szReserve2,

const char\* szReserve3 );

接口使用方法：

由解码引擎对decode函数执行后的字段进行处理

pIn为输入的字段内容

szReserve1，szReserve2，szReserve3根据逻辑使用。

2.2.14**filter函数**

功能定义：

针对经过decode函数处理过的字段做过滤检查，如果根据字段可以判断话单需要，则对话单直接过滤。

接口定义：

int32 (\*FUNC\_FILTER)( const char\* pIn,

char\* pOut,

const char\* szReserve1,

const char\* szReserve2,

const char\* szReserve3 );

接口使用方法：

由解码引擎对decode函数执行后的字段进行处理

pIn为输入的字段内容

pOut为输出内容

szReserve1，szReserve2，szReserve3根据逻辑使用。

2.2.15**解码特殊逻辑功能设计说明**

2.2.15.1**Gprs话单拆分**

话单拆分说明：

根据内容计费要求，GCDR话单包含一段对使用具体业务产生流量的数据内容，计费系统需要根据此数据内容做分业务流量的精确计费，针对此需求，系统要求解码具备对此类话单的拆分处理，拆分要求是将一条话单的每项业务流量分别拆成一条话单，以便后续业务的处理。



拆分流程说明：

1.拆分话单被程序读入，根据配置信息得到要拆分的字段，GCDR话单的流量字段存放于gprs详单的SERVICE\_USAGE字段，由字母‘N’来分割各段流量信息，举例如下：

“N4000000001,5438,4709,20080201160001,14,0,N4000000002,548,409,20080201160001,4,0”,N后以‘，’分隔字段顺序为服务代码，上行流量，下行流量，开始时间，时长，uri

2.根据拆分的分隔符‘N’判断要拆分为多少条话单，判断为拆分话单，则新建xdr详单，根据配置信息，将SERVICE\_USAGE字段的数据分拆到xdr详单的相应字段中，这些字段为xdr详单中的SERVICE\_CODE，UP\_VOLUME，DOWN\_VOLUME，START\_TIME，DURATION，URI，字段。

3.根据分隔符‘N’判断话单拆分解释后，将原有话单打错单过滤

2.2.15.2**漫游话单分拣**

语音话单漫游分解说明：

由于漫游话单和本地话单要走不同的流程处理，因此，在解码后就将漫游话单做分拣处理，以便后续流程处理方便。漫游语音话单的分拣依靠计费号码的hlr是否本地进行判断，非本地话单，则做漫游分拣。

语音话单漫游分拣流程图：



语音话单漫游分拣流程说明：

根据解码后的IMSI值组合计费用户的hlr。

根据hlr值查找hlr局数据，判断hlr是否为本地hlr，如果为本地hlr，则当作本地话单输出。

如果判断hlr为外地话单，则将文件后缀加‘.roam’进行输出处理。

2.2.15.3**Gprs话单漫游分拣**

Gprs话单漫游分拣说明：

Gprs话单的漫游判断基于计费号码的hlr和使用的SGSN Ip地址来决定，因为gprs计费话单现采集于GGSN，对一个用户而言，其在本省激活gprs pdp后漫游到外省，如果仅依靠hlr则无法判断其是否是漫游状态，所以，需要同时判断用户接入的SGSN Ip地址来判断对方是否漫游状态使用gprs。

Gprs话单漫游分拣流程图：



Gprs话单分拣流程说明：

根据hlr判断是否本地的流程与语音话单漫游分拣相同。

读取话单SGSN Ip地址到局数据中查找归属地，如果hlr判断为本地，SGSN Ip地址也同为本地，则为本地话单，将相应话单做本地保留，否则为漫游话单，做分拣输出。

SGSN地址判断中应判断话单的其实时间是否在局数据的生效日期内，如果查询不在生效日期内则打错单处理。

2.2.16**错单处理**

2.2.16.1**概述**

在计费系统中处理话单的时候，话单会因为某种原因造成不能正常计费而在系统中被当作错误话单处理。产生的错单中有些在修正错误根源之后可以被重新正常计费的，这类错单需要进行回收处理。另外，可能还需要对错单进行相关的统计分析。为了错单的回收处理及错单统计分析的方便，从而需要将错单按照预定义的类型分类入库到数据表中进行存储。

2.2.16.2**功能定义**

为了满足错单回收能出库生成计费系统能够处理的话单的需求，错单入库模块需要将错单按照系列化的方式将错单的各个字段入库，出库回收的时候才能够通过反系列化生成计费系统处理的详单。因此，新增加一个错单预处理模块，读入计费系统产生的错单，按照配置的各种时间限制及预定义的分类对错单进行处理，填入错单需要的特定字段信息并分发出入库的表名，然后将处理后的错单送给详单入库模块来完成错单的入库。

2.2.16.3**简要说明**

错单入库按业务配置作为一个单独的TS流程挂在云平台框架下。通过框架读取文件接口读取错单文件转化为SDLlist，然后将SDLlist传给错单预处理模块进行处理之后，通过详单入库模块将错单入库到数据库错单表中。

功能简要说明：

1.通过框架读取错单文件。

2.通过框架接口传入SDLlist,按条处理list中的记录。

3.按配置的时间限制及预定义的错单分类判断错单是否入库。

4.根据配置的时间限制及预定义的错单分类分发出错单入库的表名。

5.将错单需要的相关信息填入对应的详单SDL字段中。

6.将处理后的错单送详单入库模块进行入库处理。

7.详单入库模块根据填入的错单表名及错单信息将错单入库到对应的数据库错单表。

2.2.16.4**错误代码定义表**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Code | Data Type | Note |
| 错误码 | ERR\_CODE | VARCHAR2(16) | 错误代码 |
| 回收标识 | RECYCLE\_LIMIT | NUMBER(1) | 0：表示入错单表并进行回收；  1：表示不入错单表；  2：入NOTOUT表 表示此类错单不回收。 |
| 错误描述 | ERR\_DESC | Varchar(256) |  |
| 错单类型 | ERR\_TYPE | number(2) | 1预处理，2批价，3分发，4入库,5 部中心下发 6上发拦截 7解码 |

2.3漫游上发模块

2.3.1**漫游上发模块概述**

漫游上发子系统将批价后的漫游来访话单等数据文件按照集团的上发标准文件格式以及上发进度的要求形成文件，存放在指定目录下由接口子系统负责传输到全国中心。

漫入业务话单在省公司BOSS上传至集团一级BOSS系统的三期系统（语音业务）四期系统（内容计费系统）涉及的业务话单文件如下：

集团三期：D（gsm语音-漫游下发-国内漫游），ID，Q，R，RD（商旅服务业务），SD，I（国际漫游），IG（gprs下发），B（gsm语音-漫游下发-漫游呼转）

智能网：VPMN，VC，PPS，OLYVC

集团四期：MM（MMS彩信业务-下发），STM，CM，M（下发梦网），MREAD，BSMS



B文件是结算系统上传的。

PPS文件已经合并到D文件内上传。

BSMS文件各省也是结算上传，河南特殊。

M文件是分省结算，同时上传31个省。

2.3.2**功能定义**

根据最新Openbilling60的架构能力，将漫游上发处理功能拆分三个功能域： 漫游分拣、漫游检验、漫游生成。

2.3.2.1**漫游分拣**

数据分拣子模块处理批价后的话单文件，根据不同上发文件要求定义的话单特征（如漫游类型、用户类型、对端类型的组合），将符合这些分拣条件的话单输出到不同的文件中，以不同的文件名后缀展示，供后续模块使用。

功能要点：以话单SDL为输入；分拣规则通过LUA引擎定义；输出话单SDL支持输出到不同的后缀名文件中；支持一条输入，多条输出。即支持一条话单输出到2个以上的业务中；此模块可作为一个节点配置在批价长流程里。

2.3.2.2**漫游检验**

漫游检验子模块以分拣后（不过分拣的业务则是批价后）的文件数据为输入，按照不同的上发类型的业务要求，对源数据进行字段级的合法性校验，并按照全国中心规范进行字段标准化，拦截不需要的话单或错单，最后以自定义SDL的形式落地，供漫游生成模块使用。

功能要点：在新版框架下，以单独的TS模块呈现；以话单SDL为输入；检验规则通过LUA引擎定义；输出话单SDL经过整理输出模块内定义的体记录SDL；需要输出话单时间、费用、时长、流量作为统计要素，以内部定义SDL形式输出，供后续汇总模块使用；此模块可以启动多个；根据业务规则生成体记录（业务规则由LUA引擎定义）。

2.3.2.3**漫游生成**

以漫游检验模块的输出作为输入源，按照不同的上发类型及上发周期要求，整理上发文件体记录，统计上发文件信息，形成上发文件头尾记录，定时生成符合全国中心格式要求和命名规范的标准格式文件，而后由传输程序上发集团中心

功能要点：在新版框架下，以单独的TS模块呈现；输入话单格式为自定义SDL；检查系统时间是否到达上发时刻；根据业务规则生成头记录（生成规则由LUA引擎定义）；根据业务规则生成尾记录（生成规则由LUA引擎定义）；将尾记录添加到预上发文件中形成上发文件，输出至上发目录；根据业务规则生成统计信息（统计规则由LUA引擎定义）；根据业务规则分析话单的最早与最迟时间；根据业务规则实现延迟话单的过滤管理；根据业务规则识别错单重传话单（据2012集团考核）；Later\_Stat数据产生和输出；实现查重功能；此模块按照业务配置启动，只能启动一个。

2.3.3**漫游上发内外部结构关系图**



rat长流程的输出话单是漫游上发模块的输入，漫游上发输出上发给集团，漫游上发的备份话单是漫入话单入库的输入源。

2.3.4**漫游上发内部结构关系图**

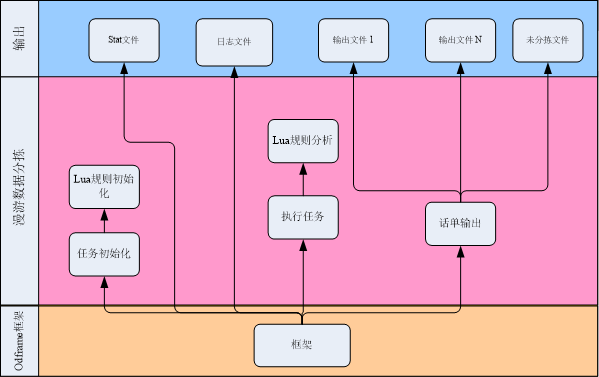


每类业务的漫游检验模块可以配置为多进程（以多个ts呈现），而漫游生成模块只能单进程单线程来实现。

2.3.5**漫游上发功能模块简要说明**

2.3.5.1**漫游分拣**

内部逻辑图：



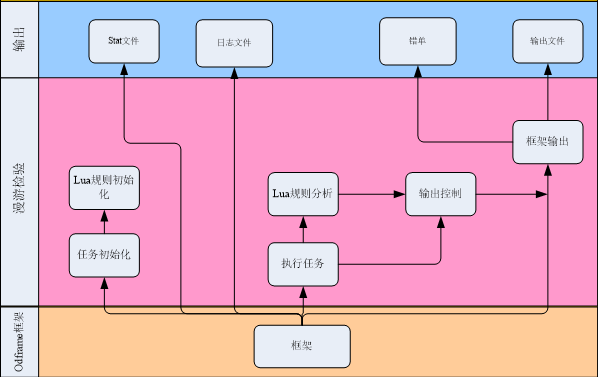
分拣模块处理话单SDL的List结构，根据LUA的分析结果填写业务后缀名称，这里加载LUA 模板是多个的，业务后缀在程序中以列表形式管理。

功能时序如下：读取配置文件；读取LUA模板；调用基础库编译模板，形成处理话单的调用入口函数；分析输入的SDL的List结构；逐条处理SDL，循环调用LUA函数；根据LUA返回的业务类型组成的业务列表，复制SDL输出。

其中模板的功能逻辑如下：读取SDL关键字段；分析字段内容，获得话单归属业务类型；将话单归属对业务类型填写到指定字段。

2.3.5.2**漫游检验**

内部逻辑图：



漫游检验upc模块处理话单SDL的List结构，根据LUA的分析结果，形成体记录和输出统计信息，这里的LUA模板实现业务逻辑（上发字段的生成，上发字段合理性检查），输出统计信息应该包括：话单开始时间、话单结束时间、时长、流量、费用（基本费、长途费1、长途费2）

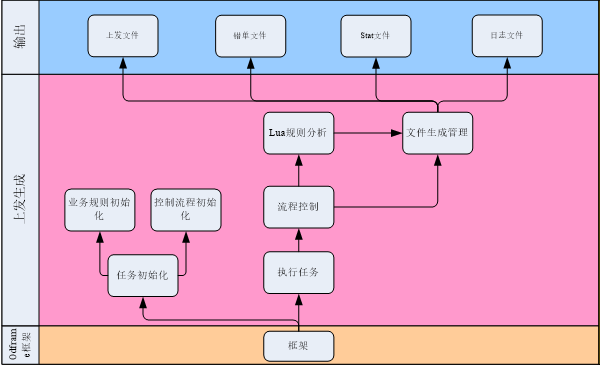
功能时序如下：读取配置；读取LUA模板；调用基础库编译模板，形成处理话单的调用入口函数；分析输入的SDL的List结构；逐条处理SDL，调用模板；

输出漫游上发内部定义的SDL结构给下一个节点漫游汇总。

其中模板的主要功能时序如下：读取话单关键字段；按照集团检验规则，逐个字段转换出上发使用的字段；按照集团检验规则，逐个错误码检验字段；符合错误信息的进行打错单处理；正确话单输出体记录信息，填写到输出的内部定义SDL上；整理出统计信息字段内容填写到内部定义的SDL上。

2.3.5.3**上发生成**

内部逻辑图：



这里处理经过漫游检验的新SDL数据，将体记录记录到指定文件，并记录当前的累计信息。同时根据话单处理时刻确定上发的文件名、头记录信息、尾记录信息，组织上发文件在上发时刻生成。

功能要点：读取配置；读取LUA模板；加载计算上发时刻计算库以及函数，向模板注册记录信息；按照处理流程检查上发处理环境；首次运行初始化配置文件和进度信息文件、累计信息文件、体记录保存信息文件；再次运行跳过初始化，直接处理漫游检验过的SDL结构list；检查是否到达上发时刻，如果否，返回模块接口。如果是继续生成文件处理；生成文件后清理上发运行环境，更新需要初始化的标签。

逻辑流程图：



2.3.6功能实现说明

2.3.6.1**漫游分拣**

功能定义：

读取sdl话单，利用lua模版对各个字段按照分拣条件逐一分析，分拣条件是一系列字段取值范围的集团；根据LUA的分析结果填写业务后缀名称，由框架负责输出到不同文件中。

处理流程：

从sdl读取各字段的值，判断各字段组合是否符合分拣条件；如果符合则填写相应后缀，不符合分析下一类文件分拣条件，如果不符合任一文件则保留在原文件中。

数据流程图：



类定义：

Class CSorterAlgo

Class CSorterConfigDef

Class LuaFileConfig

Class LuaFuncConfig

Class CSorterConfig: public NAME\_SPACE\_OD\_FRAME CAppConfigBase

接口对象：

File： sorter\_base.sdl

#include "xdr\_def.sdl"

module MSorterApp

{

interface ISorterApp

{

int32 sorter(inout MXdr::SXdrList listXdr);

}

}

2.3.6.2**漫游检验**

功能定义：

利用lua模板业务规则分析，解析话单实现业务处理（字段检错、格式化、统计等）；分析某些字段合法性时需要XC接口调用（局数据访问），如分析对端手机号码的归属，需要访问hlr信息；最话单各字段进行分析，生成符合检错规则的体记录。

数据流程图：



实现说明：

根据集团文件格式要求生成体记录，利用lua模板对输入的sdl话单进行处理，分析其字段内容是否符合集团检错的规则（如果不符合，需要作错单处理）；对字段作格式化处理；形成体记录，输出自定义sdl供输出处理；

这里的LUA模板实现业务逻辑（上发字段的生成，上发字段合理性检查）；自定义sdl除了包含上发体记录以外，还应该包括：话单开始时间、话单结束时间、时长、流量、费用（基本费、长途费1、长途费2）。

业务分析的分析接口包含了查询XC静态数据的分析功能，同时定义了丰富的全局统一的分析参数。漫游上发的检验模块复用业务分析的这些代码可以统一分析代码的统一试图（变量定义统一），可以统一模板风格。

类定义：

CRplGlobal CRplConfig

说明：与业务分析全局模板和配置文件对象相同。

接口对象：

File： checker\_base.sdl

#include "xdr\_def.sdl"

#include "upload\_base.sdl"

module MCheckerApp

{

interface ICheckerApp

{

int32 checker(in MXdr::SXdrList listXdr, outMUpload::SUploadList listUpload);

}

}

Fil：upload\_base.sdl

module MUpload

{

struct SUpload

{

string<1024> body\_record;

string<8> file\_type;

int64 volume;

int64 duration;

int64 start\_time;

int64 stop\_time;

int64 cfee;

int64 lfee1;

int64 lfee2;

}

list vector< SUpload > SUploadList;

}

2.3.6.3**上发生成**

数据流程图：



2.3.6.3.1**实现说明**

漫游生成处理上一模块（上发校验）生成自定义话单的List结构作为输入源；调用LUA的分析，输出包括文件名/头记录/统计信息累加，定时生成符合集团要求的上发文件；LUA模板实现业务逻辑：根据话单开始时间统计最早/最迟通话时间、累加时长、流量、费用信息。

处理流程：

上发任务初始化：

1.读取配置信息，目录信息，源文件匹配的正则表达式定义，数据生成周期等信息；

2.数据访问接口初始化，数据生成模块使用的局数据由静态数据管理中心（XC）统一提供，需要先注册到ODAC才能访问其封装的局数据；

3.根据配置的是否查重，对查重接口作初始化工作；查重由查重库提供的查重接口调用；

4.进度信息的读取；数据生成是要求定时生成的，所以每处理完一个上发文件，需要记录该进度信息，来控制上发文件的序号以及生成时间；

5.业务规则的初始化；上发文件名，头、体、尾记录是按照一定的规则处理，而这些规则都是由业务规则模板定义，需要对该模板作初始化分析。



2.3.6.3.1.1**流程管理**

运行模式的控制，输入、输出控制，业务规则的分析调用等。

处理流程：

1.运行模式主要有：测试处理模式：生成指定的文件日期、序号的文件，处理完成，程序退出；后台运行模式：通过计算当前需要生成的文件，文件处理完成，程序继续运行，等待至下一个文件生成时间。

2.输入控制，读取输入文件，转换成所需字段信息。

3.将上一环节生成的记录输出，输出到待上发文件或错单文件，同时需要输出统计、监控等信息。



2.3.6.3.1.2**业务规则分析**

功能定义：用于生成文件名、文件的头、尾、体记录，统计话单总数，统计总费用等。

处理流程：

1.组织文件名：根据输入的文件类型，文件日期、文件的序号组织符合要求的文件名；如D文件的名称：D0505001.371（[文件类型][4位日期，月日][3位序号].[3位省代码]）。

2.组织头记录：根据集团文件格式要求生成头记录，主要包括头标记、被访局代码、归属局代码、文件序号、产生日期、版本号等信息。

3.组织尾记录：根据集团文件格式要求生成尾记录，主要包括尾标记、被访局代码、归属局代码、文件序号、通话次数、通话开始结束时间、以及相关费用的统计值。

4.组织体记录：获取自定义sdl中提记录部分的信息。



2.3.6.3.1.3**文件生成管理**

功能定义：

完成上发文件的输出，错单输出，日志输出以及管理信息（含有监控、告警、一次延迟等）的输出。

处理流程：

1.对业务规则分析处理的结果，如头、体、尾记录以及错单记录需要分别输出到上发文件或者错单文件；

2.对处理输入文件的stat信息输出到stat文件；

3.处理的日志需要输出到日志文件，供查询分析等；



2.3.6.3.2**主要功能**

说明：

统计文件信息，组织生成头尾记录；延时话单拦截，包括一次延时和二次延时；根据配置定时生成上发文件，控制上发进度；根据配置项进行漫游上发模块内的查重功能；程序读写备份不依赖框架，框架只负责调用接口。

优点;

新版本实时处理话单，定时上发，可以准确预见上发时间。

类定义：

CRplGlobal

接口对象：

File： roamin\_base.sdl

/\* $Id: roamin\_base.sdl 231842 2012-11-02 08:42:49Z laigc $ \*/

#include "xdr\_def.sdl"

#include "upload\_base.sdl"

module MRoaminApp

{

interface IRoaminApp

{

int32 roamin();

}

}

2.3.6.3.3**配置文件**

相关xml配置部分：

  <roamin\_info  isEdit="1">

<lua\_file\_out\_path  isEdit="1">$BILL\_HOME/template</lua\_file\_out\_path> ------ 模版路径

<file\_type  isEdit="1">D</file\_type> 上发文件类型

<this\_prov\_code  isEdit="1">210</this\_prov\_code>上发省代码

<deal\_date  isEdit="1">20121115</deal\_date>首次（单次）处理日期

<deal\_no  isEdit="1">1</deal\_no> 首次（单次）处理编号

<run\_mode  isEdit="1">real</run\_mode> 处理模式（单次和实时）

<prov\_list  isEdit="1"/> 处理省代码（M文件有效）

<uploader\_cycle  isEdit="1">0</uploader\_cycle> 循环周期（天、月）

<count\_per\_cycle  isEdit="1">96</count\_per\_cycle>一天上发文件数量

<init\_number  isEdit="1">0</init\_number> 首个上发文件编号

<number\_step  isEdit="1">1</number\_step> 上发编号步长

<reup\_num\_list  isEdit="1">59;61;66;72</reup\_num\_list>重传文件编号

<file\_regex\_rule  isEdit="1">[a-zA-Z]+\*.D</file\_regex\_rule> 正则表达式

<input\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/in</input\_dir> 输入目录

<back\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/bak</back\_dir>备份目录

<reup\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/reup/</reup\_dir>重传目录

<convert\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/convert/</convert\_dir> 临时工作目录

<uploader\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/send/</uploader\_dir> 上发文件生成目录

<running\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/running/</running\_dir> 进度信息跟踪目录

<stat\_dir  isEdit="1">$DATA01/upfilegen/upf\_d/stat/</stat\_dir> 统计信息目录

<error\_dir  isEdit="1">$ERR\_RAW1/upfilegen/upf\_d/</error\_dir> 错单输出目录

<error\_code  isEdit="1">E999</error\_code>

<later\_stat\_file  isEdit="1">$DATA01/upfilegen/upf\_d/stat/later.stat</later\_stat\_file> 延迟信息统计文件

<delete\_later  isEdit="1">1</delete\_later> 延迟拦截开关

<max\_file\_once  isEdit="1">500</ max\_file\_once > 一次扫描最大输入文件数

<time\_silce  isEdit="1">1</time\_silce > 输入目录为空时，系统休息时间片

<reup\_later\_delete  isEdit="1">1</reup\_later\_delete>

<before\_mins  isEdit="1">115</before\_mins> 二次延迟拦截时间

<before\_mins\_fwd  isEdit="1">115</before\_mins\_fwd>

<later\_once\_mins  isEdit="1">27</later\_once\_mins> 一次延时拦截时间

<max\_later\_once  isEdit="1">200000</max\_later\_once> 一次延时拦截门限

<filter\_later\_mode  isEdit="1">0</filter\_later\_mode> 一次延时拦截模式（门限内上发或门限内打错单）

<number\_2\_adjust  isEdit="1">-1:0;</number\_2\_adjust>上发文件最终生成时间调整

<checkdupflag  isEdit="1">1</checkdupflag> 查重开关

<indexpath  isEdit="1">$DATA01/upfilegen/upf\_d/index</indexpath> 查重索引目录

<dupfield  isEdit="1">1:17;33:70;228:246</dupfield> 查重键值

<expr\_file  isEdit="1">$BILL\_HOME/template/upload/roam\_d.lua</expr\_file> 上发生成模版目录

<expr\_func  isEdit="1">roamin\_d\_main</expr\_func>执行模版主函数

              </roamin\_info>

1. 相关测试

3.1测试内容

贵州解码文件按地市拆分，能否正确处理话单。

3.2测试相关配置数据

修改花溪中心以及贵安中心的解码相关配置。打开流程编辑，修改ts中xml的一些字段：

**由** <Hplmn2JudgeFlag>2</Hplmn2JudgeFlag>

<center\_type\_list>

<center\_type\_table\_type\_pair>

<center\_name>A</center\_name>

<center\_list>850;851;852;853;854;999</center\_list>

</center\_type\_table\_type\_pair>

<center\_type\_table\_type\_pair>

<center\_name>B</center\_name>

<center\_list>855;856;857;858;859</center\_list>

</center\_type\_table\_type\_pair>

</center\_type\_list>

**修改为** <Hplmn2JudgeFlag>2</Hplmn2JudgeFlag>

<center\_type\_list>

<center\_type\_table\_type\_pair>

<center\_name>A</center\_name>

<center\_list>852;855;856;857;859;999</center\_list>

</center\_type\_table\_type\_pair>

<center\_type\_table\_type\_pair>

<center\_name>B</center\_name>

<center\_list>850;851;853;854;858</center\_list>

</center\_type\_table\_type\_pair>

</center\_type\_list>

3.3测试用例

验证话单是否正确处理

3.3.1**贵州-花溪中心**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case\_ID | 1 | 所属业务 | gsm | 所属模块 | decode | |
| 所属项目 | VerisBilling(gz) | | | | | |
| 最后修改者 | 赵立伟 | 最后修改日期 | 2016/7/29 | | 有效性 | T |
| 测试点描述 | 验证话单是否由正确的中心处理 | | | | | |
| 条件（DB/MDB,CFG） | | | | | | |
| 测试条件配置：  1.查询流程hx\_de\_gsm\_hw\_cs  2.在流程编辑里打开xml在 <ProvCode>851</ProvCode>后添加下列字段，如图所示：     1. 点击完成并保存 2. 将此流程配置更新 | | | | | | |
| 输入(CDR/XDR) | | | | | | |
| IMG_256 | | | | | | |
| 输出(XDR, DB/MDB) | | | | | | |
| IMG_256 | | | | | | |
| 测试步骤 | | | | | | |
| 1. 创建读取目录：$DATA01/decode/hx\_de\_gsm\_hw\_cs   输出目录：$DATA01/rating/rat\_gsm  错单目录：$ERR\_RAW1/decode/hx\_de\_gsm\_hw\_cs  错单目录（错单规整）：$ERR\_SPLIT1/decode/hx\_de\_gsm\_hw\_cs  日志目录：$BILL\_LOG1/decode/hx\_de\_gsm\_hw\_cs   1. 将话单放入到读取目录：$DATA01/decode/hx\_de\_gsm\_hw\_cs 2. 启动hx\_de\_gsm\_hw\_cs流程 3. 观察输出目录的结果 | | | | | | |

3.3.2**贵州-贵安中心**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case\_ID | 1 | 所属业务 | gsm | 所属模块 | decode | |
| 所属项目 | VerisBilling(gz) | | | | | |
| 最后修改者 | 赵立伟 | 最后修改日期 | 2016/8/5 | | 有效性 | T |
| 测试点描述 | 验证话单是否由正确的中心处理 | | | | | |
| 条件（DB/MDB,CFG） | | | | | | |
| 测试条件配置：  1.查询流程ga\_de\_gsm\_hw\_cs  2.在流程编辑里打开xml在 <ProvCode>851</ProvCode>后添加下列字段，如图所示：    3.点击完成并保存  4.将此流程配置更新 | | | | | | |
| 输入(CDR/XDR) | | | | | | |
| IMG_256 | | | | | | |
| 输出(XDR, DB/MDB) | | | | | | |
|  | | | | | | |
| 测试步骤 | | | | | | |
| 1. 创建读取目录：$DATA01/decode/ga\_de\_gsm\_hw\_cs   输出目录：$DATA01/rating/rat\_gsm  错单目录：$ERR\_RAW1/decode/ga\_de\_gsm\_hw\_cs  错单目录（错单规整）：$ERR\_SPLIT1/decode/ga\_de\_gsm\_hw\_cs  日志目录：$BILL\_LOG1/decode/ga\_de\_gsm\_hw\_cs  2.将话单放入到读取目录：$DATA01/decode/ga\_de\_gsm\_hw\_cs  3.启动ga\_de\_gsm\_hw\_cs流程  4.观察输出目录的结果 | | | | | | |

3.4测试结果评价

将花溪中心话单的输出结果与贵安中心话单的输出结果对比，完全吻合，测试通过。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **功能点** | **测试用例及数据** | **预期结果** | **实测结果** | **测试日期** | **测试人** |
| 贵州-解码文件按地市拆分需求话单处理的正确性 | 详见章节3.3.1和3.3.2 | 详见章节3.3.1和3.3.2 | 和预期一致 | 2016/7/29 | 赵立伟 |

致 谢

光阴似箭，日月如梭，转瞬之间漫长的大学四年学习生涯即将完结，不由的感叹自己的大学就这样结束了，回首以往的四年经历，这其中有军训时的吃苦训练，也有平时生活中和同学们的嬉戏打闹，这一切都是那么快乐，给我留下了许许多多美好的回忆。想想这几年还是有许多遗憾，有许多目标没有完成，但人生就是这样，没有什么都是完美的总会有一些小小的缺陷。这四年来帮助过我的人实在是太多了。我要感谢这几年来教育过我的所有老师们，是你们让我学习了许多专业知识，也教会了我如何成为一名合格的大学生。我要感谢我的辅导员，是你们在我生活学习上出现问题时及时帮助了我。同时我也要感谢和我一起度过这四年的同学们，正是因为有了你们，才会让我这四年的大学生活变得丰富多彩。我还要感谢我的学校指导老师，他在我的毕业设计上给予了我非常大的帮助，让我在迷茫的时候找到了正确的方向。我还要感谢我实习期间帮助过我的导师们，你们总是非常耐心地指导我，及时纠正我的错误并悉心指导，导师们在工作上是良师益友，在生活上更是好朋友。我还要感谢我的亲人们，是他们在背后默默支持我，让我更加有动力。最后我要感谢我的母校东南大学，感谢母校为我创造的学习环境，也感谢母校所有的老师们，是你们让我成长，感谢东南大学！

参考文献

[1] 黄远方.基础数据计费系统开发[J].《软件导刊》，2012（10）

[2] 蒋晓娟.在线计费系统分析[J].《中国新技术新产品》，2012（24）

[3] 李辉.在线计费系统建设方案分析[J].《科技创新与应用》，2013（28）

[4] 张冬冬，电信计费系统的研发与分析[D]:硕士学位论文.长春：吉林大学软件工程，2003

[5] 蒋东兴，刘启新，白冰，戚丽.校园网络计费系统研究与实现.中图分类号：TP393.1

[6] Jiang Dongxing, Liu Qixin, Bai Bing, Qi Li.The Research and Design of Campus Network Accounting System.CLC number:TP393.1

[7]Bruce Tate.Persistence strategies.IBM  http://www.ibm.com/developerworks/

[8]中国电信集团公司，《中国电信数据及多媒体业务计费帐务系统技术规范》

[9]赵洪涛，高萍.电信运营的在线计费系统[J].黑龙江科技信息，2010(22)

[10]朱一珍.在线计费在未来电信计费解决方案的应用[J].广东通信技术，2006(5)

[11]李连祥，刘晓亮，董斌.3G融合在线计费系统的实现方案[J].电信科学，2006(5)

[12]刘逢源，吴纯青.3GPP在线计费系统的几种典型架构研究与分析[J].电脑知识与技术(学术交流)，2007(9)

[13]田宇.基于策略的电信增值业务计费子系统的设计与实现[D]:硕士学位论文.成都：电子科技大学软件工程,2014

[14]戚玉峰.在线计费系统(OCS)关键技术的研究与实现[D]:硕士学位论文.南京：南京邮电大学,2011