SSD-TECH

inboxHandler Application Manual

inboxHandler Manual: 1.0

Created: 30 March 2018

Update: 16 April 2020

Systems Solutions and Development Technologies Sdn. Bhd.  
Kuala Lumpur  
Malaysia

Confidentiality: Distribution of this document is restricted. Holders of this document must ensure the confidentiality of the content and under no circumstances is this document to be shown or released to anyone without the prior consent of SSD-TECH.

Copyright: This document and its contents belongs to SSD-TECH it may not be copied, photocopied, scanned or redistributed without written consent from SSD-TECH.

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision Number | Date | Remarks | Introduced by |
| 1.0 | 11 April 2018 | Incomplete Initial version. Done until Configuration items. | Raiyan Ferdous |
| 2.0 | 30 April 2018 | Initial version | Md. Kamrul Hasan Shourov |
| 3.0 | 03 May 2018 | Adding detailed result code description | Md. Kamrul Hasan Shourov |
| 4.0 | 13 May 2018 | Changing graceful shutdown process mechanism | Md. Kamrul Hasan Shourov |

Contents

[1.0 Introduction 9](#_Toc513981288)

[2.0 Executive Summary 9](#_Toc513981289)

[3.0 System Architecture 9](#_Toc513981290)

[4.0 CGW Application Requirements 9](#_Toc513981291)

[5.0 CGW Configuration 10](#_Toc513981292)

[Configuration Items 12](#_Toc513981293)

[Server Specific configuration 12](#_Toc513981294)

[CGWID 12](#_Toc513981295)

[UserID 12](#_Toc513981296)

[LISTENING\_PORT 12](#_Toc513981297)

[COMMAND\_LISTENING\_PORT 12](#_Toc513981298)

[LOAD\_PACKAGE\_TABLE 13](#_Toc513981299)

[NO\_OF\_THREADS 13](#_Toc513981300)

[PARALLEL\_BILLING\_REQUEST 13](#_Toc513981301)

[REGISTRATION\_OUTPUT\_ENABLED 13](#_Toc513981302)

[CACHE\_ENABLED\_FOR\_WALLET 13](#_Toc513981303)

[RETRIEVE\_SUBSCRIPTION\_INFO\_USING\_IN\_MEM\_DB 13](#_Toc513981304)

[CGW Request Transformation Configuration 14](#_Toc513981305)

[DISCARD\_PREFIX\_OBD 14](#_Toc513981306)

[CHECK\_DISCARD\_PREFIX\_OBD 14](#_Toc513981307)

[MSISDN\_LENGTH 14](#_Toc513981308)

[COUNTRY\_CODE 14](#_Toc513981309)

[Rate Plan Configuration 14](#_Toc513981310)

[REFUND\_SESSION 15](#_Toc513981311)

[REGION\_CHECK\_ENABLE 15](#_Toc513981312)

[CALL\_TYPE\_ID\_CHECK 15](#_Toc513981313)

[DIRECT\_DIALING\_CHECK 15](#_Toc513981314)

[PACKAGE\_RANGE\_CHECK\_ENABLE 16](#_Toc513981315)

[PACKAGE\_LOADING\_AFTER 16](#_Toc513981316)

[DefaultPackageID 16](#_Toc513981317)

[MinPulseSize 16](#_Toc513981318)

[DefaultTimeSlotId 16](#_Toc513981319)

[DefaultCallTypeId 16](#_Toc513981320)

[DefaultTPS 16](#_Toc513981321)

[Persistent Database 16](#_Toc513981322)

[HOST 17](#_Toc513981323)

[PORT 17](#_Toc513981324)

[USER 17](#_Toc513981325)

[PASSWORD 17](#_Toc513981326)

[DB 17](#_Toc513981327)

[MAX\_CONN 17](#_Toc513981328)

[MAX\_RETRY 17](#_Toc513981329)

[In Memory Database 17](#_Toc513981330)

[REDIS\_HOST 17](#_Toc513981331)

[REDIS\_PORT 17](#_Toc513981332)

[REDIS\_PASSWORD 17](#_Toc513981333)

[REDIS\_DB 17](#_Toc513981334)

[REDIS\_MAX\_CONN 17](#_Toc513981335)

[REDIS\_MAX\_RETRY 18](#_Toc513981336)

[Billing Interface Configuration 18](#_Toc513981337)

[IN\_INTEGRATED 18](#_Toc513981338)

[DEFAULT\_BILLING\_HOST 18](#_Toc513981339)

[DEFAULT\_BILLING\_PORT 18](#_Toc513981340)

[PACKAGE\_CHECK\_ENABLED\_SESSION 18](#_Toc513981341)

[PACKAGE\_CHECK\_ENABLED\_SPECIFIC 18](#_Toc513981342)

[PACKAGE\_CHECK\_ENABLED\_WITH\_BALANCE 19](#_Toc513981343)

[NUMBER\_OF\_RETRY\_TO\_FIND\_PACKAGE 19](#_Toc513981344)

[PACKAGE\_HOST\_IP 19](#_Toc513981345)

[PACKAGE\_HOST\_PORT 19](#_Toc513981346)

[RECV\_TIMEOUT 19](#_Toc513981347)

[SHOULD\_USE\_REMARKS 19](#_Toc513981348)

[Log Configuration 19](#_Toc513981349)

[LOG\_LEVEL 20](#_Toc513981350)

[LOG\_FOR\_ANY\_MSISDN 20](#_Toc513981351)

[LOG\_INSTANCE 20](#_Toc513981352)

[LOG\_MSISDN 20](#_Toc513981353)

[LOG\_DIRECTORY 20](#_Toc513981354)

[LOG\_FILE\_CREATION\_INTERVAL 20](#_Toc513981355)

[LOG\_FILE\_LINE\_SEPARATOR 20](#_Toc513981356)

[CDR Configuration 21](#_Toc513981357)

[NUMBER\_OF\_CDR\_PROCESSING\_THREADS 21](#_Toc513981358)

[CDR\_WRITE\_IN\_FILE\_ENABLE 21](#_Toc513981359)

[IMMEDIATELY\_SAVE\_SUCCESS\_CDR 21](#_Toc513981360)

[CDR\_DIRECTORY 21](#_Toc513981361)

[CDR\_FILE\_CREATION\_INTERVAL 21](#_Toc513981362)

[CDR\_DUPLICATION\_CHECK\_ENABLE 22](#_Toc513981363)

[CDR\_DUPLICATION\_LAST\_RESPONSE 22](#_Toc513981364)

[INSERT\_INTO\_CRD\_FOR\_TPS 22](#_Toc513981365)

[INSERT\_INTO\_CRD\_FOR\_ERROR\_IN\_APPID\_OR\_PASSWORD 22](#_Toc513981366)

[CDR\_FIELD\_SEPARATOR 22](#_Toc513981367)

[CDR\_LINE\_SEPARATOR 22](#_Toc513981368)

[CDR\_BATCH\_SIZE 22](#_Toc513981369)

[MySQL-Redis SYNC application (RDSS) configuration 22](#_Toc513981370)

[SYNC\_APP\_QUEUE\_NAME 23](#_Toc513981371)

[SUBSCRIPTION\_INFO\_SEPARATOR 23](#_Toc513981372)

[SUBSCRIPTION\_INFO\_KEY\_PREFIX 23](#_Toc513981373)

[SUBSCRIPTION\_INFO\_KEY\_SEPARATOR 23](#_Toc513981374)

[WALLET\_INFO\_VALUE\_SEPARATOR 23](#_Toc513981375)

[WALLET\_INFO\_KEY\_PREFIX 23](#_Toc513981376)

[WALLET\_INFO\_KEY\_SEPARATOR 23](#_Toc513981377)

[SECONDARY\_WALLET\_INFO\_KEY\_PREFIX 23](#_Toc513981378)

[SECONDARY\_WALLET\_INFO\_KEY\_SEPARATOR 23](#_Toc513981379)

[Log Server configuration 23](#_Toc513981380)

[LOG\_SERVER\_MAX\_CONTEXTS = 2000 23](#_Toc513981381)

[LOG\_SERVER\_ENABLE 23](#_Toc513981382)

[LOG\_SERVER\_IP 23](#_Toc513981383)

[LOG\_SERVER\_PORT 24](#_Toc513981384)

[LOG\_SERVER\_RECONNECT\_INTERVAL 24](#_Toc513981385)

[LOG\_SERVER\_CONCURRENT\_SENDER 24](#_Toc513981386)

[LOG\_SERVER\_MAX\_CONTEXTS 24](#_Toc513981387)

[6.0 Start CGW Application 24](#_Toc513981388)

[7.0 Result codes 24](#_Toc513981389)

[8.0 Result Code Description 25](#_Toc513981390)

[2001 25](#_Toc513981391)

[9004 25](#_Toc513981392)

[9005 25](#_Toc513981393)

[9010 25](#_Toc513981394)

[9012 25](#_Toc513981395)

[9016 25](#_Toc513981396)

[9017 25](#_Toc513981397)

[9021 25](#_Toc513981398)

[9023 25](#_Toc513981399)

[9024 25](#_Toc513981400)

[9025 25](#_Toc513981401)

[9028 26](#_Toc513981402)

[9029 27](#_Toc513981403)

[9030 27](#_Toc513981404)

[9031 27](#_Toc513981405)

[1012 27](#_Toc513981406)

[9.0 Request format 27](#_Toc513981407)

[CGW 27](#_Toc513981408)

[Parameter description 27](#_Toc513981409)

[Specific charge request 29](#_Toc513981410)

[Session based request 29](#_Toc513981411)

[IN Requester 29](#_Toc513981412)

[Application interface 29](#_Toc513981413)

[Web service interface 30](#_Toc513981414)

[Parameter description 30](#_Toc513981415)

[10.0 Response format 31](#_Toc513981416)

[CGW 31](#_Toc513981417)

[Format of response 31](#_Toc513981418)

[Parameter description 31](#_Toc513981419)

[IN Requester 32](#_Toc513981420)

[11.0 Command Server 33](#_Toc513981421)

[Sample table reloading 33](#_Toc513981422)

[Sample configuration loading 34](#_Toc513981423)

[Sample rate context 35](#_Toc513981424)

[Sample rate information 35](#_Toc513981425)

[Sample graceful shutdown process 36](#_Toc513981426)

[Notes on CDR batch processing 36](#_Toc513981427)

[12.0 Package finding steps 37](#_Toc513981428)

[13.0 Package loading process 37](#_Toc513981429)

[14.0 Package information finding from REDIS server 37](#_Toc513981430)

Terminologies

CBI Charging and billing interface

CGW Charging Gateway

IN Intelligent Network

RDSS Relational Database Syncing System

WS Web Services

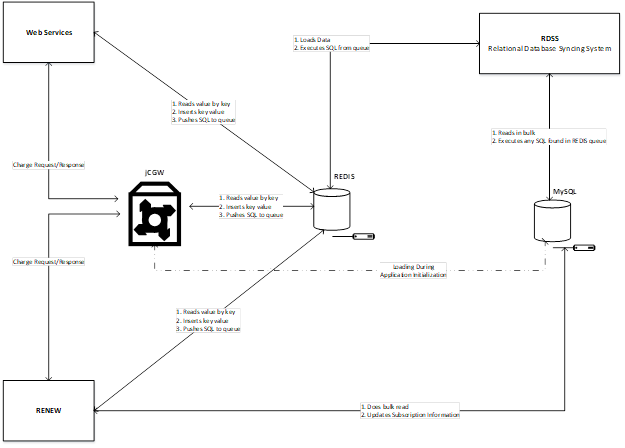
# Introduction

This document will define the SSD-TECH inboxHandler application architecture, functionalities, deployment and operational procedures. The Technology Operations team must comply to this document for successful deployment and operations of inboxHandler Application.

# Executive Summary

This version of inboxHandler application uses Inter Process Communication Ques(IPCQ) as primary storage and MySQL as secondary storage. Here you can take full advantages of RDSS application. With RDSS, CGW application can avoid any direct communication with MySQL and save a huge time during each transaction and make itself more responsive and performant to the end users. (see “Utilize RDSS with your application” section for more details)

# System Architecture



# CGW Application Requirements

1. JAVA 8
2. MySQL 5.7.x
3. REDIS 4.x.x

# inboxHandler Configuration

InboxHandler is highly dependent on its configuration file. The configuration is written in smpp.ini file. Smpp.ini file is also used as the configuration file for sms gateway application. But all the parameters of smpp.ini file are NOT necessary for inboxHandler. Among the parameters only 18 of them are used in this application.

[Inboxhandler Configuration]

CDE\_URL->

http://192.168.108.129/SMSServices\_4\_0/core/cde.php?mn=%s&msg=%s&sc=%s&rempc=%s&op=%s&tid=%s&retry=%d

NO-REPLY-STRING->no-reply,+OK

[Database related information]

DATABASE\_HOST->localhost

DATABASE\_USERNAME->root

DATABASE\_PASSWROD->Nopass!234

DATABASE\_NAME->smsgw

TPS->500

[MakeQueOrFail thread enable configuration]

IS\_REQUE\_ENABLE->0

RETRYCOUNT->0

RETRY\_DELAY->10 SECOND

[SMS Sender Thread Config]

APPEND\_SHORTCODE\_PREFIX->1

[Log writing specification]

SMSLOG\_ENABLED->1

LOG\_LEVEL->255

LOG\_DESTINATION->10

LOG\_HOST->localhost

LOG\_PORT->3470

LOG\_COMPONENT->SMSGW

[Command process]

COMMAND\_PROCESS\_PORT\_CDE->3556

## Configuration Items

The configuration items for inboxHandler can be grouped in six major class. They are described one by one below:

## Inboxhandler Configuration

This consists of the following values:

CDE\_URL->

http://192.168.108.129/SMSServices\_4\_0/core/cde.php?mn=%s&msg=%s&sc=%s&rempc=%s&op=%s&tid=%s&retry=%d

NO-REPLY-STRING->no-reply,+OK

The name/value pairs of are described below:

### CDE\_URL

Represents the url of cde. This php API is hit from CallCDEUrlBatch thread function. This API reply the corresponding message back in the application which is stored in SMSOUTBOX\_QUE and used later in updating the smsoutbox table.

### NO-REPLY\_STRING

Represents the predefined reply message for no valid reply from the CDE\_URL hit-back. The values must be in comma separated form.

## Database related information

The database configuration information consists of following items:

DATABASE\_HOST->localhost

DATABASE\_USERNAME->root

DATABASE\_PASSWROD->Nopass!234

DATABASE\_NAME->smsgw

TPS->500

The name/value pairs of are described below:

### DATABASE\_HOST

This parameter represents the IP address of the hosting server.

### DATABASE\_USER\_NAME

It represents the user name of the server user from which the database is operated.

### DATABASE\_PASSWORD

It represents the password of the database.

### DATABASE\_USER\_NAME

It specifies the name of the database.

MakeQueOrFail thread enable configuration

This consists of the following values:

IS\_REQUE\_ENABLE->0

RETRYCOUNT->0

RETRY\_DELAY->10 SECOND

### IS\_REQUE\_ENABLE

This parameter provides the option of re-queueing the message status in smsinbox table which have been FAILED. Following values are possible:

* 1 – Enable : This value activates the RetryForFailedSMS thread function.
* 0 – Disable : This value hinders the application to start the above mentioned thread.

**Default value: 1**

### REGION\_CHECK\_ENABLE

Flag to indicate if region will be checked against **region** column of **ratemaster** table. **regionurl** table will be used to generate expected url.

Format: <hostname>:<port>/url

MSISDN and ServiceId will be appended by CGW.

Following values are possible:

* 1 – Enable
* 0 – Disable

**Default value: 0**

### CALL\_TYPE\_ID\_CHECK

Flag to indicate if **CallTypeId** column of table **calltype** value will be checked of not. Following values are possible:

* 1 – Enable
* 0 – Disable

**Default value: 0**

### DIRECT\_DIALING\_CHECK

Flag to indicate if direct dialing feature is enabled at Call Handler end. If enabled, requested bno (See section to know about CGW request) will be matched against all short codes of subscriptiongroup table, returned that matched short code and treated as bno for further processing. Reason for this is that, Subscriber’s pressed options will be added with bno and if **REDIS server is used** for wallet information, then short code will be matched exactly with requested bno. Following values are possible:

* 1 – Enable
* 0 – Disable

**Default value: 0**

### PACKAGE\_RANGE\_CHECK\_ENABLE

If enabled, both **StartMSISDN** and **EndMSISDN** will be used while searching into **package/specificpackage** table using requested MSISDN, otherwise only **StartMSISDN** will be used. Following values are possible:

* 1 – Enable
* 0 – Disable

**Default value: 0**

### PACKAGE\_LOADING\_AFTER

If LOAD\_PACKAGE\_TABLE is 1, then this field will try to reload package table next day after passing PACKAGE\_LOADING\_AFTER **minutes** from 12:00:00 AM.

If LOAD\_PACKAGE\_TABLE is 0, it will just re-set current package table prefix into application memory from REDIS server.

### DefaultPackageID

This value will be used if no **PackageID** is found in **package/specificpackage** table.

Example: DefaultPackageID = Prepaid

### MinPulseSize

Minimum allowed pulse size. This value will be used with steppulse column of ratepulse table to find desired pulse size. ***MinPulseSize value should be greater than 0.***

**Default value: 1**

### DefaultTimeSlotId

This value will be used if no **TimeSlotID** is set in **timeslot** table for requested application id.

**Default value: AnyTime**

### DefaultCallTypeId

This value will be used if no **CallTypeID** is set in **calltype** table for requested ano and bno.

**Default value: AnyCall**

### DefaultTPS

This value will be used if no tps limit is set in **tps** table for requested application id and timeslot id.

**Default value: 1000**

## Persistent Database

These are used for MYSQL database configuration. Consists of the following values.

HOST = localhost

PORT = 3306

USER = root

PASSWORD = nopass

DB = cgw\_3\_0\_8

MAX\_CONN = 100

MAX\_RETRY = 3

### HOST

Server IP where CGW database has been deployed.

### PORT

Database listening port.

### USER

Database user name.

### PASSWORD

Database user’s password.

### DB

CGW database name. Current database name: **cgw\_3\_0\_8.**

### MAX\_CONN

Maximum number of active Database connections.

### MAX\_RETRY

Maximum number of retry before a query execution will be treated as failed.

## In Memory Database

Configuration values for in memory database(REDIS) information.

MEM\_HOST = localhost

MEM\_PORT = 6379

MEM\_PASSWORD = foobared

MEM\_DB = 0

MEM\_MAX\_CONN = 100

MEM\_MAX\_RETRY = 3

### REDIS\_HOST

Server IP where REDIS server has been installed.

### REDIS\_PORT

REDIS server listening port.

### REDIS\_PASSWORD

REDIS server’s password.

### REDIS\_DB

REDIS server DB index.

### REDIS\_MAX\_CONN

Maximum number of active Database connections.

### REDIS\_MAX\_RETRY

Maximum number of retry before a query execution will be treated as failed.

## Billing Interface Configuration

Billing interface configuration consists of the following values:

IN\_INTEGRATED = 1

DEFAULT\_BILLING\_HOST = localhost

DEFAULT\_BILLING\_PORT = 3650

PACKAGE\_HOST\_IP = 192.168.246.252

PACKAGE\_HOST\_PORT = 3650

PACKAGE\_CHECK\_ENABLED\_SESSION = 0

PACKAGE\_CHECK\_ENABLED\_SPECIFIC = 0

PACKAGE\_CHECK\_ENABLED\_WITH\_BALANCE = 0

NUMBER\_OF\_RETRY\_TO\_FIND\_PACKAGE = 3

SHOULD\_USE\_REMARKS = 0

RECV\_TIMEOUT = 5000

All the formats are explained below:

### IN\_INTEGRATED

Flag to indicate if IN Requester is integrated with CGW or not. Following values are possible:

* 1 – Enable
* 0 – Disable

***This value largely depends on CGW building process. If no IN Requester is being integrated at building time, then setting 1 will not work. Consult the CGW team if you need IN Requester integrated CGW.***

### DEFAULT\_BILLING\_HOST

Default billing host if no billing node is found in **billingnode** table.

* Applicable only if IN\_INTEGRATED is **0** and **NULL/Empty** value in **BillingHost** column (DB v3.0.6/v3.0.7) or **URL** (DB v3.0.4) of **billingnode** table.

### DEFAULT\_BILLING\_PORT

Default billing port if not billing node is found in **billingnode** table.

* Applicable only if IN\_INTEGRATED is **0** and **NULL/Empty** value in **BillingHost** column (DB v3.0.6/v3.0.7) or **URL** (DB v3.0.4) of **billingnode** table.

### PACKAGE\_CHECK\_ENABLED\_SESSION

Flag to indicate if package information will be queried at Operator end for session-based charging.

Following values are possible:

* 1 – Enable
* 0 – Disable

### PACKAGE\_CHECK\_ENABLED\_SPECIFIC

Flag to indicate if package information will be queried at Operator end for specific charging.

Following values are possible:

* 1 – Enable
* 0 – Disable

### PACKAGE\_CHECK\_ENABLED\_WITH\_BALANCE

Flag to indicate if user’s current balance will be checked against deductible amount before sending transaction request to IN Requester.

Following values are possible:

* 1 – Enable
* 0 – Disable

### NUMBER\_OF\_RETRY\_TO\_FIND\_PACKAGE

How many times it will retry to find package. ***If integrated IN Requester is being used, then this value will be ignored.***

### PACKAGE\_HOST\_IP

Host address to query about subscriber’s balance and package id. ***If integrated IN Requester is being used, then this value will be ignored.***

### PACKAGE\_HOST\_PORT

Host port to query about subscriber’s balance and package id. ***If integrated IN Requester is being used, then this value will be ignored.***

### RECV\_TIMEOUT

Timeout value while waiting to receive data from Socket interface. ***If integrated IN Requester is being used, then this value will be ignored.***

### SHOULD\_USE\_REMARKS

This field is used to determine whether the remarks field (11th field of the incoming request to CGW) would be used by CGW to form the request string to IN Requester. Follow [section](#_CGW) for CGW’s request format and [section](#_IN_Requester) for billing interface’s request format. Following values are possible:

* 1 – Enable
* 0 – Disable

## Log Configuration

This configuration values are used by the application for logging purpose. Values are like:

LOG\_LEVEL = 4

LOG\_FOR\_ANY\_MSISDN = 0

LOG\_INSTANCE = CGW1

LOG\_MSISDN = 1823208399, 1922113354, 1944113354, 1966113354, 1988113354, 1959771233, 1913740177, 1933113354, 1977113354, 1999113354

LOG\_DIRECTORY = E:/LogDirectory/cgw/logs

LOG\_FILE\_CREATION\_INTERVAL = 600

LOG\_FILE\_LINE\_SEPARATOR = [13, 10]

Description of these values are given below:

### LOG\_LEVEL

Determines which level(s) of log will go to the destination. Followings are possible:

* 0 – OFF
* 1 – FATAL
* 2 – ERROR
* 3 – WARN
* 4 – INFO
* 5 – DEBUG
* 6 – ALL
* Other numeric value – DEBUG

### LOG\_FOR\_ANY\_MSISDN

Flag to indicate if transaction log is enabled for all MSISDNs. Followings are possible value:

* 1 – Enabled
* 0 – Disabled

***If enabled, LOG\_MSISDN values will be ignored***

### LOG\_INSTANCE

Descriptive value that will be used while writing specific log for LOG\_MSISDN.

**Default value: CGW1**

### LOG\_MSISDN

Separated formatted log will be generated for the specified MSISDNs. **Every MSISDN will be separated by Comma (,). There is no limit about how many MSISDN can be used.**

Example: **LOG\_MSISDN = 1830677123,1847184823,1858980299,1844170396,1844075152,1850666790**

### LOG\_DIRECTORY

Directory where log is generated only for LOG\_MSISDN. ***LOG\_DIRECTORY must be created and must have write permission before running CGW application.*** Log file naming format:



New log file will have generated after every 10 minutes for a specific MSISDN.

### LOG\_FILE\_CREATION\_INTERVAL

Time interval between two log file creation. **Time unit: SECOND**

### LOG\_FILE\_LINE\_SEPARATOR

Individual log line separator.

**Default value: \r\n**

## CDR Configuration

CDR related configuration are added here. Sample key-values are:

NUMBER\_OF\_CDR\_PROCESSING\_THREADS = 1

CDR\_WRITE\_IN\_FILE\_ENABLE = 1

IMMEDIATELY\_SAVE\_SUCCESS\_CDR = 1

CDR\_DIRECTORY = E:/LogDirectory/cgw/cdr

CDR\_FILE\_CREATION\_INTERVAL = 10

CDR\_DUPLICATION\_CHECK\_ENABLE = 1

CDR\_DUPLICATION\_LAST\_RESPONSE = 0

INSERT\_INTO\_CRD\_FOR\_TPS = 1

INSERT\_INTO\_CRD\_FOR\_ERROR\_IN\_APPID\_OR\_PASSWORD = 1

CDR\_FIELD\_SEPARATOR = |

CDR\_LINE\_SEPARATOR = \r\n

CDR\_BATCH\_SIZE = 10

Description of these values are given below:

### NUMBER\_OF\_CDR\_PROCESSING\_THREADS

Total number of threads for CDR batch processing.

Note that, **setting 0 (Zero) will disable CDR batch processing. Value greater than 0 (Zero) will store generated CDR into IN Memory DB**.

### CDR\_WRITE\_IN\_FILE\_ENABLE

If it is 1 then CDR will be written in File system instead of Database table. If 0, then CDR will be stored in **cdrintegrated** table. Following values are possible:

* 1 – Enable
* 0 – Disable

### IMMEDIATELY\_SAVE\_SUCCESS\_CDR

Flag to indicate if Success CDR will be immediately written into DB/File (Depends on **CDR\_WRITE\_IN\_FILE\_ENABLE** property value**)** instead of storing into IN Memory DB**.**

Note that, if **NUMBER\_OF\_CDR\_PROCESSING\_THREADS** contains **0 (Zero)**, then it’s value will be **ignored**.

### CDR\_DIRECTORY

Directory path where the CDR record files will be stored. ***CDR\_DIRECTORY must be created and must have write permission before running CGW application***

### CDR\_FILE\_CREATION\_INTERVAL

It will act as

1. Time interval between two CDR file creation if **CDR\_WRITE\_IN\_FILE\_ENABLE** is **1.**
2. Sleep time between CDR batch processing if **NUMBER\_OF\_CDR\_PROCESSING\_THREADS** is greaterthan **0 (Zero)** and **CDR\_WRITE\_IN\_FILE\_ENABLE** is **0.**

**Time unit: Second**

### CDR\_DUPLICATION\_CHECK\_ENABLE

If set to 1 then CDR duplicate check will be done by CGW application. Otherwise if set to 0 then no duplicate checking will be done.

### CDR\_DUPLICATION\_LAST\_RESPONSE

If CDR\_DUPLICATION\_CHECK\_ENABLE=1 and CDR\_DUPLICATION\_LAST\_RESPONSE=1 then last response will be send for this duplicate CDR.

### INSERT\_INTO\_CRD\_FOR\_TPS

If enabled, then CDR will be generated if TPS is exceed. Following values are possible:

* 1 – Enable
* 0 – Disable

### INSERT\_INTO\_CRD\_FOR\_ERROR\_IN\_APPID\_OR\_PASSWORD

If enabled, then CDR will be generated if authentication failed. Following values are possible:

* 1 – Enable
* 0 – Disable

### CDR\_FIELD\_SEPARATOR

Field separator.

**Default value: |**

### CDR\_LINE\_SEPARATOR

Individual CDR separator.

**Default value: \r\n**

### CDR\_BATCH\_SIZE

If NUMBER\_OF\_CDR\_PROCESSING\_THREADS >= 0 and CDR\_WRITE\_IN\_FILE\_ENABLE = 0, then at a time this number of CDR will be stored into database. Apart from this case, it’s value will be ignored.

**Default value: 10**

## MySQL-Redis SYNC application (RDSS) configuration

Sample values are given below.

SYNC\_APP\_QUEUE\_NAME = rdss:queue

SUBSCRIPTION\_INFO\_SEPARATOR = \|

SUBSCRIPTION\_INFO\_KEY\_PREFIX = 0$

SUBSCRIPTION\_INFO\_KEY\_SEPARATOR = :

WALLET\_INFO\_VALUE\_SEPARATOR = \|

WALLET\_INFO\_KEY\_PREFIX = 1$

WALLET\_INFO\_KEY\_SEPARATOR = :

SECONDARY\_WALLET\_INFO\_KEY\_PREFIX = 2$

SECONDARY\_WALLET\_INFO\_KEY\_SEPARATOR = :

Description of the values are below:

### SYNC\_APP\_QUEUE\_NAME

The name of the RDSS queue where the MySQL queries are passed for execution. Please match this with the corresponding value of RDSS app **config.json** file.

### SUBSCRIPTION\_INFO\_SEPARATOR

**Default value: ‘\\|’**

### SUBSCRIPTION\_INFO\_KEY\_PREFIX

This is the table prefix of **subscriberservices** table of CGW database. Match this with RDSS **config.json** and append an additional $ sign after it.

### SUBSCRIPTION\_INFO\_KEY\_SEPARATOR

**Default value: ‘:’**

### WALLET\_INFO\_VALUE\_SEPARATOR

**Default value: ‘\\|’**

### WALLET\_INFO\_KEY\_PREFIX

The prefix of wallet table. Match this value with RDSS **config.json** like SUBSCRIPTION\_INFO\_KEY\_PREFIX.

### WALLET\_INFO\_KEY\_SEPARATOR

**Default value: ‘:’**

### SECONDARY\_WALLET\_INFO\_KEY\_PREFIX

This is the prefix of another format of the wallet table. Also match this value with RDSS config file.

### SECONDARY\_WALLET\_INFO\_KEY\_SEPARATOR

**Default value: ‘:’**

## Log Server configuration

They are used for showing statistics of the application in a remote log server. CGW application sends its status data to that server in a certain interval. The values look like below:

LOG\_SERVER\_ENABLE = 0

LOG\_SERVER\_IP =

LOG\_SERVER\_PORT = 0

LOG\_SERVER\_CONNECT\_TIMEOUT = 1000

LOG\_SERVER\_RECONNECT\_INTERVAL = 1000

LOG\_SERVER\_CONCURRENT\_SENDER = 0

### LOG\_SERVER\_MAX\_CONTEXTS = 2000

### LOG\_SERVER\_ENABLE

Flag to indicate if log monitoring is enabled or not. Following values are possible:

* 1 – Enable
* 0 – Disable

### LOG\_SERVER\_IP

Log server IP.

### LOG\_SERVER\_PORT

Log server port.

### LOG\_SERVER\_RECONNECT\_INTERVAL

Maximum number of milliseconds to wait before re-connecting to Log server.

**Default value: 1000**

### LOG\_SERVER\_CONCURRENT\_SENDER

Maximum number of threads that will be used to send state context to Log server.

**Default value: 0**

### LOG\_SERVER\_MAX\_CONTEXTS

Maximum number of state context that CGW will store. Beyond that new state context will be dropped.

**Default value: 2000**

# Start CGW Application

Execute following query in the command prompt/terminal to start CGW:



Current Major version: 1  
Current Minor version: 4

# Result codes

Below are the result codes that can be only generated by CGW. CGW also forwards IN Requester’s result codes. So, apart from below codes, any other codes will be treated as IN Requester’s codes.

|  |  |
| --- | --- |
| Result Code | Description |
| 2001 | Charging Success |
| 9004 | Wrong Request Format |
| 9005 | Wrong Response Format |
| 9010 | DB Query Failed |
| 9012 | Blank Amount Returned |
| 9016 | MSISDN Invalid Package |
| 9017 | Session Timeout |
| 9021 | TPS limit is exceed. |
| 9023 | ARI Server Disconnect |
| 9024 | ARI Server Send Invalid Response |
| 9025 | Duplicate CDR id is being detected. |
| 9028 | No rate id found in the current context. |
| 9029 | Wallet info parsing error while loading from session information. |
| 9030 | No session information found for requested CDR Id. |
| 9031 | No wallet configuration found in serviceinfo table. |
| 1012 | Authentication Failed |

# Result Code Description

### 2001

Charging is successful.

### 9004

Wrong formatted request has been sent to CGW interface. Follow [section](#_CGW) to know about the request format of CGW interface.

### 9005

Wrong formatted response has been returned from IN Requester’s Billing interface. Follow [section](#_IN_Requester_1) to know about the response format of IN Requester’s Billing interface.

### 9010

Database query execution failed after retrying [MAX\_RETRY](#_MAX_RETRY) times.

### 9012

There is no amount configuration found corresponding to the given rate id. Amount configuration is loaded from **ratepulse** table into CGW. Follow [section](#_Same_rate_information) to find out amount configuration.

### 9016

Package id “INVALID” is returned from Balance-package query interface. Currently only BanglaLink IN Requester returns this type of response.

### 9017

At session interim call, if wallet’s balance is finished then this result code will be generated.

### 9021

TPS [Transaction per second] will exceed the mentioned limit in **value** column of **tps** table for the mentioned application id and time slot id [**TimeSlotID** column of **timeslot** table].

### 9023

If IN Requester is not integrated, then socket communication will be used to connect balance-package query interface [Ex. ARI server]. If no socket connection can be established with ARI server, then this result code will be generated.

### 9024

Balance-package query interface’s response format is:

**<Package Id>|<Balance>|<Result code>**

If IN Requester return wrong formatted or insufficient response, then this result code will be generated.

### 9025

At successful transaction, a key-value is stored in REDIS server:  
***dcdr.<CDR id> -> “<CDR id> <Result code> <Charging type> <Sequence no> <PulseSize>”***

Now, if another transaction will use the same CDR id, then this result code will occur.

Note that, **a successful CDR id will be kept for 24 hours.**

### 9028

No rate id found in the current context.

Context is described below:

#### First phase

* When CGW loads ratemaster table into application space, Current time is between ActivationStart and ActivationEnd columns.
* Service Id which is provided into CGW request should be matched with ServiceID column of ratemaster table.
* If subscription group id is provided into CGW request [It’s an optional parameter], then it will be matched with SubscriptionGroupID column of ratemaster table.
* Charging Type which is provided into CGW request should be matched with ChargingType column of ratemaster table.
* Package Id [Follow [section](#_Package_finding_steps) to know how to find package id] should be matched with PackageID column of ratemaster table.
* Time slot id [Follow **timeslot** table to find TimeSlotId] should be matched with TimeSlotID column of ratemaster table.
* Call Type Id [Follow **calltype** table to find CallTypeID] should be matched with CallTypeID column of ratemaster table.

#### Second phase

* Query subscriberservices and wallet table to find current state of the subscriber.

Possible states:

* na -> 0
* deregistered -> 1
* registered -> 2
* registered no balance -> 3
* not registered balance -> 4
* ingraceperiod -> 5

This state will be matched with SubscriptionStatus column of ratemaster table. Note that, if SubscriptionStatus column contains ‘na’, then Subscriber’s current state will be ignored. Also, in this phase, query exception can be occurred. If this happens, then Subscriber’s current state will be unknown and rate id cannot be determined.

#### Third phase

* If subscription group id is not provided, then in [Second phase](#_Second_phase), subscription group id will be retrieved. It should be matched with SubscriptionGroupID column of ratemaster table. Note that, if SubscriptionGroupID columns contains ‘na’, then retrieved SubscriptionGroupID will be ignored.
* RegisteredServices column of ratemaster table should be among subscriber’s current registered [Registered, InGracePeriod, RenewalFailed included] subscription group ids. Note that, if RegisteredServices columns contains ‘na’, then this checking will be ignored.
* If channel is provided/retrieved from subscriberservices table, then it will be matched with channel column of ratemaster table.

If any step is unmatched, then this result code will be generated.

### 9029

In session-based charging, a serialized session information is mapped against given CDR id in REDIS server. The serialization and deserialization are totally internal to CGW. So, if this result code occurs, consult with Developer.

### 9030

This result code will only generate for session-based charging requests. At session initiation, a serialized session information is mapped against given CDR id in REDIS server. Subsequent interim and stop requests will must need to use previously given CDR id at session initiation, otherwise this code will be given.

### 9031

No wallet configuration found in **serviceinfo** table.

### 1012

Either application id or application password is not matched with ***applicationid*** and ***passwd*** columns of ***applicationinfo*** table.

# Request format

## CGW

appid=<Application Id>&apppass=<Application password>&cmdid=<Charging type>&cmdparam=<ANO>|<BNO>|<Direction>|<Service Id>|<CDR Id>|<Sequence No>|<Consumed unit since last request>|<Remarks (Optional)>|<Subscription Group Id (Optional)>|<Channel (Optional)>

### Parameter description

|  |  |
| --- | --- |
| Parameter name | Description |
| Application Id | Authentication parameter. Given value should be matched with **applicationid** column of **applicationinfo** table. Case sensitive. |
| Application password | Authentication parameter. Given value should be matched with **passwd** column of **applicationinfo** table [Equality check will be done only if **requirepass** is **TRUE**]. Case sensitive. |
| Charging type | Indicate what type of transaction has been requested. Following charging types are possible:   * SPECIFIC\_CHARGE * SESSION\_START * SESSION\_INTERIM * SESSION\_STOP   Note that, following mapping is used for Call handler:   * AUTH\_CALL -> SESSION\_START * REAUTH\_CALL -> SESSION\_INTERIM * END\_CALL -> SESSION\_STOP |
| ANO | Calling number/Subscriber’s mobile number. |
| BNO | Called number/Short code. Use ‘na’ if no BNO is needed. |
| Direction | It indicates between ANO and BNO which will be used to charge.  Two values are possible:  0 ----> ANO <- ANO, BNO <- BNO  1 ----> ANO <- BNO, BNO <- ANO |
| Service Id | **ServiceID** column of **ratemaster** table. Case insensitive. |
| CDR Id | Numeric or Alphanumeric id. Maximum length is 50. |
| Sequence No | For specific charge request, it will always be 0.  For session-based charge request, it will start from 0 and for every request within same session it’s value should be incremented by 1. |
| Consumed unit since last request | For specific charge request, it will always be 0.  For session-based charge request, it will indicate how many pulse have been allocated for previous charging request.  For SESSION\_STOP request, this value will be used to refund balance. |
| Remarks (Optional) | **Remarks** column of **subscriptiongroup** table. Case insensitive. |
| Subscription Group Id (Optional) | **SubscriptionGroupID** column of **subscriptiongroup** table. Case insensitive. |
| Channel (Optional) | It indicates which source transaction request has been came from. Case insensitive. |

### Specific charge request

appid=test&apppass=test&cmdid=SPECIFIC\_CHARGE&cmdparam=1985996717|na|0|test|1234567891234567|0|0|na|test|na

### Session based request

* Initiation

appid=test&apppass=test&cmdid=SESSION\_START&cmdparam=1985996717|na|0|test|1234567891234567|0|0|

* Continuation

appid=test&apppass=test&cmdid=SESSION\_INTERIM&cmdparam=1985996717|na|0|test|1234567891234567|0|60|

.

.

.

* Stop

appid=test&apppass=test&cmdid=SESSION\_STOP&cmdparam=1985996717|na|0|test|1234567891234567|0|39|

## IN Requester

### Application interface

**CDRID[ServiceID]|sequenceno|requesttype**|**ano**|**bno**|**direction**|**rateid**|**amount**|**consumedunitsincelastrequest|requestedUnit**

**Example**

19645016012015120930[MusicDaily]|0|SPECIFIC\_CHARGE|88001716033678|555|0|Music\_Daily\_subscription|100|0|180

If REFUND\_SESSION is enable the request string is like

**CDRID[ServiceID]|sequenceno|requesttype**|**ano**|**bno**|**direction**|**rateid**|**amount**|**consumedunitsincelastrequest|requestedUnit|refundAmount**

**Example**

19645016012015123533[MusicDaily]|0|SPECIFIC\_CHARGE|88001716033678|555|0|Music\_Daily\_subscription|100|0|180|0

If remarks are enabled, then remarks and applicationId separated by a space are also appended in the end.

**CDRID[ServiceID]|sequenceno|requesttype**|**ano**|**bno**|**direction**|**rateid**|**amount**|**consumedunitsincelastrequest|requestedUnit|remarks applicationId**

**Example**

701116042014110507[LSBNYB]|0|SESSION\_START|8801778412765|2666|0|Info\_100paisapermin\_Registered|100|0|60|bl\_ren\_mwmw vsdp

### Web service interface

**chargingid**=%s[%s]&**sequenceno**=%d&**requesttype**=%s&**ano**=%s&**bno**=%s&**direction**=%d&**rateid**=%s&**amount**=%d&**consumedunitsincelastrequest** =%d&**pulseSize**=%d

**Example**

chargingid=701116042014110507[LSBNYB]&sequenceno=0&requestType=SESSION\_START&ano=8801778412765&bno=2666&direction=0&rateid=Info\_100paisapermin\_Registered&amount=100&consumedunitsincelastrequest=0&pulseSize=60

If remarks are enabled, then remarks and applicationId are also appended in the end.

**chargingid**=%s[%s]&**sequenceno**=%d&**requesttype**=%s&**ano**=%s&**bno**=%s&**direction**=%d&**rateid**=%s&**amount**=%d&**consumedunitsincelastrequest** =%d&**pulseSize**=%d&**purpose**=%s+%s

**Example**

chargingid=701116042014110507[LSBNYB]&sequenceno=0&requestType=SESSION\_START&ano=8801778412765&bno=2666&direction=0&rateid=Info\_100paisapermin\_Registered&amount=100&consumedunitsincelastrequest=0&pulseSize=60&purpose=bl\_ren\_mwmw+vsdp

### Parameter description

|  |  |
| --- | --- |
| Parameter name | Description |
| CDRID | Unique ID for each request. It will be same for successive requests in session-based charging. |
| ServiceID | Denotes the service Id of the service. It is taken from the **ServiceId** field of **RateMaster** table. |
| SequenceNo | Sequence number will start from 0 and will increase for each interim request in session-based charging. It will be an integer. |
| RequestType | It will be one of SESSION\_START, SESSION \_INTERIM, SESSION \_STOP, SPECIFIC\_ CHARGE, SPECIFIC\_RE CHARGE. Same as CmdId. |
| ANO | calling party number |
| BNO | called party number |
| Direction | charging direction of the call. If Direction is 0, ANO will be charged and if Direction is 1, BNO will be charged. |
| RateID | RateID for specific anounts |
| Amount | Amounts in 1/100 of paisa. 100 will mean 1 paisa. |
| ConsumedUnitSinceLastRequest | It is the consumed unit since the last interim charging request in session-based charging. It will be an integer. |
| RequestedUnit | It is the requested unit for charging. It is taken from the pulseSize column of the ratePulse table. In case of session-based charging, it is actually the time duration of the session in seconds. For specific charging, it is always 60. |
| Remarks(Optional) | Additional information for billing purpose needed by the operators. It is taken from the Remarks field of the SubscriptionGroup table if SHOULD\_USE\_REMARKS is set to 1 in the charging gateway config.ini file. |
| ApplicationId(Optional) | Application Id of the charging gateway taken from the configuration database. |
| refundAmount(Optional) | If REFUND\_SESSION is enable, then refund is at the end of the request |

# Response format

## CGW

### Format of response

<TransactionID><SPACE><Resultcode><SPACE><SequenceNumber><SPACE><CommandId><SPACE><ReservedUnit><SPACE><SubscriptionStatus(Optional)>

\*\*“<>” sign is used here to indicate the separation of fields. In the actual response string, the fields are separated only by a space.

**Example**

701116042014110507 2001 0 2 20

### Parameter description

|  |  |
| --- | --- |
| Parameter name | Description |
| TransactionID | CDR ID |
| Resultcode | Result code will be generated from two sources –   * CGW itself * IN Requester   If result code is 2001, the charging is successful. Follow [section](#_Result_code_description) for all allowable result code that can be generated from CGW itself. |
| Sequence Number | Sequence number sent by the requesting application. |
| Command Id | Apart from below mappings, same command Id sent by the requesting application will be used.   * AUTH\_CALL -> SESSION\_START * REAUTH\_CALL -> SESSION\_INTERIM * END\_CALL -> SESSION\_STOP * CHARGE\_SPECIFIC\_AMOUNT -> SPECIFIC\_CHARGE |
| ReservedUnit | Determines the unit reserved from the IN for session-based charging. |
| Subscription Status (Optional) | If REGISTRATION\_OUTPUT\_ENABLED is set to 1, then this field presents the subscription status. For, specific charging it’s always 0. For session based charging its values depend on the users’ registration status. If the user is registered, then it is 2 and if the user is deregistered then it is 1. |

## IN Requester

<**TransactionId**[**ServiceType**]> <**ResultCode**> <SequenceNumber> < CommandId> < **ReservedUnit**>

\*\*“<>” sign is used here to indicate the separation of fields. In the actual response string, the fields are separated only by a space.

**Example**

19645008012015111501[MusicDaily] 2001 0 1 120

Or

19645008012015111501[MusicDaily] 2001 0 SPECIFIC\_CHARGE 120

|  |  |
| --- | --- |
| Parameter | Description |
| TransactionID | CDR ID |
| ServiceType | Denotes the service Id of the service. It is taken from the ServiceId field of RateMaster table. |
| Resultcode | If result code is 2001, the charging is successful. |
| Sequence Number | Sequence number sent by the requesting application |
| CommandId | cmdid or value for that cmdid. For SESSION\_START=2, SESSION\_INTERIM=3,SESSION\_STOP=4, SPECIFIC\_CHARGE=1,REFUND=5,BALANCE=6. |
| ReservedUnit | Determines the reservation unit for session-based charging. |

# Command Server

A formatted command can be sent to COMMAND\_LISTENTING\_PORT for administrative purposes. Following commands are allowed –

|  |  |
| --- | --- |
| Command | Description |
| [reload]|[tables]|[table(s) name] | This command will reload table(s) that is/are already loaded by CGW at starting time. Allowable tables:   * applicationinfo * tps * billingnode * timeslot * ratemaster * ratepulse * subscriptiongroup * calltype |
| [reload]|[|[conf]|[in] | This command will reload IN Requester’s configuration file. ***Note that, this command only applicable if IN Requester is integrated with CGW.*** |
| [show]|[data]|[ratecontext|(Service Id)|(SubscriptionGroupId)|(ChargingType)|(PackageId)] | This command will show rate context for given values. |
| [show]|[data]|[rateid|(Rate Id)] | This command will show rate information for given rateid (rateid column of ratepulse table). |
| [1] | For graceful shutdown purpose. |

### Sample table reloading

#### Single table reloading

reload|tables|applicationinfo

Loading table [applicationinfo]. OK.

#### Multiple table reloading

reload|tables|tps|ratemaster|calltype

Loading table [tps]. OK.Loading table [ratemaster]. OK.Loading table [calltype]. OK.

Note that,

* reload|tables|subscriptiongroup command will internally load three more tables:
* serviceinfo
* regionurl
* servicepayer

So, if any data has been changed in any of these tables, then reload|tables|subscriptiongroup command will need to be executed.

* If subscriptiongroup, ratemaster and ratepulse tables mentioned in the same command, then for consistency reason, these tables need to be followed the below order:

subscriptiongroup -> ratemaster -> ratepulse

reload|tables|subscriptiongroup|ratemaster|ratepulse

#### Wrong table reloading

reload|tables|subscriberservices

Loading table [subscriberservices]. Error.

### Sample configuration loading

* IN Requester Integrated

reload|conf|in

Configuration loaded.

* IN Requester not integrated

reload|conf|in

Dummy BI contains no loadable configuration.

### Sample rate context



### Sample rate information



### Sample graceful shutdown process



This action will activate graceful shutdown procedure –

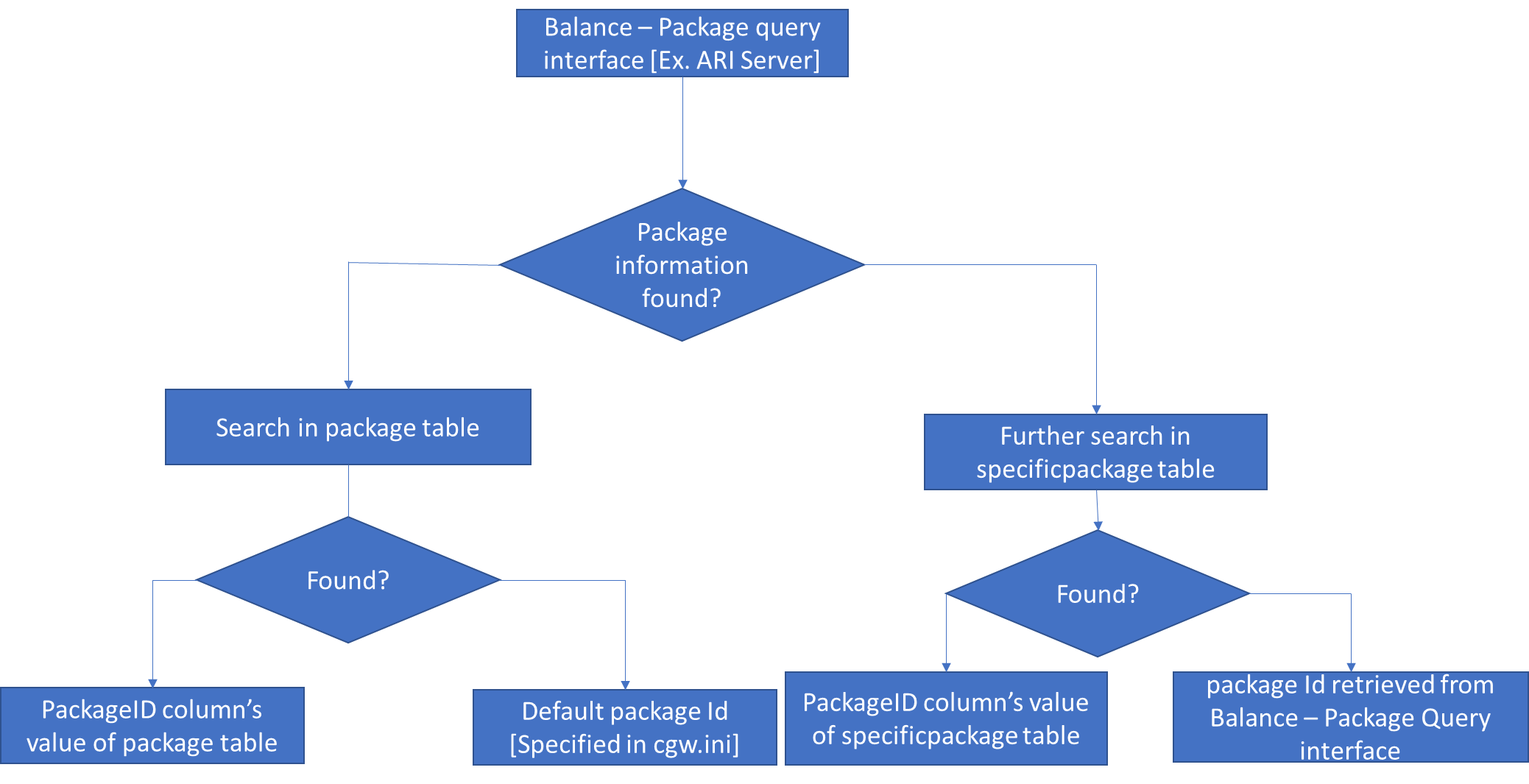
* Stop listening server.
* Finish all pending transactions.
* Stop package loading thread.
* CDR batch processing thread will store pending CDRs into DB/File.
* Stop Log monitoring server.
* Cleaning all resources (DB connections, Http connections etc.).

### Notes on CDR batch processing

If CDR batch processing is enabled, then gracefully shutdown is a **MUST**. If not, then at most **NUMBER\_OF\_CDR\_PROCESSING\_THREADS x 1** secondsCDRs will be gone. So, reducing **NUMBER\_OF\_CDR\_PROCESSING\_THREADS** value will reduce the catastrophe, but you have make sure that **NUMBER\_OF\_CDR\_PROCESSING\_THREADS** value isenough to process generated CDRs.You can query into REDIS about pending CDRs **-**

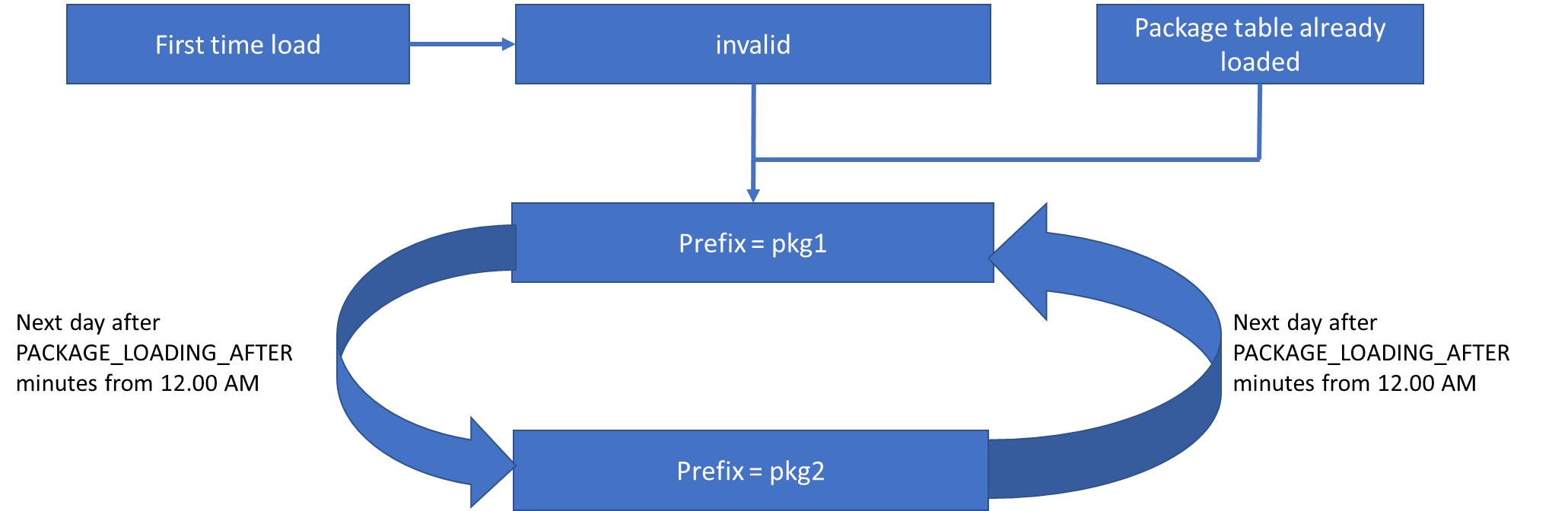


# Package finding steps



Follow [section](#_Package_information_finding) to know how to retrieve from REDIS server.

# Package loading process

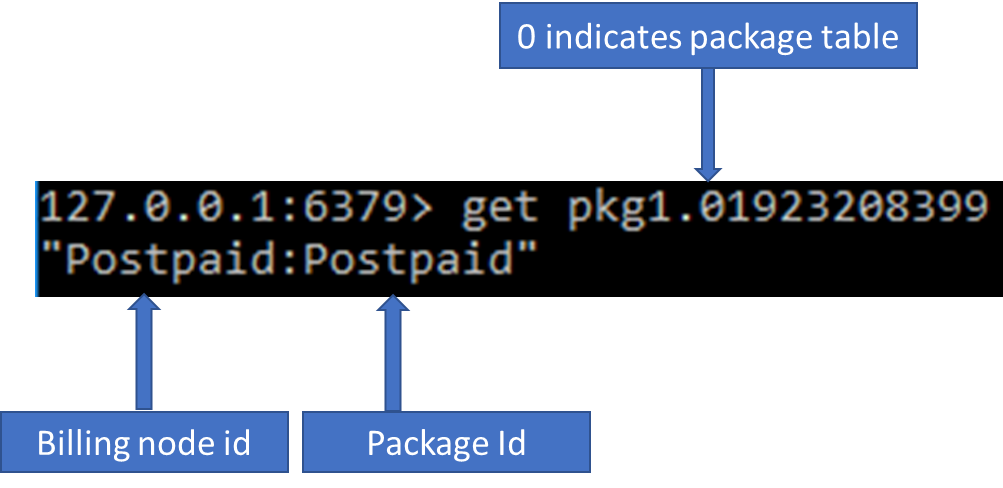


# Package information finding from REDIS server

**pkg\_tbl\_prefix** key in **Redis** DB will hold the current package table prefix.



Retrieving data from package table:



Retrieving data from specificpackage table:

