Indian National MOTECH System Mobile Kunji, Academy and Kilkari Services Architecture and Design

Location

Google Drive\National Scale Up - IVR + MOTECH\Aricent\Deliverables - MOTECH\Design

Revision History

Version	Description of changes	Changed by	Date
D0.01	First draft of the document	Aricent	01-Feb-2015
V0.1	Design of Sprint 1502 User Stories	Aricent	11-Feb-2015
V0.2	Updated as per review comments	Aricent	12-Feb-2015
V0.3	Updated as per review comments from Koshal	Aricent	18-Feb-2015
V1.0	Updated as per review comments from Rob. Final version for Sprint 1502	Aricent	02-Mar-2015
V1.01	Added sections to capture design for Sprint 1503 User Stories	Aricent	03-Mar-2015
V1.02	Updated for Sprint 1503 User Stories	Aricent	11-Mar-2015
V1.03	Added more details in sections 3 – 9. Minor updates in other sections.	Aricent	25-Mar-2015
V1.04	Fixed Chapter Numbering	Aricent	26-Mar-2015
V1.05	Incorporated Rob's comments	Aricent	31-Mar-2015
V2.0	Final version for Sprint 1503	Aricent	02-Apr-2015
V2.01	Updated for Sprint 1504 User Stories	Aricent	11-Apr-2015

Table of Contents

Indian National MOTECH System 1
Architecture and Design 1
Location 2
Google Drive\National Scale Up - IVR + MOTECH\Aricent\Deliverables - MOTECH\Design 2
Revision History 2
Table of Contents 3
1 Introduction 10
1.1 Overview 10
1.2 Purpose 11
1.3 Scope 11
1.4 Definitions, Acronym List, Abbreviations 12
1.5 References 13
[1] NMS System Requirement Specification 13
[2] NMS MOTECH-IVR Interface Specification 13
[3] MOTECH Platform Documentation: http://docs.MOTECHproject.org/en/latest/ 13
1.6 Assumptions 14
1.7 Open Issues 15
2 Architecture Overview 16
2.1 Interface Layer 17
2.2 Business Layer 18
2.3 Base Layer – MOTECH Platform components 20

2.4 Data Layer 21
3 Mobile Academy Module 23
3.1 Overview 23
3.2 Events Consumed 23
3.2.1 Course Content CSV Upload Success 23
3.3 Service Provided to IVR 26
3.3.1 Get User Details 27
3.3.2 Set Language Location Code of an MA *Subscriber* 30
3.3.3 Get Course Structure 31
3.3.4 Get Course Version 31
3.3.5 Get Bookmark and Score 32
3.3.6 Save Bookmark and Score 33
3.3.7 Save Call Details 3433
3.3.8 SMS Delivery Notification 36
4 Mobile Kunji Module 37
4.1 Overview 37
4.2 Events Consumed 37
4.2.1 Content Data CSV Upload Success 37
4.3 Services Provided to IVR 38
4.3.1 Get User Details 38
Refer 8.3.1 for details. 42
4.3.2 Set Language Location Code 42
4.3.3 Save Call Details 42
5 Kilkari Module 44

5.1 Overview....44
5.2 Events Consumed....45
5.2.1 Successful Upload of Child MCTS data. 45
5.2.2 Successful Upload of Mother MCTS data. 46
5.2.3 Successful Upload of Content Metadata. 48
5.2.4 Scheduled Event for Deletion of Deactivated/Completed Kilkari Subscription Records 49
5.3 Service Provided to IVR....50
5.3.1 Get Subscriber Details....50
5.3.2 Create Subscription 52
5.3.3 Deactivate Subscription....54
5.4 Service Provided to Kilkari OBD Module 54
5.4.1 Deactivate Subscription....54
5.4.2 Get Content File Name 5554
5.4.3 Get scheduled Subscriptions for OBD....55
6 Kilkari Out-Bound Dialer Module....58
6.1 Overview....58
6.2 Events Consumed....59
6.2.1 Scheduled Event for preparation of OBD TargetFile 59
6.2.2 Scheduled Event for invoking OBD TargetFile notification API 62
6.3 Service Provided to IVR....6364
6.3.1 CDR File Notification 64
6.3.2 FileProcessedStatus Notification....64
6.4 Assumption for Corner Cases:....65

7 Master Location Data Module 67
7.1 Overview 67
7.2 Events Consumed 67
The details of the corresponding CSV files is given in 11.2.7.2 68
7.2.1 Event Handling 68
7.3 Services 69
7.3.1 Location Data Service 69
7.3.2 Language Location Code Service 72
8 Front Line Worker Module 74
8.1 Overview 74
8.2 Events Consumed 74
8.2.1 FLW CSV Upload Successful 74
8.2.2 FLW Whitelist CSV Upload Successful 76
8.3 Services 76
8.3.1 Process User Details 76
The FLW State is set to ANONYMOUS-FLW 8081
8.3.2 Update Language Location Code given the MSISDN 81
8.3.3 Validate Operator 81
9 Utility Module 82
9.1 Overview 82
9.2 Services 82
9.2.1 Bulk Upload Log Service 82
9.2.2 CSV Data Validation and Parsing 83
10 Database Design 85

10.1 MA, MK, Kilkari Service Database 85
10.1.1 Master Location Data 85
10.1.2 Location – Circle – Language Mapping Data 8887
10.1.3 FLW Data 9290
10.1.4 WhiteListUsers 9591
10.1.5 Mobile Academy Data 9892
10.1.6 Mobile Kunji Data 107101
10.1.7 Kilkari Data 111105
10.1.8 Utilities Data 129115
10.1.9 KilkariObd Data 134118
10.2 Reporting Database Design 140124
10.3 Transactional DB to Reporting DB ETL 141125
11 Operations & Maintenance 142126
11.1 Configuration Management 142126
11.1.1 System Configuration Parameters 142126
11.1.2 Service Configuration Parameters 144128
11.2 Data Upload 149133
11.2.1 Requirements 149133
11.2.2 Data Upload Design 150134
11.2.3 CSV Upload Status Summary 151135
11.2.4 Rejected CSV Records Log 151135
11.2.5 CSV Upload Data History Tracking and Auditing 152136
11.2.6 CSV Upload Transaction Management 152136
11.2.7 Data File Formats 153137

11.2.8 Data Validation 162146
11.3 Reporting 170154
11.4 Logging 170154
11.4.1 MA, MK, Kilkari Logs 170154
11.4.2 Log Levels 170154
11.5 Exception Handling 171155
11.5.1 Design Options 171155
11.6 Security 172156
11.6.1 Data Security 173157
11.6.2 Disk Encryption 173157
11.6.3 Access Security 173157
12 High Availability and Deployment 174158
12.1 Load Balancer Design 174158
12.2 Database High Availability 174158
12.3 Event Sub-System High Availability 177161
12.3.1 Startup 177161
12.3.2 Master failure 178162
12.4 Scheduler Sub-System High Availability 180164
12.5 Reporting System High Availability 180164
12.6 Physical / Deployment View 180164
12.7 Deployment Scripts 180164
12.7.1 NMS Web Tier SW Installation 180164
13 Appendix 181165
13.1 Mobile Academy Course Structure and Content CSV File Sample 181165

1 Introduction

1.1 Overview

National MOTECH System (**NMS**) is a system that shall deliver three maternal and child health IVR services, namely Mobile Kunji, Mobile Academy and Kilkari; at a pan India level via a Toll Free, centralized long-code.

- **Mobile Academy** service is an inbound IVR mobile training course on reproductive, maternal, newborn and child health (RMNCH) for Front Line Workers (FLWs), designed to expand their knowledge of life-saving preventative health and enhance their capacity to communicate and engage effectively with families. FLWs can access the course from any phone by dialing a toll free long code, and complete it at their convenience.
- **Mobile Kunji** service includes an IVR based mobile service and a printed deck of illustrated cards on a ring, which together communicate essential audio-visual information on pregnancy and newborn health. Each card carries a unique long code and a card number printed on it. User has to dial the long code and enter the card number to access the specific audio content. Mobile Kunji is designed for use during counseling sessions with families and seeks to build support for healthy practices within families and communities.
- **Kilkari** service is an IVR subscription service that delivers time-sensitive audio information about maternal and child health to the mobile phones of husbands, their pregnant wives, and mothers of young children for upto 72 weeks, linked to the woman's stage of pregnancy or and child's age. The service covers the critical time period – where the most deaths occur - from the 2nd trimester of pregnancy until the child is one year old.

IVR services shall be powered by an open-source platform called **MOTECH** (**Mobile Technology for Community Health**). The MOTECH platform has been developed by the Grameen Foundation, a not-for-profit organization headquartered in the United States. The MOTECH platform combines the integration capabilities of an Enterprise Service Bus (ESB) with a flexible open source application development framework.

1.2 Purpose

This document provides an architectural overview of the Mobile Academy, Mobile Kunji, and Kilkari services, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the decisions which have been made while designing the system.

1.3 Scope

Scope of this document is to capture the high level design of Mobile Academy, Mobile Kunji, and Kilkari services.

The document is being written in phases. Design of the User Stories contained in each Sprint shall be added incrementally to this document.

1.4 Definitions, Acronym List, Abbreviations

Abbreviation	Description
CDR	Call Detail Record
FLW	Front Line Worker
IVR	Interactive Voice Response system
KK	Kilkari
LLC	Language Location Code
MA	Mobile Academy
MCTS	Mother Child Tracking System
MDS	MOTECH Data Services
MK	Mobile Kunji
MOTECH	Mobile Technology for Community Healthcare
NMS	National MOTECH System
OBD	Out-Bound Dialer
SMS	Short Messaging Service

Formatted Table

Formatted Table

Formatted Table

Formatted Table

1.5 References

[1] NMS System Requirement Specification

[2] NMS MOTECH-IVR Interface Specification

[3] MOTECH Platform Documentation: http://docs.MOTECHproject.org/en/latest/

1.6 Assumptions

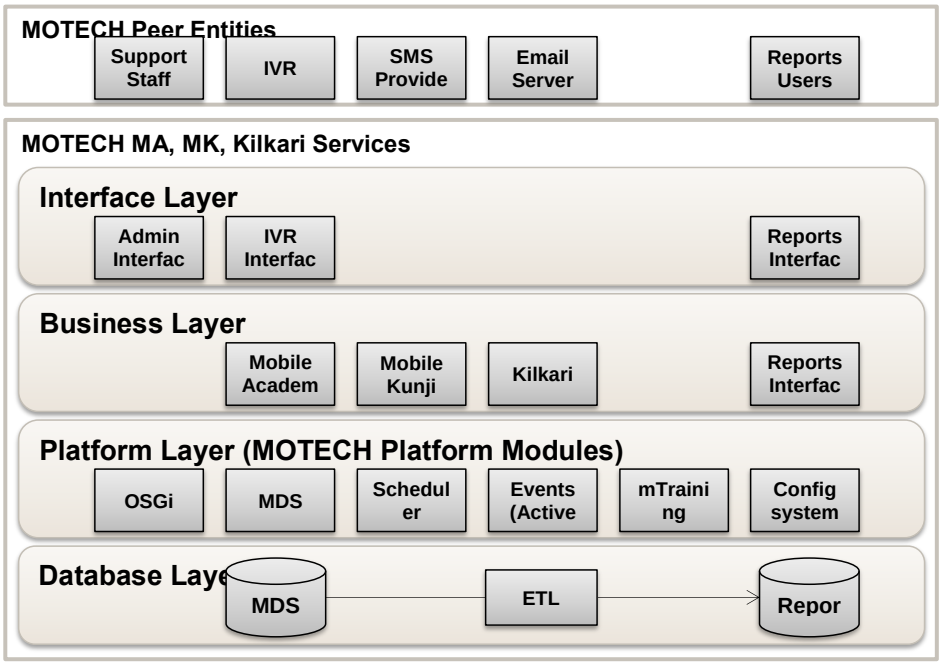
None

1.7 Open Issues

None

2 Architecture Overview

The following diagram presents the logical view of the NMS Mobile Academy, Mobile Kunji and Kilkari services software:



2.1 Interface Layer

MOTECH Mobile Kunji, Mobile Academy and Kilkari Service SW provides the user interfaces to perform various functionalities – e.g. upload master data, administers to configure, view and modify parameters, to define filters and generate different reports among others:

- **Data Upload Interface**: Provides an interface to the NMS administrators for uploading the following data among others into the NMS database:-
    - **Master location data**: the list of States, Districts, Taluka, Blocks, Health Facility (PHC/CHC), Sub-center & Village.
    - **FLW Data**: the registration details (MSISDN, Name, Designation & location data) of the FLWs.
    - **MCTS data**: the woman's Last Menstrual Period (LMP) or the child's data of birth (DOB).
    - **State-Language Mapping**: The mapping between states and languages.

  The Data Upload User Interface design is described in section 11.2.

- **Configuration Interface**: Provides an interface to administrators for viewing and modifying NMS configuration parameters that includes :
    - **System Parameters** (e.g. IP address, DB location, FTP/SFTP parameters)
    - **Business Logic parameters** related to NMS service (e.g. Capping rules, Language settings, etc.)

  The Configuration User Interface design is described in section 11.11~~11.110.1~~.

- **Reports Interface**: Pulls data from MOTECH to deliver online, real time reports on the take up and usage of MK, MA and Kilkari services.

  Reports Interface design is described in section 11.3.

- **IVR Interface**: MA, MK and Kilkari services provide an HTTP request-response based interface towards IVR.

2.2 Business Layer

The Business Layer consists of Mobile Academy, Mobile Kunji and Kilkari business logic for providing the maternal and child health IVR services:-

- **Mobile Academy** service is an inbound IVR mobile training course on reproductive, maternal, newborn and child health (RMNCH) for Front Line Workers (FLWs), designed to expand their knowledge of life-saving preventative health and enhance their capacity to communicate and engage effectively with families. FLWs can access the course from any phone by dialing a toll free long code, and complete it at their convenience.

  Mobile Academy design and use cases are described in section 3.

- **Mobile Kunji** service includes an IVR based mobile service using a long code and a printed deck of illustrated cards on a ring, which together communicate essential audio-visual information on pregnancy and newborn health. Each card carries a unique card number printed on it. User has to dial the Mobile Kunji long code and enter the card number to access the specific audio content. Mobile Kunji is designed for use during counseling

sessions with families and seeks to build support for healthy practices within families and communities.

Mobile Kunji design and use cases are described in section4.

- **Kilkari** service is an IVR subscription service that delivers time-sensitive audio information about maternal and child health to the mobile phones of husbands, their pregnant wives, and mothers of young children for upto 72 weeks, linked to the woman's stage of pregnancy or and child's age. The service covers the critical time period – where the most deaths occur - from the 2nd trimester of pregnancy until the child is one year old.

  Kilkari Service design is described in section 5 and 6

Additionally, some of the logic common to the services shall be implemented in the following modules:

- **Master Location Data**: This module shall provide APIs to MA, MK and Kilkari for retrieving location related information. Refer section 6 for details.

- **Front Line Worker**: This module shall provide APIs to MA, MK for retrieving and updating common FLW parameters. Refer section 8 for details.

- **Utility:** This module shall provide APIs to MA, MK and Kilkari for common tasks such as data validation and logging CSV upload status. Refer section 9 for details.

2.3 Base Layer – MOTECH Platform components

MA, MK, Kilkari services are built on top of Mobile Technology for Community Health (MOTECH) Platform.

MOTECH platform wraps several well-known open source systems, and augments and exposes their features to the other components. The main functions of the core are to wrap ActiveMQ (which provides the message queue) and present an internal pub/sub like event interface to the module and implementation layers. The core also provides a module loading environment (OSGi), an interface to the Scheduler, and access to the database.

MA, MK, Kilkari modules within MOTECH are self-contained bits of functionality that are loaded into the server via the OSGi host.

MOTECH is designed to be horizontally scalable with multiple MOTECHs all acting as workers connecting to the same message queue. A core design principle of the MOTECH platform is that the server should be stateless across requests to allow for horizontal scalability. It is expected that code running within the MOTECH server should perform a single action per request and then return. The module should never persist any state in memory or local disk and expect that state to be available to later requests.

MA, MK, Kilkari services shall use the following MOTECH Platform Modules:-

- **Events**: To aid in the development of stateless services, the MOTECH engine provides a pub/sub like event system. (The event system follows the publish-subscribe pattern but does not implement the standard Java pub/sub protocol.) It helps to decouple emitters of events from the modules that wish to consume them.

- **Scheduled Events & Timers**: To assist in the development of a stateless event-based server, the MOTECH platform provides access to a flexible scheduling system. Using the open source Quartz engine, MOTECH can easily schedule events for future consumption.

- **MOTECH Data Services (MDS)**: MOTECH Data Services is a flexible data modeling system that allows users to define and share custom schemas without code, and provides auditing and revision tracking. It is a JDBC-based user configurable database abstraction layer on top of a standard SQL database. It provides generated POJOs and OSGi service interfaces for the data objects, generated CRUD events, and generated user interface for data browsing and editing. In a future release it will also support auto-generation of REST APIs.

- **mTraining**: Provides data containers and APIs for defining mobile (e.g. SMS or IVR-based) training courses and tracking user enrollment and progress

- **SMS**: Provides a basic specification for integrating the MOTECH platform with an SMS provider to send/receive SMS messages.

- **Email**: Provides APIs for sending and logging email messages.

2.4 Data Layer

MySQL shall be used for storing MA, MK, Kilkari data as well as data required for generation of reports.

Database design is described in section 10.

3 Mobile Academy Module

3.1 Overview

Mobile Academy (MA) service enables a Front Line Worker (FLW) to take up a health course. When a user dials the MA long code from her mobile phone, the call lands at IVR. MA exposes various web APIs which are invoked by IVR for data exchange and to facilitate the user to take MA course.

Following database entities are used by MA for providing service to FLW users:

- ***Configuration*** (refer 10.1.5.1)
- ***Content Data*** (refer 10.1.5.2)
- ***Course Content*** (refer 10.1.5.3)
- ***FLW Usage Data*** (refer 10.1.5.4)
- ***Call Details*** (refer 10.1.5.5)
- ***Content logs*** (refer □)

3.2 Events Consumed

Mobile Academy subscribes to receive events for the following topics:

3.2.1 Course Content CSV Upload Success

- NMS Administrator/Support Person shall upload MA Content Data into NMS via the NMS Web Interface. The CSV file provided by NMS Support person shall be uploaded into a temporary table. Once the CSV file is uploaded, MOTECH Platform shall generate this event which shall be received by MA Event Handler. [Refer section 11.2]
- On receiving this event,

- MA shall check if MA course structure is already available in mTraining. If it is not available then MA course structure shall be created in mTraining as described in section 3.2.1.1
- Subsequently, the records in the temporary table shall be processed and the MA Content Tables shall be populated as described in section 3.2.1.2.
- Records once processed are deleted from the temporary table.
- If Validation of ChildCsv record fails or it's mapping to Subscriber entity fails or record is rejected during processing , then Kilkari shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason using API detailed in 9.2.1.1.
- After the processing of all csv uploaded records, Kilkari shall update the CSV Upload summary in *BullkUploadStatus* entity. Kilakri shall use the API detailed in 9.2.1.2.

3.2.1.1 mTraining Course Structure Population

- MA Course shall have 11 Chapters each having 4 Lessons and 4 Questions. There shall be constants defined in MA module for number of Chapter, Lessons and questions. These constants shall be used for populating the course structure in mTraining using mTraining service APIs.
- MA shall generate names of these chapters, lessons and questions and shall create course structure in mTraining tables (using mTraining APIs).
- The naming convention for content will be as follows:
  - Chapter – Chapter<index no.> eg. Chapter01, Chapter02,…..,Chapter11
  - Lesson – Lessons<index no.> eg. Lesson01, Lesson02,….., Lesson04
  - Questions – Question<index no.> eg. Question01, Question02,…,Question04
  - Quiz – Quiz

- Once the course is created in mTraining tables, its state will be marked as INACTIVE

3.2.1.2 Content Data Population

- MA shall read each record from CSV upload table and shall validate it as per rules given in 11.2.8.4.
- The validation and bulk upload error handling is done using the ParseDataHelper and BulkUploadError services respectively. These services are provided by Utilities module (refer 9.2.2, 9.2.1).
- If the LLC in record is already found in Content Data table then record is considered as modification record else it is considered as record for addition.
- In case of addition, it will also be validated that the CSV contains complete set of records for entire course for a particular LLC. Else that LLC will be rejected. On successful validation, a new record shall be created in Content Data Table (refer 10.1.5.2) which will be a copy of record in temporary table. This table shall contain Location-wise MA content information (audio file names, duration, etc.) and will be later used for data reporting purpose.
- The type of content shall be determined based on the naming convention for content name. The details on naming convention are available in 11.2.7.5.1
- The index number will also be determined from content name if content is of type chapter, lesson or question.
- Based on the type of content and index number of the content, the record shall be created/modified in MA content tables (refer 10.1.5.3~~10.1.5.39.1.4.3~~). These tables shall contain the MA Chapter, Lesson, Quiz content information. Data in these tables along with

the data in mTraining tables shall be used to create the MA Course Structure file that shall be shared with IVR.

- Once the course content is completely populated in content tables, the state of course will be marked as *ACTIVE* in mTraining course table.
- In case of modification, it is possible to change following for a given Circle, LLC and Content Name:
    - Audio file name :
    - Content ID
    - Content Duration
    - Answer option for a question
- If *audio file name* or *answer option for a question* has changed then it needs to be validated that same *audio file name* or *answer option for a question* has been received in CSV file for all existing LLCs in the system since the course structure is same across all LLCs.
- Every such modification is considered as a modification in course structure. Hence for every such modification the *modificationDate* field in mTraining course table should be updated. In order to do so, the course object will be retrieved and will be updated with same object in database. This will generate a new value for *modificationDate* field. This field will further serve as version of MA course in *Get Course Version* API (refer 3.3.4).

3.3 Service Provided to IVR

Mobile Academy exposes a service for data exchange between IVR and Mobile Academy module. The design description of the services interface APIs is given in following sections.

3.3.1 Get User Details

- MA shall expose *Get User Details* API for fetching the user details given the MSISDN, operator and circle from where call has originated. (refer [2], section 2.2.1)
- These parameters (circle, operator and MSIDN) shall go through data type validation and shall be used to further retrieve user details from FrontLineWorker service API (refer 8.3.1)
- The FrontLineWorker service shall return following user details to MA:
    - Whether FLW exists already
    - Is default LLC applicable. This means circle for FLW is not available or is mapped to multiple LLCs so default LLC value shall be used to prompt FLW for entering her LLC preference.
    - Value of default LLC
    - Circle
    - State level capping for MA usage
    - Deployment status in that state
    - Whether the FLW is white-listed
- MA needs to provide following parameters for an FLW to the IVR. Out of these parameters, the value of 'circle' provided by FrontLineWorker Service is used as is; other parameters are determined by MA (as described in the following sections):
    - Circle
    - LLC (refer section 3.3.1.1)
    - Max Allowed Usage (refer section 3.3.1.2)
    - Current Usage (refer section 3.3.1.3)

- Max Allowed End Of Usage Prompt Limit (refer section 3.3.1.4)
- End Of Usage Prompt Counter (refer section 3.3.1.5)

3.3.1.1 Language Location Code Selection

- Given the MSISDN of the user, the LLC of the user shall be determined using the FrontLineWorker service API (refer 8.3.1)
- If neither LLC nor default LLC is received from FrontLineWorker Service then MA shall return DEFAULT_LANGUAGE_LOCATION_CODE to the IVR. This code is used by IVR to determine the language in which the user should be prompted to enter her language preference.

3.3.1.2 Maximum Allowed Usage Determination

- The maximum allowed usage shall be filled based on the capping type configured in parameter CAPPING_TYPE (refer 11.1.2.3.1).
  - If it is configured as 'No Capping', there will not be any usage limit for all users who dial MA long code.
  - If it is configured to 'National Capping' then there will be a single cap for all Indian states for available free talk-time configured in NATIONAL_CAP_VALUE.
  - If it is configured to 'State-wise Capping' then the capping will be as per the usage limit defined for geographical state of the user (refer 11.2.7.6.2).

3.3.1.3 Current Service Usage for MA User

This usage will be maintained by MA in FLW Usage table (refer section 10.1.5.4). These details will be populated while saving the call details for an FLW [Refer 3.3.7].

If lastAccessTime of an FLW in this table falls in previous months then

- endOfUsagePromptCounter will be reset to zero
- currentUsageInPulses will be reset to zero.

Formatted: List Paragraph, Bulleted + Level: 1 + Aligned at: 2,54 cm + Indent at: 3,17 cm

3.3.1.4 Maximum number of End of Usage messages

- This is determined from configuration parameter MAX_ALLOWED_END_OF_USAGE_PROMPT (refer 11.1.2.3.1).
- This indicates the maximum number of times end-of-usage message can be played to the user.
- If the usage limit for the user exhausts during the call or in new call, end-of-usage message is played to her and a counter is maintained for number of times this message has been played to her.
- During subsequent calls, if number of times this message has been played to her is less than MAX_ALLOWED_END_OF_USAGE_PROMPT then this message is again played else the call is simply dropped.

3.3.1.5 End of Usage Message Count

- This *endOfUsagePromptCounter* shall be retrieved from the MA in FLW Usage table (refer section 10.1.5.4). The counter is updated while saving the call details for an FLW.
- More details are available in section 3.3.7

3.3.1.6 State-wise Service Deployment **Status**

- The list of Indian states where MA service is deployed will be maintained by Master Location Data module.
- When FLW service shall be invoked for getting the user details of an FLW, the service will also determine whether MA service is deployed in that state and will return the deployment status to MA.

- If the service is not deployed in FLW's state, appropriate error message shall be returned to IVR (r~~R~~efer FLW service 8.3.1~~8.3.17.3.1~~).

3.3.1.7 FLW Whitelisting

- Once the state of FLW is determined, it will be responsibility of FLW module to check if the subscriber is white-listed in that state.
- If so, same information will be returned to MA while getting user details.
- If the user is not whitelisted, appropriate error message shall be returned to IVR. (Refer FLW service 8.3.1)

3.3.2 Set Language Location Code of an MA *Subscriber*

- *MA sh*all expose *Set User Language Location Code* API for setting the LLC preference of a FLW (refer [2], section 2.2.7).
- The LLC and MSISDN received in the request will be passed to FrontLineWorker Service API (refer 8.3.2) for setting the Language Location Code preference of the user.

3.3.3 Get Course Structure

- MA shall expose *Get MA Course* API to serve course structure to IVR (refer [2], section 2.2.2).
- The structure of course is independent of the Language Location Code of the user. On receiving the request, MA shall read the course object from *mTraining* tables (using mTraining module APIs). It shall check whether:
    - The course object exists in mTraining table using mTraining APIs. If the course object does not exist, error response shall be returned to IVR indicating that course is not yet populated.
    - State of course is ACTIVE. If state of course is not ACTIVE, error response shall be returned to IVR indicating that course upload is in progress.
- If state of course is ACTIVE, it shall read content names and content audio files names from course content tables.
- Having retrieved above course data, MA shall generate course structure in JSON format using *GSON* library. The JSON shall then be returned to IVR.

3.3.4 Get Course Version

- MA shall expose *Get MA Course* Version API to serve the course version to IVR (refer [2], section 2.2.3).
- On receiving this request, MA shall check whether:
  - The course object exists in mTraining table using mTraining APIs. If the course object does not exist, error response shall be returned to IVR indicating that course is not yet populated.
  - State of course is ACTIVE. If state of course is not ACTIVE, error response shall be returned to IVR indicating that course upload is in progress.
  - If state of course is ACTIVE, then *modificationDate* parameter value (in seconds) in mTraining course object shall be returned to user as course version.

3.3.5 Get Bookmark and Score

- MA shall expose *Get Bookmark With Score* API to get the bookmark of the user whose call is ongoing. (refer [2], section 2.2.4).
- The MSISDN of the user will be received in request and it will be validated using FLW service (refer 8.3) that MSISDN exists in FLW records. If MSISDN does not exist then appropriate error response shall be returned to the IVR..
- On receiving the request, MA shall retrieve following information from mTraining bookmark table:
  - Actual bookmark. This will be stored in *progress* map against key *bookmark_id*
  - Scores by chapter. This refers to the scores achieved by user per chapter. This will also be stored in *progress* map with key being chapter no.

- A sample of *progress* map in mTraining Bookmark table is given below-

```
{
        "bookmark_id" -> "Chapter05_Question01",
        "1" -> 1,
        "2" -> 4,
        "3" -> 3,
        "4" -> 2
        }
```

3.3.6 Save Bookmark and Score

MA shall expose *Save Bookmark With Score* API to save the bookmark and chapter scores of a user. ~~This will be invoked only after disconnect of the call~~(refer [2], section 2.2.5).~~.~~

- ~~There will be a single JSON containing bookmark and chapter scores of a user which will be received in request. The scores of only those chapters shall be provided in the request which have been attempted till the bookmark point.~~

- The MSISDN of the user will be received in request and it will be validated using FLW service (refer 8.3~~8.37.3~~) that MSISDN exists in FLW records. If MSISDN does not exist then appropriate error response shall be returned to the IVR.
- Each node in the course structure which can be bookmarked has a unique identifier which is already shared with IVR as part of MA course structure. Same identifier shall be sent by IVR in this request. However, it will not be validated at MA whether bookmark is valid or not. Whatever is shared by IVR will be saved as it is with MA.

- MA does not use this bookmark internally. It is solely for communicating the IVR the bookmark location in MA course.
- The scores per chapter will also be sent by IVR in this request. The bookmark as well as scores for the user shall be saved in *progress* map in mTraining bookmarks table. A sample of *progress* map is given in section 3.3.5.

3.3.7 Save Call Details

- This API shall be invoked once the call gets disconnected. The IVR shall invoke this API and shall provide call related data and course content related logs in the Save Call Details request. (refer [2], section 2.2.6).
- The course content related logs will be in the form of list of details for each content file being listened to by the user.
- The data related to call shall be saved in Call Details table [Refer 10.1.5.5] while data related course shall be saved in Content logs table [Refer]
- The data in both these tables will be used only for reporting purpose.
- A new record will be created in Call Details table whenever 'Save Call Details' shall be invoked and data shall be stored as follows in Call Details table:
  - callId – The identifier of the call. This will be sent by IVR in requestrequestrequest.
  - callStartTime – The start time of the call. This will be sent by IVR in request
  - callEndTime – The time when the call terminated. This will be sent by IVR.
  - circle – provided by IVR
  - operator – provided by IVR

- msisdn – calling number of the user. The last 10 digits of the calling number received in request will be taken as MSISDN of the user.
- flwId – given the calling number, FLW service shall return the flwId of the user. Same will be saved here.
- callStatus – provided by IVR
- callDisconnectReason – provided by IVR.

- A new record shall be created in content logs table for each content being listened to and shall be stored as follows
  :
  - callId – The identifier of the call. This will be same as one saved above in call details table.
  - contentName – the name of content being played. This will be sent by IVR after picking from MA course structure.
  - contentFile – the name of the content file being played. This will be sent by IVR after picking from MA course structure.
  - languageLocationCode – This will be determined from MSISDN of the user using FLW service.
  - startTime – This refers to the timestamp when the IVR started to play this content to the user. This is provided by IVR.
  - endTime- This refers to the timestamp when IVR stopped playing the content to the user either due to completion of audio file content or due to call disconnect. This is provided by IVR.
  - completionFlag – This flag indicates if the content being played to the user has been listened to completely.
  - courseStartTime – This is the time when user start taking MA course. This will be same as start time of chapter 1 lesson 1. If the user attempts chapter 1 lesson 1 multiple

times then the timestamp whichever is older will be taken as course start time. This will also be saved in FLW Call Usage table.

- courseEndTime – This is the time when user completed the MA course. This will be same as end time of chapter 11 question 4 provided completion flag is trutruetru for this content. If the course is not yet completed, then this field will not be populated. If the user attempts chapter 11 question 4 multiple times then timestamp which is newer will be taken as course end time. This will also be saved in FLW Call Usage table.
- correctAnswerReceived – This field shall be significant only in case content type is question and shall indicate whether a particular question has been answered correctly by the user. This shall be further used in reporting to find score of a particular FLW for a chapter during a course attempt.

  This field is required in addition to scores by chapter being maintained for a user because scores by chapter gives latest score whereas reports generation needs all scores of a user in multiple course attempts.

- Besides following call logs will be saved only in FLW Usage table (refer 10.1.5.4):
  - lastAccessTime – time of last call of FLW to MA.
  - currentUsageInPulses – This refers to the talk time which the user has already consumed while taking MA course. This time will be accumulated at MA based on callDurationInPulses received from IVR for a particular call.
    ~~reset to zero every month.~~
  -
  - endOfUsagePromptCounter – This refers to number of times the End Of Usage prompt counter has been played to the user. The same is provided by IVR in totality in request.

3.3.8 SMS Delivery Notification

<< details will be added in sprint 1505>>

4 Mobile Kunji Module

4.1 Overview

- FLW access the Mobile Kunji service by dialing an IVR long code and entering a 2 digit number of one of the MK cards. The deck of cards along with the IVR service communicates essential audio-visual information on pregnancy and newborn health.
- Mobile Kunji module stores MK service data such as location information, service usage details of all MK subscribers
- Mobile Kunji module exposes an interface to IVR to fetch information necessary for it to provide Mobile Kunji service.
- Following database entities are used by this module to store MK related information:
    - *Configuration* [Refer 10.1.6.1]
    - *Content Data* [Refer 10.1.6.2]
    - *Call Detail data* [Refer 10.1.6.3]
    - *FLW Detail data* [Refer 10.1.6.4]

4.2 Events Consumed

Mobile Kunji module subscribes to receive success events for the following topics:

4.2.1 Content Data CSV Upload Success

- This event is raised by MDS, when MobileKunji Content metadata CSV file is uploaded successfully in *ContentUploadCsv* table.
- The event contains the list of ids of successfully uploaded records and name of the user who has uploaded the file.

- Format of CSV file is specified in section 11.2.7.5.2
- On receiving this event:
    - Mobilekunji reads records from ContentUploadCsv table one by one for processing.
    - Mobilekunji uses static methods provided by Utility module for CSV data validation and parsing (refer 9.2.2.1).
    - For each CSV record that passes validation, Mobile Kunji creates a record in ContentUpload entity.
    - If Validation of any CSV record fails or it's mapping to ContentUpload entity fails or record is rejected during processing, then Mobile Kunji module shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason using API detailed in 9.2.1.1.
    - After the processing of all CSV uploaded records, Mobile Kunji shall update the CSV Upload summary in *BullkUploadStatus* entity using the API described in 9.2.1.2.

4.3 Services Provided to IVR

Mobile Kunji exposes a service for data exchange between IVR and Mobile Kunji module. The design description of the services interface APIs is given in following sections.

4.3.1 Get User Details

- MK shall expose *Get User Details* API for fetching the user details given the MSISDN, operator and circle from where call has originated. (refer [2], section 3.2.1)

- These parameters (circle, operator and MSIDN) shall go through data type validation and shall be used to further retrieve user details from FrontLineWorker Service (refer 8.3.1)
- The FrontLineWorker service shall return following user details to MK:
  - Whether FLW exists already
  - Is default LLC applicable. This means circle for FLW is not available or is mapped to multiple LLCs so default LLC value shall be used to prompt FLW for entering her LLC preference.
  - Value of default LLC
  - Circle
  - State level capping for MK usage
- If FLW send isCreated as true i.e. FLW created by FronltLineWorker, MK update the record in FlwDetail if it exist or create a new record if it does not exist
- If FLW send isCreated as false i.e. FLW exist in FronltLineWorker entity,MK does not update FlwDetail entity.
- MK needs to provide following parameters for an FLW to the IVR. Out of these parameters, the value of 'circle' provided by FrontLineWorker Service is used as is; other parameters are determined by MK (as described in the following sections):
  - Circle
  - Language Location Code (refer)
  - Max Allowed Service Usage (refer 4.3.1.2)
  - Current Service Usage (refer 4.3.1.3)
  - Maximum number of End Of Usage messages (refer 4.3.1.4)
  - End Of Usage Prompt Counter (refer 4.3.1.5)

- State Wise deployment status (refer 4.3.1.6)
- WhiteListing status (refer 4.3.1.7)

4.3.1.1 Language Location Code Selection

- *UserProfileDetailsService* API provided by FrontLineWorker is used to determine LLC of the user (refer 8.3.1).
- MK checks the "Is default LLC" flag returned by the FrontLineWorker API, if it true then MK sends *defaultLanguageLocation* code returned by the API to IVR. IVR shall then prompt the user to enter the Language Location Code.
- If flag is false, MK sends Language Location Code returned by FrontLineWorker to IVR.

4.3.1.2 Max Allowed Usage Determination

- The maximum allowed usage shall be filled based on the capping type configured in parameter CAPPING_TYPE.
- If it is configured to 'No Capping', there will not be any usage limit for a user who dials MK.
- If it is configured to 'National Capping' then there will be a single cap for all Indian states for available free talk-time configured in NATIONAL_CAP_VALUE.
- If it is configured to 'State-wise Capping' then the capping will be as per the usage limit defined for geographical state of the user (refer 11.2.7.6.1).

4.3.1.3 Current Service Usage Determination

- MK performs a lookup based on *msisdn* (provided by IVR) in FLWDetail table (refer 10.1.6.4) and if the FLW record exists then,

- If *lastCallActivity* month is same as the current month, then the retrieved *currentUsage* value is returned to IVR.
- If the month is different, then the monthly service usage is reset to 0 and is returned to IVR.

- In case lookup fails, then MK sends DEFAULT_CURRENT_USAGE (0) to IVR.

4.3.1.4 Maximum number of End of Usage messages

- This is determined from configuration parameter MAX_ALLOWED_END_OF_USAGE_PROMPT (refer 11.1.2.3.2).
- This indicates the maximum number of times end-of-usage message can be played to the user.

4.3.1.5 End of Usage Message Count

- MK performs a lookup based on *msisdn* (provided by IVR) in FLWDetail table (refer 10.1.6.4) and if the FLW record exists then *endOfUsagePrompt* value is returned to IVR.
- If lookup fails, then MK sends DEFAULT_END_OF_USAGE_PROMT (value=0) to IVR.

4.3.1.6 State-wise Service Deployment Status

- *UserProfileDetailsService* API provided by FrontLineWorker is used to determine state-wise deployment.
- FLW API throws and exception ServiceNotDeployedException and, in this case, MK rejects the call with failure response.
- Refer 8.3.1 for details.

4.3.1.7 Subscriber Whitelisting

- *UserProfileDetailsService* API provided by FrontLineWorker is used to determine whiteListing enablement.
- FLW API throws and exception FlwNotInWhiteListException and, in this case, MK reject the call with failure response.
- Refer 8.3.1 for details.

~~4.3.3~~4.3.2 Set Language Location Code

- MK shall expose *Set User Language Location Code* API for setting the LLC preference of a FLW (refer [2], section 3.2.3).
- The LLC and MSISDN received in the request will be passed to FrontLineWorker Service *UserProfileDetailsService* API (refer 8.3.2) for setting the Language Location Code preference of the user.

4.3.4~~4.3.3~~ Save Call Details

- MK receives save call details request from IVR (refer [2], section 3.2.2).
- It uses *UserProfileDetailsService* provided by FrontLineWorker (refer 8.3.1) to validate the *operator* value provided from IVR.
- API used *: void validateOperator(String operatorCode) throws DataValidationException;*
- In case of validation failure, MK sends an error response to IVR.
- In case of success, MK saves the call related data (provided by IVR) in *CallDetail* and *FLWDetail* entities.

- A new entry is created in *CallDetail* table for each 'Save Call Details' coming from IVR.
- *welcomeFlag*, *currentUsage* and *endOfPromtCounter* are updated the *FLWDetail* record (corresponding to *msisdn* value in request)

5 Kilkari Module

5.1 Overview

- Kilkari module supports the creation / deactivation of Beneficiary Subscription via IVR.
- It supports the creation and update of Beneficiary details and subscription via MCTS Bulk Upload procedure.
- Kilkari shall provide Service APIs (as per section 5.4) that shall be used by Kilkari Out-Bound Dialer Module (refer 6) for preparing OBDBDOBD target file and during CDR file processing.
- Following are the MDS Entities that Kilkari shall utilize:
    - ***Subscriber***: This entity shall store the personal details of a Kilkari Beneficiary. It shall also contain a list of references to the Subscriptions, which belongs to the beneficiary.
    - ***Subscription***: This entity shall contain the details (like pack Name, status etc.) of Subscription for which beneficiary has subscribed.
    - ***SubscriptionMeasure***: This entity contains a record for subscription's transient information: status, weekNumber, messageNumber and a reference to the corresponding subscription record. This is needed for reporting purpose.
    - ***ContentUpload***: This entity shall contain the Kilkari Content Metadata information and shall be used to send the language and content file name to IVR system.
    - ***Configuration***: This entity shall contain the configuration parameters relevant to Kilkari service. Refer 11.1.1.5 for configuration parameter details.
    - ***ActiverUsers***: This entity shall contain the count of Active/PendingActivation subscriptions in the system.

- Refer Section 10.1.7 for DB schema of the above mentioned MDS entities.

5.2 Events Consumed

Kilkari subscribes to receive events for the following topics:

5.2.1 Successful Upload of Child MCTS data.

- This event is raised by MDS, when Child MCTS data CSV file is uploaded successfully in ChildCsv table.
- The event contains the list of ids of successfully uploaded records and name of the user who has uploaded the file. Refer section 11.2 for details on Data Upload design
- Kilkari shall read the records from ChildCsv table one by one for processing. Refer 11.2.7.3.2 for format of Child MCTS data CSV.
- Kilkari shall use the static methods provided by Util Bundle, in section 9.2.2 for CSV data validation and parsing, to validate ChildCsv record and map it to Subscriber entity Validation rules are mentioned in 11.2.8.2
- Kilkari shall also validate the location of the beneficiary provided in the CSVRecord using the Location Service provided by Masterdata Module (refer 7.3.1.1).
- If the validation passes and mapping is successful then, Kilkari shall
    - Create a new *Subscription* if the count of currently active subscriptions is less than the allowed value and there is no existing Subscription record for 72WeeksPack with Status as Active or PendingActivation having
        - msisdn matching the one in CSVRecord.
        - mctsId matching the one in CSVRecord.

      (Refer 5.3.2.1 on how to count and check against the number of currently active subscriptions in the system)

- Create a new *Subscriber* if there is no existing *Subscriber* having
  - msisdn matching the one in CSVRecord.
  - childMctsId matching the one in CSVRecord.
- Update the *Subscriber* record(MSISDN, childMctsId, motherMctsId, Location, name and DOB) if there is an existing record
  - Having childMctsId same as the one in CSVRecord.
  - Having motherMctsId same as the one in CSVRecord.
  - Having null or empty childMctsId and msisidn matching the one in CSVRecord .
- Update an existing *Subscription's* status as Deactivated, if in CSVRecord.
  - EntryType is Death i.e. child death is reported.
  - DOB is modified (also create a new subscription in this case with new DOB)
  - motherMCTSid matches to the existing *Subscription* (having status as Active/PendingActivation), create new *Subscription* as per child DOB.
- Reject the record, if Child Death is reported for a non-existing subscriber with reason as "Child death reported for Non Existing Subscriber".

- If Validation of ChildCsv record fails or it's mapping to Subscriber entity fails or record is rejected during processing , then Kilkari shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason using API detailed in 9.2.1.1.
- After the processing of all csv uploaded records, Kilkari shall update the csv Upload summary in *BullkUploadStatus* entity. Kilakri shall use the API detailed in 9.2.1.2.

5.2.2 Successful Upload of Mother MCTS data.

- This event is raised by MDS, when Mother MCTS data CSV file is uploaded successfully in MotherCsv table.

- The event contains the list of ids of successfully uploaded records and name of the user who has uploaded the file. Refer section 11.2 for details on Data Upload design
- Kilkari shall read the records from MotherCsv table one by one for processing. Refer 11.2.7.3.1 for format of Mother MCTS data CSV
- Kilkari shall use the static methods provided by Util Bundle, in section 9.2.2, for CSV data validation and parsing to validate MotherCsv record and map it to Subscriber entity, Validation rules are mentioned in 11.2.8.1.
- Kilkari shall also validate the location of the beneficiary provided in the CSVRecord using the Location Service provided by Masterdata Module, refer section 7.3.1.1.
- If the validation passes and mapping is successful then, Kilkari shall
    - Create a new Subscription if the count of currently active subscriptions is less than the allowed value and there is no existing Subscription record for 48WeeksPack with Status as Active or PendingActivation having
        - msisdn matching the one in CSVRecord.
        - mctsId matching the one in CSVRecord.

      (Refer 5.3.2.1 on how to count and check against the number of currently active subscriptions in the system)
    - Create a new *Subscriber* if there is no existing *Subscriber* having
        - msisdn matching the one in CSVRecord.
        - motherMctsId matching the one in CSVRecord.
    - Update the *Subscriber* record (MSISDN, motherMCTSId, Location, name , age and LMP) if there is an existing record
        - Having motherMctsId same as the one in CSVRecord.
        - Having null or empty motherMctsId and MSISDN matching the one in CSVRecord.

- Update an existing *Subscription's* status as Deactivated, if in CSVRecord
  - Number of outcome is 0 i.e. stillbirth is reported
  - EntryType is Death i.e. mother death is reported.
  - Abortion is not "none" i.e. abortion is reported.
  - LMP is modified (also create a new subscription in this case).
- Reject the record, if MotherDeath/Abortion/StilBirth is reported for a non-existing subscriber with reason as "MotherDeath/Abortion/StilBirth reported for Non Existing Subscriber"

- If Validation of MotherCsv record fails or it's mapping to Subscriber entity fails or record is rejected during processing, then Kilkari shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason, using API detailed in 9.2.1.1.
- After the processing of all csv uploaded records, Kilkari shall update the csv Upload summary in *BullkUploadStatus* entity. Kilkari shall use the API detailed in 9.2.1.2.

5.2.3 Successful Upload of Content Metadata.

- This event is raised by MDS, when Kilkari Content metadata CSV file is uploaded successfully in *ContentUploadCsv* table.
- The event contains the list of ids of successfully uploaded records and name of the user who has uploaded the file. Refer 11.2.7.5.3 for format of CSV file.
- Kilkari shall read the records from ContentUploadCsv table one by one for processing.
- Kilkari shall use the static methods provided by Util Bundle, in section 9.2.2, for CSV data validation and parsing to validate ContentUploadCsv record and map it to ContentUpload entity. (Validation rules are mentioned in 11.2.8).
- If the validation passes and mapping is successful then, Kilkari shall:

- Find if there is an existing record in *Circle* entity corresponding to the circleCode value (present in the CSVRecord), using API provided by Master Location Data module.
- If found, then check if there is an existing record in *LanguageLocationCode* entity corresponding to circleCode and languageLocationCode values (present in the CSVRecord), using API provided in 7.3.2.2.
- Reject the csvContent and log error in *BulkUploadError* entity with erroneous record details and error reason "circleCode Invalid" or "languageLocationCode invalid", if either Circle or LanguageLocationCode record is not found as per previous steps.

- If Validation of ContentUploadCsv record fails or it's mapping to ContentUpload entity fails or record is rejected during processing, then Kilkari shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason, using API detailed in 9.2.1.1.
- After the processing of all csv uploaded records, Kilkari shall update the csv Upload summary in *BullkUploadStatus* entity. Kilakri shall use the API detailed in 9.2.1.2.

5.2.4 Scheduled Event for Deletion of Deactivated/Completed Kilkari Subscription Records

Kilkari Module shall listenlistenschedule a cron job to thetheget a daily event, using Motech Scheduler Service, for deletion of Deactivated/Completed Kilkari Subscription records.

OnOnKilkari shall listen to this event and
~~Subscription records.~~

~~O~~on receiving the event Kilkari shall

- Find the list of Distinct Subscribers from Subscription records having with status as Completed or Deactivated and modification date is 6 weeks older than the current date.
- Delete all the Subscriptions with status as Completed or Deactivated and where modification date is 6 weeks older than the current date.

- For each of the Subscribers found in earlier step check if there is no associated subscription, then delete that subscriber record as well.

5.3 Service Provided to IVR

- Kilkari exposes following APIs to IVR System:
  - Get Subscriber details: To Fetch Subscriber details using UserDetailsService.
  - Create Subscription: To create new Subscription using subscriptionService.
  - Deactivate Subscription: To deactivate an existing Subscription using subscriptionService.

5.3.1 Get Subscriber Details

- Kilkari shall expose Get Subscriber Details API, detailed in section 4.2.1 of [2], to be invoked by IVR to
  - Determine the language and location preference for a registered. See 5.3.1.1.
  - Determine the language and location preference for an anonymous user. See 5.3.1.2.
  - Determine list of Active / PendingActivation Subscription pack names. See 5.3.1.3.

5.3.1.1 Language Selection for a Registered User

- Kilkari shall first find the registered Subscriber record with MSISDN matching the calling number,
  - Then determine the languageLocationCode for the subscriber as the value of languageLocationCode field, if present in the record.
  - Else it shall determine the languageLocationCode using Subscriber's State and District information using the LanguageLocationCode service. Refer section [7.3.2.1]

- If languageLocationCode is not determined from Subscriber record or from its location information then Kilkari shall send failure response to IVR with HTTP Status code as "500- internal server error" with failure reason as "LanguageLocationCode not found for Subscriber's state and district"

5.3.1.2 Language Selection for an Anonymous Kilkari Subscriber

- If user is not registered and then Kilkari shall use the CircleCode present in the API request.
- If the CircleCode is present in this API then Kilkari shall use API provided by LanguageLocationCode Service (refer section [7.3.2.4]) to determine the unambiguous LanguageLocationCode corresponding to the Circle (corresponding to CircleCode). (If there is only one LanguageLocationCode corresponding to a CircleCode then only it is considered unambiguous).
- If CircleCode is present in API but LanguageLocationCode for its Circle is ambiguous, then
  - Kilkari shall determine the DefaultLanguageLocationCode value corresponding to the Circle (corresponding to CircleCode) using API detailed in [7.3.2.5].
  - IVR shall use this DefaultLanguageLocationCode to play language selection prompt to user.
  - User shall provide the LanguageLocationCode DTMF Input.
  - IVR shall then send the user selected LanguageLocationCode in Create Subscription Request API (section 5.3.2), which shall be saved in the Subscriber record.
- If CircleCode is Unknown in this API and DefaultLanguageLocation code for Circle is not determined in above step, then
  - Kilkari shall determine the LanguageLocationCode for language selection as National Default LanguageLocationCode. Refer section 11.1.2.3.3 for details on this configuration parameter.
  - IVR shall use this LanguageLocationCode to play language selection prompt to user.

- User shall provide the LanguageLocationCode DTMF Input.
- IVR shall send the user selected LanguageLocationCode in Create Subscription Request API (section 5.3.2), which shall be saved in the Subscriber record.

5.3.1.3 Determination of Active / PendingActivation subscription pack names

- Kilkari shall determine list of Active/PendingActivation subscription packs, if any for the Subscriber with MSISDN matching calling number.
- IVR shall deny new subscription, if Kilkari reports an existing Active/PendingActivation subscription for the pack corresponding to the LongCode dialed by user.

5.3.2 Create Subscription

- Kilkari shall support the subscription creation for a new or an existing beneficiary through IVR long code.
- Kilkari shall also expose the *Create Subscription Request API,* detailed in section 4.2.3 of [2], to be invoked by IVR System to
  - Create a new Subscriber with LMP/DOB as date of call (and MCTSId, name, age, location, as null), if there is no existing Subscriber record with MSISDN matching the calling number.
  - Update an existing Subscriber record (with LMP/DOB information as date of call, mother & child MCTSId as null) in NMS database if MSISDN matches the calling number, but having no any corresponding Active/PendingActivation subscription pack matching the pack corresponding to dialed long code.
  - Create a new Subscription record in NMS database if there is no existing subscription record with MSISDN and SubscriptionPack (having status as Active/PendingActivation) which matches the calling number and pack corresponding to dialed long code.

5.3.2.1 Count of Active Subscriptions

- Kilkari shall maintain the count of Subscriptions with status as Active or PendingActivation.
- If this count is equal or more than the value of configurable parameter NMS_KK_MAX_ALLOWED_ ACTIVE _BENEFICIARY_COUNT, any new subscription request shall be rejected.
- Implementation:
    - The total number of currently active Kilkari Subscriptions shall be stored in a table, refer 10.1.7.4, having only 1 row).
    - The value of this field shall be incremented (decremented) whenever a Kilkari subscription is created (deactivated/Completed). SQL UPDATE query shall be used to increment / decrement the value of this counter, so that the counter is incremented / decremented atomically. Atomic update shall ensure that value of active subscriptions count used by any node is consistent with the NMS database.
    - For each new subscription via CSV upload, the value of the total number of Kilkari subscribers shall be retrieved from DB if it exceeds NMS_KK_MAX_ALLOWED_ ACTIVE _BENEFICIARY_COUNT, the record (and all subsequent new subscription records in the CSV) shall be rejected. Kilkari shall also create an error log entry in *BulkUploadError* entity with CSV record details and error reason as “Active Subscription Count Exceeded”.
    - For each new subscription via IVR, the value of the total number of Kilkari subscribers shall be retrieved from DB if it exceeds NMS_KK_MAX_ALLOWED_ ACTIVE _BENEFICIARY_COUNT, the request shall be rejected. Kilkari shall respond the “create subscription API” request with “Internal server error” and failure reason as “Active Subscription Count Exceeded”
    - Parallel subscription requests (parallel CSV Uploads or IVR requests), may result in the total number of Kilkari subscribers exceeding NMS_KK_MAX_ALLOWED_ ACTIVE _BENEFICIARY_COUNT by a small number. This may be prevented by having some

additional checks, however, since this is a rare scenario, these checks shall be omitted in order to avoid increasing code complexity.

5.3.3 Deactivate Subscription

- Kilkari shall expose the "Deactivate Subscription API", detailed in section 4.2.4 of [2], to allow the beneficiary to deactivate Subscription to his pack via IVR.
- Kilkari shall find the Subscription record corresponding to the SubscriptionId present in the "Deactivate Subscription API".
- If found it shall change the status of the Subscription to Deactivated.
- If record is not found corresponding to the SubscriptionId, then Kilkari shall just return the HTTP OK status to the API invoker and capture a warning log for the same

5.4 Service Provided to Kilkari OBD Module

5.4.1 Deactivate Subscription

- Signature: *void deactivateSubscription(Long subscriptionId, DeactivationReason reason)*
- This API retrieves the Subscription record as per the *subscriptionId*, then it updates the status of this record as "Deactivated" with deactivationReason = *reason*. For reason as "NUMBER_IN_DND", Kilkari deactivates the subscription only if channel in the subscription record is "IVR",
- In case the record is not found for the *subscriptionId*, it just creates a Warning Log in the log file.

5.4.2 Get Content File Name

- Signature: *String getContentFileName(String contentName, Interger languageLocationCode)*
- This API retrieves Kilkari's ContentUpload entity record for the given value of the contentName and languageLocationCode. and returns the content file name.
- If no record is found it shall return the value as null.

5.4.3 Get scheduled Subscriptions for OBD

- Signature: *List<Subscription> getScheduledSubscriptions();*
- This API shall return the list of Subscriptions for which a Kilkari OBD message shall be delivered on the current date. It shall also change the state of packs to "completed" if week number computed is for last week of pack. It shall also change the status of subscription from "pendingActivation" to "Active".

- Kilkari shall find Active/PendingActivation subscriptions that could be eligible for OBD message delivery as below, (queries checks if today is day 1 or day 3 of week for any subscription with reference to its start date):
    - If NumberOfMessagePerWeek == 1, then “select * from subscription where (status = Active or pendingActivation) and ((startDate – Today’sDate) % 7 = 0)”.
    - Else If NumberOfMessagePerWeek == 2, then “select * from subscription where (status = (Active or pendingActivation) and ((startDate – Today’sDate) % 7 = 0 or 3)”.
- For each subscription Kilkari shall compute
    - The week number for which message is to be sent for a pack based on DOB or LMP as
        - Start week number of 48 Weeks Pack + (Today’s date - DOB) / 7
        - Start week number of 72 Weeks Pack + (Today’s date – (LMP+3)) / 7
    - The message number for which message is to be sent for based on value of configuration parameter NumberOfMessagePerWeek.
        - Set message number = 1.
        - If (NumberOfMessagePerWeek == 2) and ((startDate – Today’sDate) % 7 == 3) then set message number = 2. [This mean if today is day 4 (starting from 0 as first day) of week for a subscription, then message 2 shall be delivered].

- If Week Number computed is greater than DurationOfPackInWeeks (72 or 48, respectively) then update the status to “Completed” and remove this subscription from list.
- If Week Number computed is equal to DurationOfPackInWeeks (72 or 48, respectively) then update the status to “Completed”.
- If Week Number computed is less than DurationOfPackInWeeks (72 or 48, respectively) and the status of a pack is pendingActivation then update it to “Active”.
- Update the subscription in DB.
- Update the WeekNumber and MessageNumber in KilkariSubscriptionMesasure entity (refer 10.1.7.3).

6 Kilkari Out-Bound Dialer Module

6.1 Overview

- Kilkari Out-Bound Dialer (KilkariObd) module supports preparation of OBD target file and processing of the CDR files.
- This module is separated from Kilkari module in order to keep the business logic for OBD preparation and CDR processing, which is specific to the IVR vendor, as independent of subscription logic.
- Outbound call flow shall be as follow:
    - KilkariObd shall prepare the OBD target file for
        - Fresh calls for subscription having a weekly message scheduled for current date
        - The calls that have failed on the previous attempt and are eligible for retry on current date.
    - KilkariObd shall perform a secure copy of the OBD target file to a location that is accessible by IVR.
    - KilkariObd shall notify IVR System to pick and process the OBD Target file.
    - On successful processing of OBD Target file IVR shall notify the processing completion to KilkariObd.
    - Then after making the calls as per OBD target file, during social hours, IVR shall do a secure copy of CDR Summary and Detail files to the same location (at which OBD target file was kept by KilkariObd).
    - IVR shall notify creation of KilkariObd with details of CDR Summary and details.
    - KilkariObd shall start the process again for next day's date.

- Following are the MDS Entities that KilkariObd shall utilize:
    - ***OutboundCallFlow***: This entity shall contain status corresponding to the processing of OBD target file and CDR file. KilkariObd shall use the information from this table to get the name of the CDR files to be used for OBD target preparation. This table shall also aid in monitoring the Outbound Call flow.
    - ***OutboundCallDetail:*** This entity shall contain the call details of the OBD message delivery calls requested in OBD target file. The records in this table shall be used in generating Kilkari reports.
    - ***OutboundCallRequest***: This entity shall contain the OBD call definition records, where each record shall have the parameters required by IVR to make calls for OBD message delivery.
    - For more details on schema of these entities refer 10.1.9.

6.2 Events Consumed

6.2.1 Scheduled Event for preparation of OBD TargetFile

- KilkariObd shall create a job in MOTECH Scheduler/Batch, if it is not already created, to get a daily event, at 1 minute past midnight, for preparing OBD TargetFile. The details and format of the OBD target file are detailed in 4.4.1 of [2].
- On receiving the event KilkariObd module shall
    - Invoke the ***GetScheduledActiveSubscriptions*** API, provided by Kilkari SubscriptionService in section 5.4.3, to get the list of active subscription for which OBD messages are to be delivered.
    - Delete all the records from OutboundCallRequest, to ensure that it doesn't contain the call requests from older date.

- For each subscription record present in the retrieved list, KilkariObd shall create a record in ***OutboundCallRequest*** entity with
  - RequestId as Subscription id.
  - ServiceId as value of configuration parameter "NMS_KK_FRESH_OBD_SERVICE_ID".
  - Msisdn as msisdn mentioned in subscription record
  - Priority as value of configuration parameter "NMS_KK_FRESH_OBD_PRIORITY".
  - LanguageLocationCode as languageLocationCode present in subscription record.
  - Circle as circleCode returned by languagelocationCode service API detailed in section 7.3.2.4.
  - WeekId as "<weekNumber>_<messageNumber>", where weekNumber and messageNumber are used from Subscription record.
  - ContentFileName as value returned by GetContentFileName API for content name "W<WeekId>". Refer section 5.4.2 for details on API GetContentFileName, provided by Kilkari ContentUploadService.

    (if filename is returned as Null, then create an Error Log for the subscription and don't add this in the ***OutboundCallRequest***)
  - retryDayNumber = 0
  - CallFlowUrl and Cli shall be set as null.
- KilkariObd shall find the ***OutboundCallFlow*** record where creation date is the date of previous day.

- For ObdFileName present in the found record, KilkariObd shall read and parse the corresponding CDRSummary CSV file:
  - Create new records in ***OutboundCallRequest*** entity for each CDRSummary record, with field values same as in CDRSummary record (except that retryDayNumber shall become retryDayNumber +1), if CDRSummary record is having :
    - Final Status as "Rejected".
    - Final status as "Failed", "StatusCode" as other than "OBD_FAILED_INVALIDNUMBER" or "OBD_DNIS_IN_DND" and "retryDayNumber" less than
      - "3" for "NMS_KK_MSGS_PER_WEEK" as 1
      - "1" for "NMS_KK_MSGS_PER_WEEK" as 2
  - For records having final status as "Failed" and "StatusCode" as either "OBD_FAILED_INVALIDNUMBER" or "OBD_DNIS_IN_DND", invoke the DeactivateSubscription API (section 5.4.1) for the subscriptionId (as value of "RequestId"). For "OBD_DNIS_IN_DND" deactivation is done only for subscription with channel as "IVR".
  - For records having final status as "Failed" and statusCode as "OBD_FAILED_NOATTEMPT" create an entry in OutboundCallDetail with call start, answer and end time as null.
- On completion of CDR Summary File processing, export the data in ***OutboundCallRequest*** entity into a CSV file named as specified in section 4.1.5 of [2]. Place the file so created at location configured as value of "NMS_KK_OBD_FILE_LOCAL_PATH".
- Read and parse the CDRDetail csv file corresponding to the ObdFileName, and create records in ***OutboundCallDetail*** entity for each call detail record present in csv file.

Update the status of record found from ***OutboundCallFlow*** to **"CDRFilesProcessed".** And invoke the CDRFileProcessedStatus notification with "cdrFileProcessingStatus" as "8000" (FILE_PROCESSED_SUCCESSFULLY).

(In case of any error in processing CDRDetail or Summary file, KilkariObd shall send failureReason in CDRFileProcessedStatus notification to IVR. Refer 4.3.2 in [2]. KilkariObd shall implement wait/retry logic for the response to this API as per section 6.2.2.1)

- Create a new entry in ***OutboundCallFlow*** with file name of newly created CSV file, MD5 checksum of the file, number of records in the file and status as "**OutboundCallRequestFileCreated**".
- Perform SCP of this OBD Csv file from "NMS_KK_OBD_FILE_LOCAL_PATH" to the location "NMS_KK_OBD_FILE_PATH" on server identified by IP "NMS_KK_OBD_FILE_SERVER_IP". Change the status in ***OutboundCallFlow*** for this file name as "**OutboundCallRequestFileCopied**".

6.2.2 Scheduled Event for invoking OBD TargetFile notification API

- KilkariObd shall create a job in MOTECH Scheduler/Batch, if it is not already created, to get a daily event for submitting the target file to IVR by invoking TargetFile notification API.
- On receiving the event KilkariObd module shall
  - Find the ***OutboundCallFlow*** record where creation date is the date of current day.

    (if no such record found, create a warning log that no OBD target created for current date and return from the event handler)
  - Invoke TargetFile notification API, refer 4.3.1 in [2], with "filename", checksum and no. of records as present in the ***OutboundCallFlow*** record.

(KilkariObd shall implement wait/retry logic for the response to this API as per section 6.2.2.1)

6.2.2.1 Invocation of API provided by IVR

- KilkariObd shall invoke the following APIs provided by IVR System:
    - "TargetFile notification"
    - "CDRFileProcessedStatus notification"
- KilkariObd shall use "NMS_KK_IVR_IP" and "NMS_KK_IVR_PORT" configuration parameters to invoke these APIs.
- These APIs are Offline API as defined in Chapter 6:" HTTP Timeout Categories" of [2].
- KilkariObd shall implement the Exponential Back-off algorithm, detailed as below, for retrying the aforementioned APIs after timeouts:
    - After invoking an Offline API KilkariObd HTTP client shall wait for the response with timeout value of "NMS_OFFLINE_API_INITIAL_INTERVAL_MILLIS".

      (At first attempt retry count is 0).
    - If the response is not received, KilkariObd HTTP client shall check if number of retries is less than "NMS_OFFLINE_API_MAX_RETRIES", then invoke the Offline API again, compute the new timeout value = "NMS_OFFLINE_API_RETRY_MULTIPLIER" * (previous_timeout).
    - In case of failure, retry till max retries are attempted.

6.3 Service Provided to IVR

- KilkariObd exposes following APIs to IVR System:

- CDR File Notification: To notify the availability of CDR file for OBD Calls.
- FileProcessedStatus Notification: To notify processing completion of CDR Detail and summary files.

6.3.1 CDR File Notification

- Kilkari shall expose "CdrFileNotification API", detailed in section refer 4.2.6 in [2], to allow IVR to notify NMS that Call Data Record File is ready for processing.
- On invocation of this API KilkariObd shall
  - Validate the parameters of the API including the the name format of the CDR files.
  - Update the status in ***OutboundCallFlow*** entity, corresponding to the ObdFileName (derived from CDRFiles name) as "**CDRFileNotificationReceived**".

6.3.2 FileProcessedStatus Notification

- KilkariObd shall expose "fileProcessdStatus Notification API", detailed in section refer 4.2.7 in [2], to allow IVR to notify NMS that OBD target file submitted has been processed.
- KilkariObd shall validate the parameters of the API:
  - If validation passes and failure reason is not present in the API then KilkariObd shall
    - Update the status in ***OutboundCallFlow*** entity, corresponding to the ObdFileName as ***"*OutboundCallRequestFileProcessedAtIvr".**
    - Delete all the records from ***OutboundCallRequest.***
  - If validation passes and failure reason is present in this API, KilkariObd shall then
    - Export the CSV file from ***OutboundCallRequest*** to location ***'"***NMS_KK_OBD_FILE_LOCAL_PATH.

- Update the checksum & records count in ***OutboundCallFlow*** along with status as **"OutboundCallRequestFileProcessingFailedAtIvr".**
- Perform SCP of this OBD Csv file from "NMS_KK_OBD_FILE_LOCAL_PATH" to the location "NMS_OBD_FILE_PATH" on server identified by IP "NMS_OBD_FILE_SERVER_IP"
- Shall invoke TargetFileNotification API, refer 4.3.1 in [2], with "filename" , checksum and no. of records as present in the ***OutboundCallFlow*** record.

- In case validation fails, KilkariObd shall send failure response to IVR.

6.4 Assumption for Corner Cases:

- In case NMS system is down (for maintenance or otherwise) at the time when scheduled event was supposed to be received and comes up later i.e if one or more previous events is/are missed then:
  - The calls those were supposed to be scheduled during down time won't be scheduled in next fresh or retry base.

- This module is separated from Kilkari module in order to keep the business logic for OBD preparation and CDR processing, which is specific to the IVR vendor, as independent of subscription logic.
- Outbound call flow shall be as follow:
    - KilkariObd shall prepare the OBD target file for
        - Fresh calls for subscription having a weekly message scheduled for current date
        - The calls that have failed on the previous attempt and are eligible for retry on current date.
    - KilkariObd shall perform a secure copy of the OBD target file to a location that is accessible by IVR.
    - KilkariObd shall notify IVR System to pick and process the OBD Target file.
    - On successful processing of OBD Target file IVR shall notify the processing completion to KilkariObd.
    - Then after making the calls as per OBD target file, during social hours, IVR shall do a secure copy of CDR Summary and Detail files to the same location (at which OBD target file was kept by KilkariObd).
    - IVR shall notify creation of KilkariObd with details of CDR Summary and details.
    - KilkariObd shall start the process again for next day's date. KilkariObd shall use the information from this table to get the name of the CDR files to be used for OBD target preparation. This table shall also aid in monitoring the Outbound Call flow. The records in this table shall be used in generating Kilkari reports.if it is not already createdDelete all the records from OutboundCallRequest, to ensure that it doesn't contain the call requests from older date.

67 Master Location Data Module

6.17.1 Overview

- Master Location Data module is responsible for storing location information received via CSV upload into the NMS database.
- It provides a service interface to other modules for querying location data.
- Master Location Data module uses following MDS entities:
    - ***State***
    - ***District***
    - ***Taluka***
    - ***Village***
    - ***HealthBlock***
    - ***HealthFacility***
    - ***HealthSubFacility***
    - Refer section 10.1.1, 10.1.2 for more details on these entities

6.27.2 Events Consumed

Master Location Data module subscribes to receive events from MDS for the following topics:

- State CSV upload successful
- District CSV upload successful
- Taluka CSV upload successful
- Village CSV upload successful

- Health Block CSV upload successful
- Health Sub-Facility CSV upload successful
- Circle CSV upload successful
- Operator CSV upload successful
- Language Location Code CSV upload successful

The details of the corresponding CSV files is given in 11.2.7.2

6.2.17.2.1 Event Handling

- The event is raised by MDS, when the CSV file is uploaded successfully in the temporary table.
- The event contains the list of ids of successfully uploaded records and name of the user who has uploaded the file.
- Master Location Data module shall read the records from temporary table one by one for processing.
- Master Location Data module shall use the static methods provided by Utility module for CSV data validation and parsing.
- Additionally, since the location data is hierarchical in nature Master Location Data module verifies whether the parent(s) of the location entity exist in the database or not. E.g. the State corresponding to the *stateCode* in a District CSV record must exist in the database for the district record to be processed.
- If the validation passes and mapping is successful then, Master Location Data module shall create an entry in the corresponding permanent table.
- If validation of any CSV record fails or it's mapping to corresponding entity fails or record is rejected during processing, then Master Location Data module shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason using API detailed in 9.2.1.1.
- After the processing of all csv uploaded records, Master Location Data module shall update the CSV Upload summary in *BullkUploadStatus* entity using the API described in 9.2.1.2.

6.37.3 Services

Master Location Data module exposes the following services to other modules:

6.3.17.3.1 Location Data Service

- This service consists of the following APIs:

6.3.1.17.3.1.1 Validate Location

- Signature : *boolean validateLocation(Long stateId, Long districtId);*
- This API verifies whether the State and District for the given StateId and DistrictId exist in the database.
- API returns boolean response after validating stateId and districtId.

6.3.1.27.3.1.2 Get State by Code

- Signature: *State getStateByCode(Long stateCode);*
- This API returns the State record for the given State Census Code.

6.3.1.37.3.1.3 Get District by Code

- Signature: *District getDistrictByCode(Long stateId, Long districtCode);*
- This API returns the District record for the given s*tateId* (State table key) and District Census code.

6.3.1.47.3.1.4 Get Taluka by Code

- Signature : *Taluka getTalukaByCode(Long districtId, Long talukaCode);*

- This API returns the Taluka record for given *districtId* (District table key) and Taluka Census code.

6.3.1.57.3.1.5 Get Health Block by Code

- Signature : *HealthBlock getHealthBlockByCode(Long talukaId, Long healthBlockCode);*
- This API returns the HealthBlock record for given *talukaId* (Taluka table key) and HealthBlock Census code.

6.3.1.67.3.1.6 Get Health Facility by Code

- Signature : *HealthFacility getHealthFacilityByCode(Long healthBlockId, Long healthFacilityCode);*
- This API returns the Health Facility record for given *healthBlockId* (HealthBlock table key) and Health Facility Census Code

6.3.1.77.3.1.7 Get Health Sub-Facility by Code

- Signature : *HealthSubFacility getHealthSubFacilityByCode(Long healthFacilityId, Long healthSubFacilityCode);*
- This API returns the Health Sub-Facility record for given *healthFacilityId* (HealthFacility table key) and health Sub-Facility Census Code.

6.3.1.87.3.1.8 Get Village by Code

- Signature : *Village getVillageByCode(Long talukaId, Long villageCode);*
- This API returns the Village record for given the *talukaId* (Taluka table key) and Village Census Code.

6.3.1.97.3.1.9 Get MA Capping by Code

- Signature : *Integer getMaCappingByCode(Long stateCode);*
- This API returns the State Level capping value for MA health service given the state identifier. The capping value is internally retrieved from the State table.

6.3.1.107.3.1.10 Get MK Capping by Code

- Signature : *Integer getMkCappingByCode(Long stateCode);*
- This API returns the State Level capping value for MK health service given the state identifier. The capping value is internally retrieved from the State table.

7.3.1.11 Get MK Service Deployed by Code

- Signature : *Boolean getMkServiceDeployedByCode(Long stateCode);*
- This API returns the State Level deployment status of MK services from state table.
- State table needs to be updated to change the deployment status of service.

7.3.1.12 Get MA Service Deployed by Code

- Signature : *Boolean getMkServiceDeployedByCode(Long stateCode);*
- This API returns the State Level deployment status of MA services from state table.
- State table needs to be updated to change the deployment status of service.

7.3.1.13 Get KK Service Deployed by Code

- Signature : *Boolean getMkServiceDeployedByCode(Long stateCode);*

- This API returns the State Level deployment status of KK services from state table.
- State table needs to be updated to change the deployment status of service.

7.3.1.14 Get WhiteListing Status by Code

- Signature : *Boolean getWhiteListingEnableStatusByCode(Long stateCode);*
- This API returns the enable status of whitelisting on a state level from state table.
- State table needs to be updated to change the whiteList enablement for users in the state.

~~6.3.3~~7.3.2 Language Location Code Service

This service provides LLC values to other modules via the following APIs:

6.3.3.17.3.2.1 Get Record by Location

- Signature : *LanguageLocationCode getRecordByLocationCode(Long stateCode, Long districtCode)*
- This API returns the language location code record (refer 10.1.2) for the given stateCode and districtCode
- Returns “null” if record is not found.

6.3.3.27.3.2.2 Get Record By Circle Code And LangLocCode

- Signature : *LanguageLocationCode getRecordByCircleCodeAndLangLocCode(String circleCode, Integer langLocCode)*

- This API returns the language location code record for the given *circleCode* and *languageLocationCode*.
- Returns “null” if record is not found.

6.3.3.37.3.2.3 Get LanguageLocationCode By Location

- Signature : *Integer getLanguageLocationCodeByLocationCode(Long stateCode, Long districtCode);*
- This API returns the value of language location code for the given location (state, district).
- Returns “null” if record is not found.

6.3.3.47.3.2.4 Get LanguageLocationCode By Circle Code

- Signature : *Integer getLanguageLocationCodeByCircleCode(String circleCode);*
- This API checks whether for the given circle code, there exists a unique Language Location Code.
- “null” is returned if there is no unique LanguageLocationCode or if there exists no entry in the LanguageLocationCode table for the given circleCode,
- Else it returns the value of language location code for the given circle code.

6.3.3.57.3.2.5 Get Default LanguageLocationCode By CircleCode

- Signature : *Integer getDefaultLanguageLocationCodeByCircleCode(String circleCode);*
- This API returns the value of default language location code for the given circle code.

78 Front Line Worker Module

7.18.1 Overview

- Front Line Worker module is responsible for storing FLW information received via CSV upload into the NMS database.
- It provides a service interface to other modules for querying and for updating FLW data.
- Front Line Worker module module uses following MDS entities:
    - ***FLW Data*** (refer 10.1.3)

7.28.2 Events Consumed

Front Line Worker module subscribes to receive success events for the following topics:

7.2.18.2.1 FLW CSV Upload Successful

- This event is raised by MDS, when FrontLineWorker MCTS data CSV file is uploaded successfully in FrontLineWorker CSV table.
- The event contains the list of ids of successfully uploaded records and name of the user who has uploaded the file.
- Front Line Worker module shall read the records from FrontLineWorker CSV table one by one for processing.
- Front Line Worker module shall use the static methods provided by Utility module for CSV data validation and parsing (refer 9.2.2), to validate the CSV record and map it to FrontLineWorker entity (validation rules are mentioned in 11.2.8.3).
- Thereafter, it is checked whether the FLW corresponding to the CSV record data exists in the NMS database. Refer 8.2.1.1
- If the FLW record exists in the database and its state is **INVALID-FLW**, then the CSV record is rejected since modification is not allowed in this case

- If no FLW record exists in the database then a new record is created in the FrontLineWorker table. The FLW state in this new record shall be set to **INACTIVE-FLW** by default.
- The fields in the FLW record in the database (newly created / already existing) shall be set / updated as per the data present in the CSV record.
- FLW lookup for msisdn in entity WhiteListUsers (Refer 10.1.4).
- The FLW state in the new / existing record shall be updated on the basis of the *isValid* field of the CSV record as described in 8.2.1.2
- If Validation of FrontLineWorker Csv record fails or it's mapping to FrontLineWorker entity fails or record is rejected during processing, then FrontLineWorker Module shall create an error log entry in *BulkUploadError* entity with erroneous record details and error reason using API detailed in 9.2.1.1.
- After the processing of all CSV uploaded records, FrontLineWorker module shall update the csv Upload summary in *BullkUploadStatus* entity. FrontLineWorker module shall use the API detailed in 9.2.1.2.
- Format of CSV file is specified in section 11.2.7.4.1

7.2.1.18.2.1.1 Check Existence of the FLW

- FLW record is retrieved from the database as per the following logic:
    - If the NMS FLW Identifier is available in the CSV, then the FLW record corresponding to the NMS FLW Id is selected.
    - Else if the NMS FLW Identifier is not available in the CSV:
        - The FLW record corresponding to the given MCTS FLW Id and the state code is searched and, if found, selected.
        - Else, the FLW record corresponding to the given MSISDN is searched and, if found, selected.

7.2.1.28.2.1.2 Update of FLW State

- If the state of the FLW record in the database is **INACTIVE-FLW** and**:**
    - If the *isValid* in the CSV record is equal to FALSE, then the state of the existing record is updated to **INVALID-FLW**

- Else if the state of the FLW record in the database is **ANONYMOUS-FLW** and**:**
  - If the *isValid* is not present in the CSV record or is equal to TRUE, then the state of the existing record is updated to **ACTIVE-FLW**
  - If the *isValid* in the CSV record is equal to FALSE, then the state of the existing record is updated to **INVALID-FLW**
- Else, if the state of the FLW record in the database is **ACTIVE-FLW** and**:**
  - If the *isValid* in the CSV record is equal to FALSE, then the state of the existing record is updated to **INVALID-FLW**

8.2.2 FLW Whitelist CSV Upload Successful

- FLW handles the Csv upload of FLW White List Data (refer 11.2.7.6.3) and ~~based on msisdn,~~ it will update the ~~IsInWhiteList parameter in FLW table~~table WhiteListUsers.
~~If Msisdn is not present or is INVALID, FLW rejects the record.~~

Formatted: Don't add space between paragraphs of the same style, Bulleted + Level: 1 + Aligned at: 0 cm + Indent at: 0,63 cm

~~7.4~~8.3 Services

FrontLineWorker module exposes the following service APIs to other modules:-

7.4.1~~8~~.3.1 Process User Details

- Signature : *UserProfile processUserDetails(String msisdn, String circleCode, String operatorCode, ServicesUsingFrontLineWorker service) throws DataValidationException;*
- Process User Detail API is used by MA and MK modules to serve the Get User Detail request coming from IVR.
- Circle, Operator, MSISDN and Service Type (MA or MK) are the input parameters.
- The *UserProfile* object returned by this API contains the following information:

- Whether FLW record for the given *msisdn* exists or not
- The NMS generated Identifier of the FLW
- The Language Location Code for the given circle (if unique, else the default LLC for the circle)
- Circle
- State level capping value for the given *service*

- This API does the following validation and throws a DataValidationException exception in case the validation fails.
  - Mandatory parameters are present (Circle, Operator, MSISDN and Service Type)
  - *Operator* exists in NMS system
- FrontLineWorker module retrieves the FrontLineWorker record for the given *msisdn:*
  - If there are multiple FLW records for the given *msisdn*, then the FLW record with state other than **INVALID-FLW** is considered. If there is no FLW record with state other than **INVALID-FLW** then the first record in the list is considered.
- The FLW User Profile information is determined on the basis of the FLW State as follows:
  - **INACTIVE-FLW:**
    - FLW from the state code checks whether state level service deployment is enabled or not in State table. If enable, it will serve the request otherwise it will throw an Exception ServiceNotDeployedException.
    - FLW from the stateCode checks whether state level white listing is enabled or not in state table. If enabled and msisdn exist in WhiteListUsers table , then it will service the request otherwise it will throw an exception FlwNotInWhiteListException.

- If Whitelisting is disable, it will serve the request.
- Refer Masterdata section 7.3.1.11,□,□ and □ for APIs used to process.
- The Language Location Code value, for the state and district code values in the FLW record, is queried (using Master Location Data module Language Location service API, refer 7.3.2.3)
- If found, the LLC value is stored in the FLW record and, additionally, the FLW state is updated to **ACTIVE-FLW**
- If LLC is not found, a *DataValidationException* exception is raised.

- **ACTIVE-FLW:**
  - FLW from the state code checks whether state level service deployment is enabled or not in state table.. If enabled, it will serve the request otherwise it will throw an Exception ServiceNotDeployedException.
  - FLW from the stateCode checks whether state level white listing is enabled or not in state table. If enabled and msisdn exist in WhiteListUsers table then it will service the request otherwise it will throw an exception FlwNotInWhiteListException.
  - If Whitelisting is disable, it will serve the request.
  - Refer Masterdata section 7.3.1.11,□,□ and □ for APIs used to process.
  - If LLC exists in the retrieved FrontLineWorker record, it is returned.
  - Else if LLC does not exist, and the default LLC exists, the default LLC is returned.
  - If neither LLC nor the default LLC is present in the FrontLineWorker record, a DataValidationException exception is raised.
- **ANONYMOUS-FLW:**

- FLW from the circle identifies the state code and if it is unique it checks whether state level service deployment is enabled or not in state table. If enable, it will serve the request otherwise it will throw an Exception ServiceNotDeployedException.
- If stateCode is not unique and if in any state service is deployed, it will serve the request otherwise it will throw an Exception ServiceNotDeployedException.
- FLW from the circle identifies the state code and if it is unique it checks whether state level white listing is enable dor not. If enable and msisdn exist in WhiteListUsers table then it will service the request otherwise it will throw an exception FlwNotInWhiteListException.If Whitelisting is disable, it will serve the request.
- If stateCode is not unique and if all the state is having white listing enabled then and msisdn exist in WhiteListUsers table , then it will service the request otherwise it will throw an exception
- In case in any state whiteListing is disabled, FLW serve the request.
- Refer Masterdata section 7.3.1.11,□,□ and □ for Api used to process.
- If LLC exists in the retrieved FrontLineWorker record, it is returned.
- Else if LLC does not exist, and the default LLC exists, the default LLC is returned.
- If default LLC does not exist, LLC is queried based on the given circle code using Master Location Data module Language Location service API (refer 7.3.2.4)
- In case there are multiple states in the given circle, the default language location code of the circle is returned (queried using Master Location Data module Language Location service API described in 7.3.2.5

- State-wise Capping Value: In each of these FLW states, the State-wise capping value is retrieved from the State table on the basis of the state code (present in the FLW record).

- In case FLW state is equal to **INVALID-FLW** or there is no FrontLineWorker record for the given *msisdn*, then Front Line Worker module:
    - Creates new entry in FrontLineWorker table.
    - FLW module, from the circle, identifies the state code and if it is unique it checks whether state level service deployment is enabledd or not. If enable, it will serve the request otherwise it will throw an Exception ServiceNotDeployedException.
    - If stateCode is not unique and if in any state service is deployed, it will serve the request otherwise it will throw an Exception ServiceNotDeployedException.
    - FLW module, from the circle, identifies the state code and if it is unique it checks whether state level white listing is enabled or not. If enabled and msisdn exist in WhiteListUsers table, then it will service the request otherwise it will throw an exception FlwNotInWhiteListException.If Whitelisting is disabled, it will serve the request.
    - If stateCode is not unique and if all the state is having white listing enabled and msisdn exist in WhiteListUsers table then it will service the request otherwise it will throw an exception
    - In case in any state whiteListing is disable, FLW serve the request.
    - Refer Masterdata section 7.3.1.11, , and for Api used to process.
    - LLC is queried based on the given circle code using Master Location Data module Language Location service API (refer 7.3.2.4)
    - In case there are multiple states in the given circle, the default language location code of the circle is returned (queried using Master Location Data module Language Location service API described in 7.3.2.5
    - The FLW State is set to **ANONYMOUS-FLW**

Update Language Location Code given the MSISDN

- Signature : *void updateLanguageLocationCodeFromMsisdn(Integer languageLocationCode, String msisdn) throws DataValidationException;*
- This API takes MSISDN and Language Location Code as Input
- In case no record corresponding to the given language location code exists, a DataValidationException error is raised.
- If no FLW record exists for the given MSISDN a DataValidationException is thrown.
- If 'language location code' as well as the front line worker exist, then front line worker record is updated with the new language location code.

7.4.48.3.3 Validate Operator

- Signature: *void validateOperator(String operatorCode) throws DataValidationException;*
- This API checks whether the record corresponding to the given operator code exists or not
- If it does not, then a *DataValidationException* is raised.

89 Utility Module

8.19.1 Overview

- This module shall be used for providing the common Utility services that shall be used by all other modules, if needed.
- Utility module uses following MDS entities:
    - ***BulkUploadError***: This entity shall be used to store the error and erroneous record details
    - ***BulkUploadStatus***: This entity shall be used to store the Bulk upload status summary.
    - Refer section 10.1.8 for DB schema of the above entities.
- Following are the services provided in Utility module:
    - ***Bulk Upload Log service***
    - **CSV Data Validation and Parsing** service

8.29.2 Services

8.2.19.2.1 Bulk Upload Log Service

8.2.1.19.2.1.1 Write Bulk Upload Error

- Signature : *public void writeBulkUploadErrLog(BulkUploadError bulkUploadError)*
- This API is used to write CSV record processing error information to the *BulkUploadError* table:
    - Name of the Uploaded CSV File

- Time of the upload
- Type of records uploaded (State, District, Circle etc.)
- Erroneous Record detail
- Error Category (Data Invalid, Mandatory Information Missing etc.)
- Error Description

8.2.1.29.2.1.2 Write Bulk Upload Status Summary

- Signature : *public void writeBulkUploadProcessingSummary(BulkUploadStatus bulkUploadStatus)*
- This API is used to write the bulk upload result summary in NMS database.
- Following information is written to *BulkUploadStatus* entity:
  - Name of the user who uploaded the file
  - Name of the Uploaded CSV File
  - Time of the upload
  - Number of failed record
  - Number of successful record.

8.2.29.2.2 CSV Data Validation and Parsing

8.2.2.19.2.2.1 Validate and Parse data to (String/Integer/Long/Date/Boolean):

- Utility module provides a set of static methods for CSV field validation and parsing the value of the field to relevant data type.
- The input to the Static methods is the String value, which is the value of the field received in CSV Upload.
- These APIs also take a Boolean flag as input, which indicates whether the field is mandatory.
- If field is mandatory and its value is null or blank or empty then these APIs shall raise *DataValidationException* (with reason “mandatory parameter missing”).

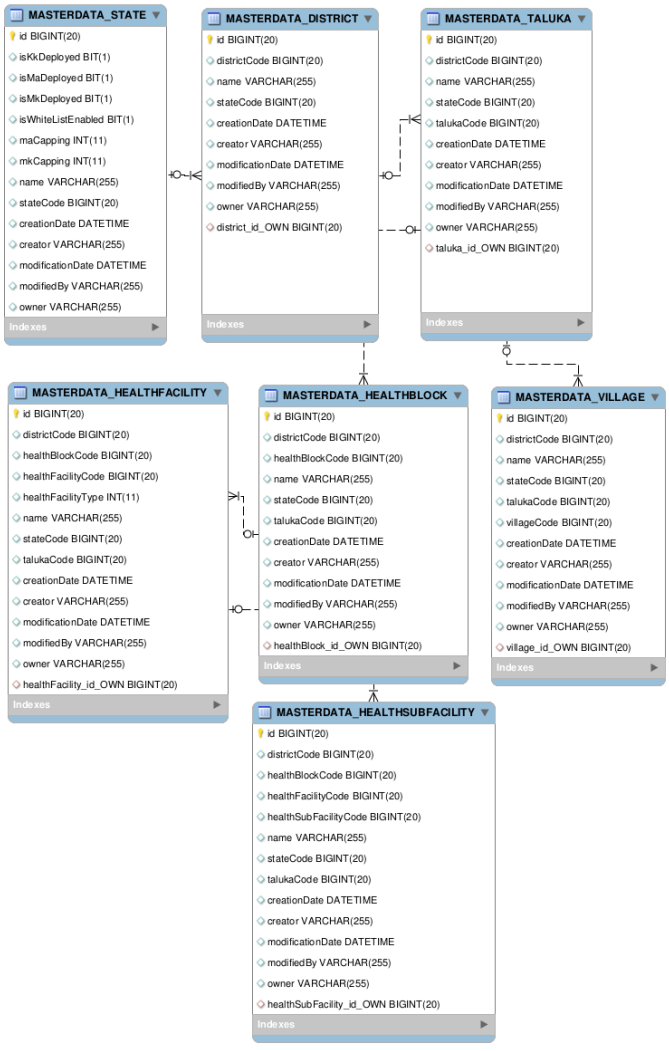
- *DataValidationException* (with reason “invalid value”) is also raised if the conversion of String to the relevant data type fails.

910 Database Design

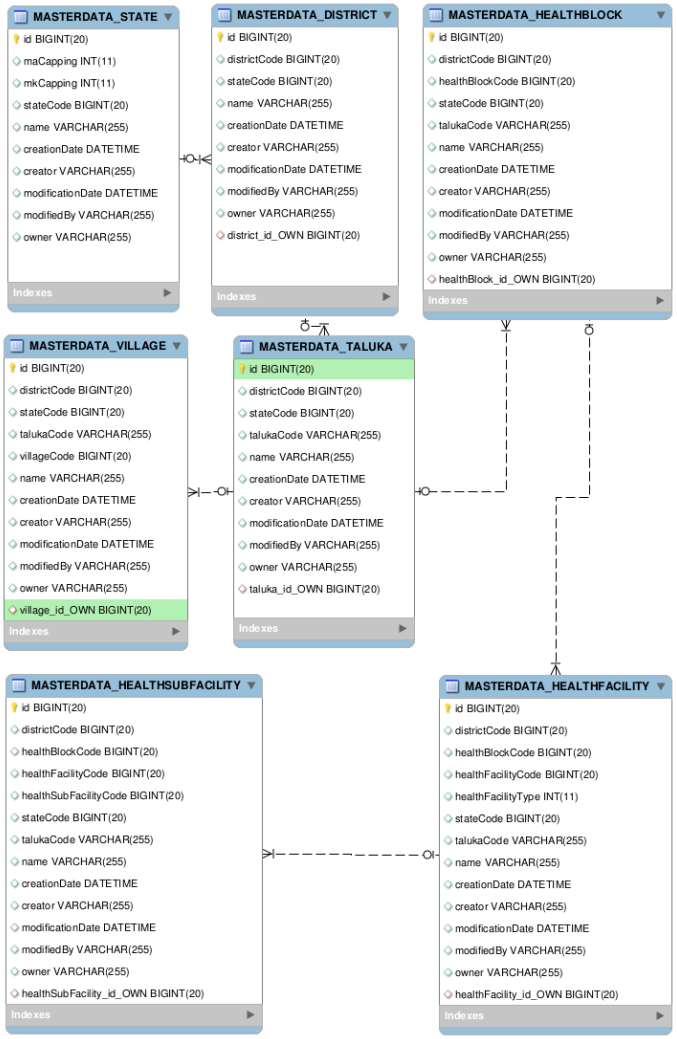
9.110.1 MA, MK, Kilkari Service Database

9.1.110.1.1 Master Location Data

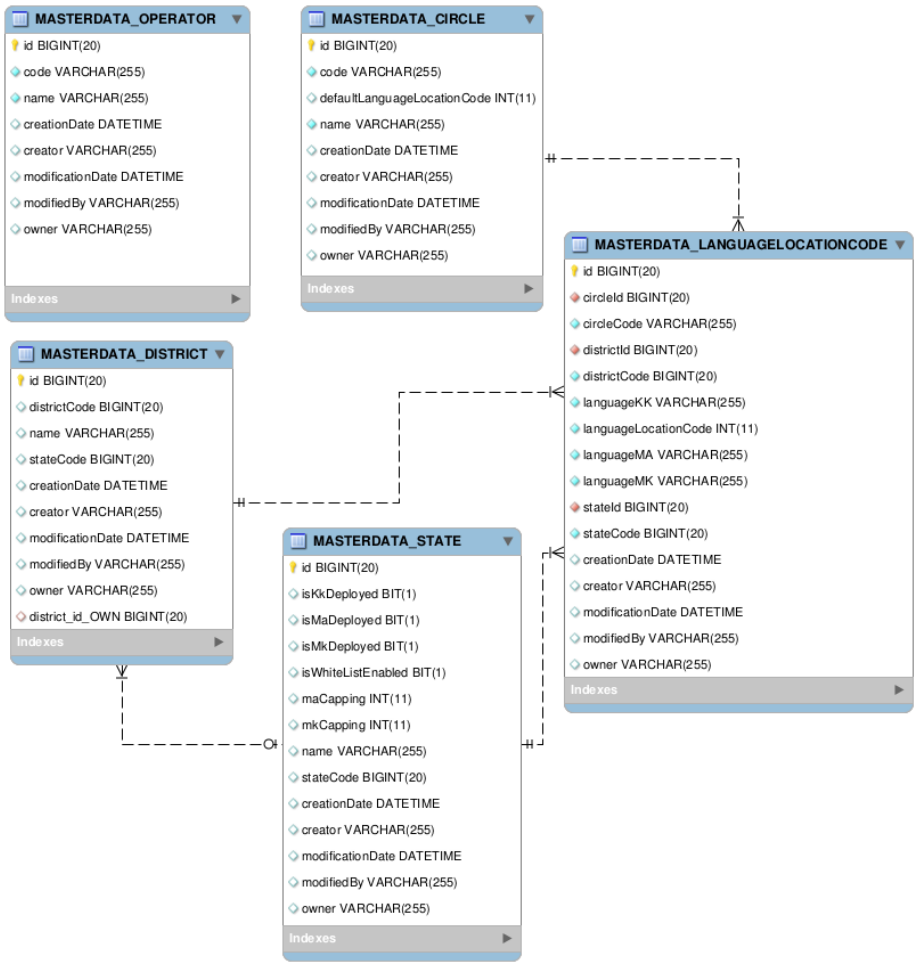
The following tables are used to store the location data entered into the NMS system via CSV upload:



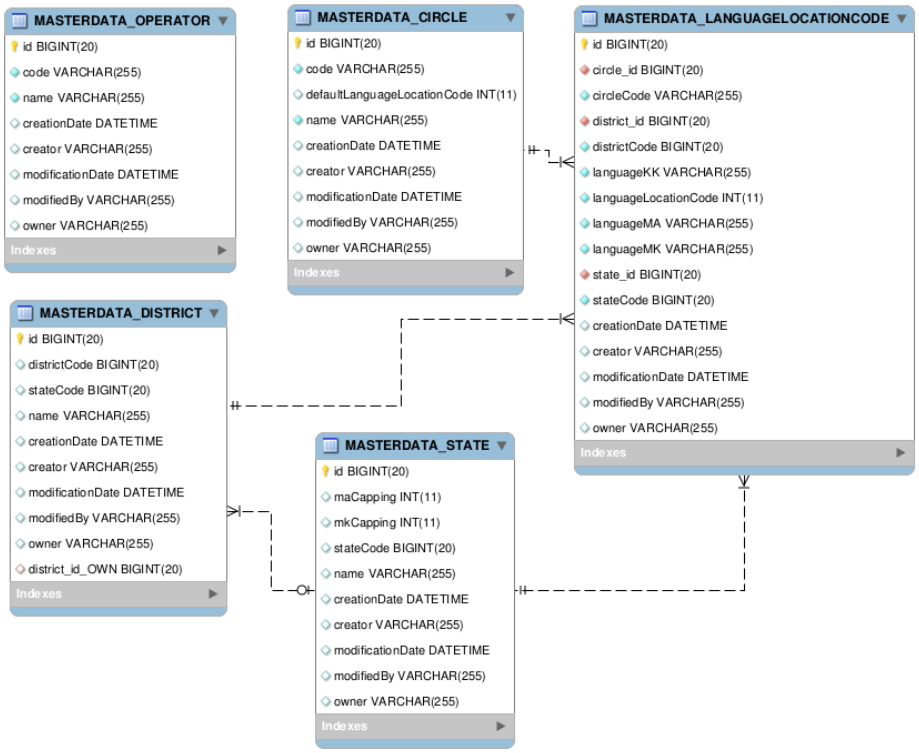
Formatted: Font: (Default) Arial


MASTERDATA_STATE
id BIGINT(20)
maCapping INT(11)
mkCapping INT(11)
stateCode BIGINT(20)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes
MASTERDATA_DISTRICT
id BIGINT(20)
districtCode BIGINT(20)
stateCode BIGINT(20)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
district_id_OWN BIGINT(20)
Indexes
MASTERDATA_HEALTHBLOCK
id BIGINT(20)
districtCode BIGINT(20)
healthBlockCode BIGINT(20)
stateCode BIGINT(20)
talukaCode VARCHAR(255)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
healthBlock_id_OWN BIGINT(20)
Indexes
MASTERDATA_VILLAGE
id BIGINT(20)
districtCode BIGINT(20)
stateCode BIGINT(20)
talukaCode VARCHAR(255)
villageCode BIGINT(20)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
village_id_OWN BIGINT(20)
Indexes
MASTERDATA_TALUKA
id BIGINT(20)
districtCode BIGINT(20)
stateCode BIGINT(20)
talukaCode VARCHAR(255)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
taluka_id_OWN BIGINT(20)
Indexes
MASTERDATA_HEALTHSUBFACILITY
id BIGINT(20)
districtCode BIGINT(20)
healthBlockCode BIGINT(20)
healthFacilityCode BIGINT(20)
healthSubFacilityCode BIGINT(20)
stateCode BIGINT(20)
talukaCode VARCHAR(255)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
healthSubFacility_id_OWN BIGINT(20)
Indexes
MASTERDATA_HEALTHFACILITY
id BIGINT(20)
districtCode BIGINT(20)
healthBlockCode BIGINT(20)
healthFacilityCode BIGINT(20)
healthFacilityType INT(11)
stateCode BIGINT(20)
talukaCode VARCHAR(255)
name VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
healthFacility_id_OWN BIGINT(20)
Indexes

9.1.210.1.2 Location – Circle – Language Mapping Data



Formatted: Font: (Default) Arial



- These tables store the mapping location to language code mapping information

- The defaultLanaguageLocationCode in Circle table is the Circle-wise default. This used to determine the Language to play the language determination prompt to caller for the case when IVR is able to determine the circle

9.1.310.1.3 FLW Data

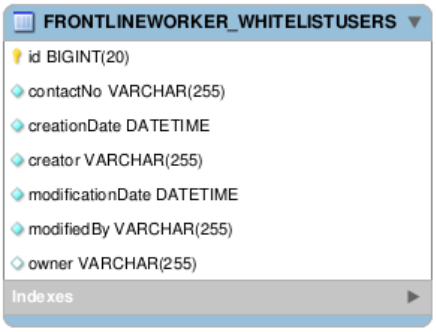
- The FLW Data table stores the FLW location and contact of all FLWs:

Formatted: Font: (Default) Arial, 12 pt

10.1.4 WhiteListUsers

The table is used to capture the list of whitelist users in NMS system.

FLW uses this table to update isInWhiteList flag in its FrontLineWorker table.

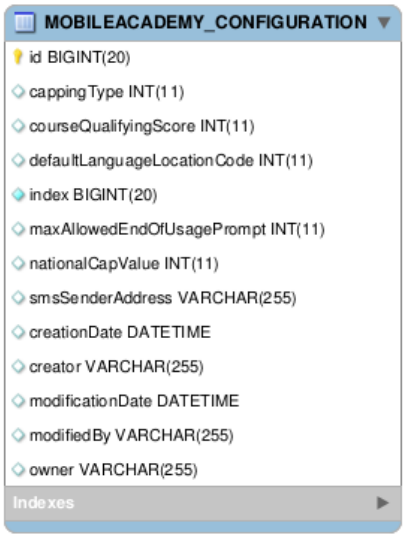


Formatted: Font: (Default) Arial

Formatted: Font: (Default) Arial, 12 pt

Mobile Academy Data

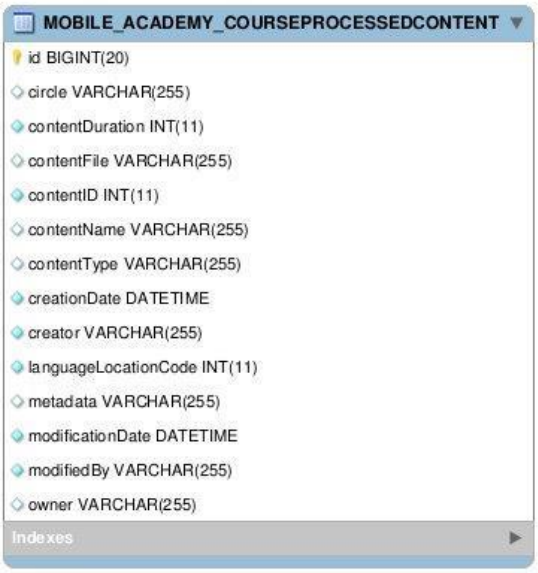
9.1.6.110.1.5.1 MA Configuration Data Table



- This table stores the configurable parameters that are needed by Mobile Academy service. Refer section 11.1.2.3.1 for details on MA configuration parameters.
- This table shall always have only one record.
- The configuration parameters shall be initialized with default values.
- User shall be able to modify the configuration parameter values via MDS UI.

9.1.6.210.1.5.2 MA Content Data Table

MA Content data is uploaded via CSV upload procedure (refer 11.2, 3.2.1) to the following table:

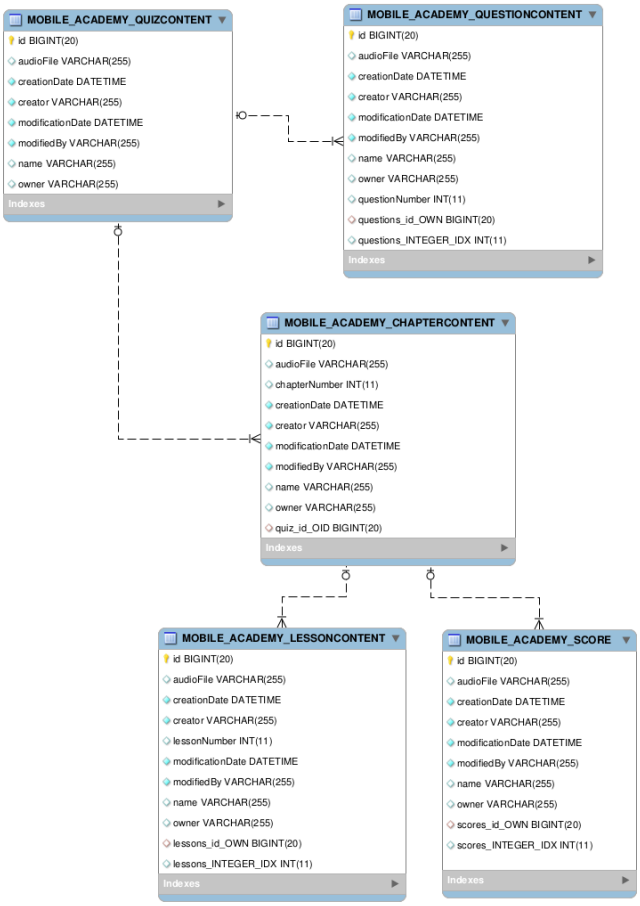


- This table shall contain Location-wise MA content information (audio file names, duration, etc.).
- The data from this table is used for generating MA Content Usage reports.

9.1.6.310.1.5.3 MA Course Content Data

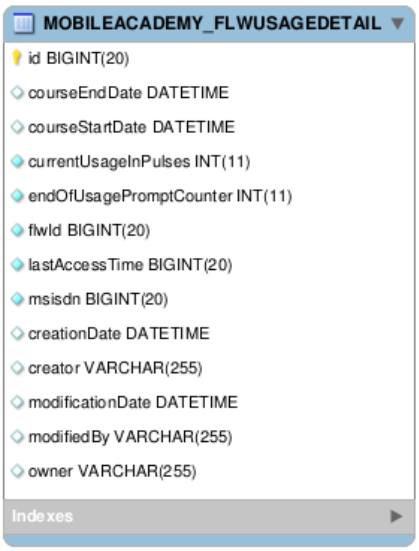
- The Mobile Academy course shall be stored in mTraining tables.
- Besides these tables, some additional tables are required for storing the content information for different course entities viz. chapter, lesson, quiz and question.
- These tables shall contain the MA Chapter, Lesson, Quiz content information. Data in these tables along with the data in mTraining tables shall be used to create the MA Course

Structure JSON that shall be shared with IVR. The schema for content tables is given

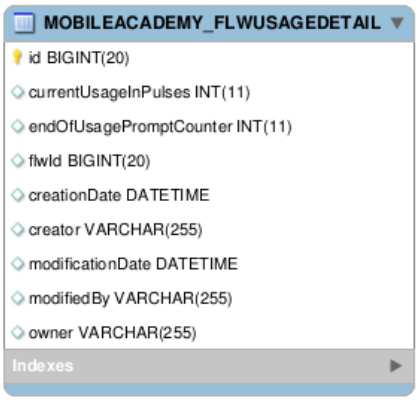

MOBILE_ACADEMY_QUIZCONTENT
id BIGINT(20)
audioFile VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
name VARCHAR(255)
owner VARCHAR(255)
Indexes
MOBILE_ACADEMY_QUESTIONCONTENT
id BIGINT(20)
audioFile VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
name VARCHAR(255)
owner VARCHAR(255)
questionNumber INT(11)
questions_id_OWN BIGINT(20)
questions_INTEGER_IDX INT(11)
Indexes
MOBILE_ACADEMY_CHAPTERCONTENT
id BIGINT(20)
audioFile VARCHAR(255)
chapterNumber INT(11)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
name VARCHAR(255)
owner VARCHAR(255)
quiz_id_OID BIGINT(20)
Indexes
MOBILE_ACADEMY_LESSONCONTENT
id BIGINT(20)
audioFile VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
lessonNumber INT(11)
modificationDate DATETIME
modifiedBy VARCHAR(255)
name VARCHAR(255)
owner VARCHAR(255)
lessons_id_OWN BIGINT(20)
lessons_INTEGER_IDX INT(11)
Indexes
MOBILE_ACADEMY_SCORE
id BIGINT(20)
audioFile VARCHAR(255)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
name VARCHAR(255)
owner VARCHAR(255)
scores_id_OWN BIGINT(20)
scores_INTEGER_IDX INT(11)
Indexes

below:

9.1.6.410.1.5.4 FrontLineWorker Usage Table



Formatted: Font: (Default) Arial

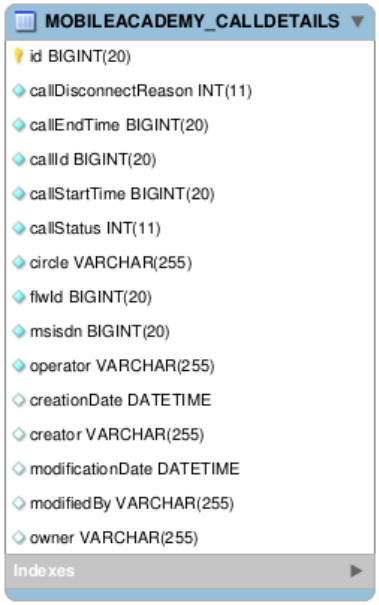


Formatted: Font: (Default) Arial

This table shall be updated while saving call details for an FLW. It shall contain the following information for an FLW:

- MA current service usage (in pulses)
- No. of times 'End of Usage' prompt has been played to the user.
- Time when MA service was last accessed by an FLW.

10.1.5.5 Call Details Table



Formatted: Font: (Default) Arial

- This table stores details regarding each call made by an FLW to MA.
- For detailed description of each field in the table, refer 3.3.7
- This will be further used for reporting purpose.
-

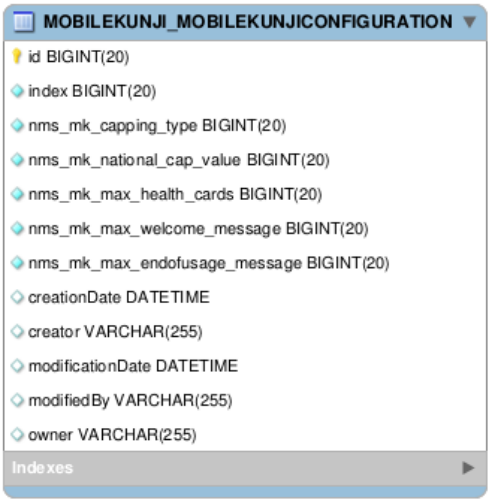
10.1.5.6 Content Logs table

- This table stores logs regarding content files being listened to by an FLW during the call.
- For detailed description of each field in the table, refer 3.3.7
- This will be further used for reporting purpose.

Formatted: Font: (Default) Arial

~~9.1.7~~10.1.6 Mobile Kunji Data

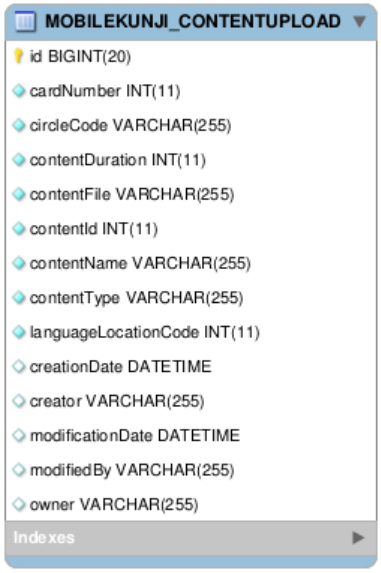
9.1.7.110.1.6.1 MK Configuration Data Table



- This table shall always have only one record.
- The configuration parameters shall be initialized with default values.
- User shall modify the configuration parameter values using MDS UI.

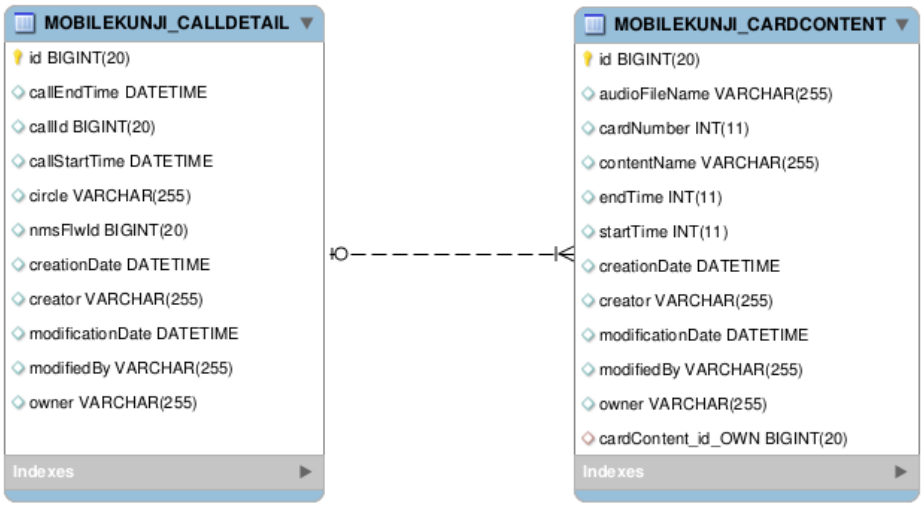
9.1.7.210.1.6.2 MK Content Data Table

MK Content data is uploaded via CSV upload procedure (refer 11.2~~11.2~~10.2) to the following table:



9.1.7.310.1.6.3 Call Detail Table

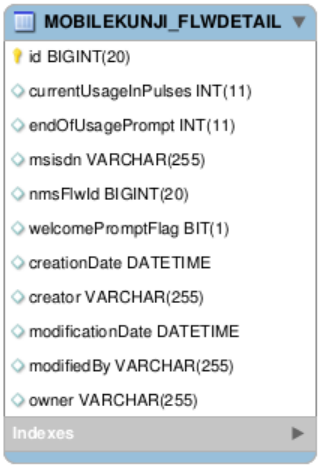
- MK call detail data is saved in the call detail table.
- The Card Content table stores MK audio file related information which is required for reporting purpose.



9.1.7.410.1.6.4 FLW Detail Table

This table shall be updated while saving call details for an FLW. It shall contain the following information for an FLW:

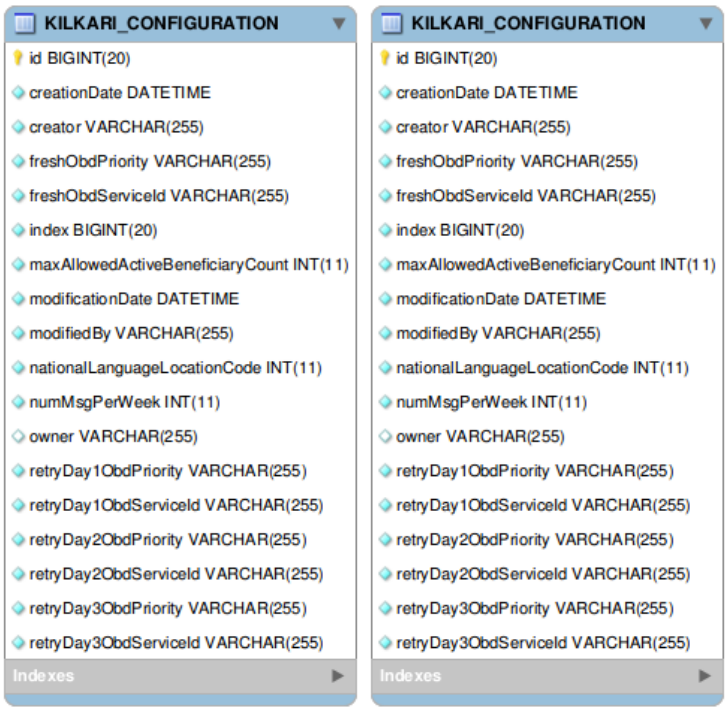
- MK current service usage (in pulses)
- No. of times 'End of Usage' prompt has been played to the user.


MOBILEKUNJI_FLWDETAIL
id BIGINT(20)
currentUsageInPulses INT(11)
endOfUsagePrompt INT(11)
msisdn VARCHAR(255)
nmsFlwId BIGINT(20)
welcomePromptFlag BIT(1)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes

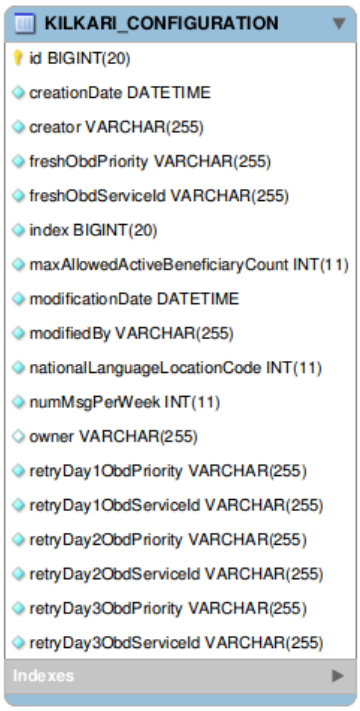
9.1.810.1.7 Kilkari Data

9.1.8.110.1.7.1 Kilkari Configuration Data

Kilkari Service configuration parameters are stored in the following table (refer 11.1.2.3.311.1.2.3.310.1.2.3.3):



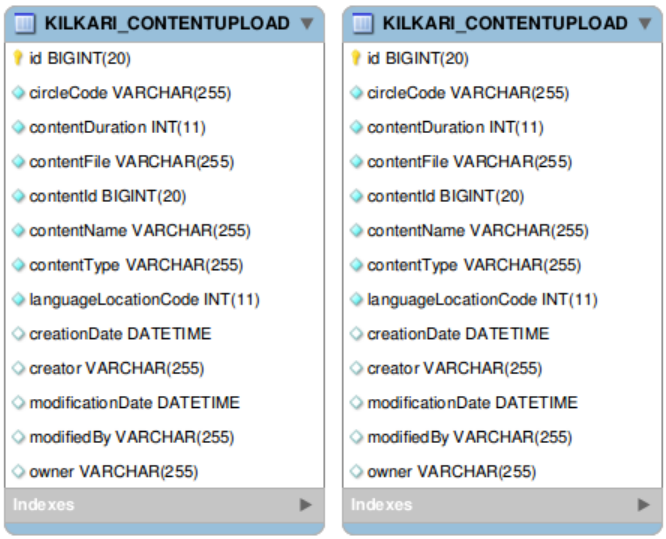
Formatted: Font: (Default) Arial



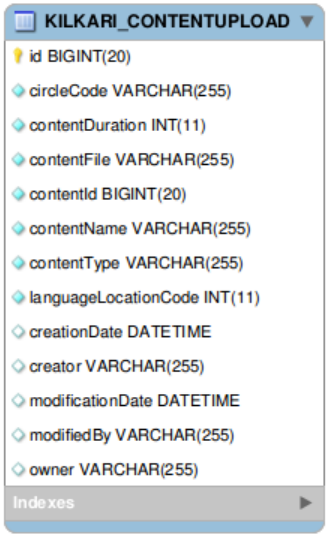
- This table shall always have only one record.
- The configuration parameters shall be initialized with default values.
- User shall modify the configuration parameter values using MDS UI.

9.1.8.210.1.7.2 Kilkari Content Data Table

Kilkari Content data is uploaded via CSV upload procedure (refer 11.211.210.2) to the following table:



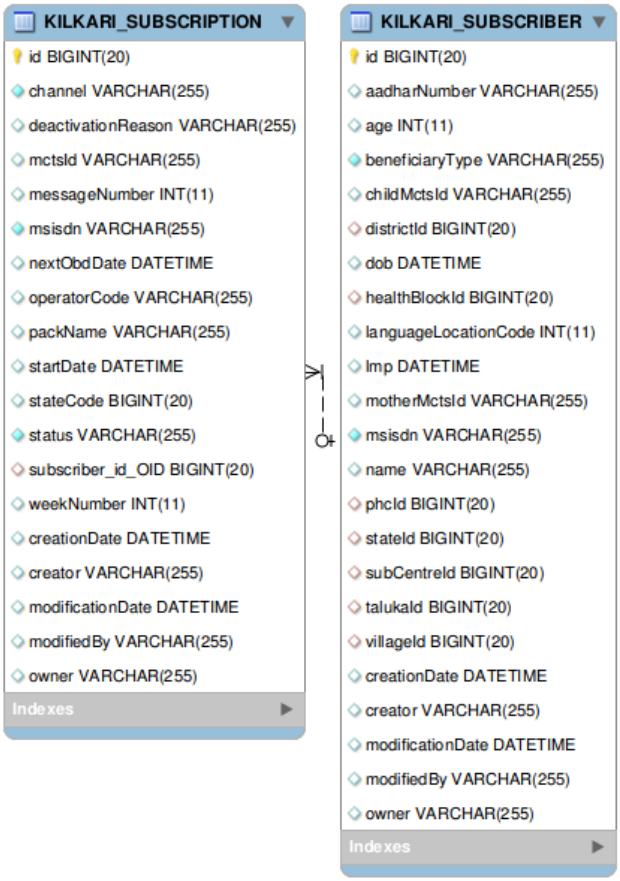
Formatted: Font: (Default) Arial



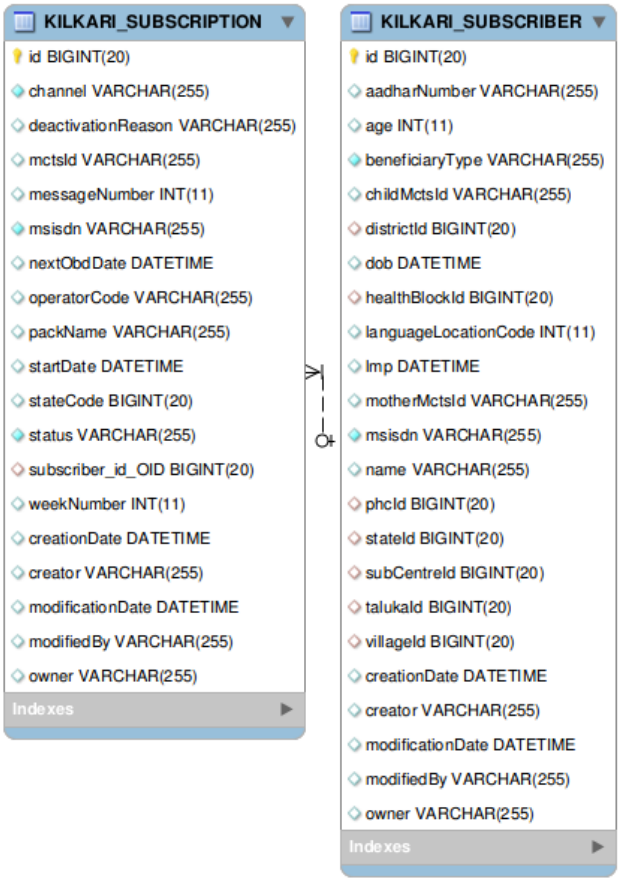
- This table shall store the mapping of content file name with Circle and languageLocationCode.
- Kilkari shall use the information from this table to send the content information to IVR.
- Kilkari shall use the content duration stored in this table to compute the percentage of the content played, for reporting purpose.

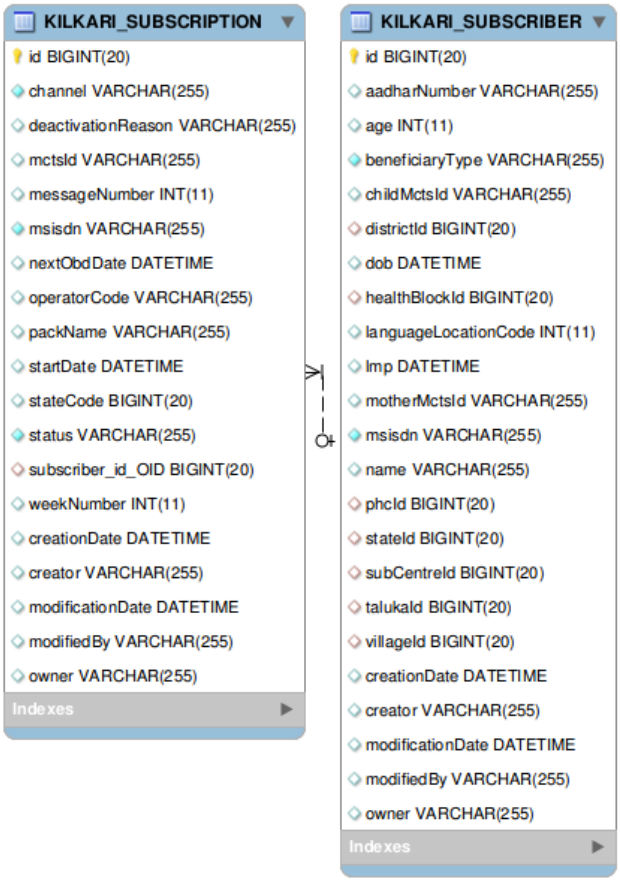
9.1.8.310.1.7.3 Kilkari Subscribers and Subscription Data

Kilkari Subscribers and Subscription Data is stored in the following table:-

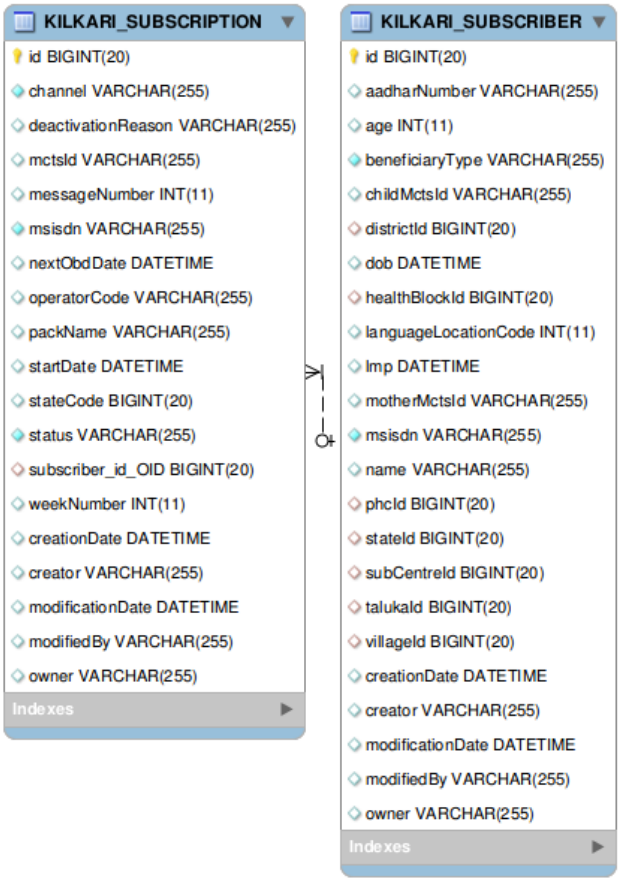


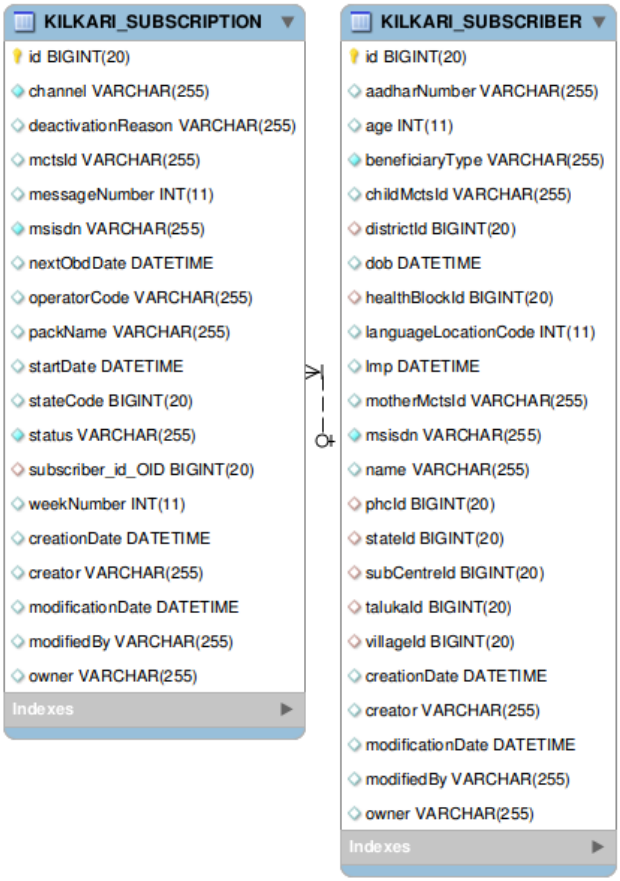
Formatted: Font: (Default) Arial, 13 pt, Bold, Font color: Custom Color(RGB(79;129;189))


KILKARI_SUBSCRIPTION
id BIGINT(20)
channel VARCHAR(255)
deactivationReason VARCHAR(255)
mctsId VARCHAR(255)
messageNumber INT(11)
msisdn VARCHAR(255)
nextObdDate DATETIME
operatorCode VARCHAR(255)
packName VARCHAR(255)
startDate DATETIME
stateCode BIGINT(20)
status VARCHAR(255)
subscriber_id_OID BIGINT(20)
weekNumber INT(11)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes
KILKARI_SUBSCRIBER
id BIGINT(20)
aadharNumber VARCHAR(255)
age INT(11)
beneficiaryType VARCHAR(255)
childMctsId VARCHAR(255)
districtId BIGINT(20)
dob DATETIME
healthBlockId BIGINT(20)
languageLocationCode INT(11)
lmp DATETIME
motherMctsId VARCHAR(255)
msisdn VARCHAR(255)
name VARCHAR(255)
phcId BIGINT(20)
stateId BIGINT(20)
subCentreId BIGINT(20)
talukaId BIGINT(20)
villageId BIGINT(20)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes

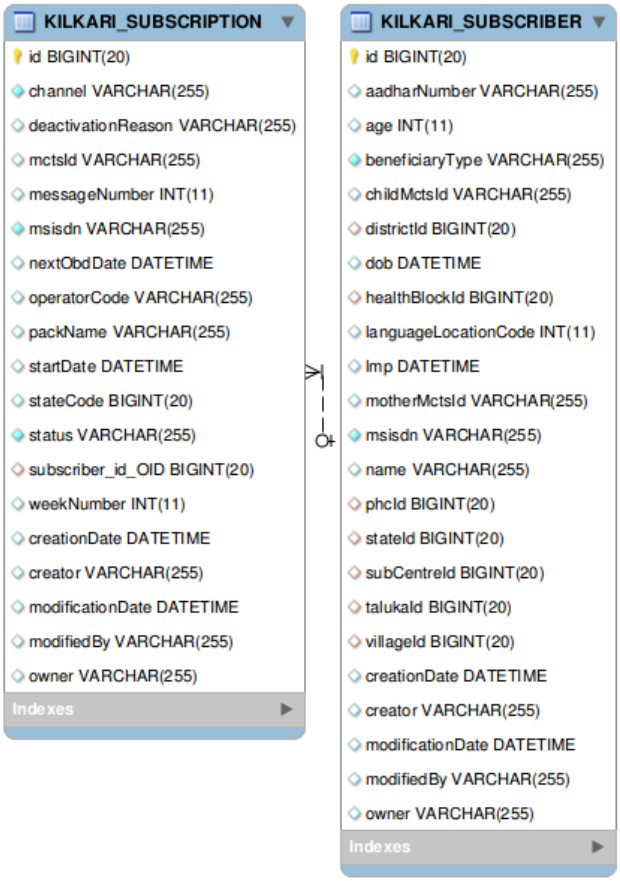


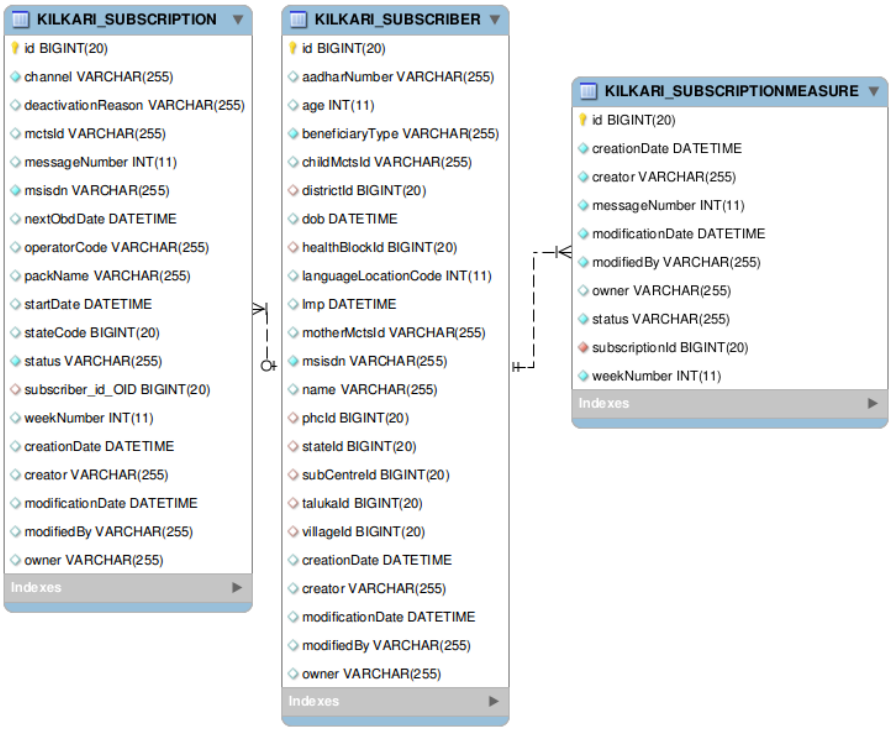
Formatted: Font: (Default) Arial, 13 pt, Bold, Font color: Custom Color(RGB(79;129;189))


KILKARI_SUBSCRIPTION
id BIGINT(20)
channel VARCHAR(255)
deactivationReason VARCHAR(255)
mctsId VARCHAR(255)
messageNumber INT(11)
msisdn VARCHAR(255)
nextObdDate DATETIME
operatorCode VARCHAR(255)
packName VARCHAR(255)
startDate DATETIME
stateCode BIGINT(20)
status VARCHAR(255)
subscriber_id_OID BIGINT(20)
weekNumber INT(11)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes
KILKARI_SUBSCRIBER
id BIGINT(20)
aadharNumber VARCHAR(255)
age INT(11)
beneficiaryType VARCHAR(255)
childMctsId VARCHAR(255)
districtId BIGINT(20)
dob DATETIME
healthBlockId BIGINT(20)
languageLocationCode INT(11)
lmp DATETIME
motherMctsId VARCHAR(255)
msisdn VARCHAR(255)
name VARCHAR(255)
phcId BIGINT(20)
stateId BIGINT(20)
subCentreId BIGINT(20)
talukaId BIGINT(20)
villageId BIGINT(20)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes



Formatted: Font: (Default) Arial, 13 pt, Bold, Font color: Custom Color(RGB(79;129;189))


KILKARI_SUBSCRIPTION
id BIGINT(20)
channel VARCHAR(255)
deactivationReason VARCHAR(255)
mctsId VARCHAR(255)
messageNumber INT(11)
msisdn VARCHAR(255)
nextObdDate DATETIME
operatorCode VARCHAR(255)
packName VARCHAR(255)
startDate DATETIME
stateCode BIGINT(20)
status VARCHAR(255)
subscriber_id_OID BIGINT(20)
weekNumber INT(11)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes
KILKARI_SUBSCRIBER
id BIGINT(20)
aadharNumber VARCHAR(255)
age INT(11)
beneficiaryType VARCHAR(255)
childMctsId VARCHAR(255)
districtId BIGINT(20)
dob DATETIME
healthBlockId BIGINT(20)
languageLocationCode INT(11)
lmp DATETIME
motherMctsId VARCHAR(255)
msisdn VARCHAR(255)
name VARCHAR(255)
phcId BIGINT(20)
stateId BIGINT(20)
subCentreId BIGINT(20)
talukaId BIGINT(20)
villageId BIGINT(20)
creationDate DATETIME
creator VARCHAR(255)
modificationDate DATETIME
modifiedBy VARCHAR(255)
owner VARCHAR(255)
Indexes



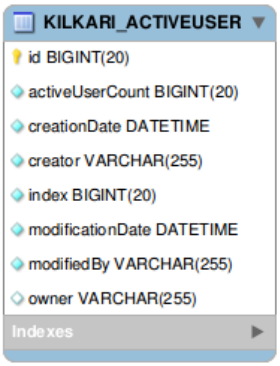
- Subscriber table stores the personal information of Kilkari beneficiaries.
- Subscriber table also maintains a list of Subscriptions which belongs to the Kilkari Beneficiary.

Formatted: Font: (Default) Arial

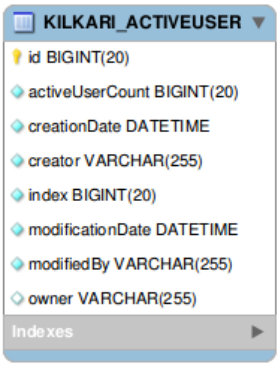
- Subscription entity shall contain the details (like pack Name, status etc.) of Subscription for which beneficiary has subscribed.
- Subscription Measure shall contain the records for each update in any of fields (status, weekNumber, messageNumber). This is for reporting purpose.

~~9.1.8.4~~10.1.7.4 Kilkari Active Subscriptions Count

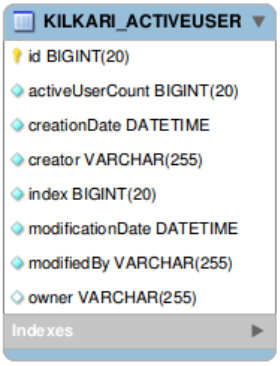
Kilkari shall store the Active Subscriptions Count in the following table:-



Formatted: Font: (Default) Arial



Formatted: Font: (Default) Arial



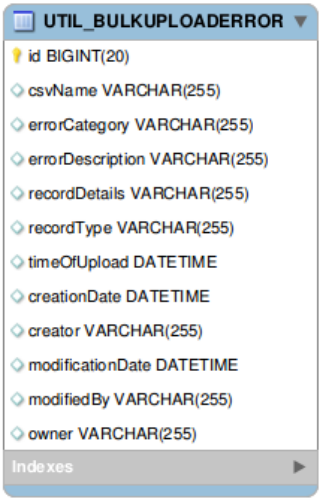
Formatted: Font: (Default) Arial

- This entity shall contain the count of Active/PendingActivation subscriptions in the system.

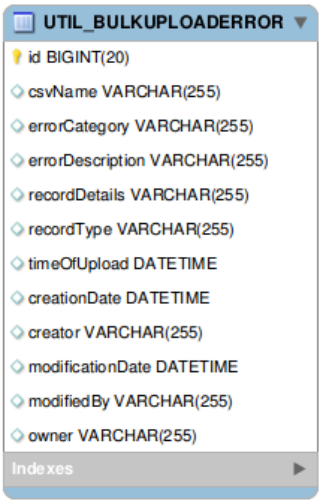
9.1.910.1.8 Utilities Data

9.1.9.110.1.8.1 Bulk Upload Error

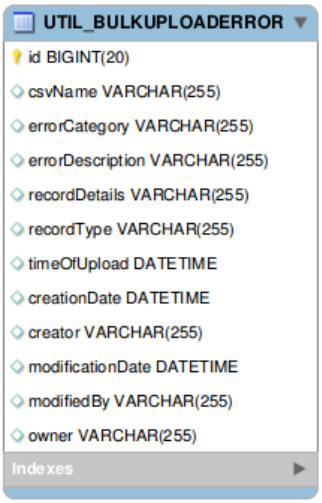
All modules shall store Bulk Upload error details in the following table, provided by Utility module:-



Formatted: Font: (Default) Arial



Formatted: Font: (Default) Arial

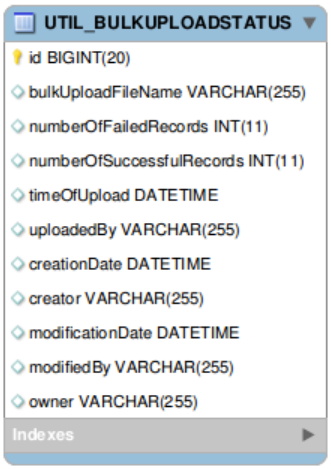


Formatted: Font: (Default) Arial

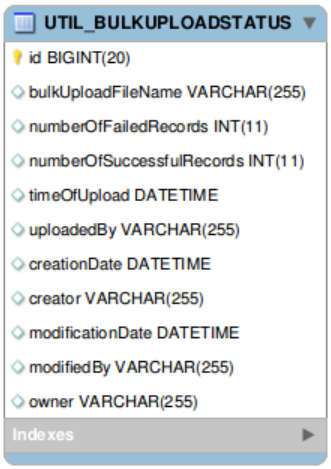
- This entity shall contain error logs for the Bulk Upload procedure.
- This entity shall contain a record for each CSVRecord for which processing fails.
- The error log shall contain the details of the erroneous record and error description, along with the name of the CSV file used for upload.

9.1.9.210.1.8.2 Bulk Upload Status

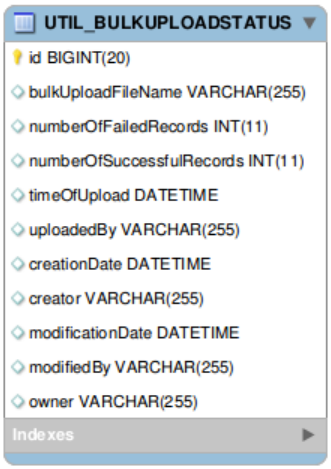
All modules shall the store the Bulk Upload status details in the following table, provided by Util module:-



Formatted: Font: (Default) Arial, 13 pt, Bold, Font color: Custom Color(RGB(79;129;189))



Formatted: Font: (Default) Arial, 13 pt, Bold, Font color: Custom Color(RGB(79;129;189))



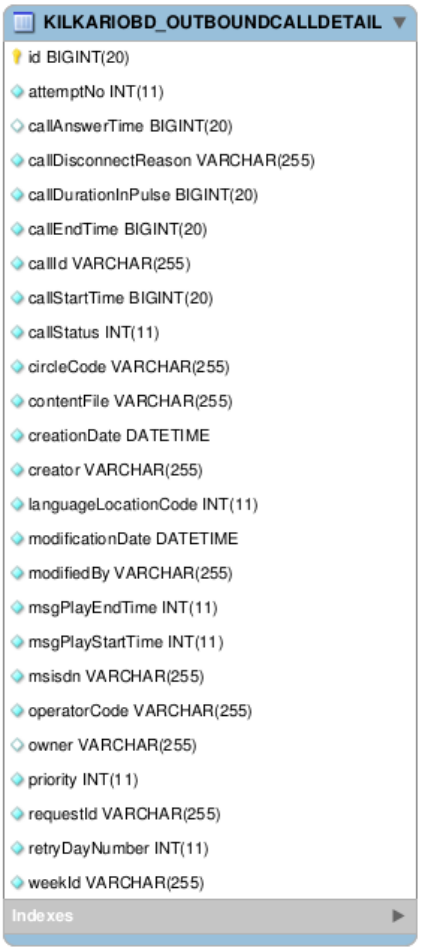
Formatted: Font: (Default) Arial, 13 pt, Bold, Font color: Custom Color(RGB(79;129;189))

- This entity shall contain the status or summary of the Bulk upload procedure.
- Summary shall include the name of the CSV file, number of record processed successfully and unsuccessfully.

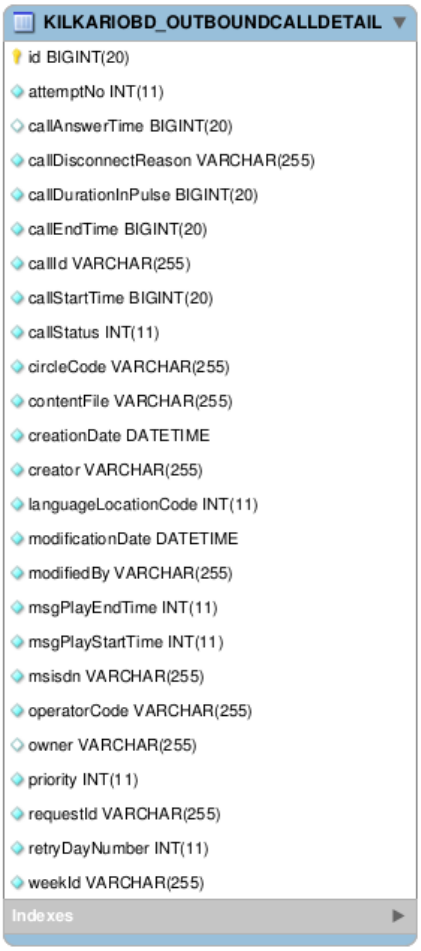
10.1.9 KilkariObd Data

10.1.9.1 Outbound Call Detail

- This entity contains the call detail records for the OBD calls placed by IVR.



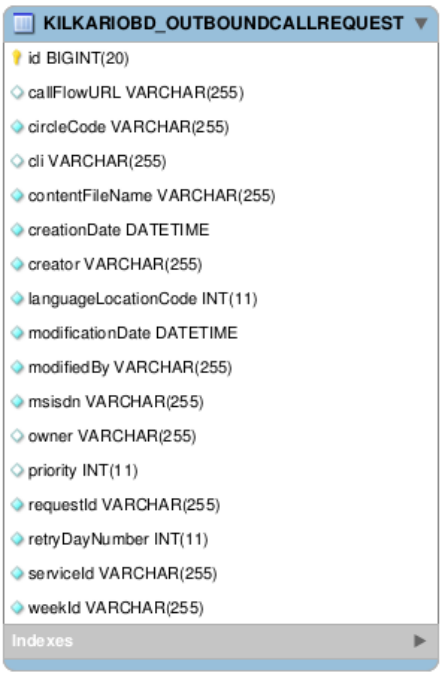
Formatted: Font: (Default) Arial



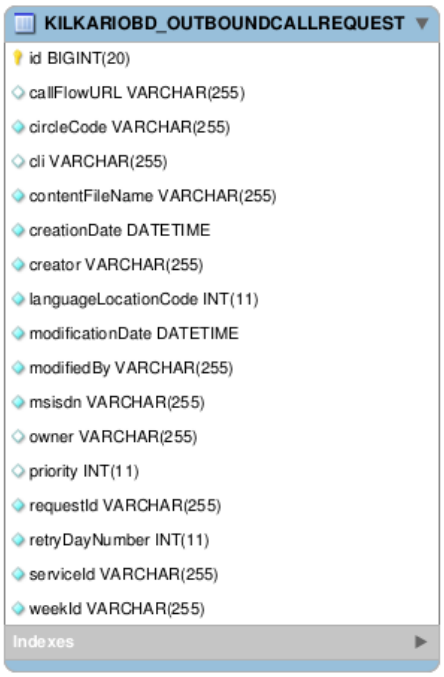
Formatted: Font: (Default) Arial

10.1.9.2 Outbound Call Request

- This entity contains the OBD call definition records, where each record shall have the parameters required by IVR to make calls for OBD message delivery



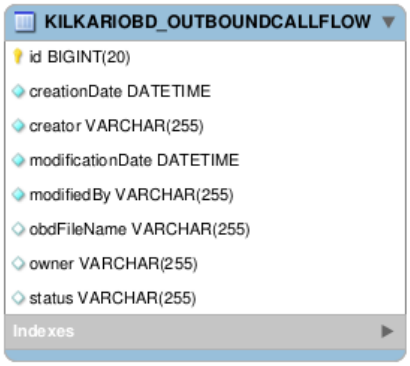
Formatted: Font: (Default) Arial



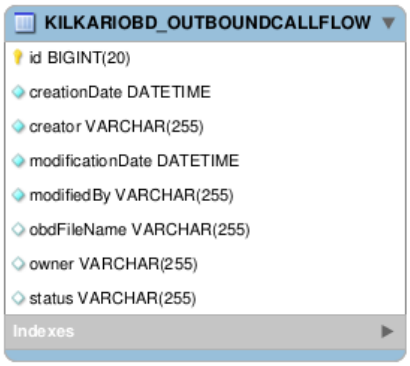
Formatted: Font: (Default) Arial

10.1.9.3 Outbound Call Flow

- This entity shall contain status corresponding to the processing of OBD target file and CDR file



Formatted: Font: (Default) Arial



Formatted: Font: (Default) Arial

9.210.2 Reporting Database Design

< ER Diagrams>

9.310.3 Transactional DB to Reporting DB ETL

1011 Operations & Maintenance

10.111.1 Configuration Management

NMS SW configurable parameters are categorized into:

1. **System Configuration Parameters**: These are parameters related to MA, MK, KK SW deployment (e.g. IP addresses, DB configuration, Transport Configuration, etc.)
2. **Service Configuration Parameters**: These are parameters related to the MA, MK, KK service business logic.

10.1.111.1.1 System Configuration Parameters

10.1.1.111.1.1.1 Requirements

NMS must support configuration of System Level Parameter for NMS services. Following are the design requirements for system level configuration:

- NMS shall store system level configuration parameters in configuration files.

10.1.1.211.1.1.2 Design Approach

10.1.1.2.111.1.1.2.1 Using MOTECH Configuration System

- MOTECH Platform Configuration System provides support for System Configuration, Bootstrap Configuration and Module Specific configuration using configuration source as File (as well as UI).
- MA, MK, Kilkari shall each have configuration parameters in property files, which shall be placed in the location from where these are loaded by MOTECH Platform Configuration System.

- MA, MK, Kilkari shall fetch these parameters values from the respective property files whenever required.
-

~~10.1.1.3~~11.1.1.3 Mobile Academy Configuration Parameters

NAME	TYPE	RANGE	DEFAULT VALUE	DESCRIPTION
NMS_ONLINE_API_HTTP_TIMEOUT_MILLIS	Integer		1000	Timeout in milliseconds for Online category APIs on NMS-IVR interface (Online APIs: APIS invoked during calls)
NMS_OFFLINE_API_INITIAL_INTERVAL_MILLIS	Integer		5000	Timeout in milliseconds for first retry in case of http timeout for Offline APIs. (Offline APIs: APIs invoked after call completion)
NMS_OFFLINE_API_MAX_RETRIES	Integer		3	Maximum number of retry attempts
NMS_OFFLINE_API_RETRY_MULTIPLIER	Integer		2	Value to be multiplied with previous retry timeout.

~~10.1.1.4~~11.1.1.4 Mobile Kunji Configuration Parameters

NAME	TYPE	RANGE	DEFAULT VALUE	DESCRIPTION
NMS_ONLINE_API_HTTP_TIMEOUT_MILLIS	Integer		1000	Timeout in milliseconds for Online category APIs on NMS-IVR interface (Online APIs: APIS invoked during calls)
NMS_OFFLINE_API_INITIAL_INTERVAL_MILLIS	Integer		5000	Timeout in milliseconds for first retry in case of http timeout for Offline APIs. (Offline APIs: APIs invoked after call completion)
NMS_OFFLINE_API_MAX_RETRIES	Integer		3	Maximum number of retry attempts
NMS_OFFLINE_API_RETRY_MULTIPLIER	Integer		2	Value to be multiplied with previous retry timeout.

~~10.1.5~~11.1.1.5 KilkariObd Configuration Parameters

NAME	TYPE	RANGE	DEFAULT VALUE	DESCRIPTION
NMS_ONLINE_API_HTTP_TIMEOUT_MILLIS	Integer		1000	Timeout in milliseconds for Online category APIs on NMS-IVR interface (Online APIs: APIS invoked during calls)
NMS_OFFLINE_API_INITIAL_INTERVAL_MILLIS	Integer		5000	Timeout in milliseconds for first retry in case of http timeout for Offline APIs. (Offline APIs: APIs invoked after call completion)
NMS_OFFLINE_API_MAX_RETRIES	Integer		3	Maximum number of retry attempts
NMS_OFFLINE_API_RETRY_MULTIPLIER	Integer		2	Value to be multiplied with previous retry timeout.
NMS_OBD_FILES_SERVER_IP	String			IP of the server where OBD target and CDR files shall reside.
NMS_OBD_FILES_PATH	String			Path of the OBD target and CDR files.

~~10.1.2~~11.1.2 Service Configuration Parameters

~~10.1.2.1~~11.1.2.1 Requirements

NMS must support configuration of Business Logic Related parameters for NMS services. Following are the design requirements for service level configuration:

- NMS shall provide UI interface to Administrators / Support Staff for viewing, modifying configuration parameter values.
- Default values for service configuration parameters shall not be hardcoded.
- Updated service configuration parameter values must be made available to all MA, MK, and KK service nodes.

10.1.2.211.1.2.2 Design Approach

10.1.2.2.111.1.2.2.1 Using MOTECH Platform MDS DB Editor UI

- Each NMS Service Module shall define MDS entities for storing its service configuration.
  - The allowed parameter values (format and range i.e. max & min) shall be specified using annotations.
  - There shall be a field named "index" in these entities having a unique constraint and the only allowed value as 1. This shall ensure that no more than one row is created in this table.
- Default Service Configuration parameter values shall be populated in the database by MA, MK, Kilkari modules (via init methods / constructors) after startup (in case the values are not present in database)NMS Administrators / Support Staff shall use MDS Web UI Interface to view, update the parameter values.
- NMS Application (MA, MK, Kilkari) shall read these config params from DB whenever required.

10.1.2.311.1.2.3 Service Configuration Parameters

10.1.2.3.111.1.2.3.1 Mobile Academy Service configuration Parameters

NAME	TYPE	DEFAULT VALUE	REMARK
NMS_MA_CAPPING_TYPE	Integer	0	0-No capping 1-National Capping 2-State wise capping
NMS_MA_NATIONAL_CAP_VALUE	Integer	NO DEFAULT	Value of national cap on MA monthly talk-time (in-pulses)

NMS_MA_MAX_END_OF_USAGE_MESSAGE	Integer	2	Maximum no. of times end of usage message can be played to the user on usage unavailability. After the expiry of this value, call will be simply dropped.
NMS_MA_COURSE_QUALIFYING_SCORE	Integer	22	Minimum score a user should achieve in order to qualify the MA course.
NMS_MA_DEFAULT_LANGUAGE_LOCATION_CODE	Integer	NO DEFAULT	Default languageLocationCode value in case circle for a user could not be determined.
NMS_MA_SMS_SENDER_ADDRESS	String	NO DEFAULT	Address to be populated in from field of SMS which will be sent on successful completion of course.

~~10.1.2.3.2~~11.1.2.3.2 Mobile Kunji Service Configurable Parameters

NAME	TYPE	RANGE	DEFAULT VALUE	DESCRIPTION
NMS_MK_CAPPING_TYPE	Integer	0-2	0	0-No capping 1-National Capping 2-State wise capping
NMS_MK_NATIONAL_CAP_VALUE	Integer		NO DEFAULT	Provides the Pan-India monthly talk-time capping value for MK service.
NMS_MK_MAX_WELCOME_MESSAGE	Integer	0 - 1	1	Maximum number of times , welcome message should be played to a user of MK.
NMS_MK_MAX_END_OF_USAGE_ MESSAGE	Integer	0-2	2	Maximum number of times , end of usage message should be played to user of MK.

~~10.1.2.3.3~~11.1.2.3.3 Kilkari Service Configurable Parameters

NAME	TYPE	RANGE	DEFAULT VALUE	DESCRIPTION
~~**NMS_KK_MSGS_PER_WEEK**~~	~~Integer~~	~~[1..2]~~	~~1~~	~~Number of Kilkari messages per week~~
NMS_KK_MAX_ALLOWED_ ACTIVE _BENEFICIARY_COUNT	Integer	> 1	9718577	Max Number of active Kilkari users at any point in the NMS system

~~**NMS_KK_FRESH_OBD_SERVICE_ID**~~	~~String~~			~~Service Id for OBD requests pertaining to fresh base of calls.~~
~~**NMS_KK_RETRY_DAY1_OBD_SERVICE_ID**~~	~~String~~			~~Service Id for OBD requests pertaining to calls for retry day1.~~
~~**NMS_KK_RETRY_DAY2_OBD_SERVICE_ID**~~	~~String~~			~~Service Id for OBD requests pertaining to calls for retry day2.~~
~~**NMS_KK_RETRY_DAY3_OBD_SERVICE_ID**~~	~~String~~			~~Service Id for OBD requests pertaining to calls for retry day3.~~
~~**NMS_KK_FRESH_OBD_PRIORITY**~~	~~Integer~~	~~[0..2]~~	~~0~~	~~Service Id for OBD requests pertaining to fresh base of calls.~~
~~**NMS_KK_RETRY_DAY1_OBD_PRIORITY**~~	~~String~~	~~[0..2]~~	~~0~~	~~Service Id for OBD requests pertaining to calls for retry day1.~~
~~**NMS_KK_RETRY_DAY2_OBD_PRIORITY**~~	~~String~~	~~[0..2]~~	~~1~~	~~Service Id for OBD requests pertaining to calls for retry day2.~~
~~**NMS_KK_RETRY_DAY3_OBD_PRIORITY**~~	~~String~~	~~[0..2]~~	~~2~~	~~Service Id for OBD requests pertaining to calls for retry day3.~~
NMS_KK_NATIONAL_DEFAULT_LANGUAGE_LOCATION_CODE	Integer		NO DEFAULT	National level default languageLocationCode value in case circle for a user is not known.

11.1.2.3.4 KilkariObd Service Configurable Parameters

NAME	TYPE	RANGE	DEFAULT VALUE	DESCRIPTION
NMS_KK_MSGS_PER_WEEK	Integer	[1..2]	1	Number of Kilkari messages per week
NMS_KK_FRESH_OBD_SERVICE_ID	String			Service Id for OBD requests pertaining to fresh base of calls.
NMS_KK_RETRY_DAY1_OBD_SERVICE_ID	String			Service Id for OBD requests pertaining to calls for retry day1.
NMS_KK_RETRY_DAY2_OBD_SERVICE_ID	String			Service Id for OBD requests pertaining to calls for retry day2.
NMS_KK_RETRY_DAY3_OBD_SERVICE_ID	String			Service Id for OBD requests pertaining to calls for retry day3.
NMS_KK_FRESH_OBD_PRIORITY	Integer	[0..2]	0	Service Id for OBD requests pertaining to fresh base of calls.
NMS_KK_RETRY_DAY1_OBD_PRIORITY	String	[0..2]	0	Service Id for OBD requests pertaining to calls for retry day1.
NMS_KK_RETRY_DAY2_OBD_PRIORITY	String	[0..2]	1	Service Id for OBD requests pertaining to calls for retry day2.
NMS_KK_RETRY_DAY3_OBD_PRIORITY	String	[0..2]	2	Service Id for OBD requests pertaining to calls for retry day3.
NMS_KK_OBD_FILE_LOCAL_PATH	String			Temporary Path where OBD file is generated on NMS node.
NMS_KK_OBD_FILE_PATH	String			Path on which OBD target shall be copied by NMS using SCP. This path is also used by IVR for copying CDR Files
NMS_KK_OBD_FILE_SERVER_IP	String			IP of the server to copy the OBD and CDR Files.

NMS_KK_OBD_IVR_IP	String	IP address of the IVR node for invoking API provided by IVR
NMS_KK_OBD_IVR_PORT	integer	Port number to invoke IVR provided APIs

~~10.2~~11.2 Data Upload

This section describes the various design options for implementing NMS Data Upload requirements; pros and cons of each option; and the selected option.

~~10.2.1~~11.2.1 Requirements

The NMS Data Upload functionality must fulfill the following requirements:

- NMS shall provide a web interface to Administrators/Support Staff for performing bulk upload (using a file containing data in Comma Separated Value format) of the following into NMS:
  - Master Location Data and Location-Language Mapping
  - FLW Registration Data
  - MCTS Mother, Child Data
  - MA, MK & Kilkari Content Data
  - State-wise Capping Data
- It should be possible to perform incremental uploads to add new records, modify existing records or delete records in NMS.

- NMS shall validate each record contained in the CSV file before storing it in its database.
- Errors found during validation must be logged.
- NMS shall maintain a history of changes made to the data.

10.2.211.2.2 Data Upload Design

- NMS Application Module shall define MDS entities corresponding to Master Location, FLW Registration and MCTS Mother, Child Data.
- Additionally, for the purpose of temporarily storing data uploaded from CSV files, NMS Application Module shall define MDS entities corresponding to Master Location, FLW Registration and MCST Mother, Child Data CSV files. Data from CSV files shall be uploaded to these tables; NMS Application shall then validate records and move them to the final tables.
- NMS Application Module shall configure MDS to generate an event after data from a CSV file has been entered into any of these temporary data upload tables. The event shall include the following information:
    - *entity_name* - the name of the entity for which this import was performed
    - *entity_class* - the fully qualified class name of the entity for which this import was performed
    - *module_name* - the name of the module from which the entity comes from (optional)
    - *namespace* - the namespace of the entity for which this import was performed (optional)
    - *csv-import.filename* - the name of the imported file

- *csv-import.created_ids* - a list of IDs for instances newly created during import
- *csv-import.updated_ids* - a list of IDs for instances updated during import
- *csv-import.created_count* - the count of instances newly created during import
- *csv-import.updated_count* - the count of instances updated during import
- *csv-import.total_count* - total count of instances created/updated by this import(sum of the created count and updated count)

- NMS Implementation Module shall subscribe to these events.
- MOTECH Web UI shall be used to perform data upload into the selected temporary data upload table.
- On receiving the CSV Upload event from MDS, NMS Application Module shall fetch records from the table, perform validation, insert (or modify or delete) the record into the appropriate final table(s) and then delete the record from the data upload table.

10.2.311.2.3 CSV Upload Status Summary

- The status (CSV filename, table name, no. of successfully processed records, no. of failed records, timestamp, etc.) of each CSV upload shall be recorded in a 'CSV Upload Status Table' (refer 10.1.8.2).

10.2.411.2.4 Rejected CSV Records Log

- Location, FLW, MCTS data uploaded via CSV upload method shall be validated before being stored in the database. All records in the CSV file that fail validation and get rejected shall be listed in a 'CSV Erroneous Records' table (refer 10.1.8.110.1.8.19.1.7.1; having CSV filename, record identifier, failure reason, timestamp, etc). It shall be ensured that

personal information of FLWs / Kilkari subscribers is not stored in the error table. The records shall instead be identified in the error table via the unique key (e.g. FLW Id, MCTS ID, MSISDN, etc.).

- The NMS Administrator / Support Staff, after performing the CSV upload, should monitor (a) CSV Upload Status Table and (b) the 'CSV Erroneous Records' table in database to check the CSV record processing status. In case any records have been rejected (as indicated in the 'CSV Erroneous Records' table in database) then these must be corrected and re-uploaded into NMS.

10.2.511.2.5 CSV Upload Data History Tracking and Auditing

It is required for NMS to maintain a history of all changes made to:

- Location Data
- FLW Data
- MCTS Mother, Child Data

To achieve this, MOTECH Platform MDS Audit and History Tracking feature shall be enabled for all entities corresponding to the above.

MOTECH Platform MDS shall then automatically maintain audit information and history of changes made to these tables. NMS Admin / Support Staff may then view track changes made to Location, FLW, and MCTS data via MOTECH web user interface.

10.2.611.2.6 CSV Upload Transaction Management

- MOTECH Platform Event Module is configured to execute each invocation of the message listener within a JMS transaction.
- Additionally, processing of all CSV records shall be performed in a single DB Transaction.

- The DB Transaction shall be linked to the JMS Transaction such that in case of an unhandled exception (e.g. DB connection loss):
  - The DB transaction shall fail, as a result all changes made to the database during processing of the CSV records shall be rolled back
  - The JMS Transaction shall be rolled back as well. This would lead the ActiveMQ broker to redeliver the message.
  - On receiving the event (re-delivery), MA / MK / Kilkari / etc. shall re-process the CSV records.

10.2.711.2.7 Data File Formats

- Data files shall be in Comma Separated Value format.
- The following abbreviations have been used in the tables given in the subsequent sub-sections.:
  - NP Not Present
  - O Optional
  - M Mandatory

10.2.7.111.2.7.1 Assumptions / Instructions for Data Upload

- Master Location Data is hierarchical in nature and, therefore, files must be uploaded in the following order (top to bottom order):
  1. State Data
  2. District Data
  3. Taluka Data
  4. Health Block Data

5. Health Sub-Facility Data
6. Village Data

FLW Data, Mother & Child Data, Location-Circle-Language Mapping Data is to be uploaded after the Master Location Data has files have been uploaded:

7. Location-Circle-Language Mapping Data:
    a. Circle Codes
    b. Operator Codes
    c. Location, Circle and Language Mapping Data
8. FLW Data
9. Mother & Child Data:

- It must be ensured that the processing of an uploaded CSV file is complete before another CSV file is uploaded.
- Parallel CSV uploads should be avoided.

10.2.7.211.2.7.2 Master Location Data CSV Files

10.2.7.2.111.2.7.2.1 State Data

Fields	Presence for NMS	Presence in MCTS	Remark
stateCode	M	M	State ID
name	M	M	Name of State
maCapping	O	NA	Capping value of MA service
mkCapping	O	NA	Capping value of MK service
IsMaDeployed	O	NA	Deployment status of MA service
IsMkDeployed	O	NA	Deployment status of MK service
IsKkDeployed	O	NA	Deployment status of KK service
IsWhiteListEnable	O	NA	White List enable status

District Data

Fields	Presence for NMS	Presence in MCTS	Remark
districtCode	M	M	District ID
name	M	M	English Name
stateCode	M	M	State ID

~~10.2.7.2.4~~11.2.7.2.3 Taluka Data

Fields	Presence for NMS	Presence in MCTS	Remark
talukaCode	M	M	Taluka ID
name	M	M	English Name
districtCode	M	M	District ID
stateCode	M	M	State ID

~~10.2.7.2.5~~11.2.7.2.4 Health Block Data

Fields	Presence for NMS	Presence in MCTS	Remark
healthBlockCode	M	M	Block ID
name	M	M	English Name
stateCode	M	M	State ID
districtCode	M	M	District Code
talukaCode	M	M	Taluka code

~~10.2.7.2.6~~11.2.7.2.5 Health Facility Data

Fields	Presence for NMS	Presence in MCTS	Remark
healthFacilityCode	M	M	FacilityID (PID)
name	M	M	English Name
stateCode	M	M	State ID
districtCode	M	M	District Code
talukaCode	M	M	Taluka code

healthBlockCode	M	M	Block ID
facilityType	M	M	Type of facility

~~10.2.7.2.7~~11.2.7.2.6 Health Sub Facility Data

Fields	Presence for NMS	Presence in MCTS	Remark
healthSubFacilityCode	M	M	Sub FacilityID
name	M	M	English Name
stateCode	M	M	State ID
districtCode	M	M	District Code
talukaCode	M	M	Taluka code
healthBlockCode	M	M	Block ID
healthFacilityCode	M	M	FacilityID (PID)

~~10.2.7.2.8~~11.2.7.2.7 Village Data

Fields	Presence for NMS	Presence in MCTS	Remark
villageCode	M	M	Village ID(VCode)
name	M	M	English Name
stateCode	M	M	State ID
districtCode	M	M	District Code
talukaCode	M	M	Taluka code

~~10.2.7.2.9~~11.2.7.2.8 Circle Codes

Fields	Presence for NMS	Presence in MCTS	Remark
name	M	NP	Name of the telecom circle
code	M	NP	Short code for the circle

~~10.2.7.2.10~~11.2.7.2.9 Operator Codes

Fields	Presence for NMS	Presence in MCTS	Remark

name	M	NP	Name of the telecom operator
code	M	NP	Short code for the Operator

~~10.2.7.2.11~~11.2.7.2.10 Location, Circle and Language Mapping

Fields	Presence for NMS	Presence in MCTS	Remark
stateCode	M	NP	
districtCode	M	NP	
circleCode	M	NP	
isDefaultLanguageLocationCode	M	NP	
languageLocationCode	M	NP	language location code to be used for MA
languageMA	M	NP	language mapped to the MAlanguageLocationCode
languageMK	M	NP	language mapped to the MKlanguageLocationCode
languageKK	M	NP	language mapped to the KilkarilanguageLocationCode

~~10.2.7.3~~11.2.7.3 MCTS Mother, Child Data CSV Files

~~10.2.7.3.1~~11.2.7.3.1 Mother Data

Fields	Presence for NMS	Presence in MCTS	Remark
stateCode	M	M	
districtCode	M	M	

talukaCode	O	M	
healthBlockCode	O	M	
phcCode	O	M	
subCentreCode	O	M	
villageCode	O	M	
IdNo	M	M	
Name	O	M	
WhomPhoneNo	M	M	This is the contact number
ImpDate	M	M	
abortion	O	O	If not present then its value will be assumed as None
outcomeNos	O	O	If not present then Null value is assumed
age	O	M	
entryType	O	M	If not present then Active shall be assumed
aadharNo	O	O	

~~10.2.7.3.2~~11.2.7.3.2 Child Data

Fields	Presence for NMS	Presence in MCTS	Remark
stateCode	M	M	
districtCode	M	M	
talukaCode	O	M	
healthBlockCode	O	M	
phcCode	O	M	
subCentreCode	O	M	
villageCode	O	M	
IdNo	M	M	This is MCTS Id
motherName	O	M	
motherID	O	M	
whomPhoneNo	M	M	This is the contact number of beneficiary
birthdate	M	M	
entryType	O	M	If not present then Active shall be assumed

10.2.7.411.2.7.4 FLW Registration Data CSV Files

10.2.7.4.111.2.7.4.1 FLW Data

FLW CSV Fields	Presence for NMS	Presence in MCTS	Remark
flwId	M	M	FLWID
nmsFlwId	O	NP	NMS FLW id
contactNo	M	M	Mobile Number
name	M	M	
type	M	M	Designation , ANM,AWW,ASHA,USHA
stateCode	M	NP	Must be provided by MCTS.
districtCode	M	M	
healthBlockCode	O	O	
talukaCode	O	M	
phcCode	O	M	
subCentreCode	O	M	
villageCode	O	M	
ashaNumber	O	NP	
isValidated	O	O	
adhaarNo	O	O	
isValid	O	NP	TRUE/FALSE. Default TRUE.

~~10.2.7.5~~11.2.7.5 Content Upload CSV Files

~~10.2.7.5.1~~11.2.7.5.1 Mobile Academy Content Data

Fields	Presence	Remark
Content ID	M	Unique identifier of the content
Circle	M	Circle code
Language Location Code	M	Language location code for the content. One circle may have multiple language location codes.
Content Name	M	Name of the course content. Naming convention defined in the following table.
Content Type	M	Type of the content – prompt or content
Content File	M	Name of the content audio file
Content Duration	M	Duration of the content audio file

Metadata	O	It is applicable only for question records. It contains a name-value pair for correct answer option of the question. The name-value pair is separated by a colon (:). It takes the format 'correctAnswer:1'

- Naming convention for MA course "Content Name":

S.No.	Content name	Naming convention
1	Lesson file in a chapter	Chapter<chapter index in two digits>_Lesson-<lesson index in two digits>
2	Lesson End Menu file	Chapter<chapter index in two digits>_LessonEndMenu-<lesson index in two digits>
3	Quiz header file in a chapter	Chapter<chapter index in two digits>_QuizHeader
4	Question file in quiz of a chapter	Chapter<chapter index in two digits>_Question<question index in two digits>
5	Correct answer file for a question in quiz	Chapter<chapter index in two digits>_CorrectAnswer< question index in two digits>
6	Wrong answer file for a question in quiz	Chapter<chapter index in two digits>_WrongAnswer< question index in two digits>
7	End menu file in a chapter	Chapter<chapter index in two digits>_EndMenu
8	Score file in a chapter	Chapter<chapter index in two digits>_Score< score in two digits>

Formatted Table

- For a given "Language Location Code", it is expected that complete MA Content information is present in the CSV file (i.e., content information for all MA Chapters, Lessons, Questions, etc.). If for a given "Language Location Code", incomplete MA Content information is present in the CSV file, all the records for the "Language Location Code" shall be rejected.

- A sample MA content data file is available at 13.1~~13.1~~12.1

~~10.2.7.5.2~~11.2.7.5.2 Mobile Kunji Content Data

Fields	Presence	Remark
contentId	M	Unique identifier of the content
circleCode	M	Circle code
languageLocationCode	M	Language location code for the content. One circle may have multiple language location codes.
contentName	M	Name of the content

contentType	M	Prompt / Content
contentFile	M	Name of the content audio file
cardNumber	M	MK Card number
contentDuration	M	Duration of the content audio file (In seconds)

~~10.2.7.5.3~~11.2.7.5.3 Kilkari Content Data

Fields	Presence	Remark
contentId	M	Unique identifier of the content
circleCode	M	Circle code
languageLocationCode	M	Language location code for the content. One circle may have multiple language location codes.
contentName	M	Name of the content
contentType	M	Prompt / Content
contentFile	M	Name of the content audio file
contentDuration	M	Duration of the content audio file (In seconds)

~~10.2.7.6~~11.2.7.6 Capping and White-Listing Data

~~10.2.7.6.1~~11.2.7.6.1 MK State-Wise Capping Data

Fields	Presence for NMS	Presence in MCTS	Remark
StateId	M	NP	Census ID of state
CapValue	M	NP	Monthly Cap Value in pulses

~~10.2.7.6.2~~11.2.7.6.2 MA State-Wise Capping Data

Fields	Presence for NMS	Presence in MCTS	Remark
StateId	M	NP	Census ID of state
CapValue	M	NP	Monthly Cap Value in pulses

11.2.7.6.3 FLW White-List Data

FLW White List CSV Fields	Presence for NMS	Presence in MCTS	Remark
~~operation~~	O	NA	~~ADD/DEL~~
contactNo	M	NA	Mobile Number

~~10.2.9~~11.2.8 Data Validation

~~10.2.9.1~~11.2.8.1 MCTS Mother Data Validation Checks

FIELD	REASON	CAUSE	ERROR STRING
IdNo (MCTS ID)	If MCTS ID is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. MCTS ID missing.
whomPhoneNo (MSISDN)	If the MSISDN is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. MSISDN missing.
	MSISDN is searched in MDS. If it is present with different MCTS ID and has a subscription to 72 weeks pack with status as Active/PendingActivation. Then new record is rejected (as same pack cannot be activated multiple times on same MSISDN).	Inconsistent Data	Upload unsuccessful. Pack Already Subscribed to MSISDN.
LOCATION (stateCode/ districtCode/ talukaCode/ healthBlockCode/ phcCode/ subCentreCode/ villageCode)	If any of the fields of location (State and /or, District) is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. State Missing. Or Upload unsuccessful. District Missing.
	If any field of the location which is present but doesn't match to location in Master Location data	Invalid Data	Upload unsuccessful. Invalid State Information. Or Upload unsuccessful.

Formatted Table

			Invalid Village Information. … Etc.
abortion	If it is present then value should be one of (MTP>12 Weeks, MTP<12 Weeks, Spontaneous, None).	Invalid Data	Upload unsuccessful. Invalid Abortion information.
entryType	If it is present then value should be one of (1-Active, 9-Death, 2-MigratedIn, 3-MigratedOut).	Invalid Data	Upload unsuccessful. Invalid Entry_Type information.
ImpDate	If LMP_Date is missing in mother record it cannot be imported	Mandatory Parameter Missing	Upload unsuccessful. LMP_Date missing.
	If the format of LMP_Date is not correct.	Invalid Data	Upload unsuccessful. Invalid LMP_Date information.
ALL	If the number of existing Active/PendingActivation subscriber exceeds the configured limit of allowed users	InconsistentData	Upload unsuccessful. Beneficiary count exceeded.

~~10.2.9.2~~11.2.8.2 MCTS Child Data Validation Checks

FIELD	REASON	CAUSE	ERROR STRING
IdNo (MCTS ID)	If MCTS ID is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. MCTS ID missing.
whomPhoneNo (MSISDN)	If the MSISDN is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. MSISDN missing.
	MSISDN is searched in MDS. If it is present with different MCTS ID and has a subscription to 72 weeks pack with status as Active/PendingActivation. Then new record is	Inconsistent Data	Upload unsuccessful. Pack Already Subscribed to MSISDN.

Formatted Table

	rejected (as same pack cannot be activated multiple times on same MSISDN).		
LOCATION (stateCode/ districtCode/ talukaCode/ healthBlockCode/ phcCode/ subCentreCode/ villageCode)	If any of the fields of location (State and /or, District) is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. State Missing. Or Upload unsuccessful. District Missing.
	If any field of the location which is present but doesn't match to location in Master Location data	Invalid Data	Upload unsuccessful. Invalid State Information. Or Upload unsuccessful. Invalid Village Information. … Etc.
abortion	If it is present then value should be one of (MTP>12 Weeks, MTP<12 Weeks, Spontaneous, None).	Invalid Data	Upload unsuccessful. Invalid Abortion information.
entryType	If it is present then value should be one of (1-Active, 9-Death, 2-MigratedIn, 3-MigratedOut).	Invalid Data	Upload unsuccessful. Invalid Entry_Type information.
lmpDate	If LMP_Date is missing in mother record it cannot be imported	Mandatory Parameter Missing	Upload unsuccessful. LMP_Date missing.
	If the format of LMP_Date is not correct.	Invalid Data	Upload unsuccessful. Invalid LMP_Date information.
ALL	If the number of existing Active/PendingActivation subscriber exceeds the configured limit of allowed users	InconsistentData	Upload unsuccessful. Beneficiary count exceeded.

10.2.9.311.2.8.3 FLW Data Validation Checks

FIELD	REASON	CAUSE	ERROR STRING
MSISDN	If the MSISDN is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. MSISDN missing.
	If the MSISDN is <10 digits then the new record cannot be imported.	Invalid Data	Upload unsuccessful. Invalid MSISDN.
	MSISDN is searched in MDS. If it is already present, corresponding FLW ID is compared with the corresponding FLW ID in MDS. If the FLW ID is different, the record is not imported.	Inconsistent Data	Upload unsuccessful. MSISDN is already present in MDS, with different FLW ID.
NAME	If FLW Name is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. FLW name missing.
DESIGNATION	If the FLW Designation is missing in the CSV, then the new record cannot be saved.	Mandatory Parameter Missing	Upload unsuccessful. FLW Designation missing.
LOCATION	If any of the fields of location (State, District) is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. FLW Location incomplete. State/ District missing
	If language is not deduced from the state-district combination, the record is not imported.	Invalid Data	Upload unsuccessful. Language could not be deduced from location information.

Formatted Table

~~10.2.9.4~~11.2.8.4 MA Data Validation Checks

FIELD	REASON	CAUSE	ERROR STRING
CONTENT ID	If contentID is missing then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content ID missing.
	If Content ID is not an integer, then the new record cannot be imported.	Invalid Data	Upload unsuccessful. Invalid Content ID.

Formatted Table

CONTENT NAME	If the content name is missing, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content Name missing.
	If content name is not in the format as specified, then the new record cannot be imported.	Invalid Data	Upload unsuccessful. Invalid Content Name.
CONTENT DURATION	If Content Duration is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content Duration missing
	If Content Duration is not numeric, then the new record cannot be imported.	Invalid Data	Upload unsuccessful. Invalid Content Duration
METADATA	If the file corresponds to main content file for a question and MetaData is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. MetaData missing
	If the file corresponds to main content file for a question and MetaData is not in the format as specified, then the new record cannot be imported.	Invalid Data	Upload unsuccessful. Invalid MetaData

~~10.2.9.5~~11.2.8.5 Kilkari Content Data Validation Checks

FIELD	REASON	CAUSE	ERROR STRING
CONTENT ID	If contentID is missing then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content ID missing.
CONTENT NAME	If the content name is missing, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content Name missing.
CONTENT TYPE	If the content type is missing, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content Type missing.

CONTENT FILE	If the content file is missing, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content File missing.
CONTENT DURATION	If Content Duration is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Content Duration missing
CIRCLE ID	If Circle is missing in the CSV, then the new record cannot be imported.	Mandatory Parameter Missing	Upload unsuccessful. Circle missing
LANGUAGE LOCATION CODE	If language location code is missing in the CSV, then the new record cannot be imported	Mandatory Parameter Missing	Upload unsuccessful. LanguageLocationCode missing

~~10.2.9.6~~11.2.8.6 Language Location Validation Checks

FIELD	REASON	CAUSE	ERROR STRING
stateCode	If StateId is not present.	Mandatory Parameter Missing	Upload unsuccessful. StateID missing.
	If this field of the location which is present but doesn't match to location in Master Location data	Invalid Data	Upload unsuccessful. Invalid State Information.
districtCode	If DistrictId is not present.	Mandatory Parameter Missing	Upload unsuccessful. DistrictId missing.
	If this field of the location which is present but doesn't match to location in Master Location data	Invalid Data	Upload unsuccessful. Invalid District Information.
circleCode	If CircleId is not present.	Mandatory Parameter Missing	Upload unsuccessful. CircleId missing.
	If this field is present but doesn't match to Circle Code in circle table	Invalid Data	Upload unsuccessful. Invalid Circle Information.

Formatted Table

languageLocationCode	if this field is not present.	Mandatory Parameter Missing	Upload unsuccessful. languageLocationCode missing
isDefaultLanguageLocationCode	If this field is not present.	Mandatory Parameter Missing	Upload unsuccessful. isDefaultLanguageLocationCode missing
ALL	If not correct datatype	InvalidData	Upload unsuccessful. Invalid <Field> Information
languageMA	if this field is not present.	Mandatory Parameter Missing	Upload unsuccessful. languageMA missing
languageMK	if this field is not present.	Mandatory Parameter Missing	Upload unsuccessful. languageMK missing
languageKK	if this field is not present.	Mandatory Parameter Missing	Upload unsuccessful. languageKK missing

10.311.3 Reporting

<Reporting Tool Evaluation and Selection, Design>

10.411.4 Logging

10.4.111.4.1 MA, MK, Kilkari Logs

MA, MK, Kilkari modules shall use the logging tool and policy as provided by MOTECH Platform:

- slf4j tool shall be used for logging
- MA, MK, Kilkari logs shall be stored in a log file.
- There shall be one log file per web-tier server (these logs shall not be aggregated).
- It shall be possible for NMS Administrators / Support Staff to view these log files via MOTECH Platform Admin User Interface or by directly accessing it on the Linux file system.
- It shall be possible for NMS Administrators / Support Staff to view and change log level of all MA, MK, and Kilkari SW modules via MOTECH Platform Admin User Interface (per-server basis).

10.4.211.4.2 Log Levels

The following table describes the information logged at various log levels (by MA, MK and Kilkari Services:

Log Level	Information Logged at this Level
FATAL	Very severe error events that will presumably lead the application to abort
ERROR	Error events that might still allow the application to continue running;
WARN	Potentially harmful situations;
INFO	CSV UploadInformational messages that highlight the progress at coarse-grained level; (e.g. module startup info; CSV upload info; etc.)
DEBUG	Per-Call Messages exchanged on the MOTECH-IVR Interface;
TRACE	Events that are most useful to debug the application;

The default log level shall be configured as **INFO** level.

10.511.5 Exception Handling

10.5.111.5.1 Design Options

10.5.1.111.5.1.1 Using a Fault Barrier

In this approach all exceptions are categorized as follows:

a) **Contingency:** An expected condition demanding an alternative response from a method that can be expressed in terms of the method's intended purpose. The caller of the method expects these kinds of conditions and has a strategy for coping with them.

b) **Fault:** An unplanned condition that prevents a method from achieving its intended purpose that cannot be described without reference to the method's internal implementation

10.5.1.1.111.5.1.1.1 Fault Handling

Fault barrier pattern shall be used in which any application component can throw a fault exception, but only the component acting as the "fault barrier" catches them. The fault barrier shall reside logically toward the top of the call stack where it stops the upward propagation of an exception before default action is triggered. For a fault occurring while processing a request from IVR, the application server shall send an appropriate error response to IVR.

The fault barrier component shall record the information contained in the fault exception for future action. Additionally, the fault barrier shall close the operation in a controlled manner.

All the participating classes must follow common conventions:

- The fault barrier method must reside at the head of a graph of method calls that traverses the participating classes.
- They must all use unchecked exceptions to signify fault conditions.
- They must all use the specific unchecked exception types that the fault barrier is expecting to receive.
- They all must catch and translate checked exceptions from lower methods that are deemed to be faults in their execution context.
- They must not interfere with the propagation of fault exceptions on their way to the barrier.

10.5.1.1.211.5.1.1.2 Contingency Handling

Checked exception type shall be used to convey the existence of a contingency condition and supply the information needed to contend with it.

10.611.6 Security

<Data Security – database encryption, access security – SSL, password protection, Firewall, etc.>

10.6.111.6.1 Data Security

10.6.211.6.2 Disk Encryption

10.6.311.6.3 Access Security

10.6.3.111.6.3.1 Firewall

10.6.3.211.6.3.2 NMS Reports Web Interface Security

10.6.3.311.6.3.3 MOTECH – IVR Interface Security

10.6.3.411.6.3.4 MOTECH – Database Interface Security

10.6.3.511.6.3.5 NMS Users, Roles, Access Rights and Passwords

1112 High Availability and Deployment

11.112.1 Load Balancer Design

11.212.2 Database High Availability

NMS MOTECH database provides crucial services to MOTECH Web servers. One or more MOTECH web servers shall be connected with MOTECH database. To meet the system requirements of high availability, the database shall be deployed in Master-Slave configuration on two servers. Master database shall be responsible for all the reads and writes, whereas slave database shall act as a hot standby of master and shall assume the role of master in case the Master fails.

Database High Availability design focuses on achieving the following two goals:

1) Transparent Routing
2) Automatic Failover.

To achieve these goals open source tool keepalived shall be configured to run at both the DB servers. Keepalived uses Virtual Router Redundancy Protocol (VRRP) which ensures that one of the nodes in the topology is assigned with Virtual IP.

Keepalived maintains state of each node in the topology. The keepalived state can be one of the following:

- MASTER
- BACKUP
- FAULT

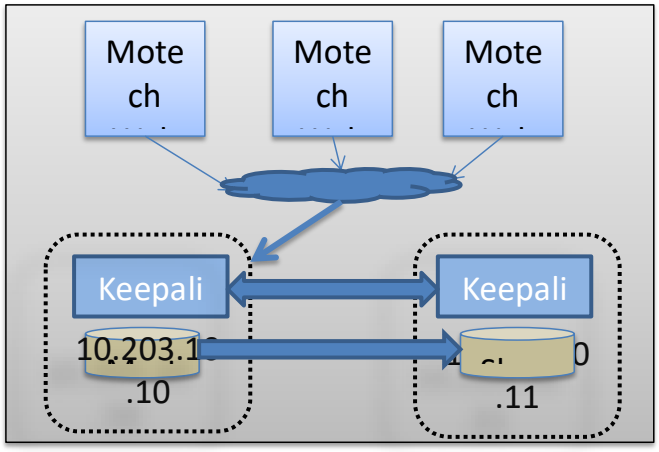
Virtual IP (VIP) shall be assigned to the node which is in MASTER state. The node configured as Backup shall listen to the multicast advertisement packets from the Master. In

case there are no advertisement packets from Master then the Backup node takes the role of Master and Virtual IP is acquired by new Master.

The database IP address shall be configured as the Virtual IP on all NMS Web Tier Nodes.

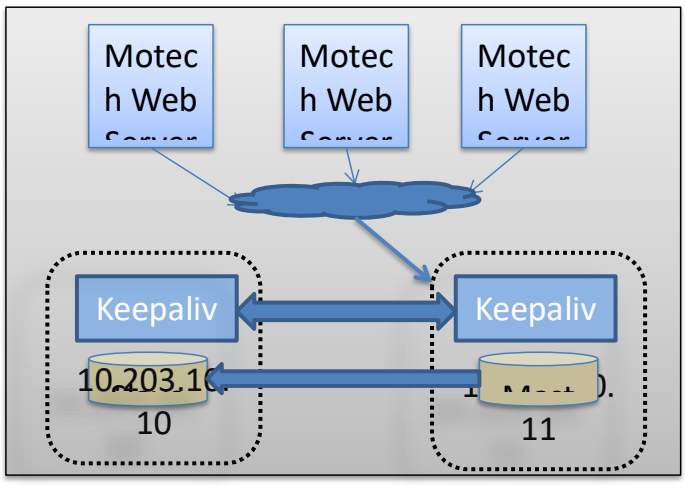
Keepalived shall be configured to use the following scripts:

- **Track Script: Keepalived** shall be configured to track MySQL services at the interval of 2 seconds. The track script will monitor MySQL service using **mysqladmin** command, If script finds that MySQL is not up, than it returns value other than zero, (0 in case MySQL is up). Once **keepalived** receives track script value other than 0 it moves keeplived state to 'FAULT' and removes the VIP if original state was 'MASTER'.
- **Notify Scripts:** These scripts are executed whenever there is transition in the **keepalived** state.
    - **Notify Master:** This script is executed when there is transition in state from 'BACKUP' to 'MASTER'. This Script shall be configured to use **mysqlrpladmin** (MySQL utility) to perform Failover from Slave to Master.
    - **Notify Backup:** This script is executed when there is transition in state from 'MASTER' to 'BACKUP'. This script shall use **mysqlreplicate** to add the server to MySQL replication topology.



The figure given above depicts DB server1 (10.203.10.10) as MASTER and DB server2 (10.203.10.11) as slave. All the MOTECH web servers are configured to connect to the database at Virtual IP and are transparent to the association of VIP with particular machine.

If Check Script detects a fault, keepalived moves the state from MASTER To FAULT in DB Server1 and assigns the VIP to DBserver2. On this state transition, Notify Master script is executed at new MASTER. The configuration after this transition is depicted in the following figure:

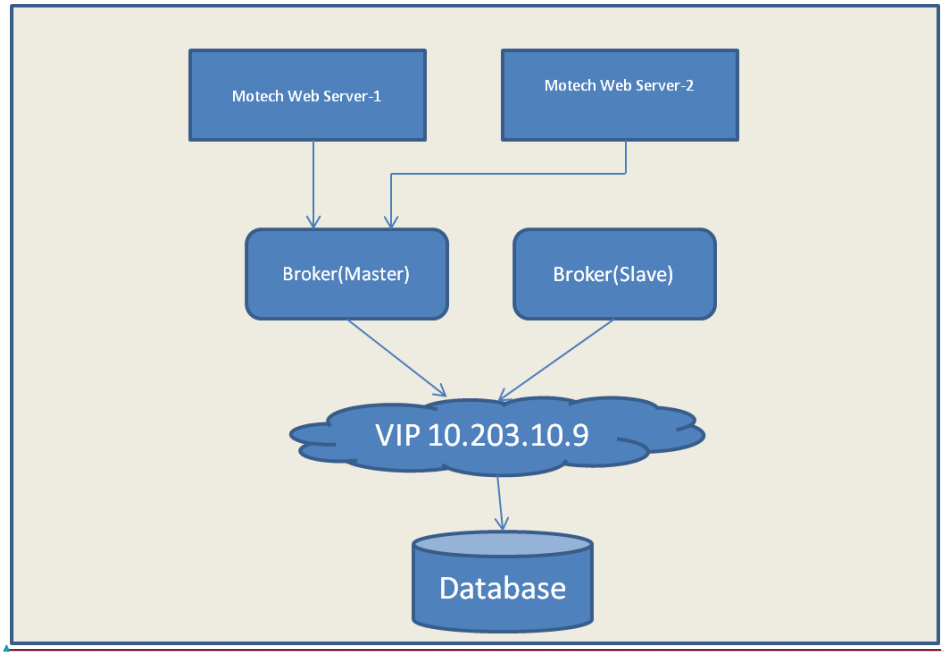


12.3 Event Sub-System High Availability

Motech Platform Event module uses ActiveMQ for sending/receiving events. JDBC shall be used as data source which is implemented using mysql and is high available. To make event system high available Master-Slave cluster is used. If Master fails, Motech shall automatically failover to a slave which will now have access (exclusive lock) to data store and can read all the messages. Hence, message are high available.

12.3.1 Startup

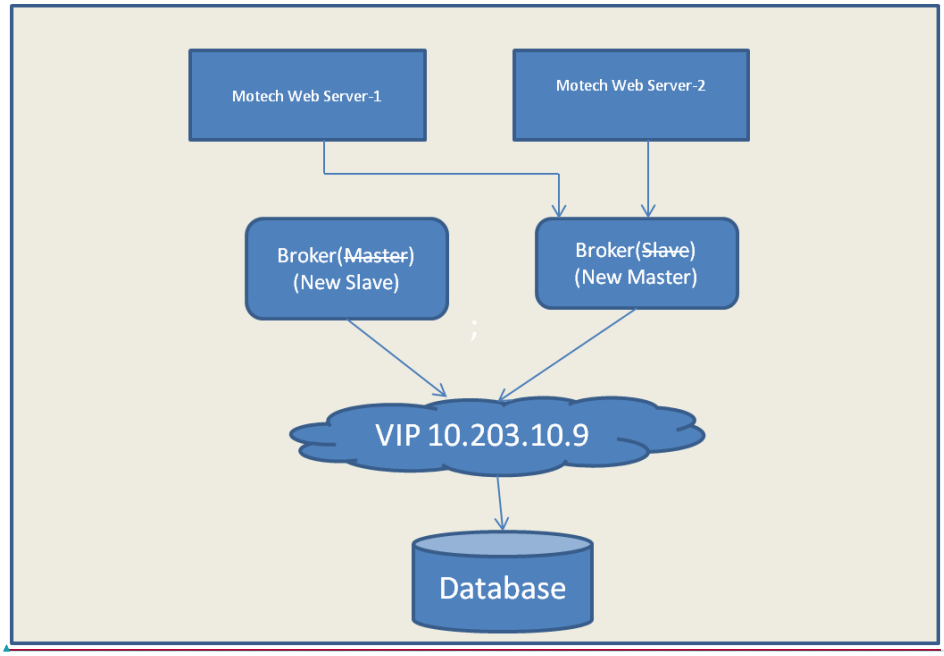
On startup one broker gets an exclusive lock on database and becomes master. All other brokers are slave.



Formatted: Font: (Default) Arial

12.3.2 Master failure

If master shuts down or fails resulting in loosing of exclusive lock to database, one of the other slaves will immediately grab the exclusive lock on the database and is promoted to be the Master. Client connection breaks with the failed Master and then failover transport tries to connect to new master.



Formatted: Font: (Default) Arial

11.312.4 Scheduler Sub-System High Availability

11.412.5 Reporting System High Availability

11.512.6 Physical / Deployment View

< The physical view depicts the system from a system engineer's point-of-view - A description of the deployment view of the architecture. It is concerned with the topology of software components on the physical layer (describes the allocation of tasks (from the Process View) to the physical nodes), as well as the physical connections between these components.>

- Physical Architecture Diagram
- Deployment Strategy – automated / manual
- Possible Manual Configuration required after automated installation – port, URL etc for connectivity with third party components (Actual values to be provided in Operations Manual)
- Folder structure setup on target environment after installation

In case installation is non-standard or complicated, a separate, short document may be produced

12.7 Deployment Scripts

11.5.112.7.1 NMS Web Tier SW Installation

1213 Appendix

12.1113.1 Mobile Academy Course Structure and Content CSV File Sample

```
contentId,circle,languageLocationCode,contentName,contentType,contentFile,contentDuration,metaData
100014,AP,170,Chapter01_Lesson01,Content,ch1_l1.wav,150,
100015,AP,170,Chapter01_LessonEndMenu01,Content,ch1_l1_op.wav,20,
100016,AP,170,Chapter01_Lesson02,Content,ch1_l2.wav,150,
100017,AP,170,Chapter01_LessonEndMenu02,Content,ch1_l2_op.wav,20,
100018,AP,170,Chapter01_Lesson03,Content,ch1_l3.wav,150,
100019,AP,170,Chapter01_LessonEndMenu03,Content,ch1_l3_op.wav,20,
100020,AP,170,Chapter01_Lesson04,Content,ch1_l4.wav,150,
100021,AP,170,Chapter01_LessonEndMenu04,Content,ch1_l4_op.wav,30,
100022,AP,170,Chapter01_QuizHeader,Content,ch1_qp.wav,40,
100023,AP,170,Chapter01_Question01 ,Content,ch1_q1.wav,20,correctAnswer:2
100024,AP,170,Chapter01_CorrectAnswer01,Content,ch1_q1_ca.wav,20,
100025,AP,170,Chapter01_WrongAnswer01,Content,ch1_q1_wa.wav,20,
100026,AP,170,Chapter01_Question02 ,Content,ch1_q2.wav,20,CorrectAnswer:1
100027,AP,170,Chapter01_CorrectAnswer02,Content,ch1_q2_ca.wav,20,
100028,AP,170,Chapter01_WrongAnswer02,Content,ch1_q2_wa.wav,20,
100029,AP,170,Chapter01_Question03,Content,ch1_q3.wav,20,CorrectAnswer:1
100030,AP,170,Chapter01_CorrectAnswer03,Content,ch1_q3_ca.wav,20,
100031,AP,170,Chapter01_WrongAnswer03,Content,ch1_q3_wa.wav,20,
100032,AP,170,Chapter01_Question04,Content,ch1_q4.wav,20,CorrectAnswer:1
100033,AP,170,Chapter01_CorrectAnswer04,Content,ch1_q4_ca.wav,20,
100034,AP,170,Chapter01_WrongAnswer04,Content,ch1_q4_wa.wav,20,
100035,AP,170,Chapter01_EndMenu,Content,ch1_end_op.wav,30,
100036,AP,170,Chapter01_Score00,Content,ch1_0ca.wav,20,
100037,AP,170,Chapter01_Score01,Content,ch1_1ca.wav,20,
100038,AP,170,Chapter01_Score02,Content,ch1_2ca.wav,20,
100039,AP,170,Chapter01_Score03,Content,ch1_3ca.wav,20,
100040,AP,170,Chapter01_Score04,Content,ch1_4ca.wav,20,
100041,AP,170,Chapter02_Lesson01,Content,ch2_l1.wav,150,
100042,AP,170,Chapter02_LessonEndMenu01,Content,ch2_l1_op.wav,20,
100043,AP,170,Chapter02_Lesson02,Content,ch2_l2.wav,150,
100044,AP,170,Chapter02_LessonEndMenu02,Content,ch2_l2_op.wav,20,
100045,AP,170,Chapter02_Lesson03,Content,ch2_l3.wav,150,
100046,AP,170,Chapter02_LessonEndMenu03,Content,ch2_l3_op.wav,20,
100047,AP,170,Chapter02_Lesson04,Content,ch2_l4.wav,150,
100048,AP,170,Chapter02_LessonEndMenu04,Content,ch2_l4_op.wav,30,
100049,AP,170,Chapter02_QuizHeader,Content,ch2_qp.wav,40,
100050,AP,170,Chapter02_Question01 ,Content,ch2_q1.wav,20,correctAnswer:2
100051,AP,170,Chapter02_CorrectAnswer01,Content,ch2_q1_ca.wav,20,
100052,AP,170,Chapter02_WrongAnswer01,Content,ch2_q1_wa.wav,20,
100053,AP,170,Chapter02_Question02 ,Content,ch2_q2.wav,20,correctAnswer:1
100054,AP,170,Chapter02_CorrectAnswer02,Content,ch2_q2_ca.wav,20,
100055,AP,170,Chapter02_WrongAnswer02,Content,ch2_q2_wa.wav,20,
```

```
100056,AP,170,Chapter02_Question03,Content,ch2_q3.wav,20,correctAnswer:1
100057,AP,170,Chapter02_CorrectAnswer03,Content,ch2_q3_ca.wav,20,
100058,AP,170,Chapter02_WrongAnswer03,Content,ch2_q3_wa.wav,20,
100059,AP,170,Chapter02_Question04,Content,ch2_q4.wav,20,correctAnswer:1
100060,AP,170,Chapter02_CorrectAnswer04,Content,ch2_q4_ca.wav,20,
100061,AP,170,Chapter02_WrongAnswer04,Content,ch2_q4_wa.wav,20,
100062,AP,170,Chapter02_EndMenu,Content,ch2_end_op.wav,30,
100063,AP,170,Chapter02_Score00,Content,ch2_0ca.wav,20,
100064,AP,170,Chapter02_Score01,Content,ch2_1ca.wav,20,
100065,AP,170,Chapter02_Score02,Content,ch2_2ca.wav,20,
100066,AP,170,Chapter02_Score03,Content,ch2_3ca.wav,20,
100067,AP,170,Chapter02_Score04,Content,ch2_4ca.wav,20,
100068,AP,170,Chapter03_Lesson01,Content,ch3_l1.wav,120,
100069,AP,170,Chapter03_LessonEndMenu01,Content,ch3_l1_op.wav,150,
100070,AP,170,Chapter03_Lesson02,Content,ch3_l2.wav,20,
100071,AP,170,Chapter03_LessonEndMenu02,Content,ch3_l2_op.wav,150,
100072,AP,170,Chapter03_Lesson03,Content,ch3_l3.wav,20,
100073,AP,170,Chapter03_LessonEndMenu03,Content,ch3_l3_op.wav,150,
100074,AP,170,Chapter03_Lesson04,Content,ch3_l4.wav,20,
100075,AP,170,Chapter03_LessonEndMenu04,Content,ch3_l4_op.wav,150,
100076,AP,170,Chapter03_QuizHeader,Content,ch3_qp.wav,30,
100077,AP,170,Chapter03_Question01 ,Content,ch3_q1.wav,40,correctAnswer:2
100078,AP,170,Chapter03_CorrectAnswer01,Content,ch3_q1_ca.wav,30,
100079,AP,170,Chapter03_WrongAnswer01,Content,ch3_q1_wa.wav,20,
100080,AP,170,Chapter03_Question02 ,Content,ch3_q2.wav,20,correctAnswer:1
100081,AP,170,Chapter03_CorrectAnswer02,Content,ch3_q2_ca.wav,20,
100082,AP,170,Chapter03_WrongAnswer02,Content,ch3_q2_wa.wav,20,
100083,AP,170,Chapter03_Question03,Content,ch3_q3.wav,20,correctAnswer:1
100084,AP,170,Chapter03_CorrectAnswer03,Content,ch3_q3_ca.wav,20,
100085,AP,170,Chapter03_WrongAnswer03,Content,ch3_q3_wa.wav,30,
100086,AP,170,Chapter03_Question04,Content,ch3_q4.wav,20,correctAnswer:1
100087,AP,170,Chapter03_CorrectAnswer04,Content,ch3_q4_ca.wav,20,
100088,AP,170,Chapter03_WrongAnswer04,Content,ch3_q4_wa.wav,20,
100089,AP,170,Chapter03_EndMenu,Content,ch3_end_op.wav,20,
100090,AP,170,Chapter03_Score00,Content,ch3_0ca.wav,20,
100091,AP,170,Chapter03_Score01,Content,ch3_1ca.wav,30,
100092,AP,170,Chapter03_Score02,Content,ch3_2ca.wav,20,
100093,AP,170,Chapter03_Score03,Content,ch3_3ca.wav,20,
100094,AP,170,Chapter03_Score04,Content,ch3_4ca.wav,20,
100095,AP,170,Chapter04_Lesson01,Content,ch4_l1.wav,150,
100096,AP,170,Chapter04_LessonEndMenu01,Content,ch4_l1_op.wav,20,
100097,AP,170,Chapter04_Lesson02,Content,ch4_l2.wav,150,
100098,AP,170,Chapter04_LessonEndMenu02,Content,ch4_l2_op.wav,20,
100099,AP,170,Chapter04_Lesson03,Content,ch4_l3.wav,150,
100100,AP,170,Chapter04_LessonEndMenu03,Content,ch4_l3_op.wav,20,
100101,AP,170,Chapter04_Lesson04,Content,ch4_l4.wav,150,
100102,AP,170,Chapter04_LessonEndMenu04,Content,ch4_l4_op.wav,30,
100103,AP,170,Chapter04_QuizHeader,Content,ch4_qp.wav,40,
100104,AP,170,Chapter04_Question01 ,Content,ch4_q1.wav,20,correctAnswer:2
100105,AP,170,Chapter04_CorrectAnswer01,Content,ch4_q1_ca.wav,20,
```

```
100106,AP,170,Chapter04_WrongAnswer01,Content,ch4_q1_wa.wav,20,
100107,AP,170,Chapter04_Question02 ,Content,ch4_q2.wav,20,correctAnswer:1
100108,AP,170,Chapter04_CorrectAnswer02,Content,ch4_q2_ca.wav,20,
100109,AP,170,Chapter04_WrongAnswer02,Content,ch4_q2_wa.wav,20,
100110,AP,170,Chapter04_Question03,Content,ch4_q3.wav,20,correctAnswer:1
100111,AP,170,Chapter04_CorrectAnswer03,Content,ch4_q3_ca.wav,20,
100112,AP,170,Chapter04_WrongAnswer03,Content,ch4_q3_wa.wav,20,
100113,AP,170,Chapter04_Question04,Content,ch4_q4.wav,20,correctAnswer:1
100114,AP,170,Chapter04_CorrectAnswer04,Content,ch4_q4_ca.wav,20,
100115,AP,170,Chapter04_WrongAnswer04,Content,ch4_q4_wa.wav,20,
100116,AP,170,Chapter04_EndMenu,Content,ch4_end_op.wav,30,
100117,AP,170,Chapter04_Score00,Content,ch4_0ca.wav,20,
100118,AP,170,Chapter04_Score01,Content,ch4_1ca.wav,20,
100119,AP,170,Chapter04_Score02,Content,ch4_2ca.wav,20,
100120,AP,170,Chapter04_Score03,Content,ch4_3ca.wav,20,
100121,AP,170,Chapter04_Score04,Content,ch4_4ca.wav,20,
100122,AP,170,Chapter05_Lesson01,Content,ch5_l1.wav,150,
100123,AP,170,Chapter05_LessonEndMenu01,Content,ch5_l1_op.wav,20,
100124,AP,170,Chapter05_Lesson02,Content,ch5_l2.wav,150,
100125,AP,170,Chapter05_LessonEndMenu02,Content,ch5_l2_op.wav,20,
100126,AP,170,Chapter05_Lesson03,Content,ch5_l3.wav,150,
100127,AP,170,Chapter05_LessonEndMenu03,Content,ch5_l3_op.wav,20,
100128,AP,170,Chapter05_Lesson04,Content,ch5_l4.wav,150,
100129,AP,170,Chapter05_LessonEndMenu04,Content,ch5_l4_op.wav,30,
100130,AP,170,Chapter05_QuizHeader,Content,ch5_qp.wav,40,
100131,AP,170,Chapter05_Question01 ,Content,ch5_q1.wav,20,correctAnswer:2
100132,AP,170,Chapter05_CorrectAnswer01,Content,ch5_q1_ca.wav,20,
100133,AP,170,Chapter05_WrongAnswer01,Content,ch5_q1_wa.wav,20,
100134,AP,170,Chapter05_Question02 ,Content,ch5_q2.wav,20,correctAnswer:1
100135,AP,170,Chapter05_CorrectAnswer02,Content,ch5_q2_ca.wav,20,
100136,AP,170,Chapter05_WrongAnswer02,Content,ch5_q2_wa.wav,20,
100137,AP,170,Chapter05_Question03,Content,ch5_q3.wav,20,correctAnswer:1
100138,AP,170,Chapter05_CorrectAnswer03,Content,ch5_q3_ca.wav,20,
100139,AP,170,Chapter05_WrongAnswer03,Content,ch5_q3_wa.wav,20,
100140,AP,170,Chapter05_Question04,Content,ch5_q4.wav,20,correctAnswer:1
100141,AP,170,Chapter05_CorrectAnswer04,Content,ch5_q4_ca.wav,20,
100142,AP,170,Chapter05_WrongAnswer04,Content,ch5_q4_wa.wav,20,
100143,AP,170,Chapter05_EndMenu,Content,ch5_end_op.wav,30,
100144,AP,170,Chapter05_Score00,Content,ch5_0ca.wav,20,
100145,AP,170,Chapter05_Score01,Content,ch5_1ca.wav,20,
100146,AP,170,Chapter05_Score02,Content,ch5_2ca.wav,20,
100147,AP,170,Chapter05_Score03,Content,ch5_3ca.wav,20,
100148,AP,170,Chapter05_Score04,Content,ch5_4ca.wav,20,
100149,AP,170,Chapter06_Lesson01,Content,ch6_l1.wav,150,
100150,AP,170,Chapter06_LessonEndMenu01,Content,ch6_l1_op.wav,20,
100151,AP,170,Chapter06_Lesson02,Content,ch6_l2.wav,150,
100152,AP,170,Chapter06_LessonEndMenu02,Content,ch6_l2_op.wav,20,
100153,AP,170,Chapter06_Lesson03,Content,ch6_l3.wav,150,
100154,AP,170,Chapter06_LessonEndMenu03,Content,ch6_l3_op.wav,20,
100155,AP,170,Chapter06_Lesson04,Content,ch6_l4.wav,150,
```

```
100156,AP,170,Chapter06_LessonEndMenu04,Content,ch6_l4_op.wav,30,
100157,AP,170,Chapter06_QuizHeader,Content,ch6_qp.wav,40,
100158,AP,170,Chapter06_Question01 ,Content,ch6_q1.wav,20,correctAnswer:2
100159,AP,170,Chapter06_CorrectAnswer01,Content,ch6_q1_ca.wav,20,
100160,AP,170,Chapter06_WrongAnswer01,Content,ch6_q1_wa.wav,20,
100161,AP,170,Chapter06_Question02 ,Content,ch6_q2.wav,20,correctAnswer:1
100162,AP,170,Chapter06_CorrectAnswer02,Content,ch6_q2_ca.wav,20,
100163,AP,170,Chapter06_WrongAnswer02,Content,ch6_q2_wa.wav,20,
100164,AP,170,Chapter06_Question03,Content,ch6_q3.wav,20,correctAnswer:1
100165,AP,170,Chapter06_CorrectAnswer03,Content,ch6_q3_ca.wav,20,
100166,AP,170,Chapter06_WrongAnswer03,Content,ch6_q3_wa.wav,20,
100167,AP,170,Chapter06_Question04,Content,ch6_q4.wav,20,correctAnswer:1
100168,AP,170,Chapter06_CorrectAnswer04,Content,ch6_q4_ca.wav,20,
100169,AP,170,Chapter06_WrongAnswer04,Content,ch6_q4_wa.wav,20,
100170,AP,170,Chapter06_EndMenu,Content,ch6_end_op.wav,30,
100171,AP,170,Chapter06_Score00,Content,ch6_0ca.wav,20,
100172,AP,170,Chapter06_Score01,Content,ch6_1ca.wav,20,
100173,AP,170,Chapter06_Score02,Content,ch6_2ca.wav,20,
100174,AP,170,Chapter06_Score03,Content,ch6_3ca.wav,20,
100175,AP,170,Chapter06_Score04,Content,ch6_4ca.wav,20,
100176,AP,170,Chapter07_Lesson01,Content,ch7_l1.wav,150,
100177,AP,170,Chapter07_LessonEndMenu01,Content,ch7_l1_op.wav,20,
100178,AP,170,Chapter07_Lesson02,Content,ch7_l2.wav,150,
100179,AP,170,Chapter07_LessonEndMenu02,Content,ch7_l2_op.wav,20,
100180,AP,170,Chapter07_Lesson03,Content,ch7_l3.wav,150,
100181,AP,170,Chapter07_LessonEndMenu03,Content,ch7_l3_op.wav,20,
100182,AP,170,Chapter07_Lesson04,Content,ch7_l4.wav,150,
100183,AP,170,Chapter07_LessonEndMenu04,Content,ch7_l4_op.wav,30,
100184,AP,170,Chapter07_QuizHeader,Content,ch7_qp.wav,40,
100185,AP,170,Chapter07_Question01 ,Content,ch7_q1.wav,20,correctAnswer:2
100186,AP,170,Chapter07_CorrectAnswer01,Content,ch7_q1_ca.wav,20,
100187,AP,170,Chapter07_WrongAnswer01,Content,ch7_q1_wa.wav,20,
100188,AP,170,Chapter07_Question02 ,Content,ch7_q2.wav,20,correctAnswer:1
100189,AP,170,Chapter07_CorrectAnswer02,Content,ch7_q2_ca.wav,20,
100190,AP,170,Chapter07_WrongAnswer02,Content,ch7_q2_wa.wav,20,
100191,AP,170,Chapter07_Question03,Content,ch7_q3.wav,20,correctAnswer:1
100192,AP,170,Chapter07_CorrectAnswer03,Content,ch7_q3_ca.wav,20,
100193,AP,170,Chapter07_WrongAnswer03,Content,ch7_q3_wa.wav,20,
100194,AP,170,Chapter07_Question04,Content,ch7_q4.wav,20,correctAnswer:1
100195,AP,170,Chapter07_CorrectAnswer04,Content,ch7_q4_ca.wav,20,
100196,AP,170,Chapter07_WrongAnswer04,Content,ch7_q4_wa.wav,20,
100197,AP,170,Chapter07_EndMenu,Content,ch7_end_op.wav,30,
100198,AP,170,Chapter07_Score00,Content,ch7_0ca.wav,20,
100199,AP,170,Chapter07_Score01,Content,ch7_1ca.wav,20,
100200,AP,170,Chapter07_Score02,Content,ch7_2ca.wav,20,
100201,AP,170,Chapter07_Score03,Content,ch7_3ca.wav,20,
100202,AP,170,Chapter07_Score04,Content,ch7_4ca.wav,20,
100203,AP,170,Chapter08_Lesson01,Content,ch8_l1.wav,150,
100204,AP,170,Chapter08_LessonEndMenu01,Content,ch8_l1_op.wav,20,
100205,AP,170,Chapter08_Lesson02,Content,ch8_l2.wav,150,
```

```
100206,AP,170,Chapter08_LessonEndMenu02,Content,ch8_l2_op.wav,20,
100207,AP,170,Chapter08_Lesson03,Content,ch8_l3.wav,150,
100208,AP,170,Chapter08_LessonEndMenu03,Content,ch8_l3_op.wav,20,
100209,AP,170,Chapter08_Lesson04,Content,ch8_l4.wav,150,
100210,AP,170,Chapter08_LessonEndMenu04,Content,ch8_l4_op.wav,30,
100211,AP,170,Chapter08_QuizHeader,Content,ch8_qp.wav,40,
100212,AP,170,Chapter08_Question01 ,Content,ch8_q1.wav,20,correctAnswer:2
100213,AP,170,Chapter08_CorrectAnswer01,Content,ch8_q1_ca.wav,20,
100214,AP,170,Chapter08_WrongAnswer01,Content,ch8_q1_wa.wav,20,
100215,AP,170,Chapter08_Question02 ,Content,ch8_q2.wav,20,correctAnswer:1
100216,AP,170,Chapter08_CorrectAnswer02,Content,ch8_q2_ca.wav,20,
100217,AP,170,Chapter08_WrongAnswer02,Content,ch8_q2_wa.wav,20,
100218,AP,170,Chapter08_Question03,Content,ch8_q3.wav,20,correctAnswer:1
100219,AP,170,Chapter08_CorrectAnswer03,Content,ch8_q3_ca.wav,20,
100220,AP,170,Chapter08_WrongAnswer03,Content,ch8_q3_wa.wav,20,
100221,AP,170,Chapter08_Question04,Content,ch8_q4.wav,20,correctAnswer:1
100222,AP,170,Chapter08_CorrectAnswer04,Content,ch8_q4_ca.wav,20,
100223,AP,170,Chapter08_WrongAnswer04,Content,ch8_q4_wa.wav,20,
100224,AP,170,Chapter08_EndMenu,Content,ch8_end_op.wav,30,
100225,AP,170,Chapter08_Score00,Content,ch8_0ca.wav,20,
100226,AP,170,Chapter08_Score01,Content,ch8_1ca.wav,20,
100227,AP,170,Chapter08_Score02,Content,ch8_2ca.wav,20,
100228,AP,170,Chapter08_Score03,Content,ch8_3ca.wav,20,
100229,AP,170,Chapter08_Score04,Content,ch8_4ca.wav,20,
100230,AP,170,Chapter09_Lesson01,Content,ch9_l1.wav,150,
100231,AP,170,Chapter09_LessonEndMenu01,Content,ch9_l1_op.wav,20,
100232,AP,170,Chapter09_Lesson02,Content,ch9_l2.wav,150,
100233,AP,170,Chapter09_LessonEndMenu02,Content,ch9_l2_op.wav,20,
100234,AP,170,Chapter09_Lesson03,Content,ch9_l3.wav,150,
100235,AP,170,Chapter09_LessonEndMenu03,Content,ch9_l3_op.wav,20,
100236,AP,170,Chapter09_Lesson04,Content,ch9_l4.wav,150,
100237,AP,170,Chapter09_LessonEndMenu04,Content,ch9_l4_op.wav,30,
100238,AP,170,Chapter09_QuizHeader,Content,ch9_qp.wav,40,
100239,AP,170,Chapter09_Question01 ,Content,ch9_q1.wav,20,correctAnswer:2
100240,AP,170,Chapter09_CorrectAnswer01,Content,ch9_q1_ca.wav,20,
100241,AP,170,Chapter09_WrongAnswer01,Content,ch9_q1_wa.wav,20,
100242,AP,170,Chapter09_Question02 ,Content,ch9_q2.wav,20,correctAnswer:1
100243,AP,170,Chapter09_CorrectAnswer02,Content,ch9_q2_ca.wav,20,
100244,AP,170,Chapter09_WrongAnswer02,Content,ch9_q2_wa.wav,20,
100245,AP,170,Chapter09_Question03,Content,ch9_q3.wav,20,correctAnswer:1
100246,AP,170,Chapter09_CorrectAnswer03,Content,ch9_q3_ca.wav,20,
100247,AP,170,Chapter09_WrongAnswer03,Content,ch9_q3_wa.wav,20,
100248,AP,170,Chapter09_Question04,Content,ch9_q4.wav,20,correctAnswer:1
100249,AP,170,Chapter09_CorrectAnswer04,Content,ch9_q4_ca.wav,20,
100250,AP,170,Chapter09_WrongAnswer04,Content,ch9_q4_wa.wav,20,
100251,AP,170,Chapter09_EndMenu,Content,ch9_end_op.wav,30,
100252,AP,170,Chapter09_Score00,Content,ch9_0ca.wav,20,
100253,AP,170,Chapter09_Score01,Content,ch9_1ca.wav,20,
100254,AP,170,Chapter09_Score02,Content,ch9_2ca.wav,20,
100255,AP,170,Chapter09_Score03,Content,ch9_3ca.wav,20,
```

```
100256,AP,170,Chapter09_Score04,Content,ch9_4ca.wav,20,
100257,AP,170,Chapter10_Lesson01,Content,ch10_l1.wav,150,
100258,AP,170,Chapter10_LessonEndMenu01,Content,ch10_l1_op.wav,20,
100259,AP,170,Chapter10_Lesson02,Content,ch10_l2.wav,150,
100260,AP,170,Chapter10_LessonEndMenu02,Content,ch10_l2_op.wav,20,
100261,AP,170,Chapter10_Lesson03,Content,ch10_l3.wav,150,
100262,AP,170,Chapter10_LessonEndMenu03,Content,ch10_l3_op.wav,20,
100263,AP,170,Chapter10_Lesson04,Content,ch10_l4.wav,150,
100264,AP,170,Chapter10_LessonEndMenu04,Content,ch10_l4_op.wav,30,
100265,AP,170,Chapter10_QuizHeader,Content,ch10_qp.wav,40,
100266,AP,170,Chapter10_Question01 ,Content,ch10_q1.wav,20,correctAnswer:2
100267,AP,170,Chapter10_CorrectAnswer01,Content,ch10_q1_ca.wav,20,
100268,AP,170,Chapter10_WrongAnswer01,Content,ch10_q1_wa.wav,20,
100269,AP,170,Chapter10_Question02 ,Content,ch10_q2.wav,20,correctAnswer:1
100270,AP,170,Chapter10_CorrectAnswer02,Content,ch10_q2_ca.wav,20,
100271,AP,170,Chapter10_WrongAnswer02,Content,ch10_q2_wa.wav,20,
100272,AP,170,Chapter10_Question03,Content,ch10_q3.wav,20,correctAnswer:1
100273,AP,170,Chapter10_CorrectAnswer03,Content,ch10_q3_ca.wav,20,
100274,AP,170,Chapter10_WrongAnswer03,Content,ch10_q3_wa.wav,20,
100275,AP,170,Chapter10_Question04,Content,ch10_q4.wav,20,correctAnswer:1
100276,AP,170,Chapter10_CorrectAnswer04,Content,ch10_q4_ca.wav,20,
100277,AP,170,Chapter10_WrongAnswer04,Content,ch10_q4_wa.wav,20,
100278,AP,170,Chapter10_EndMenu,Content,ch10_end_op.wav,30,
100279,AP,170,Chapter10_Score00,Content,ch10_0ca.wav,20,
100280,AP,170,Chapter10_Score01,Content,ch10_1ca.wav,20,
100281,AP,170,Chapter10_Score02,Content,ch10_2ca.wav,20,
100282,AP,170,Chapter10_Score03,Content,ch10_3ca.wav,20,
100283,AP,170,Chapter10_Score04,Content,ch10_4ca.wav,20,
100284,AP,170,Chapter11_Lesson01,Content,ch11_l1.wav,150,
100285,AP,170,Chapter11_LessonEndMenu01,Content,ch11_l1_op.wav,20,
100286,AP,170,Chapter11_Lesson02,Content,ch11_l2.wav,150,
100287,AP,170,Chapter11_LessonEndMenu02,Content,ch11_l2_op.wav,20,
100288,AP,170,Chapter11_Lesson03,Content,ch11_l3.wav,150,
100289,AP,170,Chapter11_LessonEndMenu03,Content,ch11_l3_op.wav,20,
100290,AP,170,Chapter11_Lesson04,Content,ch11_l4.wav,150,
100291,AP,170,Chapter11_LessonEndMenu04,Content,ch11_l4_op.wav,30,
100292,AP,170,Chapter11_QuizHeader,Content,ch11_qp.wav,40,
100293,AP,170,Chapter11_Question01 ,Content,ch11_q1.wav,20,correctAnswer:2
100294,AP,170,Chapter11_CorrectAnswer01,Content,ch11_q1_ca.wav,20,
100295,AP,170,Chapter11_WrongAnswer01,Content,ch11_q1_wa.wav,20,
100296,AP,170,Chapter11_Question02 ,Content,ch11_q2.wav,20,correctAnswer:1
100297,AP,170,Chapter11_CorrectAnswer02,Content,ch11_q2_ca.wav,20,
100298,AP,170,Chapter11_WrongAnswer02,Content,ch11_q2_wa.wav,20,
100299,AP,170,Chapter11_Question03,Content,ch11_q3.wav,20,correctAnswer:1
100300,AP,170,Chapter11_CorrectAnswer03,Content,ch11_q3_ca.wav,20,
100301,AP,170,Chapter11_WrongAnswer03,Content,ch11_q3_wa.wav,20,
100302,AP,170,Chapter11_Question04,Content,ch11_q4.wav,20,correctAnswer:1
100303,AP,170,Chapter11_CorrectAnswer04,Content,ch11_q4_ca.wav,20,
100304,AP,170,Chapter11_WrongAnswer04,Content,ch11_q4_wa.wav,20,
100305,AP,170,Chapter11_EndMenu,Content,ch11_end_op.wav,30,
```

```
100306,AP,170,Chapter11_Score00,Content,ch11_0ca.wav,20,
100307,AP,170,Chapter11_Score01,Content,ch11_1ca.wav,20,
100308,AP,170,Chapter11_Score02,Content,ch11_2ca.wav,20,
100309,AP,170,Chapter11_Score03,Content,ch11_3ca.wav,20,
100310,AP,170,Chapter11_Score04,Content,ch11_4ca.wav,20,
```