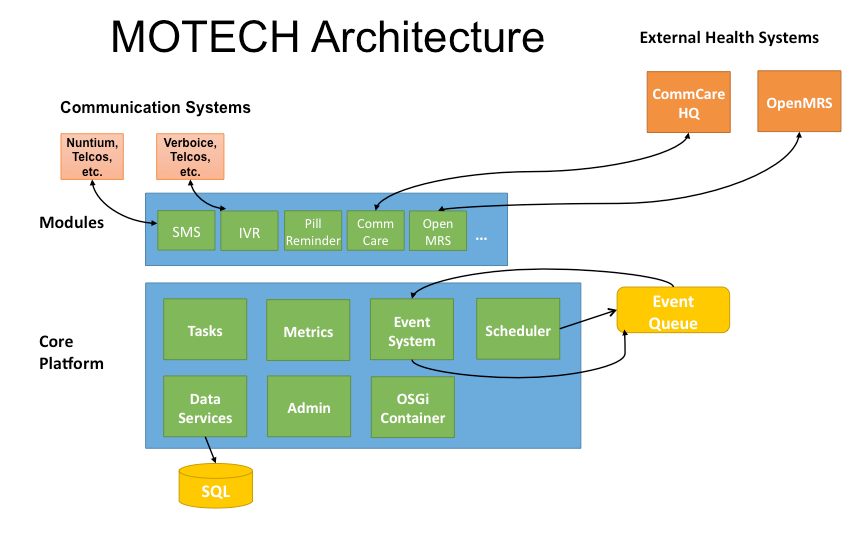
|  |
| --- |
| HCL Technologies Ltd. |
| MOTECH Framework – Solution Approach on Tamil language support in IVR |
| ***Compliant with MOTECH ver .25*** |

****

# Overview

1. **MOTECH Version considered while writing this doc is .25**
2. Set of open source technologies which address core needs of **Mobility Health (mHealth)**
3. **mHealth** - a spectrum of tools and applications to extend health services to underserved populations, helping to educate and inform patients, and helping community-based workers diagnose, treat and monitor a broad range of issues
4. **MOTECH** Suite delivers an integrated set of complementary applications that are scalable, sustainable and readily deployed for mHealth implementation
5. **General features of MOTECH applications :**
   1. Communicate information to patients by sending reminders, via **voice** or **SMS**
      * For appointments, lab visits, to take medicine.
      * To take medication
      * To take children for scheduled immunization services
   2. Collect data from patients or caregivers:
   3. Alert caregivers of the status of their patients
   4. Facilitate communication between patients, caregivers, and/or health administrators
6. **Implementations**
   1. Mobile Midwife & Nurses Application – Ghana
   2. TAMA – Urban India
   3. Ananya – Bihar, India
      1. MOBILE HANDSET APPLICATION FOR FRONT LINE HEALTH WORKERS (FLWS)
      2. IVR TRAINING TOOLS FOR FLWS
      3. KILKARI
   4. World Health Partners TB Case Management

# MOTECH Architecture



**Motech High level Architecture**

**In high level system architecture, MOTECH is logically divided into two tiers/layers**

* 1. Core Platform
  2. Module layer

1. **Core Platform**

* Wraps several well-Known open source systems and exposes their features to the layers above it
* Wraps ActiveMQ and present an event interface to the module and implementation layers
* Provides an interface to the scheduler and access to Motech Data Services (MySQL).
* provides a module loading environment (OSGi- Open Service Gateway initiative)

1. **Module Layer**

* module layer is a collection of reusable components
* Modules interact with the core platform through its APIs
* Modules interact with other modules either through their service interfaces or by consuming their events
* Modules may also register servlet controllers which allow them to respond to HTTP requests.

1. **Stateless**

* MOTECH is stateless – MOTECH server perform a single action per request and then return
* The module should never persist any state in memory and expect that state to be available to later requests

1. **Events**

* MOTECH engine provide an event system
* Any module can emit an event by calling the eventRelay and passing it a MotechEvent and a subject.
* To register for an event a module just needs to annotate a method with the list of event subjects or interest.

1. **Scheduled Events & Timers**

* MOTECH provides access to a flexible scheduling system
* Uses Quartz (open-source) application to schedule events

## Component Diagram of MOTECH system

# Module List

Following are the modules available in the MOTECH suite:

1. [**Alerts**](http://www.motechproject.org/modules/alerts.html) :- Collects alerts for users in an inbox-like container
2. [**Appointments**](http://www.motechproject.org/modules/appointments.html) :- Provides appointment scheduling and reminding
3. [**CMS Lite**](http://www.motechproject.org/modules/cms-lite.html):- Provides basic content storage and retrieval
4. [**Decision Tree**](http://www.motechproject.org/modules/decision-tree.html) :- Provides APIs for constructing an IVR decision tree
5. [**IVR**](http://www.motechproject.org/modules/ivr.html):-Provides basic specification for integrating platform with an IVR service provider; Also connects the platform IVR with an asterisk server using VoiceGlue VXML browser
6. [**Message** **Campaign**](http://www.motechproject.org/modules/message-campaign.html):-Enrolls users in message campaigns with flexible content scheduling rules
7. [**Mobile forms**](http://www.motechproject.org/modules/mobile-forms.html):-Supports configurable forms and data collection though mobile devices.
8. [**MRS (Medical Record System)**](http://www.motechproject.org/modules/mrs.html):-Provides basic specification for integrating platform with a medical record system
9. [**OpenMRS**](http://www.motechproject.org/modules/openmrs.html):-Integrates platform with OpenMRS system. OpenMRS is the database where health records are stored.
10. [**Outbox**](http://www.motechproject.org/modules/outbox.html):-A voicemail like messaging system for end users
11. [**Pill Reminder**](http://www.motechproject.org/modules/pill-reminder.html):-A flexible reminder system focused on medication
12. [**Schedule Tracking**](http://www.motechproject.org/modules/schedule-tracking.html):-Enrolls users for alerts based on complex scheduling rules
13. [**SMS**](http://www.motechproject.org/modules/sms.html):-Provides basic specification for integrating platform with an SMS provider to send/receive SMS messages
14. [**SMS HTTP**](http://www.motechproject.org/modules/sms-http.html):-Allows platform to integrate with an HTTP-based SMS gateway to send SMS messages
15. [**SMS SMPP**](http://www.motechproject.org/modules/sms-smpp.html):-Allows platform to integrate with an SMPP-based SMS gateway to send/receive SMS messages

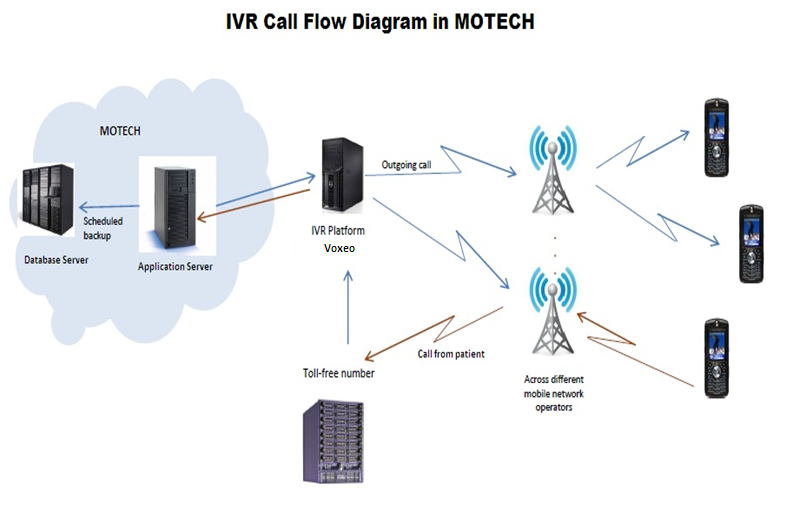
# MOTECH Services & Features

1. **Demand Generation** - for health services and health-seeking behaviors through direct messages or reminders to clients. These messages/reminders can be delivered via IVR or SMS
2. **Registration** – patient and health workers both are registered for better coordination and delivering more effective healthcare. Registration can be done in any of the following modes :
   1. via IVR
   2. via Simple forms
   3. via Mobile applications
   4. via Web UI
3. **Front Line Health Workers Empowerment** - training Front Line Workers (FLWs), help them manage their workload and facilitate more productive interactions with their patients
   1. Training Courses
   2. Medical Protocols facilitated by mobile devices
   3. Mobile Data entry
   4. Follow-up and defaulter alerts
   5. Works with available technologies like IVR, SMS, handset application, Web based UI
4. **Supply Chain Logistics** 
   1. Reporting stock outs
   2. Notification of new supply shipment
   3. Inventory management
5. **Integration -** with existing health systems , governmental or NGO databases, national ID systems, and other help systems
6. **Adherence Monitoring –** checking if advised actions are followed and intervening if not.
7. **Hosting**

* Self-hosted
* Hosted in the cloud

# IVR Support in MOTECH

IVR means **Interactive Voice Response**. IVR is used in MOTECH mainly for patients to interact with system for different issues. Patients, in rural areas, will not have the access to mobile application of MOTECH and may not have the literacy skills to navigate written information in the application. For them, IVR can be more user friendly mode to gather and provide information. Patients will receive interactive voice messages encouraging them to pursue health-seeking behaviors. Pregnant women and mothers can receive IVR calls that remind them of healthy behaviors, and encourage them during their pregnancy and baby’s first year to follow the schedule of care.

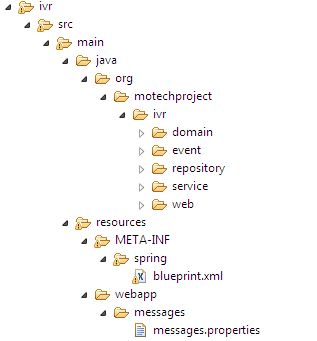


## Base IVR Module in MOTECH

1. MOTECH base IVR module provides the base integration functionality and enables any other IVR (VXML/CCXML complaint only) to integrate into MOTECH platform
2. It has a Config class which holds IVR provider configuration, represents how the IVR module interacts with an IVR provider.
3. It also has an interface OutboundCallService which is used to initiate an outgoing call and its implementation class OutboundCallServiceImpl which generates & sends an HTTP request to an IVR provider to trigger an outbound call.

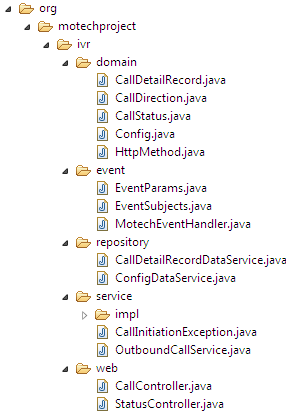
***In short the IVR module of Motech has been generalized to support any VXML/CCXML complaint IVR framework.***

The below directory structure represent the **Base IVR** **module** inside the **MOTECH** framework –



**IVR Module in MoTech**

**File view on Base IVR module inside the MOTECH framework –**



**Classes & Files in Base IVR Module of MoTech**

## High level Design of MOTECH Base IVR module

The latest version of Motech (version 0.25) has been developed in such a way that it integrates with the IVR frameworks, which are VXML/CCXML compatible.

Any VXML/CCXML IVR service provider can be integrated with Motech through the IVR module provided by Motech and hence any task can be scheduled for providing Outbound and inbound call features.

Motech IVR

domain

* CallDetailRecord.java
* CallDirection.java
* CallStatus.java
* Config.java

Motech IVR Module

event

* EventParams.java
* MotechEventHandler.java

repository

CallDetailRecordDataService.java

ConfigDataService.java



service

* OutboundCallService.java
* OutboundCallServiceImpl.java

**MOTECH IVR Module**

**Package Information:**

**Domain** –

* Contains Config class which holds IVR provider configuration, represents how the IVR module interacts with an IVR provider.
* Contains CallDetailRecord class which holds all the details regarding the IVR call such as Config object, Timestamp, Call Direction, and Call Status etc.
* Contains Java Enums as Call Status, Call Direction, HttpMethod

**Event** –

* Contains final Classes as EventParams and EventSubjects which holds Possible Event payloads and event subjects.
* Contains MotechEventHandler class which listens to the ivr\_initiate\_call MotechEvent and calls initiateCall.

**Repository** –

* Contains two classes CallDetailRecordDataService and ConfigDataService which contains MDS generated CallDetailRecord and ConfigDataService database queries.

**Service** –

* Contains an interface OutboundCallService which is used to initiate an outgoing call and its implementation class OutboundCallServiceImpl which generates & sends an HTTP request to an IVR provider to trigger an outbound call.

# Support of a third-party IVR framework Integration in MOTECH

## The latest MOTECH framework (0.25) has been modified to integrate VXML/CCXML complaint IVR framework.

## Example: KooKoo IVR Framework Integration with Motet

Kookoo is an India based IVR service provider which gives its own proprietary standards for generating XML response hence it does not support VXML/CCXML. So Kookoo cannot be integrated with the Motech IVR module. As Kookoo is an India based IVR provider and provides India based Telephone numbers and good Tamil language support so we are using Kookoo as our IVR provider with Motech.

**IMP!!** As Kookoo does not support VXML/CCXML standards so a new Motech module is developed with the custom codes that will support Kobo’s specification and can respond to Kobo’s HTTP request.

**MOTECH Server**

Calls KooKoo



HTTP Request

Motech Core Platform

**KooKoo**

**IVR Server**

Call details like caller no. etc.

Custom Module Integra

Call Sent

KooKooIVR

Module

**KooKoo Tunes** (XML) Commands (like <playtext>,<playaudio> etc.)

HTTP Response

**KooKooIVR module integration with MOTECH**

**Steps on how KookooIVR module works**

1. KooKoo server gets a call from caller.
2. Make a HTTP request to MOTECH Server’s KooKooIVR Module, passing along the caller detail.
3. Caller detail, caller no. etc. are stored in HTTPRequest Object.
4. Appropriate action is taken based on the request made and appropriate method is called.
5. This method generate the XML (KooKooXML Commands) as per the request
6. These commands are passed to KooKoo Server as response.
7. KooKoo performs the action as the command and thus interact with the user.

## KooKoo IVR Execution flow in Integration with MOTECH

# KooKoo call Flow for IVR Execution

1. **HW/PL Dials the Number**

Enter the pin

C:\Users\kumari.r\Desktop\Capture.PNG

**5. Perform Action**

**4.** **Kookoo performs the action specified in the XML on your behalf an interacts with the caller**

**MOTECH Server**

**Motech Core Platform**

HTTP Request

Ex. http://www.mysite.com/myapp.jsp

**2. Kookoo Answers the call, and makes a HTTP request to the URL of Motech’s KookooIVR Module.**

**KooKoo Server**

**KookooIVR Module**

Response

KooKooXML (XML) Commands (like <playtext>,<playaudio> etc.)

3**. Motech Application’s KookooIVR Module receives the request performs some action and sends an XML response back to kookoo to perform some actions. The response XML is called Kookoo tunes.**

**Call flow between KooKoo IVR and MOTECH’s KookooIVR Module**

**Steps of the Call Flow**

Step 1: A HW/PL dials the number.

Step 2: Kookoo Answers the call, and makes a HTTP request to the URL of MOTECH’s KookooIVR module. Let us say our URL is

<http://localhost:8080/motech-platform-server/module/KookooIVR/getResponse>

Step 3: MOTECH’s KookooIVR module receives the request performs some action and sends an XML response back to kookoo to perform some actions. The response XML is called Kookoo tunes.

Step 4:Kookoo performs the action specified in the XML on your behalf and interacts with the caller.

Step 5**:** Perform Action.

# Pattern for Multi-language support in IVR

For integrating multi language in MOTECH framework, recorded audio files must be used. That is, if we want IVR messages to be played in different language, we need to record those messages in that particular language and play through IVR systems. These recorded files must be in “**WAV**” file format.

**Text to speech engine is not used in MOTECH for multi-language support**.

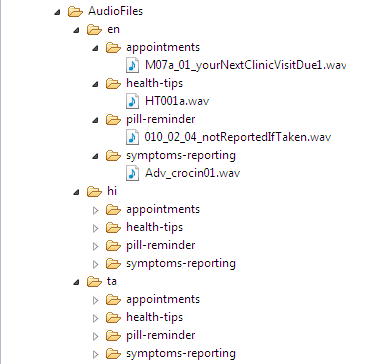
## Directory Structure of multi-language WAV files

The wav files must be kept deployed as a separate web application which will be accessible to IVR Systems. The wav files for different language must be stored in separate folders named with language code.

**Example of Language Code**

* English : ‘en’
* Hindi : ‘hi’
* Tamil : ‘ta’
* Kannada : ‘ka’
* Malayalam : ‘ml’
* Telugu : ‘te’
* Gujarati : ‘gu’
* Bengali : ‘bn’
* Marathi : ‘mr’
* Oriya : ‘ori’

Sample directory structure to store multi language WAV files in MOTECH:



## How this works

To integrate different languages in MOTECH framework, we can use wav files to play the content in a specific language. These wav files are kept deployed as **a separate web application on any public server under a directory called “AudioFiles”**.

We can create different folders for different language with language code as folder name. For example, for English we have “en” folder, for Tamil we have “ta” folder.

**B.**

Inside each language folder, the sub folders will be same. That is, directory structure inside Tamil folder will be same as that of English folder. Inside these sub folders, the wav files are residing. The name of these wav files will also be same in each language folders. Only the content of the files will be in different language. That is, “en” folder will contain the wav files recorded in English and “ta” folder will contain the wav files recorded in Tamil.

# References

* <http://www.motechproject.org/devdocs/installation.html>
* <http://motechsuite.org/index.php/installing-motech>
* <http://motechsuite.org/index.php/installing-motech>
* <http://www.motechsuite.org/>
* <http://www.motechproject.org/>
* <https://code.google.com/p/motech/>
* <http://www.kookoo.in/>

**~~ end of doc ~~**