Freedom Fone v.1.0

Installation Guide

Part 2:
Middle ware and application layer

Table of Contents

1. Data flow in Freedom Fone....1
2. Assumptions....2
3. Installation of CakePHP in production mode....2
   3.1 Install packages....2
   3.2 Download and unpack CakePHP....3
   3.3 Permissions....3
   3.4 Rewrite rules....4
   3.5 Configure CakePHP....5
      3.5.1 Change salt value....5
      3.5.2 Core path of CakePHP....5
   3.6 Database setup....5
      3.6.1 Create database for CakePHP....5
4. Freedom Fone configuration in Cake....6
   4.1 Configuration files....6
   4.2 Cronjob....9
5. Incoming events....10
   5.1 Spooler....10
      5.1.1 Create spooler database....10
      5.1.2 Provide access to spooler to CakePHP....11
   5.2 The incoming dispatcher....11
      5.2.1 Configuration file....11
      5.2.2 Event socket library (ESL)....13
   5.3 Tests available....13
   5.4 Run the incoming dispatcher....13
   5.5 Test the dispatcher....14
6. Outgoing events....14
   6.1 Configuration files....15
      6.1.1 Outgoing dispatcher....15
      6.1.2 Dialer engine....15
   6.2 Run scripts....15

1. Data flow in Freedom Fone

Let us start with a brief walk-through of the architecture of Freedom Fone v.1.0.

Incoming SMS and GSM calls enters Mobigater and is forwarded to FreeSWTICH via gsmopen. For each incoming event, the event manager of FreeSWITCH throws an event.

The incoming dispatcher is connected and authenticated to FreeSWITCH, and subscribed to a set of Freedom Fone specific events. That means that the dispatcher listens only for those events it is subscribed to, and fetches the data related to those types of event.

When an event arrives to the dispatcher, it is converted to XML format. An XSL template is applied to the XML file, in order to filter out the data needed for the application, and drop the rest of the data.

A set of rules are applied to the XML output in order to match the event with a certain application. Finally, the event data is parsed to an SQL query and inserted into an application specific table in the spooler database (spooler_in).

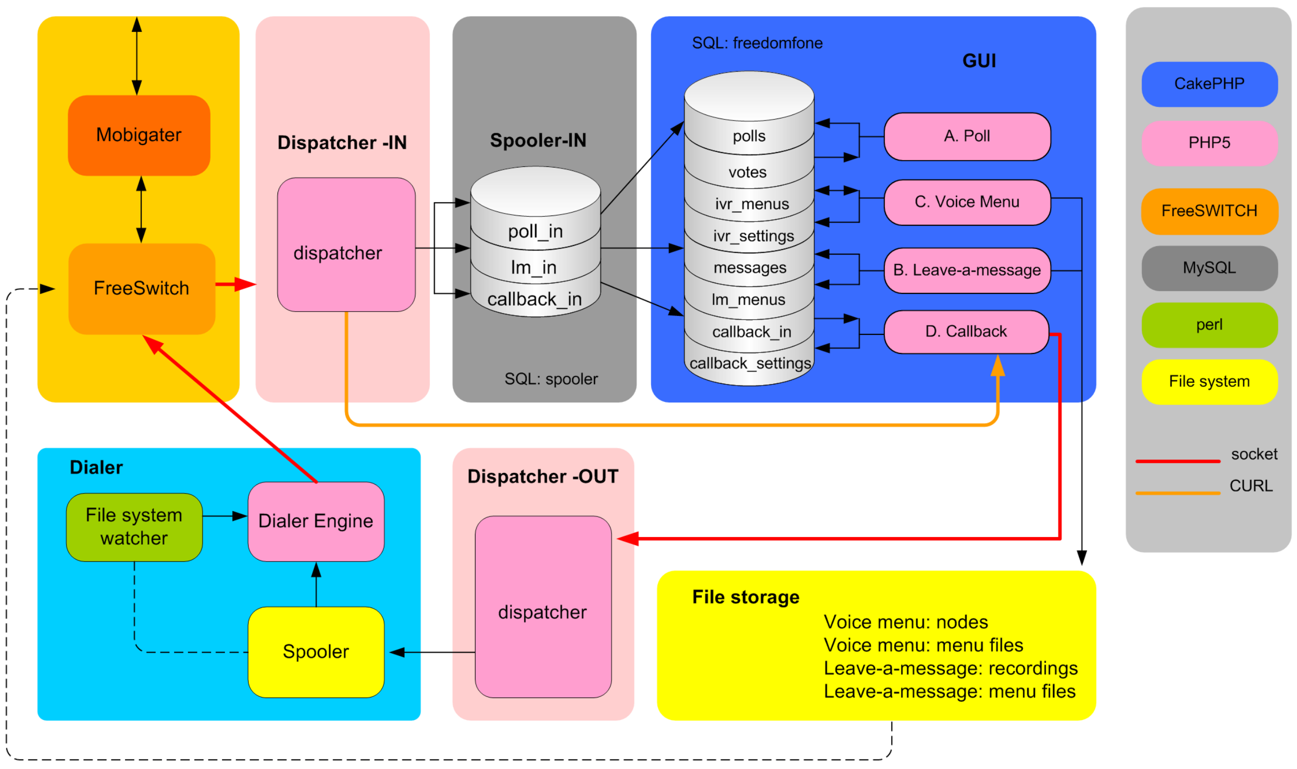
The CakePHP application fetches data from the spooler using the spooler API (*spooler_ff.php).* This can be done manually (by URL request), by crontab or via a CURL request.

The event data is finally inserted in the freedomfone database, which is a part of the CakePHP application.

Freedom Fone v.1.0 supports outgoing events for the Callback service. When an outgoing event is requested by Freedom Fone, the CakePHP application connects to the outgoing dispatcher to “order” the job.

The outgoing dispatcher is responsible for creating spooler jobs with the correct parameters for establishing a call or sending an SMS.

The dialer engine is responsible for executing the spooler jobs. The dialer engine is also responsible for managing retries of calls due to busy GSM channels or unanswered calls.



2. Assumptions

This installation guide assumes that you have checked out the Freedom Fone SVN. We will assume that you have your local SVN repository under */usr/src/pictus/*.

This installation guide will focus on the following four folders:

```
#/usr/src/pictus/gui/app
#/usr/src/pictus/dispatcher_in
#/usr/src/pictus/dispatcher_out
#/usr/src/pictus/dispatcher_dialer
```

3. Installation of CakePHP in production mode

3.1 Install packages

CakePHP requires the following packages to run Freedom Fone v.1.0: apache2, php5,php5-cli, php5-xsl, mysql-server, php5-mysql, php5-curl and lame.

```
#apt-get install apache2 php5 php5-cli php5-xsl
#apt-get install mysql-server php5-mysql php5-curl
#apt-get install lame
```

Restart the database server and the web server.

```
#/etc/init.d/mysql restart
#/etc/init.d/apache2 restart
```

3.2 Download and unpack CakePHP

Download the latest stable release of CakePHP (cakephp.org).

Unpack it and save it outside of your SVN folder.

```
#wget http://cakeforge.org/frs/download.php/734/cake_1.2.5.tar.gz
#tar -zxvf cake_1.2.5.tar.gz
```

Have a look inside of the cake folder to familiar yourself with its content

```
#cd cake_1.2.5
#ls -a

app cake README index.php vendors .htaccess
```

app is the directory where all application specific code is placed. Hence, all code specific to Freedom Fone is placed here.

The *cake* directory contains the Cake engine, and should never be touched.

The *vendors* directory contains third party code.

Copy all Cake files, except the *app* directory, to your web folder

```
#cd cake_1.2.5
#cp -Rf cake /var/www/freedomfone
#cp -Rf vendors /var/www/freedomfone
#cp -Rf index.php /var/www/freedomfone
#cp -Rf .htaccess /var/www/freedomfone
```

Make a symbolic link for the *app directory* (from the SVN) to the Freedom Fone web directory (/var/www/freedomfone).

```
#cd /var/www/freedomfone
#ln -s /usr/src/pictus/gui/app .
```

3.3 Permissions

Make sure that /app/tmp folder is writable by the webserver

```
chown -Rf www-data.www-data /var/www/freedomfone/app/tmp/
```

3.4 Rewrite rules

Freedom Fone v.1.0 requires the *rewrite module* of Apache to be enabled. Enable the module and restart the web server.

```
#a2enmod  rewrite
#/etc/init.d/apache2 restart
```

Check that the module is enabled:

```
vi /etc/apache2/mods-enabled/rewrite.load

LoadModule rewrite_module /usr/lib/apache2/modules/mod_rewrite.so
```

Open the Apache config file and make sure that the value of AllowOverride is set to ***all***.

```
# vi /etc/apache2/sites-enabled/freedomfone

        <Directory /var/www/freedomfone>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride all
        </Directory>
```

Make sure that these *three directories* contain a .htaccess file with the following data:

Directory: Cake

```
<IfModule mod_rewrite.c>
   RewriteEngine on
   RewriteRule    ^$ app/webroot/    [L]
   RewriteRule    (.*) app/webroot/$1 [L]
</IfModule>
```

Directory: cake/app

```
   <IfModule mod_rewrite.c>
    RewriteEngine on
    RewriteRule ^$ webroot/ [L]
    RewriteRule (.*) webroot/$1 [L]
    </IfModule>
```

Directory: cake/app/webroot

```
    <IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteRule ^(.*)$ index.php?url=$1 [QSA,L]
    </IfModule>
```

Restart the webserver

```
/etc/init.d/apache2 restart
```

3.5 Configure CakePHP

3.5.1 Change salt value

Change the value of 'Security.salt' in *app/config/core.php* to a salt value specific to your application (any sequence of numbers and letters, around 40 characters long)

```
#vi /var/www/freedomfone/app/config/core.php
```

Save and exit

3.5.2 Core path of CakePHP

Edit the "CAKE_CORE_INCLUDE_PATH" to reflect your setup.

```
#vi /var/www/freedomfone/app/config/core.php

if (!defined('CAKE_CORE_INCLUDE_PATH')) {
     define('CAKE_CORE_INCLUDE_PATH', '/var/www/freedomfone');
 }
```

3.6 Database setup

3.6.1 Create database for CakePHP

Create a database called "freedomfone". This is the database that CakePHP will use to store all its application specific data.

```
#mysqladmin -u root -p create freedomfone
```

Log in to it

```
#mysql -u root -p freedomfone
```

Set permissions for the user "freedomfone". Replace <password> for your password.

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER,
CREATE TEMPORARY TABLES, LOCK TABLES ON freedomfone.* TO
'freedomfone'@'localhost' IDENTIFIED BY '<password>';
```

Reload the grant table

```
mysql>flush privileges;
```

Exit mysql

```
mysql>exit;
```

Dump the database schema (located under /app/install) into your freedomfone database.

```
#mysql -u freedomfone -p freedomfone < freedomfone.sql
```

Make sure that CakePHP can connect to the database by editing the file database.php

```
#vi /var/www/freedomfone/app/config/database.php

 var $default = array(
          'driver' => 'mysql',
          'persistent' => false,
          'host' => 'localhost',
          'login' => 'freedomfone',
          'password' => '<password>',
          'database' => 'freedomfone',
          'prefix' => '',
  );
```

4. Freedom Fone configuration in Cake

4.1 Configuration files

Freedom Fone v.1.0 uses two files for configuration; core.php and config.php. Core.php includes CakePHP specific settings, while config.php contains Freedom Fone specific settings.

Let's open the config.php file and have a look at the settings.

```
#vi /var/www/freedomfoneapp/config/config.conf
```

You will find the following parameters:

```
poll_in
lm_in
callback_in

LM_SETTINGS
LM_DEFAULT
IVR_SETTINGS
IVR_DEFAULT
CALLBACK_DEFAULT
```

Start by setting the database parameters (the password) for the (incoming) spooler tables (poll_in, lm_in, callback_in).

These spooler tables are not yet created. We will take care of that soon.

```
$config['poll_in']= array(
             'host'     => 'localhost',
             'user'     => 'poll_in',
             'password' => '<password>',
             'database' => 'spooler_in',
             );


$config['lm_in']= array(
             'host'     => 'localhost',
             'user'     => 'lm_in',
             'password' => '<password>',
             'database' => 'spooler_in',
             );

$config['callback_in']= array(
               'host'     => 'localhost',
               'user'     => 'callback_in',
               'password' => '<password>',
               'database' => 'spooler_in',
               );
```

LM_SETTINGS defines path and folder names for uploaded messages and the voice menu for the Leave-a-Message application. Adjust the host variable to reflect your setup.

It is **not** recommended to change the other LM_SETTINGS. If you do so, the corresponding changes must be done in FreeSWITCH.

```
$config['LM_SETTINGS'] = array(
      'host'              => 'http://freedomfone.org/',
      'path'              => 'freedomfone/leave_message/',
      'dir_messages'      => 'messages/',
      'dir_menu'          => 'audio_menu/',
      'dir_conf'          => 'conf/'
          );
```

LM_DEFAULT defines the default fallback messages of the Leave-a-Message voice menu. If no alternative text message has been provided by the user, these are the fallback messages that will be synthesized by FreeSWITCH. You may edit these text messages as you wish.

```
$config['LM_DEFAULT']=  array(
 'lmWelcomeMessage'=> 'Welcome to Freedom Fone Leave a Message Service!',
 'lmInformMessage' => 'Record your message after the beep. To Finish, Press #',
 'lmInvalidMessage'=> 'Invalid option. Please try again.',
 'lmLongMessage'   => 'Your message is too long, to the point, please!',
 'lmSelectMessage' => 'To Play..press *. To Delete.. press 0. To Save..press  1',
 'lmDeleteMessage' => 'Your message has been deleted!',
 'lmsaveMessage'   => 'Thank you!',
 'lmGoodbyeMessage => 'Goodbye'
      );
```

IVR_SETTINGS defines path and folder names for audio files belonging to the Voice Menu application. Adjust the host variable to reflect your setup.

It is **not** recommended to change the other IVR_SETTINGS. If you do so, the corresponding changes must be done in FreeSWITCH.

```
$config['IVR_SETTINGS'] = array(
                 'host'              => 'http://freedomfone.org/',
                 'path'              => 'freedomfone/ivr/',
                 'curl'              => 'xml_curl/',
                 'dir_node'          =>  'nodes/',
                 'dir_menu'          =>  'ivr/',
                 'dir_conf'          =>  'conf/'

                 );
```

IVR_DEFAULT defines variables concerning the Voice Menus. You are allowed to modify the three Messages as you with. Do **not** edit the variable “parent_ivr”.

```
$config['IVR_DEFAULT']= array(
                 'parent_ivr'       => 'freedomfone_ivr_'.IID,
                 'ivrIndexMessage'  => 'To repeat the menu, press 9.',
                 'ivrInvalidMessage'  => 'Invalid option. Please try again.',
                    'ivrExitMessage'   => 'Thank you and good bye.');
```

CALLBACK_DEFAULT defines the FreeSWITCH extensions (defined in the dialplan) for the Voice Menu (IVR) and the Leave-a-message service.

This array also holds the default values for restricted access of the callback service (Max 10 callbacks during 24 hours).

```
$config['CALLBACK_DEFAULT']=    array(
                'sms_code'         => 'CALLBACK',
                'response_type'     =>  array (
                        '4000' =>'IVR',
                        '2000' =>'Leave-a-Message'
                        ),
                'limit_user'       => '10',
                'limit_time'       => '24');
```

The installation and configuration of the application side of Freedom Fone is now completed. You can direct your browser to http://yourdomain/freedomfone and browse through the menus. Before we can manage incoming and outgoing events, we need to setup the dispatchers, spooler and the dialer.

4.2 Cronjob

Every CakePHP application needs to pull/push data from the spooler on a regular basis. This is done through a "refresh" method in each controller.

```
#http://freedomfone.org/freedomfone/polls/refresh
#http://freedomfone.org/freedomfone/messages/refresh
#http://freedomfone.org/freedomfone/callback_in/refresh
```

These methods are called by a cronjob every minute. They can also be executed manually by requesting the pages as above[1] (a blank page will be provided as result).

Add a cronjob for the poll and the Leave-a-Message applications, that calls the refresh method every minute.

```
#crontab -e


*/1 * * * * /usr/bin/wget -O - -q -t 1 http://localhost/freedomfone/polls/refresh
```

1 Replace freedomfone.org with your own domain name.

```
*/1 * * * * /usr/bin/wget -O - -q -t 1 http://localhost/freedomfone/messages/refresh
```

For the Callback application, we will use another method (CURL) to refresh the data.

5. Incoming events

For incoming events, we need to setup the incoming dispatcher (php) and the spooler (database).

5.1 *Spooler*

5.1.1 Create spooler database

The incoming dispatcher exchanges information with the CakePHP applications through a set of database tables. For this purpose we will create a database called *spooler*

```
#mysqladmin -u root -p create spooler_in
```

Log in to the database spooler as root

```
#mysql -u root -p spooler_in
```

Set permissions for the user “dispatcher”. Replace <password> for your password.

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER,
CREATE TEMPORARY TABLES, LOCK TABLES ON spooler_in.* TO
'dispatcher_in'@'localhost' IDENTIFIED BY '<password>';
```

Reload the grant table

```
mysql>flush privileges;
```

The *spooler_ in database* contains application specific tables. Dump the database schema (/dispatcher/install) into the *spooler_ in* database.

```
#mysql -u dispatcher_in -p spooler_in < spooler_in.sql
```

The spooler contains three tables:

poll_ in: contains incoming SMS data to the Poll service

lm_ in: contains incoming data for audio messages to the Leave a Message service.

callback_ in: contains incoming data for SMS or tickles to the Callback service.

5.1.2 Provide access to spooler to CakePHP

CakePHP needs to access the *spooler* to pull data from FreeSWITCH. Each application accesses the spooler through an application specific table and user.

Grant permissions to user **poll_in** to access the table **poll_in** in the database **spooler_in** using <password>. The <password> must be replaced with the password you stated for this user/table in the CakePHP configuration file (/app/config/config.php)

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE ON spooler_in.poll_in TO
'poll_in'@'localhost' IDENTIFIED BY '<password>';
```

Do the same for the user/table lm_in and callback_in:

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE ON spooler_in.lm_in TO
'lm_in'@'localhost' IDENTIFIED BY '<password>';

mysql>GRANT SELECT, INSERT, UPDATE, DELETE ON spooler_in.callback_in TO
'callback_in'@'localhost' IDENTIFIED BY '<password>';
```

Reload the grant table

```
      mysql>flush privileges;
```

All database tables and permissions are now completed for the spooler.

5.2 The incoming dispatcher

5.2.1 Configuration file

The incoming dispatcher's configuration settings can be found in the *dispatcher_in/config.php* file. The settings that you should modify, are marked with an asterisk (*).

Parameter: ParentXML

Description: Parent tag used in the XML object created by the dispatcher.

Value: ff-event (do not change)

Parameter: DefaultXSL

Description: The default xsl filter to be used for incoming event.

Value: default.xsl (do not change)

Parameter: DirXSL

Description: Directory for xsl filters

Value: templates/ (do not change)

Parameter: LogFile

Description: Logs all events related to incoming events.

Value: ../log/dispatcher_in.log (do not change)

Parameter: LocalDomain (*)

Description: The domain of you installation. Domain name or IP address. With trailing slash.

Value: http://my.domain.org/freedomfone/ (*)

Parameter: SocketParam

Description: Socket parameters for connecting to FreeSWITCH

Values: (do not change)

```
host = 127.0.0.1
port  = 8021
pass  = ClueCon
timeout=3
stream_timeout = 0.5
```

Parameter: DispatcherDB (*)

Description: Database settings for the spooler_in database. These values must reflect the settings you just did for the spooler_in database.

Values:

```
host= localhost
user= dispatcher_in
password= <your password>
database= spooler_in
```

Parameter: AllowCURL

Description: Lists applications where the refresh method should be executed by means of a CURL request.

Value: array('callback_in' => 'callback_in');

5.2.2 Event socket library (ESL)

The incoming dispatcher uses the php class **fs_sock,** to connect to FreeSWITCH, monitor incoming events, and fetch data.

The fs_sock class is located in the ***esl*** (event socket library) directory.

```
#cd /usr/src/pictus/esl
```

5.3 Tests available

The *tests* directory (/usr/src/pictus/dispatcher_in/tests) contains scripts that facilitates testing of the system

sendSMS.php

Connects to FreeSwitch and generates 1 SMS event every second. This script is useful for testing the poll application. To run the script:

```
#php5 sendSMS.php <code> <message>
```

For example, to vote No is the “Election” poll, run

```
#php5 sendSMS.php Election No
```

5.4 Run the incoming dispatcher

The dispatcher can be found at

```
#cd /usr/src/pictus/dispatcher_in
```

To run the dispatcher, execute the following command:

```
#php5 ./dispatcher_in.php
```

The incoming dispatcher must always run. If not, incoming events to FreeSWITCH is not captured by Freedom Fone.

5.5 Test the dispatcher

Run the sendSMS.php script to auto-generate incoming SMS in FreeSWITCH.

Run the dispatcher, and see how the SMSs are received, processed and inserted into the spooler (poll_in).

6. Outgoing events

In Freedom Fone v.1.0, outgoing events are only generated by the Callback service. An outgoing event requires the following four components:

1. Outgoing dispatcher
2. Outgoing spooler
3. File System Watcher (dialer.sh)
4. Dialer engine

The code is structured as follows:

```
dispatcher_out/
dispatcher/out/dispatcher_out.php
dispatcher_out/config.php

dialer/
dialer/spooler
dialer/spooler/incoming
dialer/spooler/saved

dialer/config.php
dialer/dialer.sh
dialer/dialer_engine.php
```

The O**utgoing Dispatcher** and the **File System Watcher** are two scripts that always need to run.

The **outgoing spooler** is directory containing spooler job, and requires no configuration.

The **Dialer engine** is a script that is executed by the File System Watcher when a new spooler job is available.

6.1 Configuration files

6.1.1 Outgoing dispatcher

The *config.php* file in the *dispatcher_ out* directory contains configuration parameters for the outgoing dispatcher. If you have followed the installation guide of Freedom Fone, there is no need to alter any parameter.

```
BasePath          /usr/src/pictus/
SpoolerOutDir     dialer/spooler/incoming/
SpoolerTmpDir     /tmp/
LogFile           ../log/dispatcher_out.log

$_SpoolerOut = array(
           'host'  => '127.0.0.1',
           'port'  => '9999');
```

6.1.2 Dialer engine

The *config.php* file in the *dialer* directory contains configuration parameters for the dialer engine. If you have followed the installation guide of Freedom Fone, there is no need to alter any parameter.

```
LogFile           ../log/dialer.log
SpoolerSavedDir   dialer/spooler/saved/
BasePath          /usr/src/pictus/
Redial            5

$_SocketParam = array(
            'host'=>'localhost',
            'port'=>'8021',
            'pass'=>'ClueCon',
            'timeout'=>3,
            'stream_timeout'=>0.5
            );
```

6.2 Run scripts

Run the outgoing dispatcher:

```
      #cd /usr/src/pictus/dispatcher_out
      #php5 ./dispatcher_out.php
```

Run the file system watcher:

```
#cd /usr/src/pictus/dialer
#/bin/bash ./dialer.sh
```