



# **Marbil Broadcaster Integration Specification**

**AfriGIS**

**17 May, 2009**

Version 0.14

## Table of Contents

|   |    |
|---|----|
| Table of Contents .....                               | ii |
| Marbil.....   | 1  |
| 1. Introduction .....                                 | 1  |
| 1.1 Purpose of the Document.....                      | 1  |
| 1.2 Overview .....                                    | 1  |
| 1.3 Choosing your integration method .....            | 1  |
| 2. Marbil SOAP Integration.....                       | 3  |
| 2.1 GetAdvertNow method.....                          | 3  |
| 2.2 GetAllAdverts method.....                         | 6  |
| 2.3 Using Marbil from .NET .....                      | 6  |
| 2.4 Using Marbil from Java .....                      | 7  |
| 2.5 Using Marbil from PHP.....                        | 8  |
| 2.6 Integrating with Vodacom4me.....                  | 9  |
| 2.6.1 Rendering a text advert for XHTML browsers..... | 9  |
| 2.6.2 Rendering a text advert for WML browsers.....   | 10 |



---

|   |           |
|---|-----------|
| <b>2.7 Integrating with Vodafone Live.....</b>    | <b>10</b> |
| <b>2.7.1 Rendering a text advert in PML.....</b>  | <b>10</b> |
| <b>3. Marbil Integration by HTTP request.....</b> | <b>11</b> |
| <b>3.1 Java Example.....</b>                      | <b>12</b> |

# Marbil

## 1. Introduction

### 1.1 Purpose of the Document

This document describes how a Broadcaster can integrate with the Marbil Mobile Marketing system in order to request targeted advertisements.

### 1.2 Overview

Marbil Mobile Marketing allows you structured entry into an entirely new marketing channel. By using our network of broadcasters on different mobile bearers you can reach your target audience through simple one-line adverts or more elaborate splash screen layouts.

Marbil allows you the most targeted marketing opportunity yet. When you use Marbil you will get your message delivered directly to your customer's handset and you will get the statistics to prove it - gone are the days of spraying and praying.

Through our simple, non intrusive technology, consumers will be exposed to your adverts in the form of scrolling adverts, footnotes or splash screens while they are busy using the Broadcaster's application for the intended purpose. For example an instant messaging (IM) application: While people are talking with one another your adverts are displayed at the top in a separate window, non-intrusive.

The Marbil system facilitates both Advertisers and Broadcasters in the Mobile Marketing space. The Advertiser is the entity that wants to promote his products on the mobile marketing channel. The Broadcaster is the entity that provides mobile services and wants to add advertisements to his mobile offering.

The advertisers use a web interface to configure and set up advertising campaigns. The Broadcaster uses a SOAP web service (see [section 2](#)) or an HTTP request (see [section 3](#)) to request targeted advertisements.

Both advertisers and broadcasters can log onto <http://www.marbiladserver.co.za/ot/> to manage their accounts.

### 1.3 Choosing your integration method

Currently 2 methods of marbil integration are available to the broadcaster.

The first being the SOAP web service, which has been around for some time. SOAP (as described in detail in section 2 of this documentation) requires more

overhead and development time from the broadcaster. The advantage is a more robust system able to return one or many adverts simultaneously, as well as errors if needed. As the SOAP web service returns the parts of the advert (text, image URL, click link, etc) it forces the broadcaster to build their own HTML to render the advert. This could be seen as a pro or a con, depending on the type of platform the advert is being rendered on and the amount of control the broadcaster requires.

The second is the HTTP request method, which is new to this version of the documentation. The HTTP request method involves a provided Marbil URL (detailed in section 3 of this documentation), which the broadcaster would call with certain parameters from code. The response is the HTML of the advert itself, so all that would be required of the broadcaster is to open a connection to the URL and print the response wherever the advert should appear. This method is simpler to use, will use less bandwidth in contacting marbil and is also more flexible, because new parameters can be added at any time whilst remaining backward compatible.

## 2. Marbil SOAP Integration

The broadcaster integrates with Marbil by calling a SOAP web service. The web service is SOAP 1.1 and 1.2 compatible. The latest URL for the web service is:

<http://www.marbiladserver.co.za/Marbil/Services/Marbil080.asmx?WSDL>

The web service has two methods:

- **GetAdvertNow** – This method returns adverts and assumes immediately that the adverts have been delivered. This is the typical method to call when one advert is needed by the broadcaster.
- **GetAllAdverts** – This method returns X amount of unique adverts in an array of advert objects. This is ideal for use in a sponsored link type display, where a number of links should be rendered alongside each other.

This method will be described in detail in the following sections.

### 2.1 GetAdvertNow method

The GetAdvertNow method has the following parameters:

|          |   |
|----------|---|
| Username | Required. Your username, which is provided to you by the system administrator.  |
| Password | Required. Your password, which is provided to you by the system administrator.  |
| Amount   | Optional. Not yet defined. ( <a href="#">See 4. GetAllAdverts</a> )   |
| AgeGroup | Optional. The age for the user.   |
| Gender   | Optional. The gender of the user. This could be “male” or “female”  |
| Image    | One of the following values:<br>“Yes” – to return the image as a base 64 encoded stream.<br>“AfriGISLocal” – to return just a URL for the image.<br>Blank – To return a text advert.<br>“deviceAwareImage” – Intended for WAP broadcasters – functionally the same as AfriGISLocal, but Marbil can detect and store the users device. Currently does not support animated gifs. |

|                        |  |
|------------------------|--|
| <b>ClientRef</b>       | <b>Suggested. A unique reference to the user of your site. Currently used for statistical analysis only. Preferably the MSISDN of logged in user if available.</b>   |
| <b>ClientContext</b>   | <b>Suggested. A reference to the page or site requesting the advert. Currently used for statistical analysis only.</b>   |
| <b>ClientUserAgent</b> | <b>Suggested. Broadcaster should send the client User-Agent header to support device specific profiling.</b>   |
| <b>ClientIPAddress</b> | <b>Optional. The IP address of the user.</b>   |
| <b>Location</b>        | <p><b>Optional. Indicates the location area of interest to the user. The location can be specified as a single point (sp) ,as a bounding box (bb) or as a point with radius {ra}.</b></p> <p><b>The format for a single point request: sp{latitude, longitude}</b><br/> <b>Example: "sp{-25.7461, 28.2344}"</b></p> <p><b>The format for a bounding box request: bb{bottom left lat, bottom left long, top right lat, top right long}</b><br/> <b>Example: "bb{-34.833, 16.4549, -22.126, 32.8913}"</b></p> <p><b>The format for a point and radius request: ra{latitude,longitude, radius(km)}</b><br/> <b>Example: "ra{-25.7461, 28.2344,10}</b></p> |

The **GetAdvertNow** method returns adverts and assumes immediately that the adverts have been delivered. This method can reduce network traffic and resource consumption for high volume applications, but the broadcaster must be certain that the advert will be delivered to the end user. This method replaces the **GetAdvert** and **CompleteDelivery** methods from Marbil055 (or version 0.6 of this document).

The following is an example of a SOAP request for this method:

```
POST /marbil/services/marbil080.asmx HTTP/1.1
Host: www.marbiladserver.co.za
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://www.marbil.co.za/GetAdvertNow"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetAdvertNow xmlns="http://www.marbil.co.za/">
      <username>string</username>
      <password>string</password>
      <amount>string</amount>
      <agegroup>string</agegroup>
      <gender>string</gender>
      <race>string</race>
      <location>string</location>
    </GetAdvertNow>
  </soap:Body>
</soap:Envelope>
```

```

<category>string</category>
<image>string</image>
<clientRef>string</clientRef>
<clientContext>string</clientContext>
<clientUserAgent>string</clientUserAgent>
<clientIPAddress>string</clientIPAddress>
</GetAdvertNow>
</soap:Body>
</soap:Envelope>
    
```

The method returns an array of Advert object. Each advert contains the following fields:

|                          |   |
|--------------------------|---|
| <b>AdvertName</b>        | A short name for the advert.  |
| <b>AdvertText</b>        | The text for the advert. This is what will be shown to the user.  |
| <b>TransactionID</b>     | The ID generated for this specific request.   |
| <b>Status</b>            | The status code for this advert. This could be either "OK" or "ERROR"   |
| <b>StatusDescription</b> | A detailed error description in case the status code is "ERROR".  |
| <b>RedirectType</b>      | Redirect type can either be 1 (HTTP Internal) or 2 (HTTP External)  |
| <b>RedirectValue</b>     | Redirect value contains a URL. If an advert contains this RedirectValue the URL should be navigated to if the user clicks the advert. |
| <b>ImageData</b>         | Image data is a URL describing a direct path to the image.  |

The following is an example of a SOAP response for this method:

```

HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetAdvertNowResponse xmlns="http://www.marbil.co.za/">
      <GetAdvertNowResult>
        <Advert>
          <advertName>string</advertName>
          <advertText>string</advertText>
          <imageData>string</imageData>
          <redirectType>int</redirectType>
          <redirectValue>string</redirectValue>
          <status>string</status>
        </Advert>
      </GetAdvertNowResult>
    </GetAdvertNowResponse>
  </soap12:Body>
</soap12:Envelope>
    
```

```
<statusdescription>string</statusdescription>
<transactionId>string</transactionId>
</Advert>
<Advert>
  <advertName>string</advertName>
  <advertText>string</advertText>
  <imageData>string</imageData>
  <redirectType>int</redirectType>
  <redirectValue>string</redirectValue>
  <status>string</status>
  <statusdescription>string</statusdescription>
  <transactionId>string</transactionId>
</Advert>
</GetAdvertNowResult>
</GetAdvertNowResponse>
</soap12:Body>
</soap12:Envelope>
```

## 2.2 GetAllAdverts method

The GetAllAdverts method is designed to return a certain number of unique available adverts. It has a very similar signature to the GetAdvertsNow method, please refer to [3.GetAdvertsNow](#) for a more detailed description.

### Changes:

The “Amount” parameter can now be used to specify a maximum number of unique adverts to receive. GetAllAdverts will return this specified number of adverts within the array of adverts sent in GetAdvertsNow. If you, the broadcaster, specify a limit higher than the number of available adverts then the maximum possible unique adverts will be returned. However many unique adverts are available, you will never receive more than the “Amount” specified. The adverts are selected through the bidding system.

## 2.3 Using Marbil from .NET

The following code shows how to call the Marbil web service from C#, after a “web reference” called “Marbil” has been added using the IDE:

```
MarbilAdServer.Marbil marbil = new TestAPP.MarbilAdServer.Marbil();
MarbilAdServer.Advert[] adverts = Marbil.GetAdvertNow("BroadcasterName",
"BroadcasterPassword", "", "", "", "", "", "", "afrigisLocal", "27820001111",
"testapp");

foreach (Marbil.Advert advert in adverts)
{
    if (advert.status.Equals("ok"))
    {
```

```
String refNumber = advert.refNumber;  
String text = advert.advert;  
  
//deliver advert to the user  
//...  
if (response.status.Equals("ok"))  
{  
    //everyone is happy  
}  
}  
}
```

## 2.4 Using Marbil from Java

From Java the Marbil web service can be called by using the JAX-RPC library or the JAX-WS library.

The following code show an example on how to call the web service from Java, after a “web service reference” has been added using NetBeans 5.0:

```
String sAd = null;  
String sAdRef = null;  
try { // This code block invokes the MarbilSoap:getAdvert operation on  
web service  
    afrigis.Marbil.Marbil Marbil = new afrigis.Marbil.Marbil_Impl();  
    afrigis.Marbil.MarbilSoap MarbilSoap = Marbil.getMarbilSoap();  
    afrigis.Marbil.Advert[] adverts = MarbilSoap.getAdvert("BroadcasterName",  
"BroadcasterPassword", "", "", "", "", "", "", "afrigisLocal", "27820001111",  
"testapp").getAdvert();  
    afrigis.Marbil.Advert advert = adverts[0];  
  
    if (advert.getStatus().toUpperCase().equals("OK")) {  
        sAdRef = advert.getRefNumber();  
        sAd = advert.getAdvert();  
        //deliver advert to the user  
        //...  
    }  
    else {  
        String sError = advert.getStatusdescription();  
        logger.severe("Error retrieving ad: " + sError);  
    }  
}
```

```
    } catch(javax.xml.rpc.ServiceException ex) {  
        logger.log(Level.SEVERE, "Error", ex);  
  
    } catch(java.rmi.RemoteException ex) {  
        logger.log(Level.SEVERE, "Error", ex);  
    } catch(Exception ex) {  
        logger.log(Level.SEVERE, "Error", ex);  
    }  
}
```

## 2.5 Using Marbil from PHP

From PHP the Marbil web service can be called by using a number of open source web service libraries such as NuSOAP or PEAR. For the purpose of this documentation we will make use of the Gnome XML Library, which is included with PHP 5.

The following PHP code snippet shows a GetAdvertNow method call that prints the advert's text component:

```
<?php  
try {  
    $wsdl = "http://www.marbiladserver.co.za/Marbil/Services/Marbil080.asmx?WSDL";  
    $soap = new SoapClient($wsdl, array('features' =>  
        SOAP_SINGLE_ELEMENT_ARRAYS));  
  
    $params->username = "BroadcasterUserName";  
    $params->password = "BroadcasterPassword";  
    $params->amount = "";  
    $params->agegroup = "";  
    $params->gender = "";  
    $params->race = "";  
    $params->location = "";  
    $params->category = "";  
    $params->image = "afrigisLocal";  
    $params->clientRef = "";
```

```
$params->clientContext = "";
$params->clientUserAgent="";
$params->clientIpAddress="";

$result = $soap->GetAdvertNow($params);
$adverts = $result->GetAdvertNowResult;

$status = $adverts->Advert[0]->status;
$advertImage = $adverts->Advert[0]->imageData;
$advertText = $adverts->Advert[0]->advertText;
} catch (Exception $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
}
echo $advertText;
?>
```

## 2.6 Integrating with Vodacom4me

To integrate with Vodacom4me you first need to call the `getAdvert()` method of the Marbil web service (as described in section 3) to get an advert. Then the advert needs to be rendered as described in the following sub-sections. Then you call the `CompleteDelivery()` method of the Marbil web service (as described in section 4).

Vodacom4me passes a parameter called “TerminalType” as part of the query string. This parameter should be used to determine whether to render XHTML or WML. Also note that image adverts are only supported for XHTML browsers, therefore only request image adverts if the “TerminalType” parameter indicates XHTML.

A new advert should be requested every time that the user requests a new screen.

### 2.6.1 Rendering a text advert for XHTML browsers

Text adverts should be rendered as a scrolling marquee directly underneath the banner image of each page. This can be done by constructing the following XHTML:

```
String sAdvert = marbile.getAdvert();
```

```
sXHTML += "<p style=\"display: -wap-marquee; -wap-marquee-loop: infinite;
color:blue; background-color: white; font-size:small; font-weight: bold\">";
sXHTML += sAdvert;
sXHTML += "</p>";
```

## 2.6.2 Rendering a text advert for WML browsers

WML browsers does not support a scrolling marquee. For WML browsers you should simple render the text at the bottom of each screen.

## 2.7 Integrating with Vodafone Live

To integrate with Vodafone Live you first need to call the `getAdvert()` method of the Marbil web service (as described in section 3) to get an advert. Then the advert needs to be rendered in PML as described in the following sub-sections. Then you call the `CompleteDelivery()` method of the Marbil web service (as described in section 4).

A new advert should be requested every time that the user requests a new screen. Vodafone Live accepts PML and automatically takes care of differences between XHTML and WML browsers.

The name of the page should be passed as the *ScreenName* parameter to the *CompleteDelivery* method.

### 2.7.1 Rendering a text advert in PML

Text adverts should be rendered as a scrolling ticker directly underneath the banner image of each page. This can be done by constructing the following PML:

```
sPML += "\n<CONTAINER type=\"data\">";
sPML += "\n<TICKER dir=\"rtl\" loop=\"infinite\" style=\"scroll\"
speed=\"slow\">";
sPML += "<TEXT>" + sAdvert + "</TEXT>";
sPML += "\n</TICKER>";
sPML += "\n</CONTAINER>";
```

### 3. Marbil Integration by HTTP request.

The broadcaster can integrate with Marbil by a simple HTTP request call. The URL **required** is:  
[http://www.marbiladserver.co.za/marbil/services/marbil\\_advert.aspx](http://www.marbiladserver.co.za/marbil/services/marbil_advert.aspx)

The URL will respond by returning the entire HTML content of an image advert. All the broadcaster would be required to do is call the URL and print out the response where the advert is required to display. At this time this integration method only returns the advert in one format (ideal for a banner ad). In future releases some further flexibility will be added to cater for other commonly used advert formats, which will be accessible by sending an additional parameter.

Most basic request format:

[http://www.marbiladserver.co.za/marbil/services/marbil\\_advert.aspx?username=USERNAME&password=PASSWORD](http://www.marbiladserver.co.za/marbil/services/marbil_advert.aspx?username=USERNAME&password=PASSWORD)

This request URL should be appended with the same parameters as used in the web service call. If the parameter is marked as optional, it does not need to be sent.

|          |  |
|----------|--|
| Username | Required. Your username, which is provided to you by the system administrator.   |
| Password | Required. Your password, which is provided to you by the system administrator.   |
| Profile  | Optional. A comma separated list indicating the profile requirements for the user. Not yet defined.  |
| Amount   | Optional. Not yet defined. ( <a href="#">See 4. GetAllAdverts</a> )  |
| AgeGroup | Optional. The age for the user.  |
| Gender   | Optional. The gender of the user. This could be "male" or "female"   |
| Race     | Optional. The race of the user. Not yet defined.   |
| Location | Optional. Indicates the location area of interest to the user. The location can be specified as a single point (sp) or as a bounding box (bb).<br><br>The format for a single point request: sp{latitude, longitude}<br>Example: "sp{ 25.7461, 28.2344}"<br><br>The format for a bounding box request: bb{bottom left coord, top right coord}<br>Example: "bb{ 16.4549,-34.833,32.8913,-22.126}" |
| Day      | Optional. The day on which the advertisement will be delivered to the user, or the day to which the content applies.   |

|                 |   |
|-----------------|---|
|                 | The format should be "yyyy-mm-dd". The current date is assumed if left blank.   |
| Timeslot        | Optional. The time of the day that the advertisement will be delivered to the user. The format is "hh:mm". The default is the current time, if left blank.  |
| Category        | The category of advert that is required. For example entertainment. Automotive, information etc.  |
| Image           | One of the following values:<br>"Yes" – to return the image as a base 64 encoded stream.<br>"AfriGISLocal" – to return just a URL for the image.<br>Blank – To return a text advert.<br>"deviceAwareImage" – Intended for WAP broadcasters – functionally the same as AfriGISLocal, but Marbil can detect and store the users device. Currently does not support animated gifs. |
| ClientRef       | Suggested. A unique reference to the user of your site. Currently used for statistical analysis only. Preferably the MSISDN of logged in user if available.   |
| ClientContext   | Suggested. A reference to the page or site requesting the advert. Currently used for statistical analysis only.   |
| ClientUserAgent | Suggested. Broadcaster should send the client User-Agent header to support device specific profiling.   |
| ClientIPAddress | Optional. The IP address of the user.   |

### 3.1 Java Example

There response of the following simple java method could be printed on the part of the page that the advert is required. More advanced parameters, as described above, could be appended to the base URL as the username and password parameters are.

```
public static String getMarbilAdvertHtml(String username, String password) {  
    String html = "";  
    String marbilUrl = "http://www.marbiladserver.co.za/marbil/services/marbil_advert.aspx";  
    marbilUrl += "?username=" + username + "&password=" + password;  
    try {  
        URL url = new URL(marbilUrl);  
        HttpURLConnection request = (HttpURLConnection) url.openConnection();  
        request.setConnectTimeout(1000);  
        request.setReadTimeout(1000);  
    }  
}
```

---

```
BufferedReader reader = new BufferedReader(new InputStreamReader(request.getInputStream()));
for (String line = null; (line = reader.readLine()) != null;)
    html += line;
reader.close();
} catch (Exception ex) {
    ex.printStackTrace();
}
return html;
}
```