CIPG API INTEGRATION
Guide

1 Contents

1.1 Introduction 3
1.2 Document Overview 3
1.3 Integration Overview 3
1.3.1 Transaction Registration 3
1.3.2 Transaction Payment Request 4
1.3.3 Payment Flow 4
1.3.4 Authentication 4
1.3.5 Transaction Result 4
1.3.6 Merchant Site Redirecting 4
2 Merchant Integration Parameters 4
2.1 Explanation of the Integration Parameters: 5
3 Integration Procedures 6
3.1.1 Direct Http Post Request 6
3.1.2 XMLAPI Integration 8
3.1.3 JSON API Integration 12
3.1.4 Getting Transaction result/Response from CIPG 14
3.1.5 Summary 15

Major Version	Minor Version	Changes	Date Released
Version 1	Version 1.1	• *First Version Released*	
	Version 1.2	• *Second Version Released* • *Addition of two new Integration procedures.*	20/11/2013
	Version 1.3	• *Third Version Released* • *Replacement of old URLs with current CIPG URLs*	08/02/2013
	Version 1.4	• *Addition of Date format for registration data to CIPG*	11/03/2014
	Version 1.5	• *Addition of parameters to the return URL for approved and declined transaction URL(Section 3.1.4)*	24/03/2014
	Version 1.6	• *Addition of transaction status verification URL (Section 3.1.5)*	06/02/2014
	Version 1.7	• *Corrected the variable for CustomerLastName to CustomerLastname*	01/09/2014

1.1 Introduction

CIPG (Central Internet Payment Gateway) is an Internet payment gateway that provides online web payment solutions to UBA web merchant who are interested in receiving Debit card payments from their prospective customers' debit card to their UBA Accounts.

1.2 Document Overview

This Document provides an explanation of how UBA merchants can integrate to the CIPG application from their website, in that when their customers checkout from the website to pay, they are redirected to the CIPG Payment gateway web application to make Payment using different Debit Cards available on the payment Gateway. The integration can be implemented using any of these three options.

I. Direct HttpRequest POST API
II. XML POST API
III. JSON RPC.

This document is provided with a Test Merchant Website which contains scripts that can be used as guides during integration

1.3 Integration Overview

1.3.1 Transaction Registration

The gateway first receives a set of required parameters concerning the transaction from the merchant's website via an **Http POST Request** (these are later listed in this integration document - **Transaction Details, Merchant API Service Key and the Merchant ID*)***.

These details are used for merchant verification and also for registering the transaction on CIPG after which the Transaction ID of the registered Transaction is return back to the RPC call (depending on the integration method used by the merchant). This is the call for Transaction registration and Merchant verification.

1.3.2 Transaction Payment Request

If the merchant is successfully verified and the transaction was successfully registered, the merchant receives the Transaction ID for the transaction and then calls the second URL to pay for the transaction passing the Transaction ID as a GET parameter to the URL in another ***HttpRequest*** like: *https://ucollect.ubagroup.com/pay?id=TXNID001*.

1.3.3 Payment Flow

CIPG then displays the transaction details for the customer to verify, Displays the various card schemes available on CIPG for the customer to choose from, and requests for the Card Holder's Card Details (which includes the Card Number, CVV security code, Card Holder Name and the Card's expiry Date). After which the customer is then asked to proceed processing of the Transaction.

1.3.4 Authentication

After these, if the Card Processor then requires a second level authentication, the authenticating frame is displayed for the PIN of the customer – This is however dependent on the configuration for the Merchant or Card Scheme. Authentication is dependent on the Processor configuration for the Card.

1.3.5 Transaction Result

The result of the transaction is then displayed after Processing.

1.3.6 Merchant Site Redirecting

If the transaction was successful, CIPG redirects the customer back to the merchant's webpage indicating success (approve URL supplied during registration or return URL supplied during registration),

If the Transaction fails, it displays the result with the details indicating the reason for the failure and redirects to the merchant's website indicating failure (decline URL supplied during registration)

2 Merchant Integration Parameters

2.1 Explanation of the Integration Parameters:

- **merchantId:** This is a unique code that identifies a merchant which should have been sent to the merchant's email address after the Merchant has been registered.
- **description:** This is the description of the items that are purchased separated by comma.
- **total:** This is the total transaction amount.
- **date:** This is the date of the transaction, Please note that the date must be a timestamp in this format – dd/mm/yyyy HH:mm:ss
- **countryCurrencyCode:** This the currency in which the transaction will be processed for example 566 for Naira.
- **noOfItems:** This is the total number of items/services being purchased from the merchant website.
- **customerFirstName:** This is the first name of the customer.
- **customerLastname:** This is the last name of the customer.
- **customerEmail:** This is the email address of the customer.
- **customerPhoneNumber:** This is the contact / phone number of the customer.
- **referenceNumber:** This is the merchants transaction reference number - should be unique for each transaction.
- **serviceKey:** This is a 32 hexadecimal character set that is used for authenticating a merchant on CIPG ¨C sent to the merchant after initial registration.
- **CIPG_URL_REGISTER_POST_PARAM:** This is the URL of the CIPG payment gateway where the transaction data is being passed to be registered by HTTP Post Request.

3 Integration Procedures

Merchants can choose between four types of CIPG implementation options:

- Direct Http POST
- XML API
- JSON RPC
- XML RPC

3.1.1 Direct Http Post Request

3.1.1.1 URL to call

For registration of transaction Register URL for HTTP POST

https://ucollect.ubagroup.com/cipg-payportal/regptran

For payment of already registered Transaction URL for HTTP POST

https://ucollect.ubagroup.com/cipg-payportal/paytran?id=CTXN0001

3.1.1.2 Overview and Integration Explanation

Integration by passing the required fields individually as different post parameters using each field of the parameters as a new POSTFIELD name.

These include each of the program variables declared earlier in this document namely the: - *merchantId, description, date, countryCurrencyCode, noOfItems, customerFirstName, customerLastname, customerEmail, customerPhoneNumber, referenceNumber, serviceKey.*

These parameters are sent via an httpRequest call to the CIPG register URL to register this transaction.

If the status code of the HTTP response received is 200 then the transaction details were successfully registered and the transaction ID will be returned.

The transaction ID returned can then be used to call the payment URL.

Where "CTXN0001" is the transaction ID returned from the first RPC call.

Otherwise if the status code returned is not 200, then the registration remote procedure call was not successful, the error encountered during transaction could also be return instead of the transaction ID

3.1.1.3 HTTP Direct POST Example

3.1.1.3.1 Set up the Transaction POST Parameters

The parameter strings to send can be declared in an array like so using php programming

```
//Declare the Variables to POST to CIPG for registration of
this Transaction in an array
$post = array(
"merchantId" => CIPG_MERCHANTID,
"description" =>
$_REQUEST["description"],
"total" => $_REQUEST["amount"] *
$_REQUEST["noOfItems"], "date" =>
$_REQUEST["date"],
"countryCurrencyCode" =>
$_REQUEST["countryCurrencyCode"], "noOfItems" =>
$_REQUEST["noOfItems"],
"customerFirstName" => $_REQUEST["customerFirstName"],

customerLastname" =>
$_REQUEST["customerLastname"], "customerEmail" =>
$_REQUEST["customerEmail"], "customerPhoneNumber"
=> $_REQUEST["customerPhoneNumber"],
"referenceNumber" => $_REQUEST["referenceNumber"],
"serviceKey" => CIPG_SERVICEKEY,);
```

3.1.1.3.2 : Sending transaction POST data using PHP curl

```
//Initialise connection using PHP CURL
$ch = curl_init();

//The variable -- CIPG_URL_REGISTER_POST is being declared in
the settings.php file included
$REGISTER_CIPG_TXN_URL = CIPG_URL_REGISTER_POST_PARAM;

//set option of URL to post to
curl_setopt($ch, CURLOPT_URL, $REGISTER_CIPG_TXN_URL);
//set option of request method -----HTTP POST Request
curl_setopt($ch, CURLOPT_POST, true);
//The HTTP authentication methods to use
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_ANY);
//set to true if cipg url is via https
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);

//This line sets the parameters to post to the URL
curl_setopt($ch, CURLOPT_POSTFIELDS,   $post);
//This line makes sure that the response is gotten back to the
$response object(see below) and not echoed
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

//This line excecutes the RPC call

$response = curl_exec($ch); //and assigns the result to $response
object
 $returnCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);


//Close the stream
curl_close($ch);
```

```
//Check if there are no errors ie httpresponse == 200 -OK
if ($returnCode == 200) {

    //If there are no errors, the transaction ID is returned
    $transactionid = $response;
    //This line declares the Link to pay for this transaction
    $paylink = CIPG_URL_PAY ."?id=" .
    $transactionid; header( "Location:
    $paylink");

} else {
    //Get return Error Code, If there was an error during call
    //
    switch($returnCode){
        //200 is OK so, this should be insignificant if all is well
        case 200:
        break; default:
        //Declare the Request Error
            $result = 'HTTP ERROR -> '. $returnCode;
            break
    }
    echo
```

3.1.2 XML API Integration

This section explains how to integrate with an **httpRequest** using a XML parameter with **'transactionData'** as the name of the **Http Post Parameter**. The URL for transaction registration is *https://ucollect.ubagroup.com/cipg-payportal/regxtran*

XML format should have ***register*** as the root node after which the remaining variables can then be added as different nodes

3.1.2.1 Sending transaction XML data using PHP curl. URLs to call

For registration of transaction Register URL for HTTP POST =

https://ucollect.ubagroup.com/cipg-payportal/regxtran

Transaction parameters are sent to this URL

For payment of already registered Transaction URL for HTTP POST=

https://ucollect.ubagroup.com/cipg-payportal/paytran?id=CTXN0001

3.1.2.2 Overview and Integration Explanation

Integration by passing the required fields in an XML format with the Http parameter name **'transactionData'** used when calling the API

These include each of the program variables declared earlier in this document

namely the: - merchantId,description,date, countryCurrencyCode, noOfItems, customerFirstName, customerLastname, customerEmail, customerPhoneNumber, referenceNumber, serviceKey.

These parameters are sent via an httpRequest call to the CIPG register URL to register this transaction.

An XML response is then returned signifying the status of the transaction registration. The format of the xml is shown below.

XML Response for success:-

```
<registration><transaction id=CTXN0001/></registration>
```

XML Response for error:-

```
<registration><error message=Merchant Not
Found/></transaction></registration>
```

3.1.2.3XML API Example

3.1.2.3.1 Set up the Transaction POST Parameters in XML format

The parameter strings to send can be declared in an array using php programming

```
//Import /Include the file 'settings.php' where the URL for
CIPG.Mechant ID and Service Key is declared
require_once('settings.php');
//decalare the xml string variable
$request_xml = "<?xml version='1.0' encoding='utf-8'?>";
$request_xml.="<register>";
$request_xml.="<merchantId
>".CIPG_MERCHANTID."</merchantId>";//Variable CIPG_MERCHANTID is from
settings.php
$request_xml.="<description>".$_REQUEST['description']."</descripti
on>";
$request_xml.="<total>".$_REQUEST['amount']."</total>";
$request_xml.="<date>".$_REQUEST['date']."</date>";
$request_xml.="<countryCurrencyCode>".$_REQUEST['countryCurrencyC
ode']."</c ountryCurrencyCode>";
$request_xml.="<noOfItems>".$_REQUEST['noOfItems']."</noOfItems>";
$request_xml.="<customerFirstName>".$_REQUEST['customerFirstName'
]."</custo merFirstName>";
$request_xml.="<customerLastname>".$_REQUEST['customerLastname']."</cus
tome rLastname>";
$request_xml.="<customerEmail>".$_REQUEST['customerEmail']."</custo
merEmail
>";
$request_xml.="<customerPhoneNumber>".$_REQUEST['customerPhoneNum
ber']."</c ustomerPhoneNumber>";
$request_xml.="<referenceNumber>".$_REQUEST['referenceNumber']."<
/reference Number>";
$request_xml.="<serviceKey>".CIPG_SERVICEKEY."</serviceKey>";//Variable
CIPG_SERVICEKEY is from settings.php
$request_xml.="</register>";
```

3.1.2.3.2 : Sending transaction POST data using PHP curl

```
<?php
//The webservice URL for XML HTTP POST
$REGISTER_CIPG_TXN_URL = CIPG_URL_REGISTER_XML;//Variable
CIPG_URL_REGISTER_XML is from settings.php


//Initialize handle and set options

//Initialise connection using PHP CURL
$ch = curl_init($REGISTER_CIPG_TXN_URL);
//set the URL for XML POST to CIPG
curl_setopt($ch, CURLOPT_URL, $REGISTER_CIPG_TXN_URL);
//The HTTP authentication methods to use
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_ANY);
//set option of request method -----HTTP POST Request
```

```
curl_setopt($ch, CURLOPT_POST, true);
//set to true if cipg url is via https
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
//TRUE(1) to return the transfer(result form the webservice call) as a string of
the return value of curl_exec() instead of outputting it out directly
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
//The maximum number of seconds to allow cURL functions to execute
curl_setopt($ch, CURLOPT_TIMEOUT, 4);
//The full data to post in a HTTP "POST" operation -containing the data to send for
transaction registration
curl_setopt($ch, CURLOPT_POSTFIELDS, array('transactionData' =>
$request_xml));
//An array of HTTP header fields to set
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Connection: close'));

$resultCode;
//Execute the request
$result = curl_exec($ch);

$returnCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);

$returnError = curl_errno($ch);
//Close the curl handle
curl_close($ch);

//Check for errors ( again optional )
//Check if there are no errors
if (!$returnError) {

//Parse in the resonse XML result
$xmlresponse = simplexml_load_string($result);

if($xmlresponse->error){
    $errorMessage = $xmlresponse->error['message'];
    echo "Error Mesage".$errorMessage;
}elseif($xmlresponse->transaction){
    //Get the Transation ID
    $transactionId = $xmlresponse->transaction['id'];
    //This line declares the Link to pay for this transaction
    $paylink = CIPG_URL_PAY ."?id=" . $transactionId; header( "Location:
$paylink");
}
} else {
    //Get return Error Code, If there was an error during call

    //
    switch($returnCode){
        //200 is OK so, this should be insignificant if all is well
        case 200:
    break; default:
    //Declare the Request Error
        $result = 'HTTP ERROR -> ' . $returnCode;
    break;
    }
echo $result;
}
```

3.1.3 JSON API Integration

This section explains how to integrate with an **httpRequest** using a JSON data type in sending the transaction data. The URL for transaction registration is *https://ucollect.ubagroup.com/cipg-payportal.com/regjtran*

The JSON object is sent and a JSON object is returned as the response.

The data type of the header is set as JSON ----see below

3.1.3.1 Sending transaction JSON data using PHP curl.URLs to call

For registration of transaction Register URL for HTTP POST
https://ucollect.ubagroup.com/cipg-payportal /regjtran –
Transaction parameters are sent to this URL.
For payment of already registered Transaction URL for HTTP POST
https://ucollect.ubagroup.com/cipg-payportal/paytran?id=CTXN0001

3.1.3.2 Overview and Integration Explanation

The JSON object should declare each of the parameters as an attribute of the object to be sent.

These includes each of the program variables declared earlier in this document namely the: - *merchantId, description,date, countryCurrencyCode, noOfItems, customerFirstName, customerLastname, customerEmail, customerPhoneNumber, referenceNumber, serviceKey.*

These parameters are sent via an httpRequest call to the CIPG register URL to register this transaction.

A JSON response is then returned signifying the status of the transaction registration. The format of the json is shown below

```
JSON Response for success: - registration.transaction.id
```

JSON Response for error: -

registration.error.message

3.1.3.3 JSON API Example

3.1.3.3.1 Set up the Transaction POST Parameters in JSON format

The parameter strings to send can be declared in an array using php programming

```
//Declare the Variables to POST to CIPG for registration of
this Transaction in an array
$jsonArray = array(
    "merchantId" => CIPG_MERCHANTID,
    "description" =>
    $_REQUEST["description"], "amount" =>
    $_REQUEST["amount"],
    "total" => $_REQUEST["amount"] *
    $_REQUEST["quantity"], "date" =>
    $_REQUEST["date"],
    "countryCurrencyCode" =>
    $_REQUEST["countryCurrencyCode"], "quantity" =>
    $_REQUEST["quantity"],
    "customerFirstName" =>
    $_REQUEST["customerFirstName"], "customerLastname"
    => $_REQUEST["customerLastname"], "customerEmail"
    => $_REQUEST["customerEmail"],
    "customerPhoneNumber" =>
    $_REQUEST["customerPhoneNumber"], "referenceNumber"
    => $_REQUEST["referenceNumber"], "serviceKey" =>
    CIPG_SERVICEKEY',
);

//Check if json_encode function exists, it will exist if php
version is higher than php5.1, else it may not

//This line checks if the function to encode
JSON exists if (!function_exists('json_encode'))
{
    //import the JSON Class that contains the function----This
is if it doesn't exist
    require_once('JSON.php');
    //See file JSON.php
    $json_encode = new Services_JSON();
    //Encode the array as a PHP JSON array object
    $json = $json_encode->encode($jsonArray);
} else {
    //Encode the array as a PHP JSON array object
    $json = json_encode($jsonArray);
}
```

Note that a JSON.php file is being provided with the API which implements functions used in encoding and encoding a JSON object into a PHP array, JSON library was introduces in PHP5.2 and above

3.1.3.3.2 Sending transaction POST JSON data using PHP get_file_contents

```
//set the data to send
$context = stream_context_create(array( 'http' => array(
'method' => 'POST',
'header' => 'Content-Type: text/json', 'content' => $json)));
//echo "JSON packet to CIPG:\n" . $json . "\n";


$result = file_get_contents(CIPG_URL_REGISTER_JSON, false, $context);


//echo "Result from CIPG:\n" . var_export($result, true) . "\n"; if ($result ===
false) {
echo "There was an error, check CIPG logs\n";
} else {
//I made the second change here
//Check if json_decode function exists, will exist in php 5.2 and above
if (function_exists('json_decode')) {
//import the JSON Class that contains the function require_once('JSON.php');
$jsonService = new Services_JSON(SERVICES_JSON_LOOSE_TYPE);
$decoded = $jsonService->decode($result);
} else {
$decoded = json_decode($result, true);
}


//      echo "PHP decoded json:\n"; var_dump($decoded);
if($decoded["registration"]["transaction"]){
$transactionid = $decoded["registration"]["transaction"]["id"];
$paylink = CIPG_URL_PAY ."?id=" . $transactionid; header( "Location: $paylink");
}elseif($decoded["registration"]["error"]){ echo "Error Message====" +
$decoded["registration"]["error"]['message'];
}else
{
echo "Unspecified Error";
}
```

Set up the Transaction POST Parameters in JSON format

3.1.4 Getting Transaction result/Response from CIPG

After a complete payment, the customer clicks the finish and CIPG redirects the customer back to the merchant website with POST parameters signifying the status of the transaction, The URL redirected back to may either be the approve, decline or cancel URL.

The parameters include

1. refNo:- The Merchant reference Number of this transaction used during registration.
2. transactionId:- The ID of the transaction on CIPG.
3. status:- The Status of the Transaction.

The pattern of the url returned is as below;

approveurl?refNo=merchantTransactionId,transactionID=cipgtransactionID,status=APPROVED

declineurl?refNo=merchantTransactionId,transactionID=cipgtransactionID,status=DECLINED

For instance:-
http://www.merchant.com/cipgapprove.php?refNo=MYID101,transactionId=CTUA100011

http://www.merchant.com/cipgdecline.php?refNo=MYID101,transactionId=CTUA100011,

Where "http://www.merchant.com/cipgapprove.php" and
"http://www.merchant.com/cipgdecline.php" was provided by the merchant by the merchant

3.1.5 Transaction Verification

If the merchant wants to verify the status of a transaction at any time, the merchant can easily call the URL below with the transaction id as a parameter.

"https://ucollect.ubagroup.com/cipg-payportal/confirmation/verify?cipgtxnref=CTUA120601&mytxnref=1902864955&cipgid=MCGN00001"

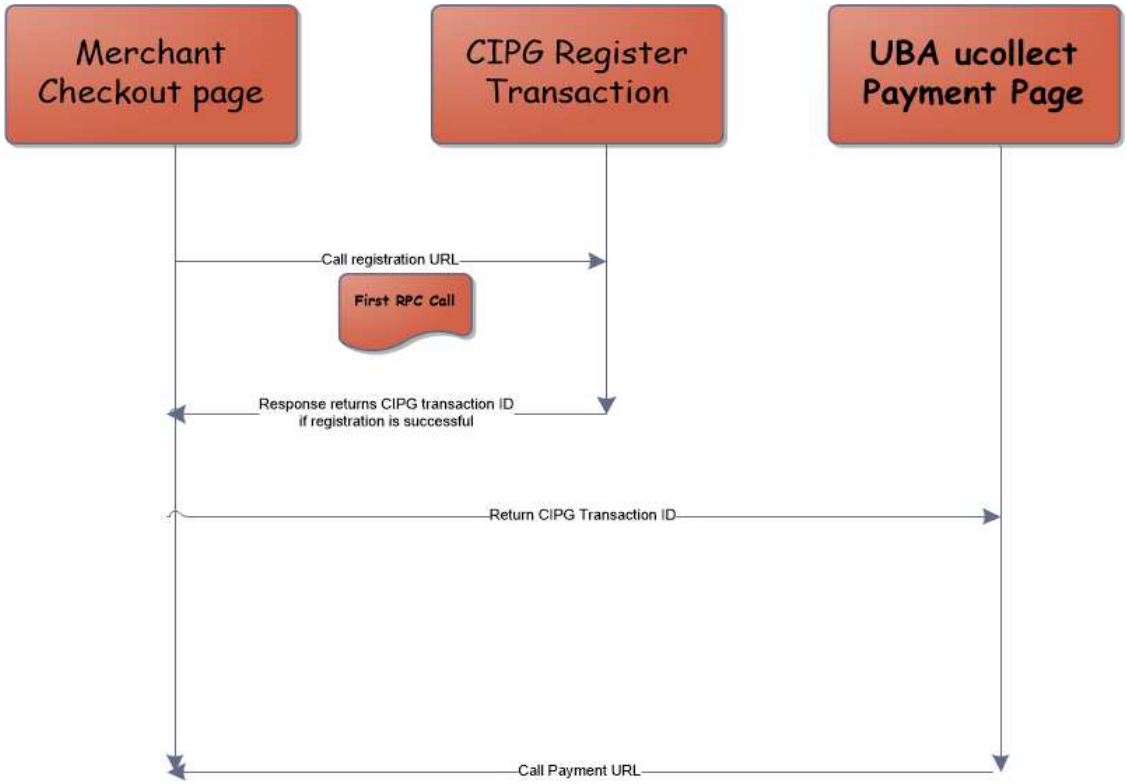
Where the parameters

- ***cipgtxnref*** is the **UBA ucollect** transaction id of the transaction sent back to the return URL.
- ***mytxnref*** is the transaction id of the transaction.
- **cipgid** is the merchant's id sent during registration on CIPG.

This call returns the status of the transaction, If approved, it returns the String "**Approved Transaction**", else it returns if the Transaction was declined or still being processed.

3.1.6 Summary

In summary, Integration with UBA CIPG involves the merchant calling the transaction Registration API, parsing the transaction data using any of the three methods explained, receiving the transaction ID after the transaction has been successfully registered on the CIPG gateway, and finally calling the payment URL with the transaction ID received. After the transaction flow, the merchant website is redirected via the URL that indicates success or failure of the transaction with the details of the transaction result as POST Data.



Sequence Diagram showing Merchant API Integration