Cheka
XML/HTTP Interface Specification

Conor Information Technologies PTY (LTD)

Prepared by Philip Foulkes

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ABBREVIATION	MEANING
GUID	Globally Unique Identifier
HTTP	Hypertext Transfer Protocol
ID	Identifier
MSISDN	Mobile Subscriber Integrated Services Digital Network Number
NGF	Next Generation Framework
SMS	Short Message Service
TTL	Time To Live
URL	Uniform Resource Locator
USSD	Unstructured Supplementary Services Data
XML	Extensible Markup Language

REVISION HISTORY

Version Number	Date	Sections Affected	Description of Revision
1.00	2017-07-03	All	Initial document

REFERENCED DOCUMENTS

DOCUMENT NAME	DOCUMENT VERSION	PUBLISHER	DOCUMENT DATE

1. Legal Notices

2. About Us

Conor was founded in 2007 and has since built up a comprehensive track record of successes as a leading South African Telco services and solutions provider. Our management team has grown in experience, kept up with the latest technology and retains a very hands on approach. Conor has assembled the products, tools, solutions, systems and most important of all, a group of talented, passionate people that enable global businesses to establish and maintain critical competitive advantage. This advantage is rooted in the ability to understand the client experience and to proactively develop services on the basis of this understanding. We take a long-term view of your growth and sustainability in an ever-evolving marketplace, only recommending Telco solutions and services that conform to strict parameters. These we back with fast and unwavering services – because we know that your business health depends on it.

We provide project implementation and consulting services, which focus on new technological trends that enable our clients to implement solutions that are stable, cost effective and of high quality. Our holistic approach includes system auditing/analysis, architecture, infrastructure establishment, application development, high availability, performance tuning and support services. Conor's quality assurance processes and proven development methodology ensures that well documented scalable solutions are delivered on time and in budget. As a company we acknowledge and value our co-workers and their families. Conor is headquartered in Pretoria, South Africa. We have a heritage of 8 years leadership in the mobile market delivering carrier class mobile data services to more than 20 customers globally, especially in Africa where we have presence in 14 countries. The solutions we offer our customers manage more than 2 Billion mobile data transactions and 60 million subscribers per day.

3. Introduction

This document describes the Cheka Extensible Markup Language (XML)/Hypertext Transfer Protocol (HTTP) interface, which allows external systems to interact with Cheka using XML/HTTP. Cheka is a service which allows mobile subscribers to purchase Bundles of voice minutes, Short Message Service (SMS), and data, as well as allowing them to opt-in and out of tariff plans. Cheka supports a number of different transaction types (e.g., Bundle Purchase, Buddy Purchase, and Balance Check), which are traditionally triggered via Unstructured Supplementary Services Data (USSD). The Cheka HTTP interface allows for interaction with the Cheka service via XML/HTTP, which in turn allows for a richer set of parameters to the supplied to the application, which allows for the Cheka application to run more flexibly.

Currently, the Cheka XML/HTTP interface allows for the following Functions:

1. Validation of intended Bundle Purchases;
2. Querying of Cheka transactions;
3. Performing Balance Check on a subscriber;
4. Obtaining the Bundle catalogue;

The Cheka XML/HTTP interface is an implementation of a generic service interface that can be used by multiple Next Generation Framework (NGF) services, and can be extended to allow for additional Functions. The Cheka interface has defined a Cheka Request Message, and a corresponding Cheka Response Message. Each Message contains a Header containing data common to both Message types, and then either a request or response body.

The listing below shows an example of a Cheka Request Message.

```
<message>
	<header>
		<timestamp>0000-00-00 00:00:00</timestamp>
		<int_guid>FFFE0101010000A6F213</int_guid>
		<reply_address></reply_address>
		<ttl>0</ttl>
		<echo_request>false</echo_request>
		<consumer>
			<name>internal</name>
			<tid>0</tid>
			<sid>0</sid>
		</consumer>
		<provider>
			<name>cheka/validate</name>
			<tid>0</tid>
```

```
                        <sid>0</sid>
                </provider>
        </header>
        <request>
                <username>conor</username>
                <password>password</password>
                <request>
                        <chekaValidate>
                                <langCode>sw</langCode>
                                <transactionId>820487828197204173</transactionId>
                                <startTime>2015-11-23 12:13:03</startTime>
                                <amsisdn>255767954443</amsisdn>
                                <transactionType>
                                        <normalPurchase>
                                                <bundleUssdId>500</bundleUssdId>
                                        </normalPurchase>
                                </transactionType>
                        </chekaValidate>
                </request>
        </request>
</message>
```

The listing below shows an example of a Cheka Response Message.

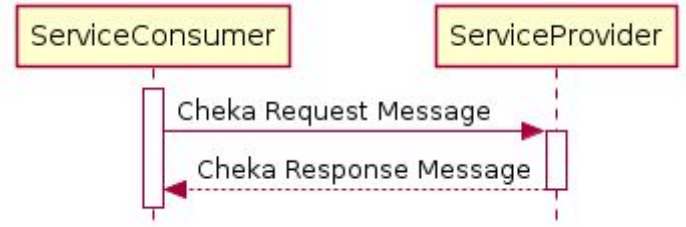
```
<message>
        <header>
                <timestamp>2015-11-23 12:13:03</timestamp>
                <int_guid>05000000000000A6F213</int_guid>
                <reply_address/>
                <ttl>0</ttl>
                <echo_request>true</echo_request>
                <provider>
                        <name>cheka/validate</name>
                        <tid/>
                        <sid/>
                </provider>
                <consumer>
                        <name>internal</name>
                        <tid/>
                        <sid/>
                </consumer>
                <result>
                        <code>9900</code>
                        <description>Success</description>
                        <details/>
                </result>
        </header>
        <response>
                <chekaValidate>
                        <transactionType>
                                <normalPurchase>
                                        <packages>
                                                <package>
                                                        <name>ChekaA1Voice</name>
                                                </package>
```

```
                                        <package>
                                                <name>ChekaA1OSMS</name>
                                        </package>
                                        <package>
                                                <name>ChekaA1OData</name>
                                        </package>
                                </packages>
                        </normalPurchase>
                </transactionType>
        </chekaValidate>
</response>
</message>
```

The general structure of these Messages will be discussed throughout this document.

4. Service Provider and Service Consumer

Within interactions between Cheka and external systems, the actors are known as Service Provider, and Service Consumer. The Cheka application is the Service Provider (providing the Cheka service), and external systems that use the Cheka service are known as Service Consumers. A Service Consumer initiates a request to Cheka to consume the Service by sending a Cheka Request Message to Cheka. For each request, Cheka responds to the Service Consumer with a Cheka Response Message. This interaction is shown in the figure below.



5. HTTP

Requests sent from a Service Consumer to Cheka are POST'ed to the HTTP server using the Uniform Resource Locator (URL) `http://<hostname>/cheka/request`.

6. Message

Both Cheka Request and Cheka Response Messages are represented with a `<message>` XML root element, as can be seen in the example listings above. The `<message>` XML element has a mandatory `<header>` child XML element, as well as either a `<request>` or a `<response>` child XML element. The `<request>` XML element is present in Cheka Request Messages, and the `<response>` XML element is present in Cheka Response Messages. The structure and contents of these elements are discussed in subsequent sections within this document.

7. Header

Both Cheka Request and Cheka Response Messages contain a Header which is represented with the `<header>` XML element. This XML element is a child of the `<message>` XML element. The Header contains data that is common to both Cheka Request and Cheka Response Messages. The Header structure is shown in the listing below.

```
<message>
	<header>
		<timestamp>2015-11-23 12:13:03</timestamp>
		<int_guid>05000000000000A6F213</int_guid>
		<reply_address/>
		<ttl>0</ttl>
		<echo_request>true</echo_request>
		<provider>
			<name>cheka / validate</name>
			<tid/>
			<sid/>
		</provider>
		<consumer>
			<name>internal</name>
			<tid/>
			<sid/>
		</consumer>
		<result>
			<code>9900</code>
			<description>Success</description>
			<details/>
		</result>
	</header>
</message>
```

7.1. Time Stamp

The `<timestamp>` field contains a timestamp indicating the time that the Message was sent (by either the Service Consumer or the Service Provider). The format of the time stamp is

`CCYY-MM-DD HH:MM:SS`. Currently, the timestamp in Cheka Request Messages is not used by Cheka, but should be included in requests for debugging purposes. Cheka Responses contain the time that the Cheka Response was sent.

7.2. GUID

Each Message Header contains a GUID. The GUID uniquely identifies transactions that are processed by a set of co-operating systems and processes. Its main purpose is to allow for transactions to be traced through the various modules they flow through. Any log entries that are generated within the context of a transaction contain the GUID, which allows for ease of troubleshooting. The GUID also allows for Cheka Requests and Cheka Responses to be easily identified from network traces. Within these Messages, the GUID is represented by the `<int_guid>` XML element within a Message Header.

The module where a transaction originates from is responsible for generating the GUID, and then passing it along to other systems or modules involved in handling the transaction. Thus, a Cheka Request Message may contain a generated GUID. The GUID is a 20 character hexadecimal number which has the format shown in the listing below.

```
[Originating System ID][Originating System Instance No][Originating Process ID][Unique ID]
```

The various fields of the GUID are explained in the table below.

Eventually, the Unique ID portion of the GUID will wrap. Given the purpose of the GUIDs (to aid with identifying and troubleshooting transactions within a short term time frame), this is acceptable. Even if a process instance is generating 20,000,000 GUIDs per day, the Unique ID portion would only wrap after about 214 days.

Field	No. Hex Digits	Description
Originating System ID	4	The first four hexadecimal digits of the GUID contain the ID allocated to the particular system that generated the GUID. For example, NGF systems have the System ID of 0x0001. Contact Conor Solutions for a System ID if you do not have one.
Originating System Instance No	2	The fifth and sixth hexadecimal digits contains the System Instance number. There may be a number of instances of a system running at a particular site, so the System Instance number is used to identify each instance of that system. These numbers are allocated at the discretion of the system generating the GUIDs.

Originating Process ID	6	These six hexadecimal digits represents the Linux Process ID of the process that generated the GUID.
Unique ID	8	The last eight hexadecimal digits are a unique 32-bit unsigned integer generated by the process that generated the GUID. These can be generated sequentially per process instance as and when needed.

The `<int_guid>` XML element may be left blank, which results in the Cheka application generating its own GUID.

7.3. Reply Address

The `<reply_address>` XML element contains the address to which the Service Provider should respond. The reply address supplied within a Cheka Request Message is not used by Cheka, but is sent back to the requester verbatim within a Cheka Response. The reply address can be up to 255 characters in length, or may be left empty if the address is not needed.

7.4. Time To Live

The `<ttl>` XML element allows Service Consumers to specify Time To Live (TTL) values for the requests they send to Service Providers. If the processing of the request begins at a time that exceeds its TTL, the request is discarded. The TTL is an integer value expressed in milliseconds. Currently, the Cheka service does not make use of the TTL value, thus its value should be set to zero in Cheka Request Messages. Cheka Response Messages will also have a TTL value of zero.

7.5. Echo Request

The `<echo_request>` XML element is a flag that indicates to a Service Provider as to whether or not it should include the original request body (contained within a Cheka Request Message) in the corresponding Cheka Response Message. This element may either have the value `true` or `false`. A value of `true` indicates to a Service Provider that it should include the original request in the corresponding Cheka Response Message. A value of `false` indicates to a Service Provider that the original request should not be included in the corresponding Cheka Response Message.

7.6. Provider

The `<provider>` element provides information used to address a particular Service Provider Function, and contains a `<name>`, `<tid>`, and `<sid>` child elements. The `<provider>` element is set by the Service Consumer, and sent to a Service Provider within a Cheka

Request Message. The Service Provider sends back the <provider> element within the corresponding Cheka Response Message. The <name> child element is used to address a particular Function, and can be up to 63 characters in length. The <tid> represents a transaction ID, and the <sid> represents a session ID .

For the Cheka service, the <provider> <name> is set to the name of the Function that should be triggered. The valid <provider> <name> values are:

1. cheka/validate: Setting the <provider> <name> to cheka/validate causes the Cheka Validation Function to be triggered;
2. cheka/queryTransaction: Setting the <provider> <name> to cheka/queryTransaction causes the Cheka Query Transaction Function to be triggered;
3. cheka/balanceCheck: Setting the <provider> <name> to cheka/balanceCheck causes the Cheka Balance Check Function to be triggered;
4. cheka/bundleCatalogue: Setting the <provider> <name> to cheka/bundleCatalogue causes the Cheka Bundle Catalogue Function to be triggered;

The actual parameters expected by the various Functions are specified within the request and response bodies of the Cheka Request and Cheka Response Messages. If there is a mismatch between the specified Function name and the request body within a received Cheka Request Message, an error result will be returned to the Service Consumer.

The <tid> and <sid> XML elements are not used by the Cheka service, and both must be left blank. These elements may be used within future versions of this interface. An example <provider> element is shown in the listing below.

```
<provider>
       <name>cheka/validate</name>
       <tid></tid>
       <sid></sid>
</provider>
```

7.7. Consumer

The <consumer> XML element is used to address a Service Consumer, and contains a <name>, <tid>, and <sid> child elements. The <consumer> element is set by the Service Consumer, and sent to a Service Provider within a Cheka Request Message. The Service Provider sends back the <consumer> element within the corresponding Cheka Response Message. The <name> child element is used to address a particular service consumer, and

may be up to 63 characters in length. The <tid> represents a transaction ID , and the <sid> represents a session ID. These ID s may be up to 36 characters in length.

For the Cheka service, the <consumer> element is not used or interpreted in any way. The <name>, <tid> and <sid> elements may be set to any values that the Service Consumer may require within the Cheka Response Message. If these values are not required, set the <name> to internal, and leave the <tid> and <sid> blank. The default <consumer> XML element is shown in the listing below.

```
<consumer>
	<name>internal</name>
	<tid></tid>
	<sid></sid>
</consumer>
```

7.8. Result

The <result> XML element is only present within the Header of a Cheka Response Message. The <result> element informs the Service Consumer of the result of the request it sent to the Service Provider. It contains the following child XML elements: <code>, <description>, and <details>.

The <code> element is an integer type that specifies whether the request was successful or not. If the request was not successful, the <code> specifies the reason for this. If the request was successful, the result code is 9900 (evenly divisible by 100). Other result codes are in the range 9901 to 9999.

The <description> element provides a short textual description of the result code, and the <details> element provides further details on the result code. The result code description and details are used to aid with troubleshooting, and are not necessarily values that are displayed to an end user. For the Cheka service, result codes are accompanied with <description>s, but the <detail> elements are left blank.

An example of a <result> element of a successful request is shown the listing below.

```
<result>
	<code>9900</code>
	<description>Success</description>
	<details/>
</result>
```

An example of a <result> element of a failed request is shown in the listing below.

```
<result>
	<code>9907</code>
	<description>Failed to parse the XML request</description>
```

```
        <details/>
</result>
```

8. Request

In addition to the <header> XML element, each Cheka Request Message contains a <request> XML element. This element contains elements common to all Cheka functions, as well as elements that are specific to the Function being triggered.

The common elements present in all requests to Cheka are shown in the listing below. The <username> and <password> fields should be populated with the username and password supplied by the service provider. Only users who are authenticated and authorised are able to trigger the Functions provided.

```
<request>
        <username>conor</username>
        <password>password</password>
        <request>
        . . .
        </request>
</request>
```

The content of the <request> sub element is dependent on the particular Function that is being triggered. The request contents for the various Functions are explained within the sections detailing the available Functions.

9. Response

In response to a Cheka Request Message, a Service Provider sends a Cheka Response Message back to the Service Consumer. This Message is sent back to the Service Consumer regardless of the result of the request. The Cheka Response Message contains a Header, as well as an application specific response body. The response body will only be populated within the Cheka Response Message if the request was successful (as indicated by the result code found in the Header). The response body within a Cheka Response Message is represented with the <response> XML element. The content of the <response> is dependent on the particular Function that was triggered. The response contents for the various Functions are explained within the sections detailing the available Functions.

10. Cheka Validation

The Cheka Validation Function allows for Cheka transactions to be validated against the Cheka business rules without actually performing the Cheka transaction itself. An external system could use this to determine whether a particular subscriber would be allowed to purchase a particular Bundle at a particular time.

10.1. Cheka Validation Request

The structure of the `<request>` element for the Cheka Validation Function is shown in the listing below. The various child XML elements that are contained within each `<request>` are detailed within this section.

```
<request>
	<username>conor</username>
	<password>password</password>
	<request>
		<chekaValidate>
			<langCode>sw</langCode>
			<transactionId>820487828197204174</transactionId>
			<startTime>2016-04-13 13:25:00</startTime>
			<amsisdn>255767954442</amsisdn>
			<transactionType>
				<normalPurchase>
					<bundleUssdId>500</bundleUssdId>
				</normalPurchase>
			</transactionType>
		</chekaValidate>
	</request>
</request>
```

10.1.1. Language Code

Each `<request>` contains a `<langCode>` child XML element. This element's value must be the two character language code (e.g., *sw*, *en*) of the subscriber that the request is related to. Currently, for Cheka validation requests, this value is not used, but should still be set (it can be set to any language code if the subscriber's language code is not readily available).

10.1.2. Transaction ID

Each `<request>` contains a `<transactionId>` child XML element. This element's value is the transaction ID that Cheka should use should it require to query Unified to perform a request. This allows for all Unified transactions (originating from different systems) relating to a single Cheka transaction to use the same transaction ID .

10.1.3. Start Time

Each `<request>` contains a `<startTime>` child XML element. This element's value represents the time that the Cheka transaction was initiated (this should be the time that the Service Consumer received the request to perform a Cheka transaction). The format of this element is CCYY-MM-DD HH:MM:SS. The supplied time value should be reasonably close to the current time. If the difference between the supplied time and the current time is beyond the configured limit, the request will fail.

10.1.4. A Party MSISDN

The `<amsisdn>` child XML element represents the Mobile Subscriber Integrated Services Digital Network Number (MSISDN) of the party for whom the Cheka request should be performed. For example, for a Cheka validation request, it is the MSISDN of the subscriber for whom the Bundle will be purchased. The MSISDN can be supplied in both national and international format. Any MSISDN supplied in national format will be converted to international format.

10.1.5. Transaction Type

The `<transactionType>` element represents the type of Cheka transaction that the request relates to. Currently, this element has a single XML child element, namely `<normalPurchase>`. The `<normalPurchase>` XML element contains further details relating to a normal Bundle purchase Cheka transaction.

The `<normalPurchase>` element contains one child XML element, namely `<bundleUssdId>`. This value is an integer value, and represents the Cheka Bundle's USSD ID (e.g., 500 for the Cheka Tsh500 Bundle). This is the Bundle USSD ID that the request will be performed against.

10.2. Cheka Validation Response

For the Cheka Validation Function, the `<response>` element has a child element called `<chekaValidate>`, which has a child element called `<transactionType>`, as shown in the listing below.

```
<response>
	<chekaValidate>
		<transactionType>
			<normalPurchase>
				<packages>
					<package>
						<name>ChekaA1Voice</name>
					</package>
					<package>
```

```
                                        <name>ChekaA1OSMS</name>
                                    </package>
                                    <package>
                                        <name>ChekaA1OData</name>
                                    </package>
                                </packages>
                            </normalPurchase>
                        </transactionType>
                </chekaValidate>
</response>
```

10.2.1. Transaction Type

The response body's `<transactionType>` element contains response details relating to the requested Cheka transaction. Currently, the only child element to this element is `<normalPurchase>`. The `<normalPurchase>` element contains response details relating to a request for a Cheka normal Bundle purchase. The child elements to `<normalPurchase>` (`<packages><package><name>`) represent the names of the Packages that make up the Bundle that was requested within the Cheka Request Message, as referred to by the Bundle USSD ID.

11. Cheka Query Transaction

The Cheka Query Transaction Function allows for a Cheka transaction to be queried. This may be useful in scenarios where a Cheka transaction has been triggered by an external system, but for some reason a response was not received from Cheka. The external system could then query the transaction to see whether it exists, and if it exists, what its status is. The Cheka application only keeps track a predefined number of days worth of transactions, so only recent transactions may be queried. Also, the transaction querying will only query a configured number of days worth of transactions, which may be less than what is stored. Enquire with the service provider as to what these values are.

11.1. Cheka Query Transaction Request

The structure of the `<request>` element for the Cheka Query Transaction Function is shown in the listing below. The various child XML elements that are contained within the `<request>` are detailed within this section. The Query Transaction Function is represented with the `<chekaQueryTrans>` element.

```
<request>
        <username>conor</username>
        <password>password</password>
        <request>
                <chekaQueryTrans>
```

```
                    <transactionId>820487828208982101</transactionId>
                </chekaQueryTrans>
        </request>
</request>
```

11.1.1. Transaction ID

The `<chekaQueryTrans>` element contains a `<transactionId>` element, which represents the transaction ID that should be queried. The format of the transaction ID is a string of digits, as shown in the example listing above.

11.2. Cheka Query Transaction Response

For the Cheka Query Transaction Function, the `<response>` element has a child element called `<chekaQueryTrans>` which contains the Cheka Query Transaction response parameters, as shown in the listing below.

```
<response>
        <chekaQueryTrans>
                <transactionId>820487828208982101</transactionId>
                <transactionFound>yes</transactionFound>
                <transactionState>0</transactionState>
        </chekaQueryTrans>
</response>
```

The child elements to `<chekaQueryTrans>` are described within this section.

11.2.1. Transaction ID

The `<transactionId>` element contains the transaction ID that was queried, as sent in the original request.

11.2.2. Transaction Found

The `<transactionFound>` element specifies whether the transaction was found or not. It will either contain the value `yes` (meaning that the transaction was found), or the value `no` (meaning that the transaction was not found).

11.2.3. Transaction State

The `<transactionState>` element specifies what state the transaction is in. This field contains an integer type. The valid state values are specified in the table below. All other values are reserved for future use.

Transaction State	Description
-1	The Cheka transaction state in not known.

0	The Cheka transaction completed successfully.
1	The Cheka transaction is currently in progress, and its final state is not yet known. The transaction's state will change once the final state is known.
2	The Cheka transaction failed in a known manner. i.e., the Cheka application knows that the transaction did not succeed. This is the final state of the transaction.
3	The Cheka transaction failed in an unknown manner. ie., the Cheka application is not sure as to whether the transaction succeeded or not, and has given up trying to determine whether it did or did not succeed. This is the final state of the transaction.

12. Cheka Balance Check

The Cheka Balance Check Function allows for a Cheka Balance Check to be performed for a particular subscriber. This allows for the subscriber's active Bundles to be viewed, as well as various balances associated with these Bundles.

12.1. Cheka Balance Check Request

The structure of the `<request>` element for the Cheka Balance Check Function is shown in the listing below. The various child XML elements that are contained within the `<request>` are detailed within this section. The Balance Check Function is represented with the `<chekaBalanceCheck>` element.

```
<request>
	<username>conor</username>
	<password>password</password>
	<request>
		<chekaBalanceCheck>
			<msisdn>255767954442</msisdn>
		</chekaBalanceCheck>
	</request>
</request>
```

The child elements of the `<chekaBalanceCheck>` element are discussed within this section.

12.1.1. MSISDN

The `<msisdn>` should contain the MSISDN of the subscriber for which the Balance Check should be performed.

12.2. Cheka Balance Check Response

For the Cheka Balance Check Function, the <response> element has a child element called <chekaBalanceCheck> which contains the Cheka Balance Check response parameters, as shown in the listing below.

```
<response>
	<chekaBalanceCheck>
		<bundles>
			<bundle>
				<name>Cheka Tsh500</name>
				<quota>0</quota>
				<subscriptionCount>8</subscriptionCount>
				<subscriptionsRemaining>-1</subscriptionsRemaining>
				<subscriptionCountResetDate>1970-01-01
00:00:00</subscriptionCountResetDate>
			</bundle>
			<bundle>
				<name>Internet Tsh200</name>
				<quota>5</quota>
				<subscriptionCount>4</subscriptionCount>
				<subscriptionsRemaining>1</subscriptionsRemaining>
				<subscriptionCountResetDate>2016-04-14
02:00:00</subscriptionCountResetDate>
			</bundle>
		</bundles>
	</chekaBalanceCheck>
</response>
```

12.2.1. Bundles

The <chekaBalanceCheck> element contains a <bundles> element which represents the Bundles that the requested subscriber is subscribed to. The <bundles> element contains zero or more <bundle> elements, where each <bundle> element represents a Bundle. The <bundle> element contains the following child element:

1. **<name>**: The <name> element is a string, and represents the name of the Bundle;
2. **<quota>**: The <quota> element is an integer, and represents the quota associated with the Bundle. The quota defines the number of times a subscriber may subscribe to the Bundle within a given Bundle subscription count cycle. A value of zero indicates that quotas are not applicable to the particular Bundle. A value greater than zero represents the Bundle's quota. Negative values are reserved for future use;
3. **<subscriptionCount>**: The <subscriptionCount> element is an integer, and represents the number of times the subscriber has subscribed to the Bundle since it was last reset;
4. **<subscriptionsRemaininge>**: The <subscriptionsRemaininge> element is an integer, and represents the number of times that the subscriber may still subscribe to the

Bundle. A value of -1 indicates that quotas are not applicable to the specific Bundle, and thus the subscriber may subscribe to the Bundle any number of times;

5. **<subscriptionCountResetDate>**: The <subscriptionCountResetDate> element represents the date and time that the Bundle subscription count for the Bundle will be reset to zero. The format of the field is CCYY-MM-DD HH:MM:SS. For Bundles that are not subjected to Bundle quotas, this field should be ignored;

13. Cheka Bundle Catalogue

The Cheka Bundle Catalogue function allows for the retrieval of a configured catalogue of bundles from USSDGW1. The bundle catalogue is a hierarchically arranged set of bundles and their properties. This allows for bundles to be configured centrally, with external systems retrieving the bundles when the catalogue configuration is updated.

The bundle hierarchy consists of up to three levels of bundle categories. Any category may contain bundles, and the two topmost bundle categories may contain sub bundle categories. This hierarchy is shown conceptually in the example below.

- Bundle Category 1: *Cheka*
  - Bundle Category 2: *All Net*
    - Bundle Category 3: *Daily*
      - Bundle 01: Bundle ID: nn
      - Bundle 02: Bundle ID: nn
      - Bundle 03: Bundle ID: nn
    - Bundle Category 3: *Weekly*
      - Bundle 04: Bundle ID: nn
      - Bundle 05: Bundle ID: nn
      - Bundle 06: Bundle ID: nn
    - Bundle Category 3: *Monthly*
      - Bundle 07: Bundle ID: nn
      - Bundle 09: Bundle ID: nn
      - Bundle 00: Bundle ID: nn
  - Bundle Category 2: *Vodacom to Vodacom*
    - Bundle Category 3: *Daily*
      - Bundle 10: Bundle ID: nn
      - Bundle 11: Bundle ID: nn
      - Bundle 12: Bundle ID: nn
    - Bundle Category 3: *Weekly*
      - Bundle 13: Bundle ID: nn
      - Bundle 14: Bundle ID: nn
      - Bundle 15: Bundle ID: nn

```
            -  Bundle Category 3: Monthly
                  -  Bundle 16: Bundle ID: nn
                  -  Bundle 17: Bundle ID: nn
                  -  Bundle 18: Bundle ID: nn
-  Bundle Category 1: SMS
      -  SMS Bundle 01: Bundle ID: nn
      -  SMS Bundle 02: Bundle ID: nn
      -  SMS Bundle 03: Bundle ID: nn
```

It is possible to configure multiple bundle catalogues on USSDGW1. Each one is a subset of the list of all of the Cheka bundles. This allows catalogues to be configured for various systems and scenarios (e.g., there could be a catalogue for a USSD menu, and a catalogue for a web portal, for instance). When requesting the catalogue, the name of the particular catalogue is specified.

13.1. Cheka Bundle Catalogue Request

The request body for the Cheka Bundle Catalogue function is shown in the listing below. A Cheka Bundle Catalogue request is represented with the `<chekaBundleCatalogue>` XML element.

```
<request>
       <username>conor</username>
       <password>password</password>
       <request>
              <chekaBundleCatalogue>
                     <key>vodafasta</key>
              </chekaBundleCatalogue>
       </request>
</request>
```

The `<chekaBundleCatalogue>` XML element contains the following sub elements:

1. **`<key>`**: The `<key>` element is a key used to lookup a particular bundle catalogue. The key values are configurable within the Cheka application, thus the value used for a particular scenario should be obtained from the service provider;

13.2. Cheka Bundle Catalogue Response

A Cheka Bundle Catalogue response snippet is shown in the listing below. The bundle catalogue is represented with the `<chekaBundleCatalogue>` XML element.

```
<response>
  <chekaBundleCatalogue>
    <bundleCategories>
      <bundleCategory>
        <name>
```

```
          <code>en</code>
          <text>Cheka</text>
        </name>
        <name>
          <code>sw</code>
          <text>Cheka</text>
        </name>
        <bundleCategories>
          <bundleCategory>
            <name>
              <code>en</code>
              <text>All Net</text>
            </name>
            <name>
              <code>sw</code>
              <text>Mitandao Yote</text>
            </name>
            <bundleCategories>
              <bundleCategory>
                <name>
                  <code>en</code>
                  <text>Daily</text>
                </name>
                <name>
                  <code>sw</code>
                  <text>Vifurushi vya Siku</text>
                </name>
                <bundles>
                  <bundle>
                    <ussdCode>*147</ussdCode>
                    <ussdId>150</ussdId>
                    <cost>150</cost>
                    <days>1</days>
                    <name>
                      <code>en</code>
                      <text>Cheka Tsh150 (Tsh 150)</text>
                    </name>
                    <name>
                      <code>sw</code>
                      <text>Cheka Tsh150 (Tsh 150)</text>
                    </name>
                    <description>
                      <code>en</code>
                      <text>EN Cheka Tsh150 description. Cost Tsh 150. Excl Tsh 127.12. Vat 18%. Vat
Amount Tsh 22.88</text>
                    </description>
                    <description>
                      <code>sw</code>
                      <text>SW Cheka Tsh150 description. Cost Tsh 150. Excl Tsh 127.12. Vat 18%. Vat
Amount Tsh 22.88</text>
                    </description>
                  </bundle>
```

The `<chekaBundleCatalogue>` XML element contains a `<bundleCategories>` XML element, which may contain a `<bundles>` and a further `<bundleCategories>` XML element. The `<bundles>` XML element represents a list of bundles within the bundle category. The

<bundleCategories> XML element represents a list of sub bundle categories within the bundle category.

Each <bundleCategories> XML element contains a list of <bundleCategory> XML elements. Each one represents a bundle category. Each <bundleCategory> contains the following sub XML elements:

1. **<name>**: The <name> XML element represents the name of the category. Each category may contain multiple <name> elements, each one representing the name of a category in a different language;
2. **<bundles>**: The <bundles> XML element represents the bundles within the category;
3. **<bundleCategories>**: The <bundleCategories> XML element represents a list of sub-bundle categories;

Each <bundles> element contains a list of <bundle> XML elements. The <bundle> XML element represents a bundle within a bundle category. Each <bundle> XML element contains the following sub XML elements:

1. **<name>**: The <name> XML element is a string that represents the name of the bundle. Each bundle may contain multiple <name> elements, each one representing the name of a bundle in a different language;
2. **<description>**: The <description> XML element is a string that represents the description of the bundle. Each bundle may contain multiple <description> elements, each one representing the description of the bundle in a different language;
3. **<ussdCode>**: <ussdCode> represents the USSD code that should be used when purchasing this bundle. The format of this field will be *[*nnn]+* where *nnn* is a number (e.g., *147, *148, *149*60);
4. **<ussdId>**: <ussdId> is a numeric value, and represents the bundle USSD ID used to identify the bundle. When purchasing the bundle, the value of <ussdCode> should be concatenated with the value of <ussdId> to form the USSD string sent to the Cheka application. The format of the string sent to the Cheka application for bundle purchase is as follows: *ussdCode*ussdId#* (for a normal bundle purchase) or *ussdCode*ussdId*MSISDN#* (for a buddy bundle purchase);
5. **<cost>**: <cost> is a numeric value representing the cost of the bundle in TSh;
6. **<days>**: <days> is a numeric value representing the number of days that the bundle is valid for;

The <name> and <description> XML elements have the following child XML elements:

1. **<code>**: <code> represents the two letter language code of the name or description (e.g., sw, en);

2. **<text>**: <text> is a string representing the actual name or description of the category or bundle;

14. Conclusion

All queries and corrections relating to this Cheka interface specification should be directed to Conor Solutions.