USSD S-Gateway XML/TCP/IP Interface

User’s Guide v1.1

Table of Contents

Introduction

LeibICT USSD S-Gateway supports applications over MAP (Mobile Application Part) (phase 2 and 3), either mobile or USSD application originated, using the message set defined in the protocol. The mobile originated data are received by the USSD application and forwarded to the HLR (Home Location Register), where as the network originated messages are forwarded to the HLR at once.

This document presents a description of the XML/TCP/IP Interface supported and managed by the LeibICT's USSD S-Gateway.

Capacity

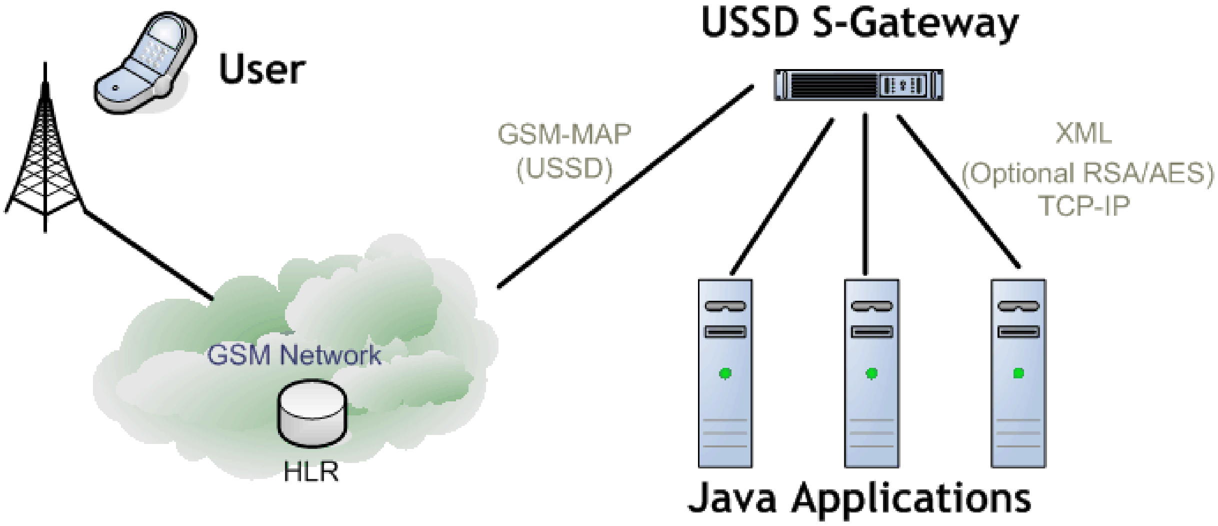
The current USSD S-Gateway version supports 65356 incoming and outgoing messages.

Each USSD transaction is identified by a dialog identifier (field named: *DialogueId*), represented by a 32bits integer. This parameter must be included in each dialogue message so the USSD application must be capable of controlling such parameter. For each outgoing message coming from the application, the latter must include a *Dialogue Id* between 0 and 65356.

The RSA encryption and decryption process is very high cpu consuming, setting up the right key length is needed to depending of the traffic estimated.

Typical system structure

The LeibICT USSD S-Gateway provides Load Balancing and Fault-Tolerance of applications:



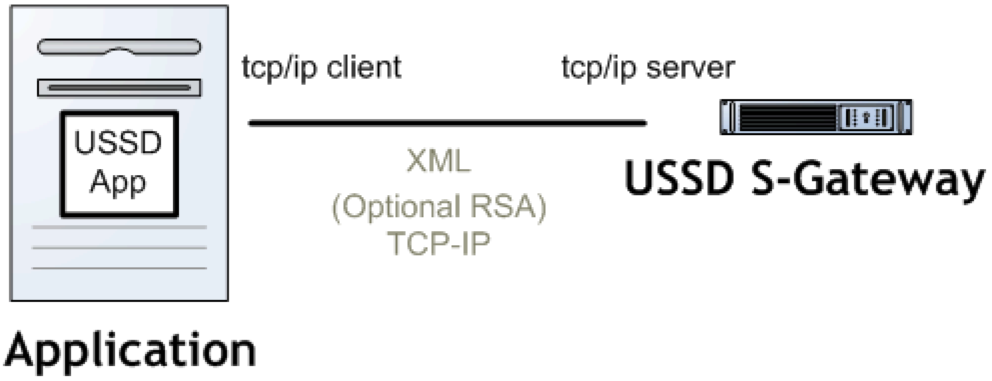
Two USSD S-Gateways can be paired to provide Fault-Tolerance on the gateway and GSM Network level.

TCP/IP Architecture

The XML/TCP/Interface consists in opened port and several XML messages being transmitted and received.

The USSD S-Gateway may send pings (a specific XML messages) within an internal pattern.

Also the application should ping the GW to test the connection.



Each XML messages is transferred using a 32bit unsigned integer valued that represents the length in octets of the XML string.

The XML string can potentially have the ending zero (\0) octet or not.

XML Messages

1. Ping/Pong

The first message the server will send is a ping, it must be replied with a pong:

```
<siggw><msg type="ping"/></siggw>
```

```
<siggw><msg type="pong"/></siggw>
```

2. Process USSD Request (invoke)

Once a subscriber starts a transaction the first messages contains the MSISDN of the subscriber and the short code entered.

```
<siggw>
        <msg type="begin">
                <user id="0"/>
                <dialog id="1"/>
                <context v="04000001001302"/>
                <user_info>
                        <object_identifier v="04000001010101"/>
                        <map operation="open">
                                <param name="destination" number="5989999909"  nai="1" npi="1"/>
                                <param name="origination" number="5989925603"  nai="1" npi="1"/>
                        </map>
                </user_info>
                <component type="invoke" id="1" op="process_ussd_request">
                        <param name="data_coding_scheme" v="15"/>
                        <param name="ussd_string" v="2A31323323"/>
                </component>
        </msg>
</siggw>
```

The USSD interaction is drived by a “dialog” each dialog is identified by a number (in the example the first “1” in red)

3. USSD Request (invoke)

The menus and prompts are sent to the subscriber using the next type of messages.

```
<siggw>
        <msg type="continue">
                <user id="0"/>
                <dialog id="1"/>
                <context v="04000001001302"/>
                <user_info>
                        <map operation="accept"/>
                </user_info>
                <component type="invoke" id="2" op="ussd_request">
                        <param name="data_coding_scheme" v="15"/>
                        <param                                   name="ussd_string"
v="4669727374204D656E750A312E20746573740A322E20696D7075740A332E20746F206
5786974"/>
                </component>
        </msg>
</siggw>
```

The invoke id (in the example, the “2” in red) must be increased every time during the same dialog.

4. USSD Request (result)

The result of a menu or prompt is received by the next type of messages:

```
<siggw>
        <msg type="continue">
                <user id="0"/>
                <dialog id="1"/>
                <component type="return_result_l" id="2" op="ussd_request">
                        <param name="data_coding_scheme" v="15"/>
                        <param name="ussd_string" v="32"/>
                </component>
        </msg>
</siggw>
```

5. Process USSD Request (result)

To end a ussd dialog the next message is used:

```
<siggw>
        <msg type="end">
                <user id="0"/>
                <dialog id="1"/>
                <component type="return_result_l" id="1" op="process_ussd_request">
                        <param name="data_coding_scheme" v="15"/>
                        <param                                    name="ussd_string"
v="596F7520656E7465726564202232"/>
                </component>
        </msg>
</siggw>
```

The final text to the subscriber is sent in red.

6. Any Time Interrogation (invoke)

To request the location of a subscriber:

```
<siggw>
        <msg type="begin">
                <user id="0"/>
                <dialog id="1"/>
                <context v="04000001001D03"/>
                <component type="invoke" id="0" op="any_time_interrogation">
                        <param name="subscriber_identity">
                                <param   name="msisdn"   number="18763808880"   nai="1"
npi="1"/>
                        </param>
                        <param name="requested_info">
                                <param name="location_information"/>
                        </param>
                        <param  name="gsm_scf_address"  number="18763808847"  nai="1"
npi="1"/>
                </component>
        </msg>
</siggw>
```

7. Any Time Interrogation (result)

The successful result to the ATI invoke:

```
<siggw>
        <msg type="end">
                <user id="0"/>
                <dialog id="0"/>
                <context v="04000001001D03"/>
                <component type="return_result" id="0" op="any_time_interrogation">
                        <param name="subscriber_info">
                                <param name="location_information">
                                        <param name="cell_glob_id_or_lai">
                                                <param
name="cell_glob_id_or_serv_area_fixed_length" mcc="506" mnc="563" lca="1" cid="4093"/>
                                        </parm>
                                </param>
                        </param>
                </component>
        </msg>
</siggw>
```

8. USSD Notify (invoke)

```
<siggw>
        <msg type="begin">
                <user id="0"/>
                <dialog id="2"/>
                <destination id="0"/>
                <context v="04000001001302"/>
                <user_info>
                        <object_identifier v="04000001010101"/>
                        <map operation="open">
                                <param  name="origination"  number="59899998931"    nai="1"
npi="1"/>
                                <param  name="destination"  number="59899256037"    nai="1"
npi="1"/>
                        </map>
                </user_info>
                <component type="invoke" id="1" op="unstructured_ss_notify">
                        <param name="data_coding_scheme" v="15"/>
                        <param name="ussd_string" v="746573742074657874"/>
                        <param name="msisdn" number="59899256037" nai="1" npi="1"/>
                </component>
        </msg>
</siggw>
```

9. Abort

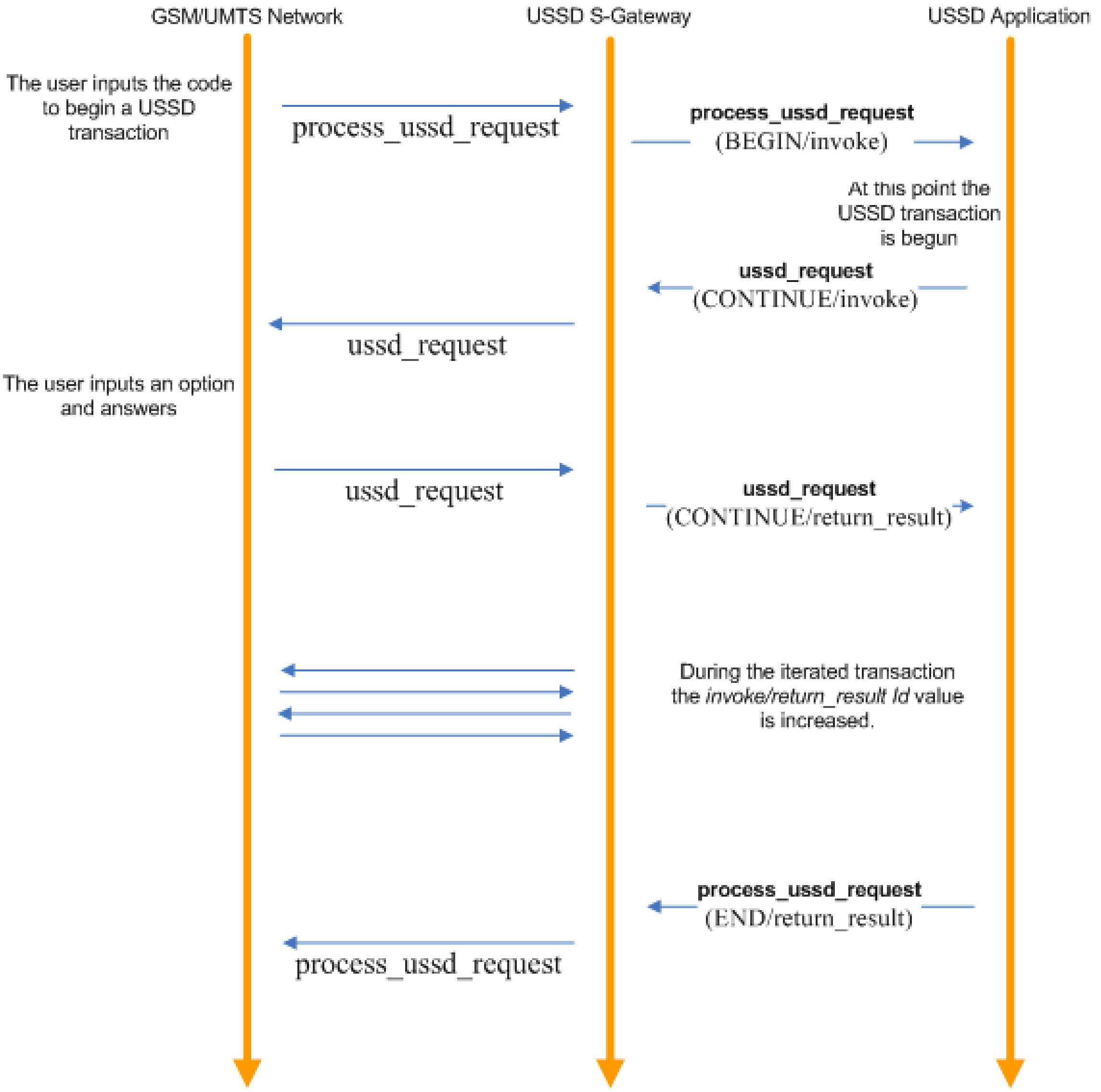
In case the subscribers end the dialog the next messages is received:

```
<siggw>
        <msg type="abort">
                <user id="0"/>
                <dialog id="1"/>
        </msg>
</siggw>
```

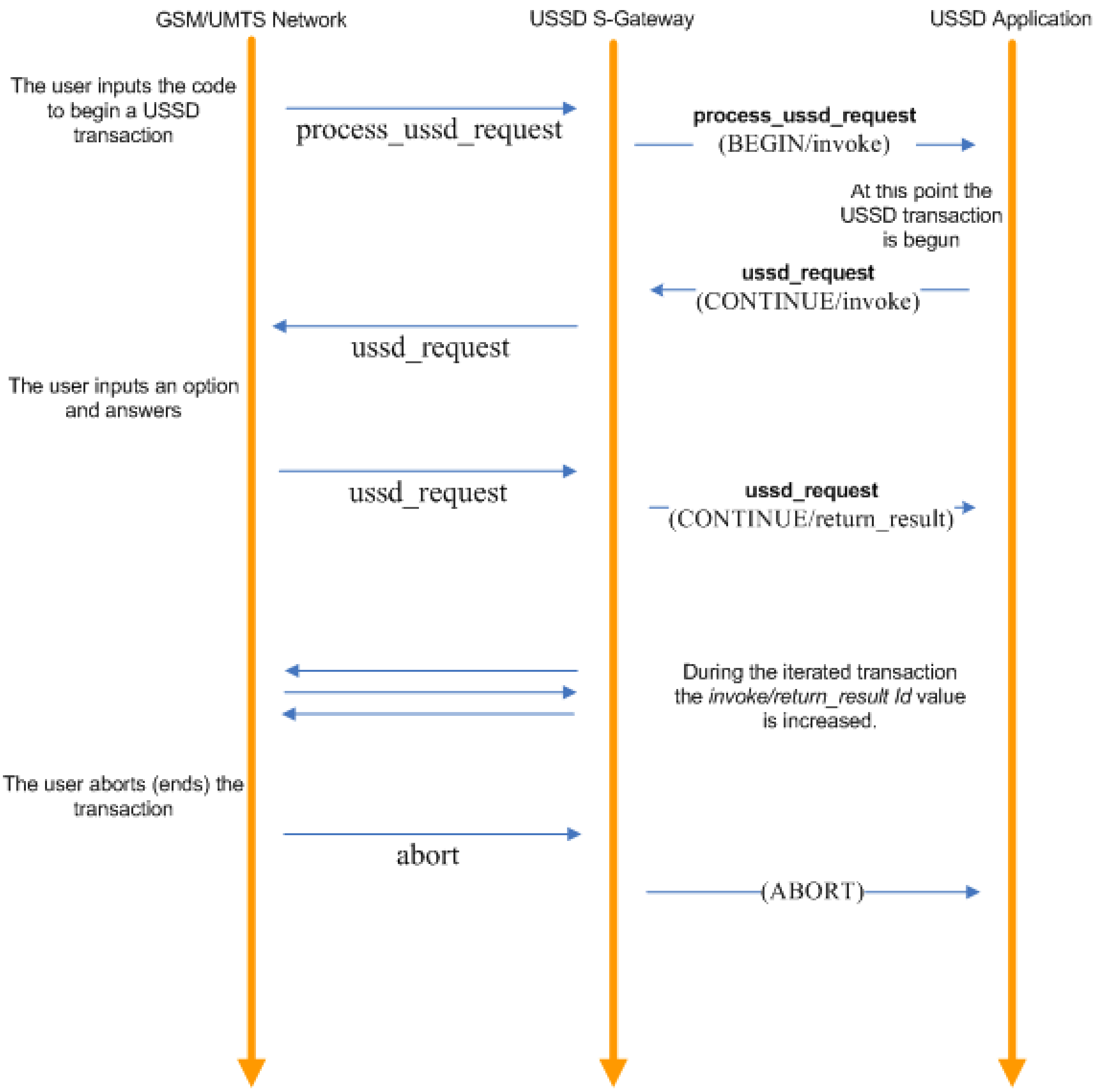
Example Message Sequences

The fallowing chapter gives example message sequences for interacting with USSDJNI Java API.

1. User Initiated USSD Transaction Ended by Application


GSM/UMTS Network
USSD S-Gateway
USSD Application
The user inputs the code to begin a USSD transaction
process_ussd_request
process_ussd_request (BEGIN/invoke)
At this point the USSD transaction is begun
ussd_request (CONTINUE/invoke)
ussd_request
The user inputs an option and answers
ussd_request
ussd_request (CONTINUE/return_result)
During the iterated transaction the invoke/return_result Id value is increased.
process_ussd_request (END/return_result)
process_ussd_request

2. User Initiated USSD Transaction Aborted by User


GSM/UMTS Network
USSD S-Gateway
USSD Application
The user inputs the code to begin a USSD transaction
process_ussd_request
process_ussd_request (BEGIN/invoke)
At this point the USSD transaction is begun
ussd_request (CONTINUE/invoke)
ussd_request
The user inputs an option and answers
ussd_request
ussd_request (CONTINUE/return_result)
During the iterated transaction the *invoke/return_result Id* value is increased.
The user aborts (ends) the transaction
abort
(ABORT)

Length of the USSD String

In GSM 0902 160 octets is stated as the maximum length for the USSD string. Due to underlying signalling layers the maximum length of the USSD string depending on the message is:

USSD operation	Max length
Begin, Invoke ProcessUSSDRequest	133
End, Result ProcessUSSDRequest	160
First Continue, Invoke USSDRequest in mobile initiated dialogue	154
Begin, Invoke USSDRequest	144
First Continue, Result USSDRequest in network initiated dialogue	154
Other messages	160