Mobile ID Client

Reference Guide

Version: 2.7

Contents

1 Introduction

The purpose of this document is to provide technical clarifications and guidelines on **WHO** has **WHAT**, **WHERE** and **HOW** to implement (in its own environment) to use the Swisscom Mobile ID Web Service.

By connecting to the Swisscom Mobile ID web service, all end-users from the participating Mobile Network Operators (MNO) - Swisscom, Orange Switzerland and Sunrise - can be addressed with Mobile ID requests.

This reference guide assumes that you are familiar with general Web Services (SOAP, RESTful, XML, JSON) and Application Server concepts and with the specific environment in which you are installing/configuring the Mobile ID clients.

⚠ A detailed technical Mobile ID platform description, the supported ETSI standards, the required Java SIM card as well as the Public-Key-Infrastructure (PKI) are out of scope.

1.1 Terms and abbreviations

Abbreviation	Definition
ℹ	Please note
⚠	Be careful, important
AP	Application Provider
DTBD	Data To Be Displayed
DTBS	Data To Be Signed
HB	Heartbeat
JSON	JavaScript Object Notation is a text-based open standard designed for human readable data interchange. Although derived from the JavaScript scripting language it is language independent. The JSON format is often used for serializing and transmitting structured data over a network connection, primarily between a server and a web application, as an alternative to XML.
LAN-I	Swisscom LAN-Interconnect Service
MID	Mobile ID platform providing the mobile signature service
MNO	Mobile Network Operator, also known as a wireless service provider, wireless carrier, cellular company, or mobile network carrier.
MSISDN	Number uniquely identifying a subscription in a GSM/UMTS mobile network
MSSP	Mobile Signature Service Provider
OTA	Over-The-Air is a technology used to communicate with and manage a SIM card without being connected physically to the card.
RESTful	Representational State Transfer is a style of software architecture for distributed systems such as the World Wide Web. It is based on the existing design of HTTP/1.0. REST-style architectures consist of clients and servers. Clients initiate requests to servers; servers process requests and return appropriate responses.
SOAP	Simple Object Access Protocol (SOAP) is a protocol specification for exchanging structured information in the implementation of Web Services relying on Extensible Markup Language (XML)
WS	A Web service (WS) is a method of communication between two electronic devices over the Web (Internet).
WSDL	The Web Services Description Language (WSDL) is an XML-based language that is used for describing the functionality offered by a Web service.
X509	PKI and Digital Certificates
XML	Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

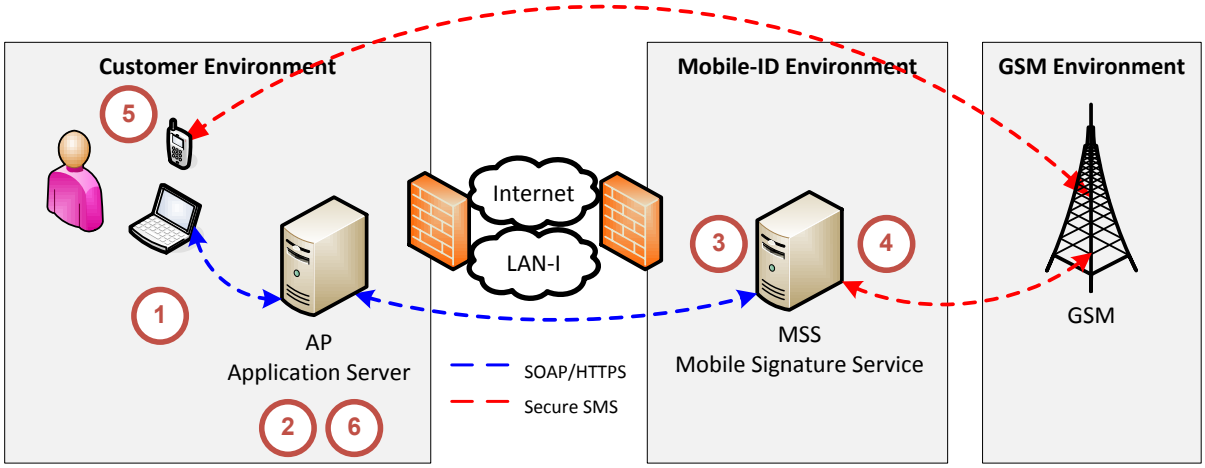
1.2 Referenced documents

[1] ETSI TS 102 204 V1.1.4 (2003-08)

2 Overview and main scenario

Before entering into more technical details, let's have a short look at the main scenario.

- Scenario - Strong Authentication:
  The end-user would like to access to a corporate application protected by Mobile ID strong authentication.



The main processing steps that are performed by the end-user and the mobile signature service are described below:

1. The end-user uses any application relying on Mobile ID for authentication, which sends a mobile signature request through the dedicated web interface (of the AP) including the personal MSISDN as input parameter to login.
2. The AP receives the end-user request, forms the contents to be signed (in accordance with the ETSI TS 102 204 standard) and forwards the request to the MID service.
3. The MID platform receives the signature request and validates the AP in accordance with the service agreement.
4. The MID platform ensures that the end-user signature request is allowed and forwards the signature request to the end-user's mobile phone.
5. The end-user gets a message on his mobile phone to sign the mobile signature request. The End-User signs the request by entering his Mobile ID PIN code.
6. After the AP has received a valid electronic signature, the end-user will be granted access to the corporate application.

3 Preconditions and assumptions

Before using the Swisscom Mobile ID web service, some provisioning steps are required.

1. The customer (your company) has an agreement with Swisscom and is provisioned on the MID platform.
    a. The connectivity (Internet or LAN-I) between the customer and Mobile ID is established.
    b. The customer has delivered their public AP IP address (or range) to configure the network infrastructure including firewalls on the MID platform side.
    c. The customer has delivered an X.509 certificate (SSL certificate[1]); to configure the mutual authentication on MID platform side.
2. The customer has received from Swisscom his AP_ID and AP_PWD, the Service Endpoint(s), the Signature Profile(s) and the relevant Swisscom CA certificates.

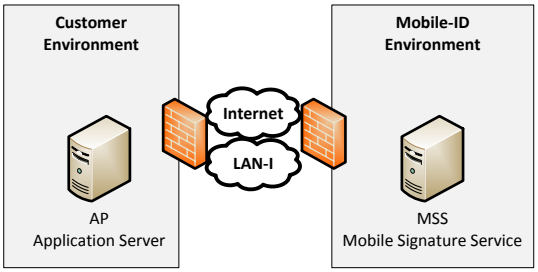
3.1 Client AP Integration

The Swisscom Mobile ID web service is securely accessible through LAN-I[2] and Internet. If not otherwise specified use the following default access configuration information:

Base-URL

Via Internet:
https://mobileid.swisscom.com

Via Swisscom LAN-I:
https://195.65.233.222[3]



Service Endpoints:

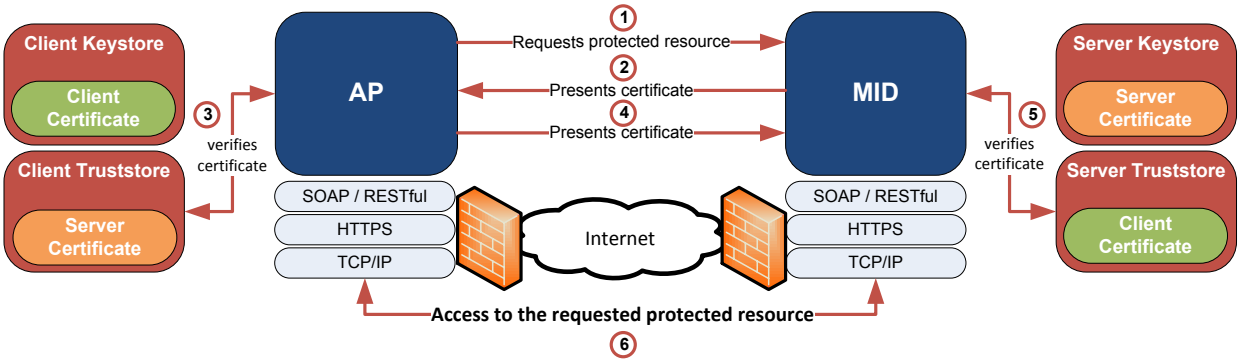
SOAP: <**Base-URL**>/soap/services/MSS_SignaturePort
<**Base-URL**>/soap/services/MSS_StatusQueryPort
<**Base-URL**>/soap/services/MSS_ReceiptPort
<**Base-URL**>/soap/services/MSS_ProfilePort

RESTful: <**Base-URL**>/rest/service

[1] The Customer is responsible to check the validity of his certificate
[2] http://www.swisscom.ch/solutions/Loesungen-Produkte/LAN-Interconnect-Service-LAN-I
[3] LAN-I works with IPs where Internet works with names.

3.2 Mutual authentication

MID requires a certificate-based mutual authentication. When using certificate-based mutual authentication, the following actions occur:



1. The client AP requests access to a protected resource on MID.
2. The MID Web-Server presents its server certificate to the client AP.
3. The client AP verifies the MID server certificate.
4. If successful, the client AP sends its client certificate to the MID server.
5. The MID server verifies the AP client credentials.
6. If successful, the MID server grants access to the protected resource requested by the client AP.

Important guidelines for the certificate-based mutual authentication:

- Authentication at the MID side does not consider any validation of a client certificate chain or restrictions of the root CA. **The client shall send only its end entity certificate**. The authentication is denied in case the client sends a bag with the full certificate chain.
- Enhanced Key Usage value of client certificates must contain Client Authentication (1.3.6.1.5.5.7.3.2). Chapter 7 contains examples on how to create self-signed certificates.
- It is critically important that all your requests to the Mobile ID Service are coming from the server controlled by you. Never make requests to the service directly from the client side (e.g. a Mobile App or JavaScript), as this will compromise your credentials, and allow unauthorised access.
- The CA that issued the MID SSL server certificate is part of global root programs[4] and should be present in most of the clients. If this is not the case, to validate the chain of trust it will be sufficient to add the "root" certificate to the client TrustStore. The intermediate CAs are returned by the MID server and may change like for any web server. Refer to chapter 5.6 for more details.

[4] References: https://cabforum.org, https://technet.microsoft.com/en-us/library/cc751157.aspx, https://www.swissdigicert.ch/sdcs/portal/page?node=services, https://www.swissdigicert.ch/sdcs/portal/open_pdf?file=english%2FDeployment_of_Swisscom_Root_CA_Certificates.pdf

4 MID Web Service

MID exposes web services based on SOAP or RESTful (JSON).

In case of SOAP, Swisscom provides the WSDL file that describes the interface methods that can be used by the AP client. The WSDL can be downloaded at http://git.io/a1vOag

4.1 HTTP/1.1 Header

You can POST signature requests via HTTP/1.1 using the SOAP or RESTful interface. The RESTful interface supports JavaScript Object Notation (JSON) as media type.

The header fields should be set as follows:

HEADER FIELD	SOAP	RESTFUL/JSON
Content-Type	text/xml;charset=UTF-8	application/json;charset=UTF-8
Accept	-	application/json

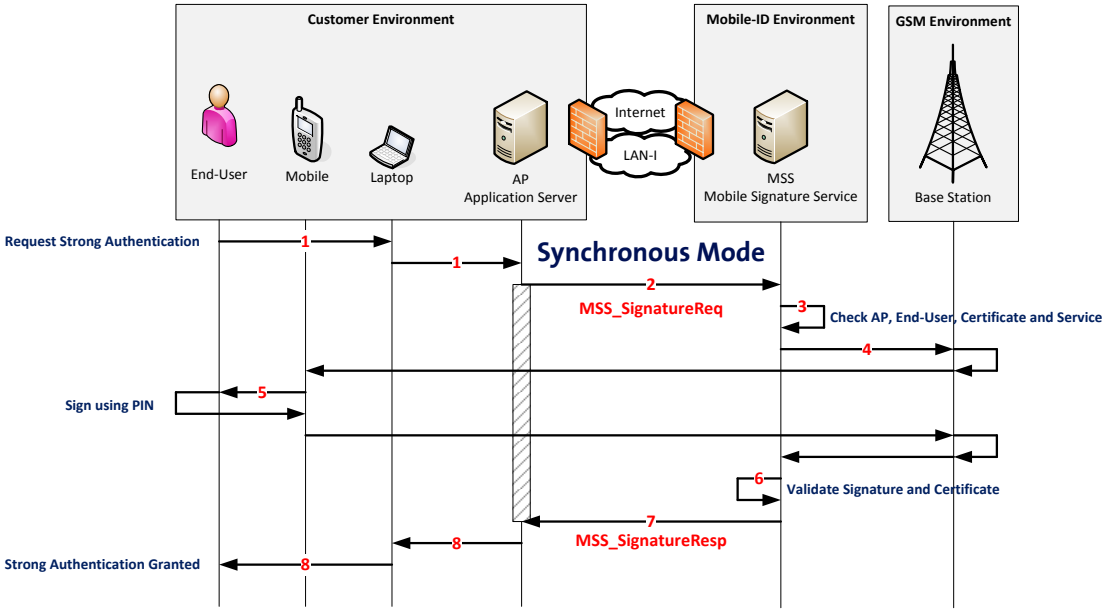
4.2 The Mobile Signature

The standard has defined them and Swisscom MID offers both synchronous and asynchronous (client-server) modes[5].

The MSS_Signature method is used to submit the mobile signature request message (MSS_SignatureReq). The result is provided within the signature response message (MSS_SignatureResp).

4.2.1 Synchronous Mode

In the synchronous mode, the AP initiates the signature request by calling MSS_SignatureReq, which is then blocked until the signature has been received or the signing times out occurs as depicted below.



1. End-User uses an application that sends an authentication request.
2. AP receives the request (performs spam protection if implemented) and sends a MSS_SignatureReq request message (MessagingMode="synch") to MID.
3. MID backend checks the credentials.
4. MID sends a signature request over GSM to SIM applet. This request is protected by the GSM network encryption and by the Mobile ID encryption.
5. SIM applet prompts the End-User to enter the Mobile ID PIN. End-User enters the PIN. The SIM applet signs the request and sends it back over GSM to MID.
6. MID receives the signature response and sends a complete response to the AP.
7. Depending on the response of MID the AP may grant or deny access to the End-User.

[5] Swisscom Mobile ID does not support the asynchronous server-server mode where the server does a call-back to the client

Request and Response Messages: Synchronous Signature

In red highlighted the most important input/output value required for a successful signature request:

SOAP MSS_SignatureReq

```
<soapenv:Envelope soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <MSS_Signature>
         <mss:MSS_SignatureReq MinorVersion="1" MajorVersion="1" MessagingMode="synch" TimeOut="80"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info>
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:DataToBeSigned MimeType="text/plain" Encoding="UTF-8">TEXT_TO_BE_SIGNED</mss:DataToBeSigned>
            <mss:SignatureProfile>
               <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
            </mss:SignatureProfile>
             <mss:AdditionalServices>
                <mss:Service>
                  <mss:Description>
                     <mss:mssURI>http://mss.ficom.fi/TS102204/v1.0.0#userLang</mss:mssURI>
                  </mss:Description>
                  <fi:UserLang>LANGUAGE</fi:UserLang>
               </mss:Service>
             </mss:AdditionalServices>
         </mss:MSS_SignatureReq>
      </MSS_Signature>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_SignatureReq

```
{
    "MSS_SignatureReq": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_PWD": "yourAP_PWD",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T12:00:00.000+01:00"
        },
        "AdditionalServices": [
            {
                "Description": "http://mss.ficom.fi/TS102204/v1.0.0#userLang",
                "UserLang": {
                    "Value": "LANGUAGE"
                }
            }
        ],
        "DataToBeSigned": {
            "Data": "TEXT_TO_BE_SIGNED",
            "Encoding": "UTF-8",
            "MimeType": "text/plain"
        },
        "MSSP_Info": {
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MessagingMode": "synch",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "SignatureProfile": "http://mid.swisscom.ch/MID/v1/AuthProfile1",
        "TimeOut": "80"
    }
}
```

SOAP MSS_SignatureResp

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_SignatureResponse>
         <mss:MSS_SignatureResp MajorVersion="1" MinorVersion="1" MSSP_TransID="h4dhx"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:MSS_Signature>
               <mss:Base64Signature>MIIIdwYJKoZIhvc...</mss:Base64Signature>
            </mss:MSS_Signature>
            <mss:SignatureProfile>
               <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
            </mss:SignatureProfile>
            <mss:Status>
               <mss:StatusCode Value="500"/>
               <mss:StatusMessage>SIGNATURE</mss:StatusMessage>
            </mss:Status>
         </mss:MSS_SignatureResp>
      </MSS_SignatureResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_SignatureResp

```
{
    "MSS_SignatureResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MSSP_TransID": "h44okl",
        "MSS_Signature": {
            "Base64Signature": "MIIIdwYJKoZIhvc..."
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "Status": {
            "StatusCode": {
                "Value": "500"
            },
            "StatusMessage": "SIGNATURE"
        }
    }
}
```

The 'Base64Signature' content is a base 64 encoded CMS signature object. It contains the DataToBeSigned (DTBS) message that has been signed by the SIM applet (during the signature process). In addition, it includes the mobile user (end entity) certificate and all related Swisscom intermediate CAs. Therefore, the AP will always be able to fully validate the signature response by itself..

Refer to chapter 5 for best practices (response handling, user mapping and signature validation) and chapter 4.2.3.2 for validation hints.

In case of errors where a specific Fault sub-code will be raised here a sample response:

SOAP Fault Response

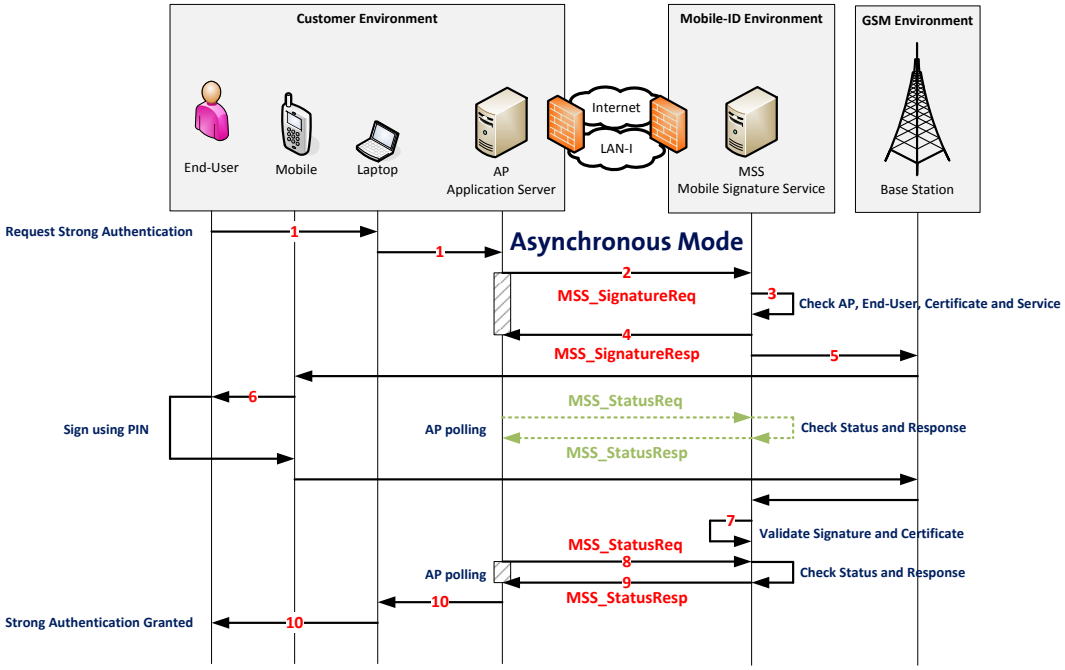
```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <soapenv:Fault>
         <soapenv:Code>
            <soapenv:Value>soapenv:Sender</soapenv:Value>
            <soapenv:Subcode xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
             xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
               <soapenv:Value>mss:_101</soapenv:Value>
            </soapenv:Subcode>
         </soapenv:Code>
         <soapenv:Reason>
            <soapenv:Text xml:lang="en">WRONG_PARAM</soapenv:Text>
         </soapenv:Reason>
         <soapenv:Detail>
            <ns1:detail xmlns:ns1="http://kiuru.methics.fi/mssp">Illegal msisdn</ns1:detail>
         </soapenv:Detail>
      </soapenv:Fault>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON Fault Response

```
{
    "Fault": {
        "Code": {
            "SubCode": {
                "Value": "_101",
                "ValueNs": "http://uri.etsi.org/TS102204/v1.1.2#"
            },
            "Value": "Sender",
            "ValueNs": "http://www.w3.org/2003/05/soap-envelope"
        },
        "Detail": "Illegal msisdn",
        "Reason": "WRONG_PARAM"
    }
}
```

4.2.2 Asynchronous Mode (client-server)

In the asynchronous mode, the AP initiates the signature request by calling MSS_Signature method, an acknowledgment response is sent back immediately by the MID to AP. After an adequate break[6] (enough time for the user to receive and sign the requests), AP starts polling using MSS_StatusQuery service to receive the response as depicted below.



1. End-User uses an application that sends an authentication request.
2. AP receives the request (performs spam protection if implemented) and sends an asynchronous MSS_SignatureReq request message to MID.
3. MID backend checks the credentials.
4. MID responds to AP with MSS_SignatureResp[7] message (acknowledgment).
5. MID sends a signature request over GSM to SIM applet. This request is protected by the GSM network encryption and by the Mobile ID encryption.
6. SIM applet receives the prompts the End-User to sign with PIN. End-User enters the PIN. SIM applet signs the request and sends it back over GSM back to MID (encrypted as well).

Meanwhile the AP sends MSS_StatusReq requests to MID. The MID replies with MSS_StatusResp[7] (status code "504 OUTSTANDING_TRANSACTION", which means that the AP will need to call again the status method).

7. MID receives the signature response.
8. AP sends next MSS_StatusReq request message to MID.
9. MID sends a complete response to AP.
10. Depending on the response of MID, the AP may grant or deny access to the End-User.

[6] Swisscom recommends to wait 10 seconds between MSS_SignatureResp and the very first MSS_StatusReq (polling)
[7] In case of an error, MID may replies with a SoapFault containing an error code as described in chapter 6.

Request and Response Messages: Asynchronous Signature

In blue the major differences compared to the sync mode.
Note that the value isn't the same between SOAP and JSON.

SOAP MSS_SignatureReq

```
<soapenv:Envelope soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <MSS_Signature>
         <mss:MSS_SignatureReq MinorVersion="1" MajorVersion="1" MessagingMode="asynchClientServer"
          TimeOut="80" xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
          xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info>
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:DataToBeSigned MimeType="text/plain" Encoding="UTF-8">TEXT_TO_BE_SIGNED</mss:DataToBeSigned>
            <mss:SignatureProfile>
               <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
            </mss:SignatureProfile>
             <mss:AdditionalServices>
                <mss:Service>
                  <mss:Description>
                     <mss:mssURI>http://mss.ficom.fi/TS102204/v1.0.0#userLang</mss:mssURI>
                  </mss:Description>
                  <fi:UserLang>LANGUAGE</fi:UserLang>
               </mss:Service>
             </mss:AdditionalServices>
         </mss:MSS_SignatureReq>
      </MSS_Signature>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_SignatureReq

```
{
    "MSS_SignatureReq": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_PWD": "yourAP_PWD",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T12:00:00.000+01:00"
        },
        "AdditionalServices": [
            {
                "Description": "http://mss.ficom.fi/TS102204/v1.0.0#userLang",
                "UserLang": {
                    "Value": "LANGUAGE"
                }
            }
        ],
        "DataToBeSigned": {
            "Data": "TEXT_TO_BE_SIGNED",
            "Encoding": "UTF-8",
            "MimeType": "text/plain"
        },
        "MSSP_Info": {
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MessagingMode": "asynch",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "SignatureProfile": "http://mid.swisscom.ch/MID/v1/AuthProfile1",
        "TimeOut": "80"
    }
}
```

The received MSS_SignatureResp message will contain the following parameters/values:

SOAP MSS_SignatureResp

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_SignatureResponse>
         <mss:MSS_SignatureResp MajorVersion="1" MinorVersion="1" MSSP_TransID="h4iof"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:SignatureProfile>
               <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
            </mss:SignatureProfile>
            <mss:Status>
               <mss:StatusCode Value="100"/>
               <mss:StatusMessage>REQUEST_OK</mss:StatusMessage>
            </mss:Status>
         </mss:MSS_SignatureResp>
      </MSS_SignatureResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_SignatureResp

```
{
    "MSS_SignatureResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MSSP_TransID": "h4iof",
        "MajorVersion": "1",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusMessage": "REQUEST_OK"
        }
    }
}
```

Request and Response Messages: Status Polling

At this point in time, AP has received the **MSS_SignatureResp** with the **MSSP_TransID="h4iof"**. The AP has to take the received **MSSP_TransID="h4iof"** and **<mss:URI>** of MSSP_ID and use it in the **MSS_StatusReq**:

SOAP MSS_StatusReq

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
   <soapenv:Body>
      <MSS_StatusQuery>
         <mss:MSS_StatusReq MinorVersion="1" MajorVersion="1" MSSP_TransID="h4iof"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info>
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
         </mss:MSS_StatusReq>
      </MSS_StatusQuery>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_StatusReq

```
{
  "MSS_StatusReq": {
    "MajorVersion": "1",
    "MinorVersion": "1",
    "AP_Info": {
      "AP_ID": "yourAP_ID",
      "AP_PWD": "yourAP_PWD",
      "Instant":"2015-01-01T12:00:00.000+01:00",
      "AP_TransID":"REF0101120000"
    },
    "MSSP_Info": {
      "MSSP_ID": {
        "URI": "http://mid.swisscom.ch/"
      }
    },
    "MSSP_TransID": "h4iof"
  }
}
```

The received MSS_StatusResp message will contain the following parameters/values:

SOAP MSS_StatusResp

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_StatusQueryResponse>
         <mss:MSS_StatusResp MajorVersion="1" MinorVersion="1"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:MSS_Signature>
               <mss:Base64Signature>MIIdwYJKoZIhvc...</mss:Base64Signature>
            </mss:MSS_Signature>
            <mss:Status>
               <mss:StatusCode Value="500"/>
               <mss:StatusMessage>SIGNATURE</mss:StatusMessage>
            </mss:Status>
         </mss:MSS_StatusResp>
      </MSS_StatusQueryResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_StatusResp

```
{
    "MSS_StatusResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MSS_Signature": {
            "Base64Signature": "MIIdwYJKoZIhvc..."
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "MobileUser": MOBILE_NUMBER,
        "Status": {
            "StatusCode": {
                "Value": "500"
            },
            "StatusMessage": "SIGNATURE"
        }
    }
}
```

Because the AP can call the status method at any time, the status value can take any value among error codes and status codes. A typical status code for this method is **504**, which means that the transaction is not yet completed and the AP will need to call **MSS_StatusReq** again (polling). Example for a MSS_StatusResp with a 504 Status Code instead of 500:

```
<mss:Status>
   <mss:StatusCode Value="504"/>
   <mss:StatusMessage>OUTSTANDING_TRANSACTION</mss:StatusMessage>
</mss:Status>
```

```
"Status": {
    "StatusCode": { "Value": "504" },
    "StatusMessage": "OUTSTANDING_TRANSACTION"
}
```

Swisscom recommends to use synchronous mode unless you have a compelling reason to use asynchronous mode. Synchronous mode will ensure that the signature response is immediately processed by the AP after is has been made available by the Mobile ID Service.
Hence the login time is usually faster compared to an asynchronous mode, just because of the polling interval.

4.2.3 Mobile Signature Additional Services (AS)

The MSS Signature supports additional services that may be requested.

```
<mss:AdditionalServices>
  <mss:Service>
    <mss:Description>
      <mss:mssURI>URI</mss:mssURI>
    </mss:Description>
    (..)
  </mss:Service>
</mss:AdditionalServices>
```

```
"AdditionalServices": [
  {
    "Description": "URI",
    (..)
  }
]
```

4.2.3.1 UserLang

URI:

```
http://mss.ficom.fi/TS102204/v1.0.0#userLang
```

⚠ The UserLang is a mandatory service for all Signature Requests.

One of the supported language (EN, DE, FR, IT) must be defined. Please refer to chapter 4.2 for an example.

It should correspond to the language of the DataToBeSigned (DTBS) text message.

4.2.3.2 Validate

⚠ This service has been marked as obsolete and should no longer be used.

4.2.3.3 Subscriber Info

URI:

```
http://mid.swisscom.ch/as#subscriberInfo
```

This AS allows any authorized[8] AP to request subscriber info to be returned in a successful MSS Signature response (either as part of the synchronous MSS Signature Response or as part of one of the MSS Status Query Response in case of asynchronously processed signatures).

As of today, the subscriber info AS contains the value of the Current Operator, with numerical code "1901". The value of this code is in the form of an "**MCCMNC**" string value, representing the Mobile Country Code (MCC) and Mobile Network Code (MNC) of the network where the Mobile User (subscriber) is currently registered at the time of the signature acquisition[9]. The AP may use this service to know, whether or not the Mobile User is currently located in Switzerland.

Example value: "22801" (228=Switzerland, 01=Swisscom)

If the AP requests the subscriber info AS without being authorized to use it, the MSS Signature Response or MSS Status Query response will contain the AS response with an empty subscriber info detail.

If the AP requests the subscriber info AS and he is authorized to use it, but the Mobile ID Service was not able to retrieve the values, then the MSS Signature Response or MSS Status Query Response will contain the AS response with the authorized code (1901), having the values filled with the string "unknown".

If the AP requested the subscriber info AS but the signature failed, no subscriber info AS related content will be returned to the AP as part of the MSS Signature Response or MSS Status Query Response.

[8] The subscriber info AS is only available to APs that are authorized to use this additional service
[9] http://en.wikipedia.org/wiki/Mobile_country_code

Request and Response Messages: MSS Signature + Subscriber Info AS

SOAP MSS_SignatureReq + Subscriber Info AS

```
      <MSS_Signature>
         <mss:MSS_SignatureReq MinorVersion="1" MajorVersion="1" MessagingMode="synch" TimeOut="80"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info>
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:DataToBeSigned MimeType="text/plain" Encoding="UTF-8">TEXT_TO_BE_SIGNED</mss:DataToBeSigned>
            <mss:SignatureProfile>
               <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
            </mss:SignatureProfile>
             <mss:AdditionalServices>
                <mss:Service>
                  <mss:Description>
                     <mss:mssURI>http://mss.ficom.fi/TS102204/v1.0.0#userLang</mss:mssURI>
                  </mss:Description>
                  <fi:UserLang>LANGUAGE</fi:UserLang>
               </mss:Service>
               <mss:Service>
                  <mss:Description>
                     <mss:mssURI>http://mid.swisscom.ch/as#subscriberInfo</mss:mssURI>
                  </mss:Description>
               </mss:Service>
             </mss:AdditionalServices>
         </mss:MSS_SignatureReq>
      </MSS_Signature>
```

JSON MSS_SignatureReq + Subscriber Info AS

```
{
    "MSS_SignatureReq": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_PWD": "yourAP_PWD",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T12:00:00.000+01:00"
        },
        "AdditionalServices": [
            {
                "Description": "http://mid.swisscom.ch/as#subscriberInfo"
            },
            {
                "Description": "http://mss.ficom.fi/TS102204/v1.0.0#userLang",
                "UserLang": {
                    "Value": "LANGUAGE"
                }
            }
        ],
        "DataToBeSigned": {
            "Data": "TEXT_TO_BE_SIGNED",
            "Encoding": "UTF-8",
            "MimeType": "text/plain"
        },
        "MSSP_Info": {
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MessagingMode": "synch",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "SignatureProfile": "http://mid.swisscom.ch/MID/v1/AuthProfile1",
        "TimeOut": "80"
    }
}
```

SOAP MSS_SignatureResp + Subscriber Info AS

```
<MSS_SignatureResponse>
   <mss:MSS_SignatureResp MajorVersion="1" MinorVersion="1" MSSP_TransID="h44okl"
    xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#" xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
      <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
       Instant="2015-01-01T12:00:00.000+01:00"/>
      <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
         <mss:MSSP_ID>
            <mss:URI>http://mid.swisscom.ch/</mss:URI>
         </mss:MSSP_ID>
      </mss:MSSP_Info>
      <mss:MobileUser>
         <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
      </mss:MobileUser>
      <mss:MSS_Signature>
         <mss:Base64Signature>MIIIdwYJKoZIhvc...</mss:Base64Signature>
      </mss:MSS_Signature>
      <mss:SignatureProfile>
         <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
      </mss:SignatureProfile>
      <mss:Status>
         <mss:StatusCode Value="500"/>
         <mss:StatusMessage>SIGNATURE</mss:StatusMessage>
         <mss:StatusDetail>
            <fi:ServiceResponses>
               <fi:ServiceResponse>
                  <fi:Description>
                     <mss:mssURI>http://mid.swisscom.ch/as#subscriberInfo</mss:mssURI>
                  </fi:Description>
                  <ns1:SubscriberInfo xmlns:ns1="http://mid.swisscom.ch/TS102204/as/v1.0">
                     <ns1:Detail id="1901" value="22801"/>
                  </ns1:SubscriberInfo>
               </fi:ServiceResponse>
            </fi:ServiceResponses>
         </mss:StatusDetail>
      </mss:Status>
   </mss:MSS_SignatureResp>
</MSS_SignatureResponse>
```

JSON MSS_SignatureResp + Subscriber Info AS

```
{
    "MSS_SignatureResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MSSP_TransID": "h44okl",
        "MSS_Signature": {
            "Base64Signature": "MIIIdwYJKoZIhvc..."
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "ServiceResponses": [ {
            "Description": "http://mid.swisscom.ch/as#subscriberInfo",
            "SubscriberInfo": {
                "Details": [ {
                    "id": "1901",
                    "value": "22801"
                } ]
            }
        } ],
        "Status": {
            "StatusCode": {
                "Value": "500"
            },
            "StatusMessage": "SIGNATURE"
        }
    }
```

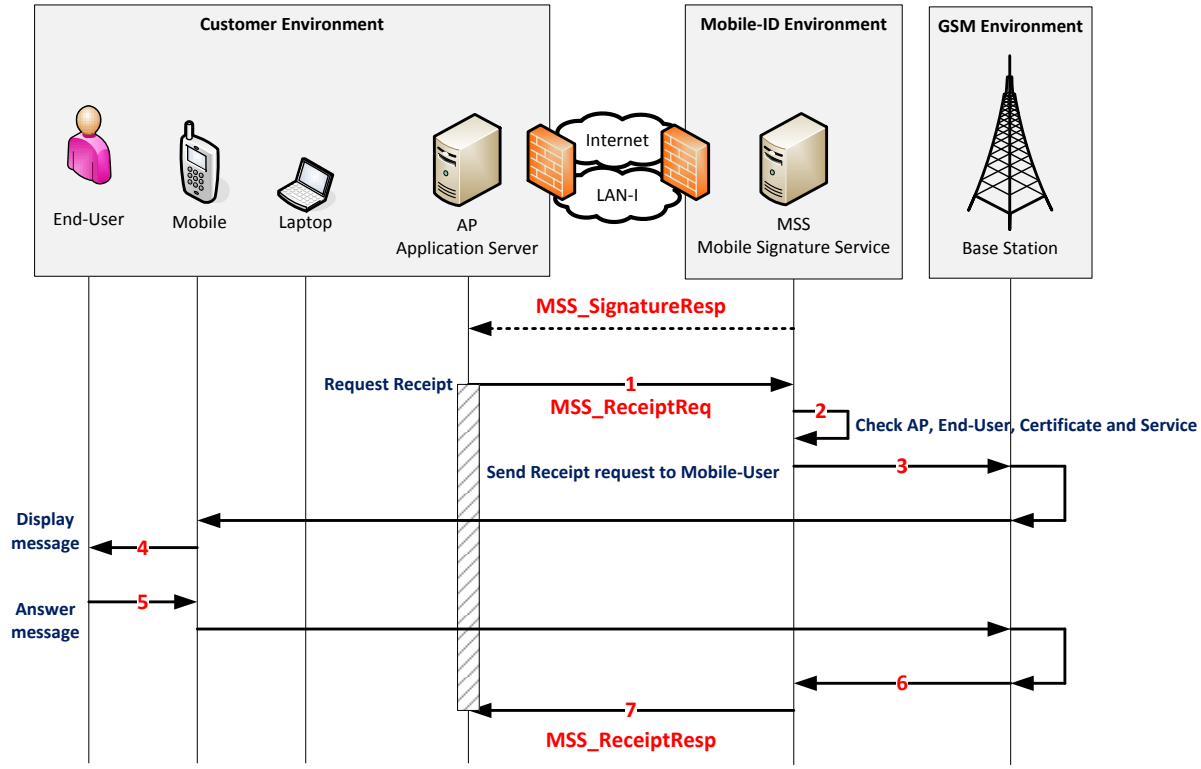
4.3 The Mobile Signature Receipt

An AP may use this operation after a successful mobile signature request in order to provide the End-User with some kind of a "receipt" that informs him of the proceeding of the mobile signature transaction.

Swisscom MID offers both synchronous and asynchronous (client-server) modes.

4.3.1 Synchronous Mode

In the synchronous mode, the AP initiates the receipt request by calling MSS_ReceiptReq, which is then blocked until the signature has been acknowledged, cancelled or the signing times out occurs. The picture depicted below, shows the successful case.



1. For each successful MSS_SignatureResp the AP can request a MSS_ReceiptReq by passing the same MSSP_TransID and same MSISDN from the MSS_SignatureResp and defining message to be displayed to the End-User. Optionally the message to be displayed can be encrypted with the public key from the MSS_SignatureResp.
2. MID backend checks the credentials.
3. MID sends a receipt request over GSM to SIM applet including the message to be displayed.
4. SIM applet displays the message to the End-User. If the message is encrypted the End-User will have to enter the Mobile ID PIN in order to decrypt it with his private key.
5. The user press “ok” or “cancel” which trigger MSS Receipt response containing the “user response”.
6. The Receipt response is sent to Mobile ID.
7. MID sends MSS_ReceiptResp to the AP containing the “User response” of the MSS Receipt request

Request and Response Messages: Receipt

At this point in time, AP has received the MSS_SignatureResp with the **MSSP_TransID="h4iof"**. The AP has to take the received **MSSP_TransID="h4iof"** and use it in the next **MSS_ReceiptReq** message.

Moreover, the AP has to set the LANGUAGE (one of EN, DE, FR, IT) which is usually the same as used in the preceding signature. All other values in the pink highlighted section (e.g. ReceiptMessagingMode, UserAck or ReceiptProfileURI) should be set exactly as shown in the example below (other values aren't supported).

SOAP Synchronous MSS_ReceiptReq

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <MSS_Receipt>
         <mss:MSS_ReceiptReq MinorVersion="1" MajorVersion="1" MSSP_TransID="h4iof"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
          xmlns:sco="http://www.swisscom.ch/TS102204/ext/v1.0.0">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info>
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
            <mss:Status>
              <mss:StatusCode Value="100"/>
              <mss:StatusDetail>
                <sco:ReceiptRequestExtension ReceiptMessagingMode="synch" UserAck="true">
                   <sco:ReceiptProfile Language="LANGUAGE">
                     <sco:ReceiptProfileURI>http://mss.swisscom.ch/synch</sco:ReceiptProfileURI>
                   </sco:ReceiptProfile>
                </sco:ReceiptRequestExtension>
              </mss:StatusDetail>
            </mss:Status>
            <mss:Message MimeType="text/plain" Encoding="UTF-8">RECEIPT_MESSAGE</mss:Message>
         </mss:MSS_ReceiptReq>
      </MSS_Receipt>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON Synchronous MSS_ReceiptReq

```
{
    "MSS_ReceiptReq": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_PWD": "yourAP_PWD",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T12:00:00.000+01:00"
        },
        "MSSP_Info": {
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MSSP_TransID": "h4iof",
        "MajorVersion": "1",
        "Message": {
            "Data": "RECEIPT_MESSAGE",
            "Encoding": "UTF-8",
            "MimeType": "text/plain"
        },
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        },
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusDetail": {
                "ReceiptRequestExtension": {
                    "ReceiptMessagingMode": "synch",
                    "ReceiptProfile": {
                        "Language": "LANGUAGE",
                        "ReceiptProfileURI": "http://mss.swisscom.ch/synch"
                    },
                    "UserAck": "true"
                }
            }
        }
}   }
```

If the user clicked “ok”, the MSS_ReceiptResp message will contain the attribute UserAck set to “true” and the attribute UserResponse with the status value “OK” (JSON format):

SOAP MSS_ReceiptResp

```
<soapenv:Envelope
  xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_ReceiptResponse>
         <mss:MSS_ReceiptResp MajorVersion="1" MinorVersion="1"
           xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
           xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
              <mss:MSSP_ID>
                <mss:URI>http://mid.swisscom.ch/</mss:URI>
              </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:Status>
              <mss:StatusCode Value="100"/>
              <mss:StatusMessage>REQUEST_OK</mss:StatusMessage>
              <mss:StatusDetail>
                <ns1:ReceiptResponseExtension ReceiptMessagingMode="synch"
                  UserAck="true"
                  UserResponse="{&amp;quot;status&amp;quot;:&amp;quot;OK&amp;quot;}"
                  xmlns:ns1="http://www.swisscom.ch/TS102204/ext/v1.0.0"/>
              </mss:StatusDetail>
            </mss:Status>
          </mss:MSS_ReceiptResp>
       </MSS_ReceiptResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_ReceiptResp

```
{
    "MSS_ReceiptResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusDetail": {
                "ReceiptResponseExtension": {
                    "ClientAck": "false",
                    "NetworkAck": "false",
                    "ReceiptMessagingMode": "synch",
                    "UserAck": "true",
                    "UserResponse": "{\"status\":\"OK\"}"
                }
            },
            "StatusMessage": "REQUEST_OK"
        }
    }
}
```

There are different types of synchronous MSS_ReceiptResp. The table below describes some possibilities:

Response scenario	UserAck	UserResponse	FaultMessage
No user response could be received within 80 seconds. It is unknown if the user got the receipt message.	false	N/A	N/A
User accepted the receipt message	true	"OK"	N/A
User cancelled the receipt message	true	"CANCEL"	"User Cancelled the request"
The receipt message was displayed for 60 seconds but the user did not respond	true	"TIMEOUT"	"Timeout waiting response"

If no user response could be received within 80 seconds, the MSS_ReceiptResp message will contain the attribute UserAck set to “false”:

SOAP MSS_ReceiptResp – No user response could be received

```
<soapenv:Envelope
 xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <MSS_ReceiptResponse>
      <mss:MSS_ReceiptResp MajorVersion="1" MinorVersion="1" xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
       xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
        <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
         Instant="2015-01-01T12:00:00.000+01:00"/>
          <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
            <mss:MSSP_ID>
              <mss:URI>http://mid.swisscom.ch/</mss:URI>
            </mss:MSSP_ID>
          </mss:MSSP_Info>
          <mss:Status>
            <mss:StatusCode Value="100"/>
            <mss:StatusMessage>REQUEST_OK</mss:StatusMessage>
            <mss:StatusDetail>
              <ns1:ReceiptResponseExtension ReceiptMessagingMode="synch" UserAck="false"
                xmlns:ns1="http://www.swisscom.ch/TS102204/ext/v1.0.0"/>
            </mss:StatusDetail>
          </mss:Status>
        </mss:MSS_ReceiptResp>
      </MSS_ReceiptResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_ReceiptResp – No user response could be received

```
{
    "MSS_ReceiptResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusDetail": {
                "ReceiptResponseExtension": {
                    "ClientAck": "false",
                    "NetworkAck": "false",
                    "ReceiptMessagingMode": "synch",
                    "UserAck": "false"
                }
            },
            "StatusMessage": "REQUEST_OK"
        }
    }
}
```

If the user cancelled the receipt message, the MSS_ReceiptResp message will contain the attribute UserAck set to “true” and the attribute UserResponse with the status value “CANCEL” (JSON format):

SOAP MSS_ReceiptResp – The user cancelled the request

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_ReceiptResponse>
         <mss:MSS_ReceiptResp MajorVersion="1" MinorVersion="1"
           xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
           xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
           <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
            Instant="2015-01-01T12:00:00.000+01:00"/>
           <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
             <mss:MSSP_ID>
               <mss:URI>http://mid.swisscom.ch/</mss:URI>
             </mss:MSSP_ID>
           </mss:MSSP_Info>
           <mss:Status>
             <mss:StatusCode Value="100"/>
             <mss:StatusMessage>REQUEST_OK</mss:StatusMessage>
             <mss:StatusDetail>
               <ns1:ReceiptResponseExtension ReceiptMessagingMode="synch"
                 UserAck="true"
                 UserResponse="{&amp;quot;status&amp;quot;:&amp;quot;CANCEL&amp;quot;}"
                 FaultMessage="User Cancelled the request"
                 xmlns:ns1="http://www.swisscom.ch/TS102204/ext/v1.0.0"/>
             </mss:StatusDetail>
           </mss:Status>
         </mss:MSS_ReceiptResp>
      </MSS_ReceiptResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_ReceiptResp – The user cancelled the request

```
{
    "MSS_ReceiptResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusDetail": {
                "ReceiptResponseExtension": {
                    "ClientAck": "false",
                    "FaultMessage": "User Cancelled the request",
                    "NetworkAck": "false",
                    "ReceiptMessagingMode": "synch",
                    "UserAck": "true",
                    "UserResponse": "{\"status\":\"CANCEL\"}"
                }
            },
            "StatusMessage": "REQUEST_OK"
        }
    }
}
```

If the user did not respond to the receipt message within 60 seconds, the MSS_ReceiptResp message will contain the attribute UserAck set to “true” and the attribute UserResponse with the status value “TIMEOUT” (JSON format):

SOAP MSS_ReceiptResp – A Timeout occurred

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_ReceiptResponse>
         <mss:MSS_ReceiptResp MajorVersion="1" MinorVersion="1"
           xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
           xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
           <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
            Instant="2015-01-01T12:00:00.000+01:00"/>
           <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00">
             <mss:MSSP_ID>
               <mss:URI>http://mid.swisscom.ch/</mss:URI>
             </mss:MSSP_ID>
           </mss:MSSP_Info>
           <mss:Status>
             <mss:StatusCode Value="100"/>
             <mss:StatusMessage>REQUEST_OK</mss:StatusMessage>
             <mss:StatusDetail>
               <ns1:ReceiptResponseExtension ReceiptMessagingMode="synch"
                 UserAck="true"
                 UserResponse="{&amp;quot;status&amp;quot;:&amp;quot;TIMEOUT&amp;quot;}"
                 FaultMessage="Timeout waiting response"
                 xmlns:ns1="http://www.swisscom.ch/TS102204/ext/v1.0.0"/>
             </mss:StatusDetail>
           </mss:Status>
         </mss:MSS_ReceiptResp>
      </MSS_ReceiptResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_ReceiptResp – A Timeout occurred

```
{
    "MSS_ReceiptResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusDetail": {
                "ReceiptResponseExtension": {
                    "ClientAck": "false",
                    "FaultMessage": "Timeout waiting response",
                    "NetworkAck": "false",
                    "ReceiptMessagingMode": "synch",
                    "UserAck": "true",
                    "UserResponse": "{\"status\":\"TIMEOUT\"}"
                }
            },
            "StatusMessage": "REQUEST_OK"
        }
    }
}
```

4.3.2 Asynchronous Mode (client-server)

The former asynchronous mode for MSS_Receipt is still supported by Mobile ID (for backward compatibility reason). However we strongly suggest you to implement the synchronous mode of MSS_Receipt.
Therefore this document will not describe the asynchronous mode of the MSS_Receipt.

4.3.3 Encrypted receipts

It is possible to encrypt the message sent over the MSS_Receipt request with the public key of the Mobile ID User. This will ensure that the message set by the AP will only be readable by the targeted End-User after typing in his Mobile ID PIN.

In order to properly encrypt the message, following actions needs to be done:

1. If the message contains special characters outside of the GSMDA character set it must be encoded as UCS-2 and prefixed with Hex 80 (0x80) [10]. This encoding is mandatory as the MID Service can not verify and convert properly the message content as it is encrypted, refer to Appendix 7.10 for encoding information.
2. Encrypt the message with the public certificate/key provided in the MSS_SignatureResp.
3. Set the MimeType to "application/alauda-rsamessage" Encoding="BASE64".

Example for a Message element with an encrypted receipt message:

```
<mss:Message MimeType="application/alauda-rsamessage" Encoding="BASE64">c..=</mss:Message>
```

4.3.4 Important notes

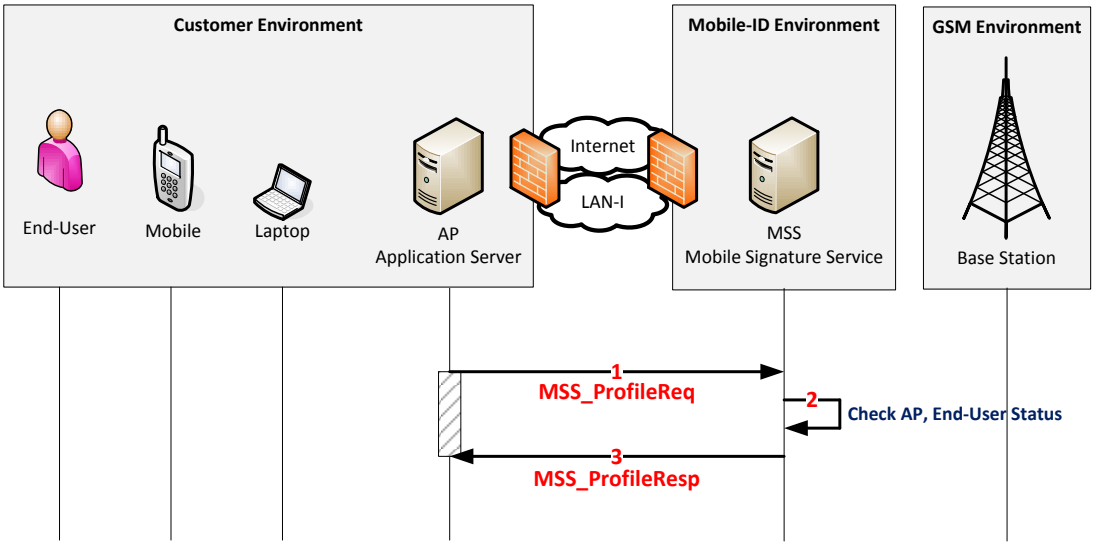
Please note the following points in regards of the MSS_Receipt:

- Only one MSS_Receipt Request can be sent for each signature transaction.
- There is the possibility to define the language of the receipt message. It is independent of the language used in the prior signature request.
- The time span between a successful signature request and issuing a corresponding MSS_Receipt is 60 Minutes. The receipt message string in the MSS_ReceiptReq is mandatory.

[10] To be also compatible with the old Mobile ID applet (V 1.200) the encoding must even be done in GSMDA.

4.4 The Mobile Profile Query

An AP may use this operation to **check the status of a Mobile ID user without end-user interaction**; the AP can use a Profile Query request for that as depicted below.



1. Before to send a signature request for authentication, the AP validates the status of a Mobile ID user by sending an MSS_ProfileReq request to Mobile ID
2. Mobile ID checks the user status and only in case of an active user, it retrieves the Profiles of the end-user.
3. Mobile ID sends back the user MSS_ProfileResp to the AP Service.

The MSS_ProfileResp details (i.e. list of Signature Profile URIs) shouldn't be of interest. You only want to know whether you will get a Fault Code Response (Mobile User will not be able to answer a signature request, i.e. due to an invalid account) or a MSS_ProfileResp (Mobile User should be able to answer a signature request)

Request and Response Messages: ProfileQuery

In red highlighted the most important input/output parameters required for a Profile Query request:

SOAP MSS_ProfileReq

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
 xmlns:ets="http://uri.etsi.org/TS102204/etsi204-kiuru.wsdl"
 xmlns:v1="http://uri.etsi.org/TS102204/v1.1.2#">
   <soap:Body>
      <ets:MSS_ProfileQuery>
         <MSS_ProfileReq MajorVersion="1" MinorVersion="1"
          xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
          xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
             <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
              Instant="2015-01-01T12:00:00.000+01:00"/>
            <mss:MSSP_Info>
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:MobileUser>
               <mss:MSISDN>MOBILE_NUMBER</mss:MSISDN>
            </mss:MobileUser>
         </MSS_ProfileReq>
      </ets:MSS_ProfileQuery>
   </soap:Body>
</soap:Envelope>
```

JSON MSS_ProfileReq

```
{
    "MSS_ProfileReq": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_PWD": "yourAP_PWD",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T12:00:00.000+01:00"
        },
        "MSSP_Info": {
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "MobileUser": {
            "MSISDN": "MOBILE_NUMBER"
        }
    }
}
```

The received MSS_ProfileQueryResponse message will contain the following parameters/values:

SOAP MSS_ProfileResp

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <MSS_ProfileQueryResponse xmlns="http://uri.etsi.org/TS102204/etsi204-kiuru.wsdl">
         <MSS_ProfileResp MajorVersion="1" MinorVersion="1" xmlns="">
            <mss:AP_Info AP_ID="yourAP_ID" AP_PWD="yourAP_PWD" AP_TransID="REF0101120000"
             Instant="2015-01-01T12:00:00.000+01:00" xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
             xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#"/>
            <mss:MSSP_Info Instant="2015-01-01T12:00:00.100+01:00"
             xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
             xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
               <mss:MSSP_ID>
                  <mss:URI>http://mid.swisscom.ch/</mss:URI>
               </mss:MSSP_ID>
            </mss:MSSP_Info>
            <mss:SignatureProfile xmlns:mss=http://uri.etsi.org/TS102204/v1.1.2#
             xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
               <mss:mssURI>http://mid.swisscom.ch/MID/v1/AuthProfile1</mss:mssURI>
            </mss:SignatureProfile>
            <mss:Status xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
             xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
               <mss:StatusCode Value="100"/>
               <mss:StatusMessage>REQUEST_OK</mss:StatusMessage>
            </mss:Status>
         </MSS_ProfileResp>
      </MSS_ProfileQueryResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

JSON MSS_ProfileResp

```
{
    "MSS_ProfileResp": {
        "AP_Info": {
            "AP_ID": "yourAP_ID",
            "AP_TransID": "REF0101120000",
            "Instant": "2015-01-01T11:00:00.000Z"
        },
        "MSSP_Info": {
            "Instant": "2015-01-01T11:00:00.100Z",
            "MSSP_ID": {
                "URI": "http://mid.swisscom.ch/"
            }
        },
        "MajorVersion": "1",
        "MinorVersion": "1",
        "SignatureProfile": [
            "http://mid.swisscom.ch/MID/v1/AuthProfile1"
        ],
        "Status": {
            "StatusCode": {
                "Value": "100"
            },
            "StatusMessage": "REQUEST_OK"
        }
    }
}
```

4.4.1 ProfileQuery Fault Response and Best Practices

The AP knows that a specific Mobile ID user is active if an MSS_ProfileResp has been received.
On the other side, the ProfileQuery supports a list of exceptional cases to the user account status. The AP will get a fault message with an appropriate subcode if the user is not in an active status.

Refer to chapter 6.2 for a list of possible fault subcodes.

4.4.2 Important notes

- Query Profile request doesn't support asynchronous mode.

5 Best Practices

5.1 Signature request

Those are the major aspects that needs to be considered when the signature request is constructed:

1) Define an unique AP_TransID.
2) Set the current time as value for "Instant" including the Timezone information.
   Example: 2015-01-01T12:00:00.000+01:00. ETSI 204 defines the Instant value to be xs:dateTime[11].
3) The triplet AP_ID, AP_TransID and Instant shall be unique, otherwise the request will be rejected.
4) MSSIDN shall be in international format without spaces. It can have a leading + char.
   Example: +41791234567
5) The UserLang shall be defined by the AP with one of the supported language (EN, DE, FR, IT)
6) The DataToBeSigned (DTBS) text, displayed on the mobile phone:
   - Shall be formatted in UTF-8[12]
   - Should contain a unique transaction ID (e.g. timestamp, customer ID, contract ID, etc.). See the example below (highlighted in **brown**).
   - Shall contain the AP-Identification in the message as requested and defined during the AP registration process. If the AP-Identification is wrong or missing, the request will be rejected with an mss:_107/INAPPROPRIATE_DATA fault response. See the example below (highlighted in **blue**).
   - Shall not exceed 239 characters. In case one of the characters is not present in the GSMDA[13] character set (for example the lowercase cedilla 'ç'), the length of the message is reduced to max. 119 characters.

     We suggest to keep the message as short as possible. See the example below.

     Example DTBS: "**Company: Login? (#234)**"

[11] http://www.w3.org/TR/xmlschema-2/#dateTime
[12] http://en.wikipedia.org/wiki/UTF-8
[13] In mobile telephony GSM 03.38 is the standard character set used in short message service .

5.2 Response handlings

It's under the responsibility of the AP to verify the response content of the MID service. Below are the most relevant points:

1) Verify that AP_TransID is the same between the request and the response
2) Verify that the MSISDN number is the same between the request and the response
3) Verify the certificate against: validity, revocation status and his trust anchor
4) Verify that the DataToBeSigned has effectively been signed
5) Manage the Status and Fault Codes with proper exception handling
6) Optionally send a receipt

5.3 User Mapping

The simplest way of mapping the AP Users to a Mobile ID User is over the MSISDN number. There are also other options by parsing the MID User certificate content:

1) The serialNumber contained in the Distinguished Name (DN)

   This attribute is assigned to a unique Mobile ID User and is kept as long as Swisscom is able to identify the same physical body behind this ID.

   Example of DN: /serialNumber=MIDCHE3QWAXYEAA2/CN=XXX/C=CH

   ⚠ Do not confuse the DN serialNumber with the unique Serial Number of the certificate itself.

2) The entire X.509 Certificate

   The entire certificate can be used to map a Mobile ID User. This will ensure that a specific card at a given time is bound to the user. In fact, when a Mobile ID User receives a new SIM card or has lost his Mobile ID PIN, a new unique certificate will be issued.

ⓘ Best practice will be to map your users against the serialNumber of the DN.

5.4 Multiple / Simultaneous signature request to the same End-User

The Mobile ID can only process one transaction at a time for a specific MSISDN end user. As long as the first request is in process, the second request will be rejected and a proper signature response error will be raised.

5.5 Instant and Timeouts

The instant, timeout and client connection timeout are different factors that need to be considered.

- The instant defines the precise time of the request
- The timeout defined as value in the request itself
- The client connection timeout defines the maximal connection time to the server itself

5.5.1 Instant handling

The Instant value defined in the request will be checked. If the AP is too far from MID Service current time then error 101 will be raised (AP Instant is too far from my current time).

5.5.2 Timeouts

Use the reference values below to set a proper timeout value:

	Transaction Timeout[14]	Client Connection Timeout[15]
MSS_SignatureReq (synchronous)	80 seconds	90 seconds
MSS_SignatureReq (asynchronous)	80 seconds	10 seconds
MSS_SignatureReq HeartBeat	5 seconds	10 seconds
MSS_StatusReq	n/a	10 seconds
MSS_ReceiptReq	n/a	90 seconds

⚠ According to the SIM Toolkit standard, any Toolkit message displayed on the mobile device will time out after 60 seconds of inactivity.

5.6 Trusted CA chains for certificate and signature validations

For all certificate related actions like mutual authentication against the MID server or validation of the signed messages in the responses a trusted CA chain must be defined.

As this chain may change over the time, the implemented solution should be flexible enough to modify this CA chain just over a generic change management process and should not require deployment of updated software

The MID signed messages as well as the SSL certificate are signed by a certificate issued under the "Swisscom Root CA 2" authority. Refer to chapter 7.6 for the related Authority information. The full certificate chain is always returned to the client, so that it will be sufficient to add the "root" certificate to the client TrustStore to validate the chain of trust. We don't recommend to store any intermediate CA certificate as they are subject to change.

[14] How long will the transaction at Mobile ID be valid. This is set via the "Timeout=" attribute in the request message.
[15] How long your AP client (socket) shall wait for a response message from Mobile ID.

6 Status and Fault codes

In general the AP should have a generic status and fault code handler to cover successful states, warnings and the system/interface errors.

From an End-User perspective the Status and Fault codes can be set into 3 different groups:

1) Green: Success
2) Yellow: End-User is in a situation where additional actions in regards to his Mobile ID must be taken (PIN locked, not yet activated...)
3) Red: AP interface to Mobile ID has troubles (wrong calls, temporary outages...)

6.1 Green: MSSP processing completed with success

This status indicates that all is fine and the signature has been processed by the MSSP according to the request.

Status code 500

Status code 500 means that a mobile signature has been successfully constructed.

Status code 502

Status code 502 means that a signature was received from the end-user and in addition the MSSP has checked and determined if the signature was valid. Refer to 4.2.3.2 for more details.

6.2 Yellow: MSSP answers where explicit error handling should be done

Here a list of Status and fault subcodes where explicit error handling with corresponding end user solution hint should apply. In the solution hint displayed to the end-user the Faultcode user assistance URL as described in 6.5 should be set to allow a direct assistance.

Status code 501

Status code 501 means that the signature request was properly constructed, but end-user certificate is revoked.

Solution hint: “A new activation is required. Please visit the Mobile ID Portal of your Mobile Network Operator”

Status code 503

Status code 503 means that the signature is invalid.

Solution hint: “A new activation is required. Please visit the Mobile ID Portal of your Mobile Network Operator. If the error persists, contact your Mobile Network Operator Support.”

Fault Subcode mss:_105

The fault subcode "mss:_105" indicates that for this subscriber number the SIM card does not support Mobile ID.

Solution hint: "A Mobile ID SIM card has to be ordered. Please visit the Mobile ID Portal of any of the participating Mobile Network Operators."

Fault Subcode mss:_208

The fault subcode "mss:208" indicates that the transaction has timed out.

Solution hint: "Please try again. If the error persists, contact your Mobile Network Operator Support."

Fault Subcode mss:_209

The fault subcode "mss:209" indicates that there was an internal problem with the OTA subsystem.

Solution hint: "Please try again. If the error persists, contact your Mobile Network Operator Support."

Fault Subcode mss:_401

The fault subcode "mss:401" indicates that the request has been cancelled by the user.

Solution hint: "Please try again. If the error persists, contact your Mobile Network Operator Support."

Fault Subcode mss:_402

The fault subcode "mss:402" indicates that the user's Mobile ID PIN is blocked.

Solution hint: "A new activation is required. Please visit the Mobile ID Portal of your Mobile Network Operator and follow the PIN forgotten link (#Faultcode user assistance URL#)"

Fault Subcode mss:_403

The fault subcode "mss:403" indicates that the Mobile ID user has been suspended.

Solution hint: "A new activation is required. Please visit the Mobile ID Portal of your Mobile Network Operator (#Faultcode user assistance URL#)"

Fault Subcode mss:_404

The fault subcode "mss:404" indicates that the subscriber number is not yet activated for Mobile ID.

Solution hint: "A new activation is required. Please visit the Mobile ID Portal of your Mobile Network Operator (#Faultcode user assistance URL#)"

Fault Subcode mss:_406

The fault subcode "mss:406" indicates that there was already an on-going transaction of the same subscriber.

Solution hint: "Please try again. If the error persists, contact your Mobile Network Operator Support."

Fault Subcode mss:_422

The fault subcode "mss:422" indicates that the certificate of the subscriber number is not valid.

Solution hint: "A new activation is required. Please visit the Mobile ID Portal of your Mobile Network Operator (#Faultcode user assistance URL#)"

6.3 Red: MSSP answers covered by generic system error handling

A generic error handler displaying adequate End-User information should handle all the other Status and Fault codes.

Solution hint: "Mobile ID cannot be used at this time. Please try again later. If the error persists, contact your Mobile Network Operator Support."

6.4 Reserved MSISDN for fault code testing

Specific MSISDN have been defined in order to raise a corresponding fault code. By placing a request to one of this number the related fault code will be raised.

The following structure is used to define the test MSISDN number: ***+41000092<3_digits_fault_code>***
A specific MSISDN has been defined for each 'Fault sub-code value' defined in 7.8.

Therefore a request placed to +41000092**401** will raise fault code **mss:_401**.

Reserved MSISDNs are available on Signature- and Profile Query Requests according the table of chapter 7.8

Here's a sample list of responses of requests placed to the specific MSISDN's:

Fault codes with specific MSISDN's

FAILED on +41000092**101** with error 101 (WRONG_PARAM: Error among the arguments of the request)
FAILED on +41000092**102** with error 102 (MISSING_PARAM: An argument in the request is missing)
FAILED on +41000092**103** with error 103 (WRONG_DATA_LENGTH: The DataToBeSigned are too large. Limitations are due to the Mobile Signature technology implemented by the MSSP)
FAILED on +41000092**104** with error 104 (UNAUTHORIZED_ACCESS: The AP is unknown or the password is wrong or the AP asks for an additional service for which it has not subscribed)
FAILED on +41000092**105** with error 105 (UNKNOWN_CLIENT: MSISDN is unknown)
FAILED on +41000092**107** with error 107 (INAPPROPRIATE_DATA: DTBD matching failed)
FAILED on +41000092**108** with error 108 (INCOMPATIBLE_INTERFACE: the minor version and/or major version parameters are inappropriate for the receiver of the message)
FAILED on +41000092**109** with error 109 (UNSUPPORTED_PROFILE: The AP has specified a Mobile Signature Profile that the MSSP does not support)
FAILED on +41000092**208** with error 208 (EXPIRED_TRANSACTION: Transaction Expiry date has been reached or Time out has elapsed)
FAILED on +41000092209 with error 209 (OTA_ERROR: The MSSP has not succeeded to contact the enduser's mobile equipment (Bad connection…))
FAILED on +41000092**401** with error 401 (USER_CANCEL: The client has cancelled the transaction)
FAILED on +41000092**402** with error 402 (PIN_NR_BLOCKED: PIN of the mobile user is blocked)
FAILED on +41000092**403** with error 403 (CARD_BLOCKED: Mobile user account has state INACTIVE)
FAILED on +41000092**404** with error 404 (NO_KEY_FOUND: Mobile user account needs to be activated)
FAILED on +41000092**406** with error 406 (PB_SIGNATURE_PROCESS: Error during the Mobile Signature process on the Mobile equipment)
FAILED on +41000092**422** with error 422 (NO_CERT_FOUND: Certificate is expired, revoked or with an invalid certification path)
FAILED on +41000092**900** with error 900 (INTERNAL_ERROR: Unknown Error)

6.5 Fault Code User Assistance

For a subset of the yellow fault subcodes defined in chapter 6.2, the mobile user is likely able to resolve the problem himself by using the Mobile ID Portal.

Fault Response Messages related to the error code in the list below will be enhanced with an additional UserAssistance object that contains the fault specific Mobile ID Portal URL. This URL will be the Portal of the Mobile Network Operator (MNO) to which the mobile user is currently subscribed to.

- mss:_402 (PIN of the mobile user is blocked)
- mss:_403 (Mobile user account has state INACTIVE)
- mss:_404 (Mobile user account needs to be activated)
- mss:_422 (Certificate is expired, revoked or with an invalid certification path)

In case the error code is mss:_105 (MSISDN is unknown), Mobile ID does not know the MSISDN. The user would need to order Mobile ID using the Portal URL of either Swisscom, Sunrise or Orange.

⚠ The AP is responsible to display an appropriate message towards the mobile user, with the provided portal URL. This in order to allow the mobile user to navigate to the MNO specific Mobile ID Portal where proper user assistance will be provided for a given problem.

An AP's client should parse and decode the Portal URL value in order to..

1) get the correct string representation of any ampersand, if present (`&` to `&`)
2) enhance the Portal URL with the AP's actual portal language (one of en, de, fr, it), i.e. `&lang=en`
   For example, if the Fault Response contains the PortalUrl as follows..

```
<sco:PortalUrl>
  https://www1.swisscom.ch/registration/online/app/MobileId?resetPin=true&amp;msisdn=41795135823
</sco:PortalUrl>
```

The URL to be used will look like as follows..

```
https://www1.swisscom.ch/registration/online/app/MobileId?resetPin=true&msisdn=41795135823&lang=en
```

Here's an example how a Fault Response with UserAssistance object will look like:

SOAP Fault Subcode with UserAssistance

```
..
    <soapenv:Fault>
       <soapenv:Code>
          <soapenv:Value>soapenv:Sender</soapenv:Value>
          <soapenv:Subcode xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
           xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
             <soapenv:Value>mss:_402</soapenv:Value>
          </soapenv:Subcode>
       </soapenv:Code>
       <soapenv:Reason>
          <soapenv:Text xml:lang="en">PIN_NR_BLOCKED</soapenv:Text>
       </soapenv:Reason>
       <soapenv:Detail>
          <ns1:detail xmlns:ns1="http://kiuru.methics.fi/mssp">PIN of the mobile user is blocked</ns1:detail>
          <sco:UserAssistance xmlns:sco="http://www.swisscom.ch/TS102204/ext/v1.0.0">
             <sco:PortalUrl>
                https://www1.swisscom.ch/registration/online/app/MobileId?resetPin=true&amp;msisdn=41795135823
             </sco:PortalUrl>
          </sco:UserAssistance>
       </soapenv:Detail>
    </soapenv:Fault>
..
```

JSON Fault Subcode with UserAssistance

```
{
  "Fault": {
    "Code": {
      "SubCode": {
        "Value": "_402",
        "ValueNs": "http://uri.etsi.org/TS102204/v1.1.2#"
      },
      "Value": "Sender",
      "ValueNs": "http://www.w3.org/2003/05/soap-envelope"
    },
    "Detail": "PIN of the mobile user is blocked",
    "Reason": "PIN_NR_BLOCKED",
    "UserAssistance": [
      {
        "PortalUrl":
          "https://www1.swisscom.ch/registration/online/app/MobileId?resetPin=true&amp;msisdn=41795135823"
      }
    ]
  }
}
```

7 Appendix

7.1 Create self-signed certificate with openssl

Below are some examples on how to create a self-signed certificate with OpenSSL, valid for 5 years.

7.1.1 Generate Key and CSR

```
$ openssl req -new -newkey rsa:2048 -nodes -rand /dev/urandom -keyout mycert.key
  -out mycert.csr -sha256 -subj '/CN=mobileid.company.ch/O=Company/C=CH'
```

7.1.2 Self-sign it and create your certificate

```
$ openssl x509 -req -days 1825 -sha256 -in mycert.csr -signkey mycert.key -out mycert.crt
```

7.1.3 Convert into PKCS#12 (if needed)

If you need a PKCS#12 file you can convert the Key and Certificate with this command:

```
$ openssl pkcs12 -export -in mycert.crt -inkey mycert.key -out mycert.p12
```

Provide the **`mycert.crt`** file to Swisscom and keep your *.key file securely stored.

7.2 Create self-signed certificate with Java keytool

Below are some examples on how to create a self-signed certificate with the Java Keytool, valid for 5 years.

7.2.1 Generate KeyStore & export the self-signed certificate

```
$ keytool -genkey -alias <alias-name> -keyalg RSA -keysize 2048 -validity 1825
  -dname 'CN=mobileid.company.ch,O=Company,C=CH' -keystore mycert.jks

$ keytool -export -alias <alias-name> -keystore mycert.jks -file mycert.crt
```

7.2.2 Root CA and Intermediate CA certificate import

```
$ keytool -keystore truststore.jks -import -file Swisscom_Root_CA_2_der.crt
$ keytool -keystore truststore.jks -import -file Swisscom_Rubin_CA_2_der.crt
```

7.2.3 Verification:

```
$ keytool -printcert -v -file mycert.crt
$ keytool -list -v -keystore keystore.jks
$ keytool -list -v -keystore keystore.jks -alias <alias-name>
```

Provide the **`mycert.crt`** file to Swisscom and keep your *.jks file securely stored.

7.3 Swisscom signed certificate

Any client certificate issued by an official CA like Swisscom SDCS http://www.swissdigicert.ch may be used as an alternative to the self-signed certificate.

The Swisscom CA Certificates can be downloaded from the Swisscom SDCS website:
http://www.swissdigicert.ch/sdcs/portal/page?node=download_ca

7.4 Sample Mobile ID Scripts

Refer to https://github.com/SCS-CBU-CED-IAM/mobileid for sample scripts to invoke the MSS_Signature, MSS_Receipt and MSS_ProfileQuery request.

7.5 Sample Mobile ID Solutions

Refer to http://swisscom.com/mid and https://github.com/SCS-CBU-CED-IAM for additional Mobile ID enabled solutions.

7.6 Swisscom Certificates Files

The CA Certificates can be found and downloaded from the Swisscom SDCS CA site:

http://www.swissdigicert.ch/sdcs/portal/page?node=download_ca

An example of a CA Certificate file for the MID service can be downloaded at http://git.io/5KMNGA

7.7 Heartbeat

If for some specific reason your service needs to get the status about the signature service (Heartbeat), the recommended way will be to poll the status over a specific Synchronous MSS_Signature request. The request must go to the special MSISDN +41000000000 and contain "Heartbeat"as DataToBeSigned.

The recommended polling interval for successful Heartbeat (HB) message is 300 seconds. In case HB wasn't successful reduce interval to 180 seconds and repeat it 3 times before declaring signature service unavailability. If signature service gets unavailability keep on polling with 180 seconds until successful HB message received again.

Example for mss:MobileUser and mss:DataToBeSigned in a MSS_SignatureReq:

```
        <mss:MobileUser>
            <mss:MSISDN>+41000000000</mss:MSISDN>
        </mss:MobileUser>
        <mss:DataToBeSigned MimeType="text/plain" Encoding="UTF-8">Heartbeat</mss:DataToBeSigned>
```

```
        "MobileUser": {
            "MSISDN": "+41000000000"
        },
        "DataToBeSigned": {
            "Data": "Heartbeat",
            "Encoding": "UTF-8",
            "MimeType": "text/plain"
        },
```

The signature service is working fine if valid fault codes will be returned. With the special MSISDN the fault code detail will be "Illegal msisdn":

SOAP Heartbeat MSS_SignatureResp

```
..
      <soapenv:Fault>
         <soapenv:Code>
            <soapenv:Value>soapenv:Sender</soapenv:Value>
            <soapenv:Subcode xmlns:mss="http://uri.etsi.org/TS102204/v1.1.2#"
                             xmlns:fi="http://mss.ficom.fi/TS102204/v1.0.0#">
               <soapenv:Value>mss:_101</soapenv:Value>
            </soapenv:Subcode>
         </soapenv:Code>
         <soapenv:Reason>
            <soapenv:Text xml:lang="en">WRONG_PARAM</soapenv:Text>
         </soapenv:Reason>
         <soapenv:Detail>
            <ns1:detail xmlns:ns1="http://kiuru.methics.fi/mssp">Illegal msisdn</ns1:detail>
         </soapenv:Detail>
      </soapenv:Fault>
..
```

JSON Heartbeat MSS_SignatureResp

```
{
    "Fault": {
        "Code": {
            "SubCode": {
                "Value": "_101",
                "ValueNs": "http://uri.etsi.org/TS102204/v1.1.2#"
            },
            "Value": "Sender",
            "ValueNs": "http://www.w3.org/2003/05/soap-envelope"
        },
        "Detail": "Illegal msisdn",
        "Reason": "WRONG_PARAM"
    }
}
```

7.8 ETSI TS 102 204 Status Codes

The ETSI TS 102 204 and 102 207 MSS Status Codes and Fault sub-codes that might appear. This list may not be complete and additional MSS Status Codes and/or Fault sub-code value may be returned by the platform.

				Operation			
MSS Status Code	**Fault sub-code value**	**StatusMessage**	**Possible causes**	**MSS_Signature**	**MSS_StatusQuery**	**MSS_Receipt**	**MSS_ProfileQuery**
100		REQUEST_OK	Request accepted by the receiver (MSSP or AP)	X		X	X
	mss:_101	WRONG_PARAM	There are several causes for WRONG_PARAM, see separate table in 7.9	X	X	X	X
	mss:_102	MISSING_PARAM	AP created bad request data	X	X	X	X
	mss:_103	WRONG_DATA_LENGTH	AP created bad request data targeted to mobile; Signature Request's DataToBeSigned (DTBS) is too long or short	X		X	
	mss:_104	UNAUTHORIZED_ACCESS	1) System does not know the calling AP_ID 2) AP_ID is known, but client's SSL client certificate does not match 3) AP_PWD field value is not set	X	X	X	X
	mss:_105	UNKNOWN_CLIENT	1) Target MSISDN is unknown, Mobile ID option needs to be ordered first.	X		X	X
	mss:_107	INAPPROPRIATE_DATA	AP created bad request data. 1) Signature Request with unsupported data format 3) DataToBeSigned is missing the expected AP-Identification	X		X	
	mss:_108	INCOMPATIBLE_INTERFACE	AP created bad request XML data with wrong MajorVersion, MinorVersion attribute values	X	X		X
	mss:_109	UNSUPPORTED_PROFILE	The AP has specified a Mobile Signature Profile that the MSSP does not support	X			
	mss:_208	EXPIRED_TRANSACTION	Transaction Expiry date has been reached or Time out has elapsed.	X	X	X	
	mss:_209	OTA_ERROR	A Problem related to the Over-The-Air (OTA) gateway that is responsible for the SMS communication with the mobile phone	X	X		
	mss:_401	USER_CANCEL	User cancelled the request at the mobile phone	X	X		
	mss:_402	PIN_NR_BLOCKED	The Mobile ID PIN is blocked	X	X	X	X
	mss:_403	CARD_BLOCKED	The Mobile User is currently suspended	X	X	X	X
	mss:_404	NO_KEY_FOUND	Target MSISDN is not yet activated	X	X	X	X
	mss:_406	PB_SIGNATURE_PROCESS	A signature transaction is already in progress	X	X		
	mss:_422	NO_CERT_FOUND	No valid certificate for the Mobile User found	X	X	X	X
500		SIGNATURE	Successful signature result	X	X		
501		REVOKED_CERTIFICATE	If the AP did request the (obsolete) validation service (see 4.2.3.2), this is a negative validation result because the certificate is revoked	X	X		
502		VALID_SIGNATURE	If the AP did request the (obsolete) validation service (see 4.2.3.2), this is a successful validation result	X	X		
503		INVALID_SIGNATURE	If the AP did request the (obsolete) validation service (see 4.2.3.2), this is a negative validation result because the signature is invalid	X	X		
504		OUSTANDING_TRANSACTION	Intermediate state during an asynchronous signature request. Please call again on the status		X		

				Operation			
MSS Status Code	**Fault sub-code value**	**StatusMessage**	**Possible causes**	***MSS_Signature***	***MSS_StatusQuery***	***MSS_Receipt***	***MSS_ProfileQuery***
			query service (keep polling the result)				
	mss:_900	INTERNAL_ERROR	1) database access problems 2) configuration problems 3) internal backend communication problems	X	X	X	X

7.9 Possible reasons for "WRONG_PARAM" reply

One of the following descriptions may be the root cause of the WRONG_PARAM reply received.

- A CounterSignature request with bad reference signature data
- A CounterSignature request with AP's signature that cannot be validated with known certificate(s) for the AP.
- An MSS_SignatureReq with DataToBeSigned having unsupported Encoding= attribute value.
- An MSS_SignatureReq processing with bad configuration
- An MSS_SignatureReq processing with badly formatted ASN.1 DER data on inputs.
- Receipt refers to unknown transaction reference
- Receipt has already been sent for referred transaction
- Receipt sending on transaction that didn't terminate with OK Signature Status
- Invalid messaging mode in MSS_SignatureReq
- Internally inconsistent MSS_MessageSignature
- XML MSS_MessageSignature verification failed.
- Did not receive XML MSS_MessageSignature from roaming partner(s).
- Duplicate message. You must use unique combination of <AP_TransID + Instant> values.
- There is no such transaction.
- Invalid timeout/validity parameter on request.
- Key already exists, and no overwrite has been requested.
- Did not find referred MSS_SignatureReq.
- Missing AP_URL
- The request tries to access a non-existing service port.
- AP Instant is too far from my current time.

7.10 UCS2/GSMDA Encoding

In mobile telephony GSM 03.38 or 3GPP 23.038 is a character set used in the Short Message Service of GSM based cell phones. It is defined in GSM recommendation 03.38 (http://pda.etsi.org). In case of Mobile ID the messages can be encoded in either GSMDA (GSM Default Alphabet) or in UCS-2.

Example of proper HEX encodings for MID for "François Müller"

- GSMDA: `4672616E096F6973204D7E6C6C6572`
- UCS2 prefixed with 80: `800046007200610060E00E7006F006900730020004D00FC006C006C00650072`

For all practical purposes you could use UTF-16BE encoding in case there is no specific UCS-2 encoder available (e.g. Java: `message.getBytes("UTF-16BE")`). Extended multinational codeplane characters whose unicode codepoint is above 0xFFFF will not work in this case.

7.11 Guidelines for test managers

This addendum describes how to test the Mobile ID enabled services from an end user perspective.

Most of the situations can be triggered with a live productive Mobile ID user:

- Answering the request displayed on the device
- Not answering or cancelling the request displayed on the device
- Locking the screen of the device before any request will be addressed to the device
- Turning off the device
- Entering wrong Mobile ID PIN or using a PIN locked Mobile ID
- Using a SIM that is not yet Mobile ID ready

Some specific situations can only be reached with the support of:

- the owner of the mobile subscription (typically a "fleet manager" in a corporate environment)
- the software developer or the integrator responsible for the Mobile ID enabled system
- the MNO issuing the Mobile ID (i.e. Swisscom)

The following table gives test managers an overview of how to trigger different Mobile ID situations for a given mobile subscription and which third party must be allocated to execute a specific test case:

				Required support			
MSS Status Code	**Fault sub-code value**	**Test requirement**	**Test procedure**	**Mobile ID User**	**Subscription owner**[16]	**Software Developer**	**Mobile Operator**
500 502 504		Functional Mobile ID	Request an authentication/signature for the given MSISDN and confirm it with the right PIN. Those status codes are all "successful" status codes. The value depends on the implementation chosen by the Software Developer (with or without (obsolete) validation, asynchronous polling)	X		X	
100		Functional Mobile ID	After a successful authentication/signature request for a given MSISDN there is the option to provide a Receipt to the given MSISDN. If the service implements such option, the delivery of the receipt will produce this status code.	X		X	
501		Functional Mobile ID where the related CA has revoked the issued certificate	With the help of the Mobile Operator revoke the corresponding Mobile ID certificate and request an authentication/signature for the given MSISDN.	X			X
503		Functional Mobile ID but problem on the signature itself	n/a This error code cannot be triggered for a given MSISDN				
	mss:_101	Access to system configuration or source code	Change the configuration of the Mobile ID enabled system with the help of your software provider to a non-valid entry according to 7.9.			X	
	mss:_102	Access to system configuration or source code	With help of a software provider remove a required parameter in the request of the Mobile ID enabled			X	

[16] Fleet Manager for corporate subscriptions, Call center for residential subscription

				Required support			
MSS Status Code	**Fault sub-code value**	**Test requirement**	**Test procedure**	**Mobile ID User**	**Subscription owner**[16]	**Software Developer**	**Mobile Operator**
			system.				
	mss:_103	Access to system configuration or source code	Configure a DTBS exceeding the maximal size of the Mobile ID system.			X	
	mss:_104	Access to system configuration or source code	Configure a wrong non-existing AP_ID for your Mobile ID enabled system.			X	
	mss:_105	MSISDN with no Mobile ID	Request a valid authentication/signature for the given MSISDN.	X			
	mss:_107	Access to system configuration or source code	Remove the defined AP-Identification from the text sent to Swisscom to display on the handset (DTBS).			X	
	mss:_108	Access to system configuration or source code	With help of a software provider create a bad request XML data with wrong MajorVersion, MinorVersion attribute values.			X	
	mss:_109	Access to system configuration or source code	With help of a software provider create a request with a non-existent profile name.			X	
	mss:_208	Functional Mobile ID	Request an authentication/signature for the given MSISDN and don't accept or cancel the request on the mobile device.	X			
	mss:_209	Functional Mobile ID	Switch off the mobile phone and request an authentication/signature for the given MSISDN.	X			
	mss:_401	Functional Mobile ID	Request an authentication/signature for the given MSISDN press cancel on the mobile device.	X			
	mss:_402	Functional Mobile ID or PIN Locked Mobile ID	Request an authentication/signature for the given MSISDN and enter the PIN as many times wrong to get it locked or use a PIN locked Mobile ID.	X			
	mss:_403	Functional Mobile ID	This error code can only be triggered with project support of the mobile operator.	X			X
	mss:_404	Account not activated	Reset the Mobile ID PIN, then send a signature or a ProfileQuery request	X	X		
	mss:_406	Functional Mobile ID	Request an additional authentication/signature for a given MSISDN where a request is already on the mobile screen.	X			
	mss:_422	Functional Mobile ID	This error code can only be triggered with project support of the mobile operator.	X			X