



Dynamic Roaming Management

Trigger Specification

© 2002 Sonera SmartTrust Ltd. All rights reserved.

Sonera SmartTrust endeavors to ensure that the information in this document is correct and fairly stated, but does not accept liability for any error or omission. The development of Sonera SmartTrust products and services is continuous and published information may not be up to date. It is important to check the current position with Sonera SmartTrust. This document is not part of a contract or license save insofar as may be expressly agreed.

Unless otherwise noted, all names of companies, products, street addresses and persons contained herein are part of a completely fictitious scenario and are designed solely to document the use of Service Platform.

SmartTrust is a trademark of Sonera SmartTrust Ltd. All the other trademarks are the property of their respective owners.

Contents

Introduction	5
About Dynamic Roaming Management	5
Target audience	5
Protocols	5
References	6
Roaming Management Services	7
Parameters common to all services.....	7
Service: simfiledownload.....	8
Example request for simfiledownload	8
Service: refresh.....	10
Example requests for refresh.....	10
Service: registeredplmn.....	12
Example requests for registeredplmn.....	13
Service responses	14
Appendix A	15
DTD for XML-RPC documents	15

Introduction

About Dynamic Roaming Management

Roaming Server enables roaming management for individual SIM cards. When a mobile station switches networks, a customer application sends an XML-RPC trigger to the Roaming Server.

The trigger document defines the actual reaction. Basically there are three types of actions:

- Updating a selected elementary file on SIM card
- Sending a SIM Application Toolkit command REFRESH to SIM card
- More complex operations; a pre-defined logical sequence based on currently visited network. The actual logical sequence is defined on the basis of customer needs.

All these actions, or services, are build on top of Service Platform, which is a framework for utilizing Delivery Platform functionality, including access management, transaction handling etc. By using this framework it is easy to extend the functionality later, if needed. For example, several different logical sequences can be implemented with little additional work afterwards.

Target audience

This document is meant for people who need to know how to generate triggers for SmartTrust dynamic roaming management application.

Protocols

Trigger generating device communicates with Roaming Server using the XML-RPC protocol. XML-RPC is a remote procedure calling protocol which uses HTTP as the transport protocol and XML as the encoding.

HTTP is defined in RFC 2616: <http://www.rfc-editor.org/rfc/rfc2616.txt>

XML-RPC encoding is defined at <http://www.xml-rpc.org/>

References

Ref	Document
[1]	XML-RPC Specification, http://www.xml-rpc.org/spec
[2]	RFC 2616 Hypertext Transfer Protocol – HTTP/1.1, http://www.rfc-editor.org/rfc/rfc2616.txt
[3]	WML specification – Wireless Internet Gateway/Delivery Platform 5.2, http://www.smarttrust.com/developerscorner/pdf/wmlspecificationwig.pdf

Roaming Management Services

Roaming management services consist of following services

- simfiledownload
- refresh
- registeredplmn

Service Platform recognizes the requested service from incoming request's methodname-parameter (see example requests later in this chapter). The actual service names can be changed, but the changes must be imported to Service Platform configuration.

In addition, more services can be implemented easily, if need for such emerges. For example, if several customer-specific logical

Parameters common to all services

These characteristics are common to all roaming management services. The target SIM has to be identified by either msisdn, imsi or iccid. Thus one of these three parameters has to be present in every request. Whatever identifier is used, it should be given in same format as it is stored in Delivery Platform's database.

Parameter	Type	Value
msisdn	String	End user's MSISDN
imsi	String	End user's IMSI
iccid	String	End user's ICCID

Due to Service Platform transaction management system, the following parameters should also be included in the request.

Parameter	Type	Value
callback-url	string	The URL to which the response is to be sent.
callback-function	string	The function which is to be called in the callback.

All requests are responded with an error code or success status to URL defined in callback-url.

Service: simfiledownload

This service can be used to download a defined content to an elementary file of SIM card; thus the trigger sender must know and include the contents of the file to be updated.

One request can be used to update one file. However, it is possible for the Service Platform to queue requests inside configured period of time so, that the amount of resulting SMS's per SIM card is minimized.

In addition to common parameters, the following parameters should be included:

Parameter	Type	Value
file	string	The elementary file to be updated. For example: -loci -plmnsel -fplmn
position	string	The offset from the beginning of the file. Default value: 0.
truncate	string	Either true or false. This parameter tells if the updated file should be truncated at the end of update or not. Default value: false
encoding	string	The content encoding. Possible values include -dec –content is decimal data -hex –content is hexadecimal -bin: <encoding> the content string is an encoded binary data. The encoding should be given after semicolon. For example: “bin; iso-8859-1”
content	string	The content to be downloaded.

Example request for simfiledownload

The following request would result as SIM card with MSISDN 1234567890 having the PLMNsel-file contents 244091FFFFF...

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
  <!-- Name of the service -->
```



```

<methodName>simfiledownload</methodName>
<params>
  <param><value>
    <struct>

      <!-- [Customer msisdn] -->
      <member>
        <name>msisdn</name>
        <value><string>1234567890</string></value>
      </member>

      <!-- [file to be updated] -->
      <member>
        <name>file</name>
        <value><string>plmnsel</string></value>
      </member>

      <!-- [should we truncate the file?] -->
      <member>
        <name>truncate</name>
        <value><string>>true</string></value>
      </member>

      <!-- [the used encoding] -->
      <member>
        <name>encoding</name>
        <value><string>hex</string></value>
      </member>

      <!-- [actual content, the example value corresponds to MCC 244/MNC
91, i.e. Sonera Finland] -->
      <member>
        <name>content</name>
        <value><string>F45B</string></value>
      </member>

      <!-- Callback parameters. -->

      <!-- URL to the XML-RPC server. -->
      <member>
        <name>callback-url</name>
        <value><string>http://trigger.sender.com/xmlrpc</string></value>
      </member>

      <!-- Name of the XML-RPC callback function. -->
      <member>
        <name>callback-function</name>
        <value><string>callback</string></value>
      </member>

    </struct>
  </value></param>
</params>
</methodCall>

```

Service: refresh

This service can be used to execute a defined SIM application toolkit command in a specified SIM card.

In addition to common parameters, the following parameters should be included:

Parameter	Type	Value
level	string	0 - SIM Initialisation and Full File Change Notification 1 - File Change Notification 2 - SIM Initialisation and File Change Notification 3 - SIM Initialisation 4 - SIM Reset
filecount	string	The number of changed files. Only relevant for REFRESH levels 1 and 2.
filelist	string	The list of changed files. The filelist format is specified in [3], Chapter 9.4 Only relevant for REFRESH levels 1 and 2.
fileloci	string	If "true", EF LOCI is included in the list of updated files. Only relevant for REFRESH levels 1 and 2.
fileplmnsel	string	If "true", EF PLMNsel is included in the list of updated files. Only relevant for REFRESH levels 1 and 2.
filefplmn	string	see above.
file...	string	see above.

The parameters starting from fileloci are convenience parameters, meaning that all file information for "File Change Notification" and "SIM Initialization and File Change Notification" can be provided in filelist-parameter. However, it may be desirable to define an own parameter for any file, that is frequently the target of update.

Example requests for refresh

The following example would result as SAT command REFRESH with "SIM Initialization and File Change Notification" –level for PLMNsel –file residing in SIM MSISDN 1234567890.

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
  <!-- Name of the service -->
  <methodName>refresh</methodName>
  <params>
    <param><value>
      <struct>

        <!-- [Customer msisdn] -->
        <member>
          <name>msisdn</name>
          <value><string>1234567890</string></value>
        </member>

        <!-- [Refresh level] -->
        <member>
          <name>level</name>
          <value><string>2</string></value>
        </member>

        <!-- [File count for SIM file change] -->
        <member>
          <name>filecount</name>
          <value><string>1</string></value>
        </member>

        <!-- [What files are updated?] -->
        <member>
          <name>fileplmnsel</name>
          <value><string>true</string></value>
        </member>

        <!-- Callback parameters. -->

        <!-- URL to the XML-RPC server. -->
        <member>
          <name>callback-url</name>
          <value><string>http://trigger.sender.com/xmlrpc</string></value>
        </member>

        <!-- Name of the XML-RPC callback function. -->
        <member>
          <name>callback-function</name>
          <value><string>callback</string></value>
        </member>

      </struct>
    </value></param>
  </params>
</methodCall>

```

The following example would result as SAT command REFRESH with "SIM Initialization and Full File Change" –level for SIM MSISDN 1234567890.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
  <!-- Name of the service -->
  <methodName>refresh</methodName>
  <params>
    <param><value>
      <struct>

        <!-- [Customer msisdn] -->
        <member>
          <name>msisdn</name>
          <value><string>1234567890</string></value>
        </member>

        <!-- [Refresh level] -->
        <member>
          <name>level</name>
          <value><string>0</string></value>
        </member>

        <!-- Callback parameters. -->

        <!-- URL to the XML-RPC server. -->
        <member>
          <name>callback-url</name>
          <value><string>http://trigger.sender.com/xmlrpc</string></value>
        </member>

        <!-- Name of the XML-RPC callback function. -->
        <member>
          <name>callback-function</name>
          <value><string>callback</string></value>
        </member>

      </struct>
    </value></param>
  </params>
</methodCall>
```

Service: registeredplmn

This service can be used to execute a specified flow of actions based on the currently registered PLMN. The actual flow of actions is basically beyond the scope of the trigger specification. It can be for example downloading a pre-defined content for PLMNsel-file based on the currently registered network. The logical flow definition results as implementation work inside Service Platform.

In addition to common parameters, the following parameters should be included:

Parameter	Type	Value
mcc	string	Mobile Country Code of the visited PLMN.
mnc	string	Mobile Network Code of the visited PLMN.

Example requests for registeredplmn

The following example would result as a defined logical operations for MSISDN 1234567890.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
  <!-- Name of the service -->
  <methodName>registeredplmn</methodName>
  <params>
    <param><value>
      <struct>

        <!-- [Customer msisdn] -->
        <member>
          <name>msisdn</name>
          <value><string>1234567890</string></value>
        </member>

        <!-- [Mobile Country Code of currently visited PLMN] -->
        <member>
          <name>mcc</name>
          <value><string>244</string></value>
        </member>

        <!-- [Mobile Network Code of currently visited PLMN] -->
        <member>
          <name>mnc</name>
          <value><string>91</string></value>
        </member>

      <!-- Callback parameters. -->

      <!-- URL to the XML-RPC server. -->
      <member>
        <name>callback-url</name>
        <value><string>http://trigger.sender.com/xmlrpc</string></value>
      </member>

      <!-- Name of the XML-RPC callback function. -->
      <member>
        <name>callback-function</name>
        <value><string>callback</string></value>
      </member>
    </value>
  </param>
</params>
</methodCall>
```

```
    </struct>
  </value></param>
</params>
</methodCall>
```

Service responses

For each service request, service platform returns a response. For all services described in previous chapter, the response format is an XML-RPC document or HTTP error status.

The response XML-RPC document contains the original request added with two parameters: transaction identifier and request status. The transaction identifier is a unique identifier generated by Service Platform as a part of the framework transaction management. It is mainly beneficial for complex services, which include several steps of communication or require phone user input (for example digital signatures). In such services the whole transaction duration can be anything from seconds to days, and it is not useful to reserve a HTTP(S) session for the whole duration.

Request status indicates, if the request has been processed correctly or if (non-HTTP – level) errors have occurred.

The additional parameters are listed in the table below.

Parameter	Type	Value
transaction-id	string	An unique id for the transaction.
status	string	Status code for transaction success. Error codes depend on the used service.

Appendix A

DTD for XML-RPC documents

The DTD for the XML documents does not accurately describe the protocol because of the limitations in the DTD format. This DTD is informative, not normative. There is no official DTD for XML-RPC.

```
<!ELEMENT methodCall (methodName, params)>
<!ELEMENT methodName (#PCDATA)>
<!ELEMENT params (param*)>
<!ELEMENT param (value)>
<!ELEMENT value (i4 | int
                 |string |dateTime.iso8601
                 |double |base64
                 |struct |array)>

<!ELEMENT i4 (#PCDATA)>
<!ELEMENT int (#PCDATA)>
<!ELEMENT string (#PCDATA)>
<!ELEMENT dateTime.iso8601 (#PCDATA)>
<!ELEMENT double (#PCDATA)>
<!ELEMENT base64 (#PCDATA)>

<!ELEMENT array (data)>
<!ELEMENT data (value*)>
<!ELEMENT struct (member+)>
<!ELEMENT member (name, value)>
<!ELEMENT name (#PCDATA)>

<!ELEMENT methodResponse (params | fault)>
<!ELEMENT fault (value)>
```