# User Guidelines for the Web Services Channel and the Related Certificate Service of OP-POHJOLA GROUP

## Payment Transfer Services

**User guide**
**August 2010**

# 1 Introduction

This guide describes the procedures and practices related to the Web Services Channel (hereinafter also the 'WSC') not covered by the common message specifications issued by the banks.

These guidelines advise on how to obtain and use the certificates required by the WSC of OP-Pohjola Group. The system is referred to herein as the WSC Certificate Service or, in shorter form, the Certificate Service.

The guidelines describe the operations and the message descriptions of the WSC and Certificate Service. In addition, the document includes instructions on software implementation and example content/messages that can be utilised in the implementation.

## 1.1 The Web Services Channel

The Web Services Channel is designed for the secure transmission of binary content between OP-Pohjola Group's corporate customer and the services of the bank (hereinafter 'the bank').

The WSC allows the customer systems to upload to and download from the Bank payment transfer content, such as C2B payment transaction files, account statements, e-invoice files, and the related notice messages.

The WSC message specifications were issued in co-operation by various bank consolidations and are available without charge on the Web site of the Federation of the Finnish Financial Services at www.fkl.fi via the search term 'Web Services'.

The WSC verifies the integrity and authenticity of the messages and application requests with XML Digital Signature Technology – i.e., by means of a digital signature. In order to trust a message or application request, the receiver must verify its signature. A public key, or, in practice, a certificate, is required for verifying and authenticating the sender's signature, hence the need for the Certificate Service – i.e., a certificates management system

## 1.2 The Certificate Service of the Web Services Channel

The purpose of the Certificate Service of the Web Services Channel is to generate and manage the certificates used for verification and authentication of signatures in the WSC.

The Certificate Service generates the required WSC certificates and is responsible for maintaining and publishing the related revocation information.

Most Certificate Service operations are performed over the WSC – i.e., from the end user's perspective through the operation of the customer's IT system.

Because of the user rights related to the certificate, the customer must at the beginning of a certificate's life cycle visit the Bank for verification of identity, thus allowing the linking of the certificate securely with the WSC username. This first authentication cannot be performed digitally.

### 1.3 Limitations

This guide does not contain any information on the customer's software; instead, it only describes the functions and operations that must be available in the software used by the customer The user can find detailed, concrete instructions in the user instructions for the relevant software.

This set of guidelines issued by the Bank is non-binding and does not constitute a legal definition of the parties' duties and obligations related to the use of the keys and certificates. The legal specification of the duties and responsibilities is presented in the terms and conditions of the Web Services Channel Agreement.

### 1.4 Source material

The common WSC guidelines have been published on the Web site of the Federation of Finnish Financial Services at www.fkl.fi, available there via the search term 'Web Services'.

### 1.5 Definitions

| | |
|---|---|
| ApplicationRequest | The service request contained in a WSC message – in practical, terms a signed XML document incorporating the required identifying information and business content. |
| Key pair | Keys used in the Public Key Infrastructure (PKI). There are always two keys, because of asymmetric encryption. See also 'Private key' and 'Public key'. |
| CA | The Certificate Authority, the organisation issuing and authenticating the certificate. |
| CA certificate | A certificate from a Certificate Authority by which the receiver of a certificate verifies the authenticity of the certificate. The authenticity of a CA certificate is verified by means of a root certificate. |
| Certificate Authority | See 'CA'. |
| Common Name | The subject field in the certificate, indicating the holder of the certificate. In the WSC Certificate Service, this field contains the username for which the certificate has been issued. |
| Public key | The public part of a key pair. This key is distributed to other parties. The public key requires no safeguarding or encryption, since it is public by definition. In most cases, a public key is distributed in the form of a certificate. |
| Root certificate | The ultimate certificate in the certificate chain, by which the authenticity of a CA certificate is verified. The root certificate is always distributed to the users separately from other certificates and via a different route, and it is in many cases included in the software installation package. |
| Application Request | An XML document by the name ApplicationRequest in the WSC, incorporating the command data for the service requested by the customer system from the Bank, any content related to the application request, |

| | |
|---|---|
| | and the digital signature required for authenticating the request. |
| pkcs10 | The standardised format of a certificate application request. |
| PKI | Public Key Infrastructure. An encryption system based on asymmetric algorithms. The system also comprises, in the form of certificates, the distribution and management of public keys. |
| Registration | The event in which the identity of the holder of a new certificate to be generated is verified. Registration ensures that the holder of the certificate is known with certainty and that authorisations can be linked with the certificate<br><br>The certificate is included in the certificate revocation list and in the blocking services so that the systems trusting the certificate will reject the certificate – i.e., deny the related authentication check. The certificate is revoked especially upon discovery or suspicion of unauthorised access to the private key. |
| Secret key | See ‘Private key’. |
| Transfer Secret | Same as a transfer key. A concept in the WSC Certificate Service. |
| Transfer key | Means by which the WSC Certificate Service verifies the authenticity of a certificate application request. The system sending the certificate application request includes the transfer key in the request. A concept in the WSC Certificate Service. |
| SOAP message | Wrapper for the WSC application requests and returned responses. SOAP messages are standardised XML documents containing, for example, security elements. |
| Subject | Certificate element providing information on the holder of the certificate. In the WSC Certificate Service, the most important information is the Common Name (CN) – i.e., the username for which the certificate was issued. |
| Certificate Service | A service from OP-Pohjola Group that generates the customer certificates required for the WSC and the related support operations, such as the blocking service. |
| Certificate | A digital document (e.g., an XML document) the primary purpose of which is to link a public key and the information on its holder with each other. In addition to these two items, the certificate contains other important information and is signed by a Certificate Authority. The Certificate Authority’s signature verifies the above information and the integrity of the certificate. |

| | |
|---|---|
| Certificate application request | A digital document sent by the customer system to the WSC, containing the customer's public key and identifier. The Bank's Certificate Service generates a certificate in conformity with the certificate application request and returns the certificate to the customer system in a response message. |
| Revocation of certificate | Mechanism for early expiry. In the event that the customer suspects or is aware of unauthorised access to the customer's private key, the customer must revoke the certificate without delay. A revoked certificate is no longer valid in the WSC. Use of revoked certificate cannot be reinstated; the customer must register a new certificate and submit a new certificate application request. |
| Web Services Channel | A service of the Bank, based on the Web Services and SOAP standards, by which the systems of the Bank's corporate customers upload to and download from the Bank binary content. |
| WSC | See 'Web Services Channel'. |
| XML Digital Signature | See 'XML signature'. |
| XML signature | Technology used for verifying the authenticity and integrity of an XML document. The signature is issued with a private key and authenticated with a public key. |
| X.509v3 | Standardised presentation format of certificates. |
| Private key | The secret part of the key pair, available only to the key holder. This key must be safeguarded against disclosure and copying with due care. Also referred to as the secret key. |

# 2 General security principles

The most critical security targets in the Certificate Service are as follows:

1. Safekeeping and use of the private key must be so arranged that only authorised persons can access and use the key. On the basis of the private key, the customer software generates the signature that allows the Bank to accept the authenticity of the content (hereinafter specifically 'Content') on trust and to authenticate the author of the Content.

2. The registration of certificates, delivery of transfer keys, and authentication of certificate application requests must be performed after a secure and reliable fashion. This procedure will ensure that the certificate is effectively generated on the basis of the public key that the customer created upon registration at the Bank.

3. The certificate blocking service (certificate revocation list) must be in operation and the information supplied by the service up to date at all times. This pertains to the Bank in particular, since the Bank uses the certificates for authentication of business Content sent by customers and thus for allowing such Content to enter processing. In the event that a customer has revoked a certificate, the Bank must not accept the signature generated by means of the secret key corresponding to the certificate in question.

The Certificate Service also involves other operations that are critical and essential where data security is concerned, but the three objects mentioned above are the most important.

## 2.1 The quality of the key pair

The customer (hereinafter 'the Customer') is responsible for generating the key pair used in the WSC. The key pair can be generated by means of dedicated software or by the Customer's system. The Customer's software can use a security module for the generation and safekeeping of the key pair.

The Bank will not participate in the generation of the key pair or be able to view or process the Customer's private key.

The Customer must ensure that the quality of the key pair is sufficient. First of all this means that the random integer used in the generation of the key must be random enough and thus cannot be reiterated. The party implementing the application that generates the key pair must ensure that the quality of the algorithm used in the generation is adequate and complies with good encryption practice.

## 2.2 Safekeeping and use of the private key

The Customer is responsible for the safekeeping of the private (secret) key and for controlling the use of the key.

The private key must not be stored in unencrypted form or its use allowed without adequate authentication.

On the basis of the private key, the Customer's software application generates in the WSC the required XML signature that allows the Bank to accept the message and the contained application request on trust and thus also the Content sent. A party having access to the private key is able to transmit over the WSC application requests and Content to the Bank, which transfer the Bank will execute in the name of the Customer linked with the private key through the certificate.

The Customer is liable in full for all transactions performed through the private key.

### 2.3 Identification of the application request in the Bank's Certificate Service

The Customer's system submits a certificate application request to the Bank's Certificate Service over the WSC.

Depending on the type of certificate application request, the Bank's service identifies and authenticates the request through the following methods. In all identification methods, protection against third-party access is provided by SSL-protected message transmission.

In the case of authentication of the certificate for a username for the first time, the element CertApplicationRequest.transferKey must contain the 16-digit transfer key received from the Bank and the element CertApplicationRequest.customerId the 10-digit username. The last digit in the transfer key is a verifier, by which the Customer's software can locally verify that the transfer key is entered correctly. The verifier is calculated with the Luhn modulo 10 algorithm.

In the case of renewal of a valid certificate, the element CertApplicationRequest must be signed with the key of the existing certificate already in use and the element CertApplicationRequest.customerId must contain the 10-digit username.

In the event that the Customer's system submits a certificate application request regarding the key pair of an existing certificate already in use, the Bank's Certificate Service will not generate a new certificate but return a copy of the existing certificate in use.

### 2.4 Certificate revocation and use of revocation list information

The Customer can revoke its certificate via the telephone number *010 2528470*.

The 10-digit WSC username or the certificate's serial number is required for the revocation.

A revoked certificate is no longer valid in the WSC and therefore cannot be reinstated. After revocation, the Customer must register a new certificate and submit to the WSC a certificate application request with the transfer key.

The Bank publishes a certificate revocation list for customers at http://wsk.op.fi/crl/ws/OP-Pohjola-ws.crl. The revocation list is updated once a day and is in force for two days at a time. The Customer's system must retrieve the revocation list and be up to date accordingly at al times. The system must check from the revocation list the status of the certificates linked with digitally signed response messages.

The Bank does not consent to any other use of the WSC Certificates but for the purposes of the WSC, and the publication or accuracy of the revocation list intended for the Bank's internal use does not place the Bank under any obligation or liability. The Bank has up-to-date information on the revocation status of Customer certificates but does not provide this service to third parties.

# 3 The Web Services Channel

The Web Services Channel is designed for the secure transmission of binary Content between a corporate customer's system and the services of the Bank.

The operation of the WSC is based on the Security and Message Specification issued through co-operation by banks operating in Finland.

The preferred mode of connection in the WSC is an SSL-protected HTTPS connection over the public Internet. The transport unit in the channel is a digitally signed SOAP message. The message contains the XML document ApplicationRequest, which is the actual service request. Also, the ApplicationRequest (i.e., the service request) is digitally signed. The ApplicationRequest contains the business Content related to the service – e.g., payment transaction files.

The WSC is designed for uploading and downloading batches. The Customer's system transmits an application request, and the WSC immediately returns a response. The Content sent is saved on the Bank side, pending processing. The processing may generate feedback files, which the Customer's system must download separately.

The production environment WSDL file is available at:

https://wsk.op.fi/wsdl/MaksuliikeWS.xml

The test environment WSDL file is available at:

https://wsk.astesti.op.fi/wsdl/MaksuliikeWS.xml

## 3.1 Operations of the Web Services Channel

The Web Services Channel is in production and used by OP-Pohjola Group in parallel with the batch transfer service. The services related to corporate payment transfer traffic will be complemented further in the course of 2010.

### 3.1.1 Uploading Content to the Bank

The business application of the Customer or the Customer's third-party administrator transmits Content to the Bank over the WSC.

The WSC performs a format validation check on the Content upon transmission and rejects any corrupted Content. The WSC will not save the corrupted Content. The WSC immediately returns an error message to the sending software, with error code 12 and the message 'Schema Validation Failed'.

Only one piece of Content can be transmitted at a time – i.e., one unit of Content per message.

We recommend compressing the Content irrespective of its size (see 'Compression of Content').

### 3.1.2 Downloading Content from the Bank

The Customer's software can download Content from the WSC, both Content sent to the channel by the Customer and downloadable Content generated by the Bank.

When downloading, the Customer must specify the Content for download by the Content reference (FileReference). The Content references will be shown upon

listing of the Content. Then the Customer can download Content by the Content reference. The Customer also receives the reference for Content sent to the channel in the response message to the Content transmission.

Only one piece of Content can be downloaded at a time.

The Content will be stored in the WSC for three months and then deleted automatically. Deletion requires no action by the Customer.

The Customer may download the same Content multiple times. The status of downloaded Content is changed from 'NEW' to 'DLD', but the Content remains viewable and downloadable.

### 3.1.3 Content size limitations

Large XML payment transaction files slow down payment processing in the Web Services Channel. For this reason, OP-Pohjola Group has decided to limit the size allowed for XML Content, in order to secure fast and reliable transmission of payment transfer content between the corporations and the Bank. The largest single C2B payment transfer Content item allowed by OP Pohjola Group is currently 50,000 payment transactions per Content item. The maximum size allowed size for other Content units is 100 Mbit.

### 3.1.4 Compression of content

We recommend that Content sent to the Bank be compressed at all times. The compression algorithm is GZIP in compliance with RFC1952. The original Content is compressed prior to Base64 encoding and writing in the element ApplicationRequest.content. The value of ApplicationRequest.compression must be 'true' after compression.

We also recommend compression in download of Content from the Bank. Setting the value of ApplicationRequest.compression = 'true' in the download request will compress the downloadable Content.

### 3.1.5 Listing of Content

The Customer's system can retrieve from the WSC a listing of available Content. The following search criteria can be used in the listing:

- The moment of saving of the Content in the channel within a given period, delimited by the date.
- Content status information
    - for Content sent by the Customer
        - WPF – pending processing ('Waiting for Processing')
        - FWD – forwarded for further processing ('Forwarded')
    - for Content downloadable by the Customer
        - DLD – downloaded ('Downloaded')
        - NEW – not downloaded ('New')
- Content type – e.g., 'pain.001.001.02' or 'pain.002.001.02'.

Content deleted by the Customer with the deleteFile operation will not be shown in the listing (see 'Deletion of Content').

Please note when generating a listing that both the Content sent to the Bank by the Customer and Content allowed by the Bank for download by the Customer are shown in the Content list. The Customer's software can select, by applying appropriate filters to the getFileList operation, the Content to be shown in the list.

### 3.1.6 Deletion of Content

The Customer's system can delete any Content sent to the WSC by the Customer. The deletion will deny forwarding of the Content for further processing.

The Customer can also delete in the WSC, via the deleteFile operation, any Content the Customer has sent to the Bank. The deletion of Content only changes the status of the Content from 'WFP' to 'DEL'. This status change only denies entry of the Content for processing and has no further consequences. The deleted Content cannot be viewed by means of the getFileList operation.

Deletion of Content may have its benefits but is only feasible within the time slot from transmission to entry into processing. For example, for SEPA C2B payment transaction Content, the time slot is 30 minutes at maximum.

This means that Content must be deleted rather a short time after transmission, since Content in processing (with the status 'FWD') cannot be deleted or cancelled in the WSC. The WSC will return an error message in the event of such an attempt.

The time the Content is pending for further processing in the WSC varies in accordance with the service and the Content type. For example, C2B payment transaction Content is processed on banking days from 7am to 6pm at 30-minute intervals.

### 3.1.7 Administrator and authorisations

The authorisation related to payment transfer content is based on the Generator role for WSC username. The so-called administrator identifier is created on the basis of the CustomerID value entered in the WSC Agreement for the username in question and of the service point number, which is a parameter for the username. This administrator identifier – i.e., the service point – must be included in the allowed senders list or as an allowed receiver of downloadable Content in the payment transfer agreement applicable to the processing and generation of the Content.

The administrator is the party entered in the payment transfer agreement as the allowed sender or receiver of the Content. The administrator has a dedicated WC Channel Agreement and the related usernames and certificates linked with the usernames.

## 3.2 Example messages and application requests

### 3.2.1 Request message

An example SOAP request message for the getFileList operation is presented below. The Base64-encoded Content elements have been shortened and the omitted parts replaced with three dots for better readability.

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
     <env:Header>
          <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
          wss-wssecurity-secext-1.0.xsd" env:mustUnderstand="1">
               <wsse:BinarySecurityToken wsu:Id="bst_Agz1d9olDWEwBt2u"
               xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
               wssecurity-utility-1.0.xsd" ValueType="http://docs.oasis-
               open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"
               EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-
               message-security-1.0#Base64Binary">MIIDmjCCAo...
```

```
                1jB0+UOw=</wsse:BinarySecurityToken>
                <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
                     <dsig:SignedInfo>
                          <dsig:CanonicalizationMethod
                          Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                          <dsig:SignatureMethod
                          Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                          <dsig:Reference URI="#Body_H2Tm3eguh8pYNnNB">
                               <dsig:Transforms>
                                    <dsig:Transform
                                    Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                                         <exc14n:InclusiveNamespaces
                                         xmlns:exc14n="http://www.w3.org/2001/10/xml-exc-
                                         c14n#" PrefixList=""/>
                                    </dsig:Transform>
                               </dsig:Transforms>
                               <dsig:DigestMethod
                               Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                               <dsig:DigestValue>FtfDFuSJmTMmoaAvOCUzhzVBsMU=
                               </dsig:DigestValue>
                          </dsig:Reference>
                     </dsig:SignedInfo>
                     <dsig:SignatureValue>D7O7643... g4VRfA==</dsig:SignatureValue>
                     <dsig:KeyInfo>
                          <wsse:SecurityTokenReference xmlns:wsse="http://docs.oasis-
                          open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
                          xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
                          wss-wssecurity-utility-1.0.xsd" wsu:Id="str_Zq2ZC4G9rKVOkaxE">
                               <wsse:Reference URI="#bst_Agz1d9olDWEwBt2u"
                               ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
                               200401-wss-x509-token-profile-1.0#X509v3"/>
                          </wsse:SecurityTokenReference>
                     </dsig:KeyInfo>
                </dsig:Signature>
                <wsu:Timestamp xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
                200401-wss-wssecurity-utility-1.0.xsd">
                     <wsu:Created>2010-02-03T08:21:47Z</wsu:Created>
                     <wsu:Expires>2010-02-03T08:22:47Z</wsu:Expires>
                </wsu:Timestamp>
           </wsse:Security>
      </env:Header>
      <env:Body wsu:Id="Body_H2Tm3eguh8pYNnNB" xmlns:wsu="http://docs.oasis-
      open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
           <cor:downloadFileListin xmlns:cor="http://bxd.fi/CorporateFileService">
                <mod:RequestHeader xmlns:mod="http://model.bxd.fi">
                     <mod:SenderId>1000000000</mod:SenderId>
                     <mod:RequestId>1265185304796</mod:RequestId>
                     <mod:Timestamp>2010-02-03T10:21:44.796+02:00</mod:Timestamp>
                     <mod:Language>FI</mod:Language>
                     <mod:UserAgent>OP Client 1.13</mod:UserAgent>
                     <mod:ReceiverId>OKOYFIHH</mod:ReceiverId>
                </mod:RequestHeader>
                <mod:ApplicationRequest xmlns:mod="http://model.bxd.fi">PD94bW...
                F1ZXN0Pg==</mod:ApplicationRequest>
           </cor:downloadFileListin>
      </env:Body>
</env:Envelope>
```

### 3.2.2 Response message

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
      <env:Header>
           <wsse:Security env:mustUnderstand="1" xmlns:wsse="http://docs.oasis-
           open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
                <wsse:BinarySecurityToken wsu:Id="bst_dq9DVAoA1G9jPpj1"
                ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-
                token-profile-1.0#X509v3" EncodingType="http://docs.oasis-
                open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary"
```

```
                xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
                wssecurity-utility-1.0.xsd">MIID2DCC... L6euA==</wsse:BinarySecurityToken>
                <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
                     <dsig:SignedInfo>
                          <dsig:CanonicalizationMethod
                          Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                          <dsig:SignatureMethod
                          Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                          <dsig:Reference URI="#Body_3Bne1wbo7t62GvzN">
                               <dsig:Transforms>
                                    <dsig:Transform
                                    Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                                         <exc14n:InclusiveNamespaces PrefixList=""
                                         xmlns:exc14n="http://www.w3.org/2001/10/xml-exc-
                                         c14n#"/>
                                    </dsig:Transform>
                               </dsig:Transforms>
                               <dsig:DigestMethod
                               Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                               <dsig:DigestValue>+MbKIB12X0CKQT2sjZMpyMfoP0I=
                               </dsig:DigestValue>
                          </dsig:Reference>
                     </dsig:SignedInfo>
                     <dsig:SignatureValue>Ytb27TZ... 6KMvQ==</dsig:SignatureValue>
                     <dsig:KeyInfo>
                          <wsse:SecurityTokenReference wsu:Id="str_ShzyCryYMTbzPcaA"
                          xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
                          wss-wssecurity-utility-1.0.xsd">
                               <wsse:Reference URI="#bst_dq9DVAoA1G9jPpj1"
                               ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
                               200401-wss-x509-token-profile-1.0#X509v3"/>
                          </wsse:SecurityTokenReference>
                     </dsig:KeyInfo>
                </dsig:Signature>
                <wsu:Timestamp xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
                200401-wss-wssecurity-utility-1.0.xsd">
                     <wsu:Created>2010-02-03T08:21:52Z</wsu:Created>
                     <wsu:Expires>2010-02-03T08:22:52Z</wsu:Expires>
                </wsu:Timestamp>
           </wsse:Security>
      </env:Header>
      <env:Body wsu:Id="Body_3Bne1wbo7t62GvzN" xmlns:wsu="http://docs.oasis-
      open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
           <cor:downloadFileListout xmlns:cor="http://bxd.fi/CorporateFileService">
                <mod:ResponseHeader xmlns:mod="http://model.bxd.fi">
                     <mod:SenderId>1000000000</mod:SenderId>
                     <mod:RequestId>1265185304796</mod:RequestId>
                     <mod:Timestamp>2010-02-03T10:21:49.740+02:00</mod:Timestamp>
                     <mod:ResponseCode>00</mod:ResponseCode>
                     <mod:ResponseText>OK.</mod:ResponseText>
                     <mod:ReceiverId>OKOYFIHH</mod:ReceiverId>
                </mod:ResponseHeader>
                <mod:ApplicationResponse xmlns:mod="http://model.bxd.fi">PD94bW...
                5zZT4=</mod:ApplicationResponse>
           </cor:downloadFileListout>
      </env:Body>
</env:Envelope>
```

### 3.2.3 Application request getFileList

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
      <CustomerId>1000000000</CustomerId>
      <Timestamp>2010-02-03T14:28:59.200+02:00</Timestamp>
      <StartDate>2009-12-05+02:00</StartDate>
      <EndDate>2010-02-05+02:00</EndDate>
      <Environment>TEST</Environment>
      <SoftwareId>software 1.01</SoftwareId>
      <FileType>pain.001.001.02</FileType>
```

```
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>r2tz0MTMCk1lIpAujuBU3PHZ8YE=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>atSRE0... GmX6g==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIDmjC... UOw=</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</ApplicationRequest>
```

### 3.2.4 Application response getFileList

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:ApplicationResponse xmlns:xd="http://bxd.fi/xmldata/">
     <xd:CustomerId>1000000000</xd:CustomerId>
     <xd:Timestamp>2010-02-03T14:29:01.702+02:00</xd:Timestamp>
     <xd:ResponseCode>00</xd:ResponseCode>
     <xd:ResponseText>OK.</xd:ResponseText>
     <xd:FileDescriptors>
          <xd:FileDescriptor>
               <xd:FileReference>7833</xd:FileReference>
               <xd:TargetId>MLP</xd:TargetId>
               <xd:FileType>pain.001.001.02</xd:FileType>
               <xd:FileTimestamp>2010-02-03T14:14:14.762+02:00</xd:FileTimestamp>
               <xd:Status>WFP</xd:Status>
          </xd:FileDescriptor>
          <xd:FileDescriptor>
               <xd:FileReference>7834</xd:FileReference>
               <xd:TargetId>MLP</xd:TargetId>
               <xd:FileType>pain.001.001.02</xd:FileType>
               <xd:FileTimestamp>2010-02-03T14:14:25.960+02:00</xd:FileTimestamp>
               <xd:Status>WFP</xd:Status>
          </xd:FileDescriptor>
     </xd:FileDescriptors>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>aI0eODdHLtUpuu2roQKSWagjJd4=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>akahsu... 3V+20eA==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIID1z... lc5gLu</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</xd:ApplicationResponse>
```

### 3.2.5 Application request getFile

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
     <CustomerId>1000000000</CustomerId>
     <Timestamp>2010-02-03T14:53:41.603+02:00</Timestamp>
     <StartDate>2010-02-03+02:00</StartDate>
     <Environment>TEST</Environment>
     <FileReferences>
          <FileReference>7834</FileReference>
     </FileReferences>
     <Compression>true</Compression>
     <SoftwareId>software 1.01</SoftwareId>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>Lw0fT+Bk31Jd17Th+3sZ7rE/FbA=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>cVYjnG... hKdFQ==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIDmj... Ow=</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</ApplicationRequest>
```

### 3.2.6 Application response getFile

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:ApplicationResponse xmlns:xd="http://bxd.fi/xmldata/">
     <xd:CustomerId>1000000000</xd:CustomerId>
     <xd:Timestamp>2010-02-03T14:53:44.376+02:00</xd:Timestamp>
     <xd:ResponseCode>00</xd:ResponseCode>
     <xd:ResponseText>OK.</xd:ResponseText>
     <xd:Compressed>true</xd:Compressed>
     <xd:CompressionMethod>RFC1952</xd:CompressionMethod>
     <xd:Content>H4sIAAAAAAA... V0wHAAA=</xd:Content>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>1P5FMN7gb/de+EbP7FVm5bNf2x8=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>hyfVx7... VQ5fQ==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIID1z... oOlc5gLu</X509Certificate>
               </X509Data>
```

```
          </KeyInfo>
     </Signature>
</xd:ApplicationResponse>
```

### 3.2.7 Application request uploadFile

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
     <CustomerId>1000000000</CustomerId>
     <Timestamp>2010-02-03T14:36:55.959+02:00</Timestamp>
     <Environment>TEST</Environment>
     <TargetId>target</TargetId>
     <Compression>true</Compression>
     <SoftwareId>software 1.01</SoftwareId>
     <FileType>pain.001.001.02</FileType>
     <Content>H4sIAAAAA... wV0wHAAA=</Content>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>qFobnrcgqQnceFifSwzGHaCw9jw=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>Tc99O/... DRBQ==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIDmj... 0+UOw=</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</ApplicationRequest>
```

### 3.2.8 Application response uploadFile

In the following example, a validation error has been detected in the pain.001.001.002 Content sent by the Customer.

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:ApplicationResponse xmlns:xd="http://bxd.fi/xmldata/">
     <xd:CustomerId>1000000000</xd:CustomerId>
     <xd:Timestamp>2010-02-03T14:07:04.629+02:00</xd:Timestamp>
     <xd:ResponseCode>12</xd:ResponseCode>
     <xd:ResponseText>Schemavalidation failed.</xd:ResponseText>
     <xd:FileType>pain.002.001.02</xd:FileType>
     <xd:Content>PD94bW... lbnQ+Cg==</xd:Content>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>Ju6PdF7uPqv0EeQPKmej0Vx5rps=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>gzOTP... zhlrDA==</SignatureValue>
          <KeyInfo>
```

```
                <X509Data>
                     <X509Certificate>MIID1zCC... lc5gLu</X509Certificate>
                </X509Data>
           </KeyInfo>
     </Signature>
</xd:ApplicationResponse>
```

ApplicationResponse.Content contains the following pain.002.001.02 Content. Further information on the Content and usage of payment feedback of this kind is available in the separate instruction concerning SEPA C2B payment transactions.

```
<?xml version="1.0" encoding="UTF-8"?>
<Document xmlns="urn:iso:std:iso:20022:tech:xsd:pain.002.001.02"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<pain.002.001.02>
<GrpHdr>
<MsgId></MsgId>
<CreDtTm>2010-02-03T14:07:04+02:00</CreDtTm>
</GrpHdr>
<OrgnlGrpInfAndSts>
<NtwkFileNm>1265198819592</NtwkFileNm>
<OrgnlMsgNmId>pain.001.001.02</OrgnlMsgNmId>
<GrpSts>RJCT</GrpSts>
<StsRsnInf>
<StsOrgtr>
<Id>
<OrgId>
<PrtryId>
<Id>1000000000</Id>
</PrtryId>
</OrgId>
</Id>
</StsOrgtr>
<StsRsn>
<Cd>NARR</Cd>
</StsRsn>
<AddtlStsRsnInf>pain.001.001.02 could not be processed - please verify structure.cvc-
datatype-valid.1.2.1: 'False' is not</AddtlStsRsnInf>
<AddtlStsRsnInf>a valid value for 'boolean'.cvc-type.3.1.3: The value 'False' of element
'BtchBookg' is not valid.Invalid</AddtlStsRsnInf>
<AddtlStsRsnInf>byte 1 of 1-byte UTF-8 sequence.</AddtlStsRsnInf>
</StsRsnInf>
</OrgnlGrpInfAndSts>
</pain.002.001.02>
</Document>
```

### 3.2.9 Application request deleteFile

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
     <CustomerId>1000000000</CustomerId>
     <Timestamp>2010-02-03T14:48:48.825+02:00</Timestamp>
     <StartDate>2010-02-03+02:00</StartDate>
     <Environment>TEST</Environment>
     <FileReferences>
          <FileReference>7833</FileReference>
     </FileReferences>
     <SoftwareId>software 1.01</SoftwareId>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
```

```
                    <DigestValue>8mzDhp5J7LpeYh4NC0NqKCpG1pU=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>dUulFB... zhOeg==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIDmj... +UOw=</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</ApplicationRequest>
```

### 3.2.10 Application response deleteFile

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:ApplicationResponse xmlns:xd="http://bxd.fi/xmldata/">
     <xd:CustomerId>1000000000</xd:CustomerId>
     <xd:Timestamp>2010-02-03T14:48:52.194+02:00</xd:Timestamp>
     <xd:ResponseCode>00</xd:ResponseCode>
     <xd:ResponseText>OK.</xd:ResponseText>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>iHS1HaxShIzpTBR+0WaPZR4YxCA=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>chSQ3... SX3ow==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIID1... c5gLu</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</xd:ApplicationResponse>
```

# 4 Certificate Service of the Web Services Channel

## 4.1 Operations of the Certificate Service

### 4.1.1 Registration of a certificate and the transfer key

In order to use the WSC, the Customer's software must have a key pair and certificate issued by the Certificate Service of OP-Pohjola Group's WSC.

The Customer's software will retrieve the certificate from the Bank's WSC.

For the purposes of retrieving the certificate, the Customer must provide the software with a transfer key, by which the Certificate Service will identify and authenticate the request submitted by the software.

The Customer receives the transfer key upon registration at the Bank. The Bank's employee performs the registration. The registration always correlates to a specific WSC username.

In conjunction with the registration, the Bank's employee verifies the identity of the Customer's representative and checks the representative's authorisation and other credentials. At the Bank, the Customer receives a printed document, which contains the WSC username and the first part of the transfer key, eight digits.

The Customer can elect to receive the second part of the transfer key via SMS message to a handset or have it sent to post to an address specified by the Customer.

Once the Customer holds both parts of the transfer key, 16 digits in total, the Customer must enter the key in the software and initiate generation of the certificate.

### 4.1.2 Generation of the key pair

The key length is 2048 bits, and the algorithm RSA; the message digest algorithm for the signature is SHA-1.

The key pair must be generated with such an algorithm and method as will ensure adequate randomness.

### 4.1.3 Safekeeping of the private key

The private key must be safeguarded against any risk of unauthorised access. The most secure repository is a physical security module – i.e., a Hardware Security Module (HSM). When a security module is applied, the private key will be generated inside the security module and cannot be removed from the module in the course of regular usage. If the private key is not contained in a security module, the key must be at least adequately encrypted.

Access to the private key must be controlled and so secure that only authorised software applications can use the key.

The holder of the private key is responsible for the use and safekeeping of the key as well as for any unauthorised use.

In the event that the user 1) suspects or 2) becomes aware of unauthorised access to the private key or 3) of unauthorised use of the key, the certificate related to the key in question must be revoked without delay in the Bank's blocking service.

### 4.1.4 Submission of a certificate application request

The Customer's software submits a certificate application request to the Bank's WSC Certificate Service. For the operation, the software requires the 16-digit transfer key entered by the Customer and the username activated for the WSC Agreement.

The software submits a certificate application request and, after having received the certificate from the Certificate Service, saves it for future use.

The public key is used for creating a certificate application request in pksc10 format.

The following two items will be entered in the subject of the certificate application request:

```
C=FI

CN=[WSC username, 10 digits]
```

There is great variety in certificate application requests, and the identification and authentication by the Bank's Certificate Service is case-specific.

When the certificate application request is based on a prior registration, the element CertApplicationRequest must contain a binary certificate application request in pkcs10 format. In addition, the element CertApplicationRequest.transferKey must contain the 16-digit transfer key. The CertApplicationRequest and SOAP message do not require signing.

When the certificate application request is based on a prior certificate, the element CertApplicationRequest.Content must contain a binary certificate application request in pkcs10 format (the binary 'Content' element must be Base64-encoded in accordance with the schema). The CertApplicatIonRequest must be signed with the key corresponding to the certificate issued to the username for which the certification is requested. The SOAP message does not require signing in this case either.

If the Customer's system submits a certificate application request using a serial number, the element CertApplicationRequest.serialNumber must contain the serial number of the certificate. The CertApplicationRequest and SOAP message do not require signing.

If the public key in the certificate application request is contained in a prior certificate for the same username, the prior certificate, even if lapsed, will be returned in the response to the request. In this case, the WSC will not show an error message and the requesting software must detect on its own terms that the copy received pertains to a lapsed certificate and that no new certificate was generated.

It is imperative that the Customer's software check, upon submitting a certificate application request, the SSL certificate of the Bank's Certificate Service issued for the domain wsk.op.fi. The check ensures that the certificate application request is effectively routed to the Bank's service.

### 4.1.5 Use of keys and certificates

The Customer's system uses the Customer's private key for issuing digital signatures. Both the application request (ApplicationRequest) and the SOAP message must be signed separately in the WSC.

The signature is performed with the private key. The signing system must include in the signature also the certificate. This certificate contains the public key

corresponding to the private key used in the signing. The receiver uses the public key to authenticate the signature.

The signature verifies that the signed message or service request has not been modified after signing and that the message or request originates from the stated sender, since only the holder of the private key can issue the signature.

The certificate is used for binding the public key and thus the key pair to the holder. In the WSC certificates, the Common Name information in the subject of the certificate containing the WSC username is the identifier for the holder.

#### 4.1.6 Certificate life cycle and renewal

The Customer's software uses the generated or received key for digital signing of messages and application requests in the WSC. In addition, the software must include in each signature the certificate associated with the key in question received from the Bank's Certificate Service.

The certificate is valid for two years. The certificate must be renewed before the expiry of the prior certificate. The certificate can be renewed no sooner than 60 calendar days before the expiry of the prior certificate. In the event that the certificate lapses before a new certificate is obtained, the Customer must start the registration process from the beginning.

The Customer must also monitor the validity period and renew the certificate in time. The Customer's software will renew the certificate automatically. The software detects the end date of the certificate each time the certificate is used.

A new key pair must be generated for the renewed certificate. If the certificate renewal request is submitted by means of the key pair of the prior certificate, the Certificate Service will return only a copy of the prior certificate.

The certificate renewal request is similar to the request for obtaining a new certificate, with the exception that a transfer key is not used in the renewal (CertApplicationRequest.TransferKey) and the CertApplicationRequest is signed with the private key for which the username holds a valid certificate. Thus the verification of the authenticity of the renewal request is in the Certificate Service based on the immediately precedent certificate for the username, valid at the time of submission of the request.

#### 4.1.7 Retrieval and use of revocation information

The Customer's system must retrieve the certificate revocation list from the Certificate Service and check the revocation status of trusted certificates against the revocation list. In practical terms, this means that the software must check the Bank's service certificates contained in the response message.

The Certificate Service generates the revocation list once a day, and the list is valid for two days at a time. The Certificate Service may also generate a new blocking list when a certificate is revoked – i.e., outside the normal update schedule.

#### 4.1.8 Premature expiry of the certificate

In the event that the Customer suspects or is aware of unauthorised access to the Customer's private key, the Customer must revoke the certificate without delay via the telephone number *010 2528470*.

Please see the revocation instructions referred to herein above.

### 4.2 Message descriptions for the Certificate Service

This section describes the messages and application requests used in the WSC Certificate Service.

The structure of SOAP messages and the address of the Certificate Service are available in a WSDL file.

The production environment WSDL file is available at:

https://wsk.op.fi/wsdl/MaksuliikeWS.xml

The test environment WSDL file is available at:

https://wsk.astesti.op.fi/wsdl/MaksuliikeWS.xml

#### 4.2.1 SOAP Messages and WSDL

The WSDL file describes the SOAP message structure.

The SOAP message does not require signing in the Certificate Service, and authenticity is verified by signature at the application request level only (CertApplicationRequest) – and in certain cases, not even there.

#### 4.2.2 Application requests and schemas

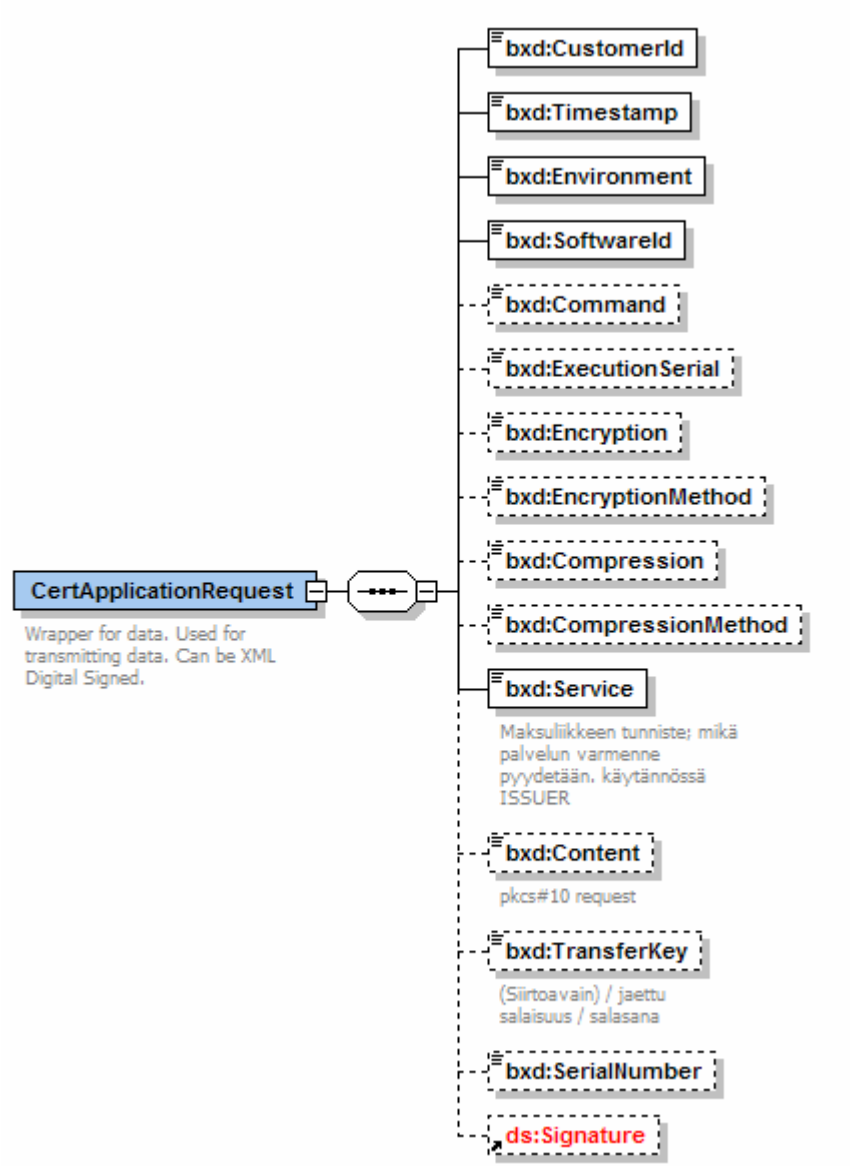
The XML Schema files describe the application request and application response wrapped in the message.

The WDSL for the Certificate Service is available at:
https://wsk.op.fi/wsdl/MaksuliikeWS.xml

The location and address of the schema files will be indicated at a later date.

The application request submitted by the Customer is called the CertApplicationRequest and the application response returned by the Bank the CertApplicationResponse.

#### 4.2.2.1 CertApplicationRequest




The main elements to be entered in the application request in the case of a certificate application request are as follows:

*CustomerId* – the WSC username of the party requesting a certificate, 10 digits

*Content* – certificate application request in pkcs10 format, Base64-encoded

*TransferKey* – transfer key (16 digits) in the case of submission of the first certificate application request by the username

*Signature* – XML signature in the case of certificate renewal

In addition, there are a few mandatory items:

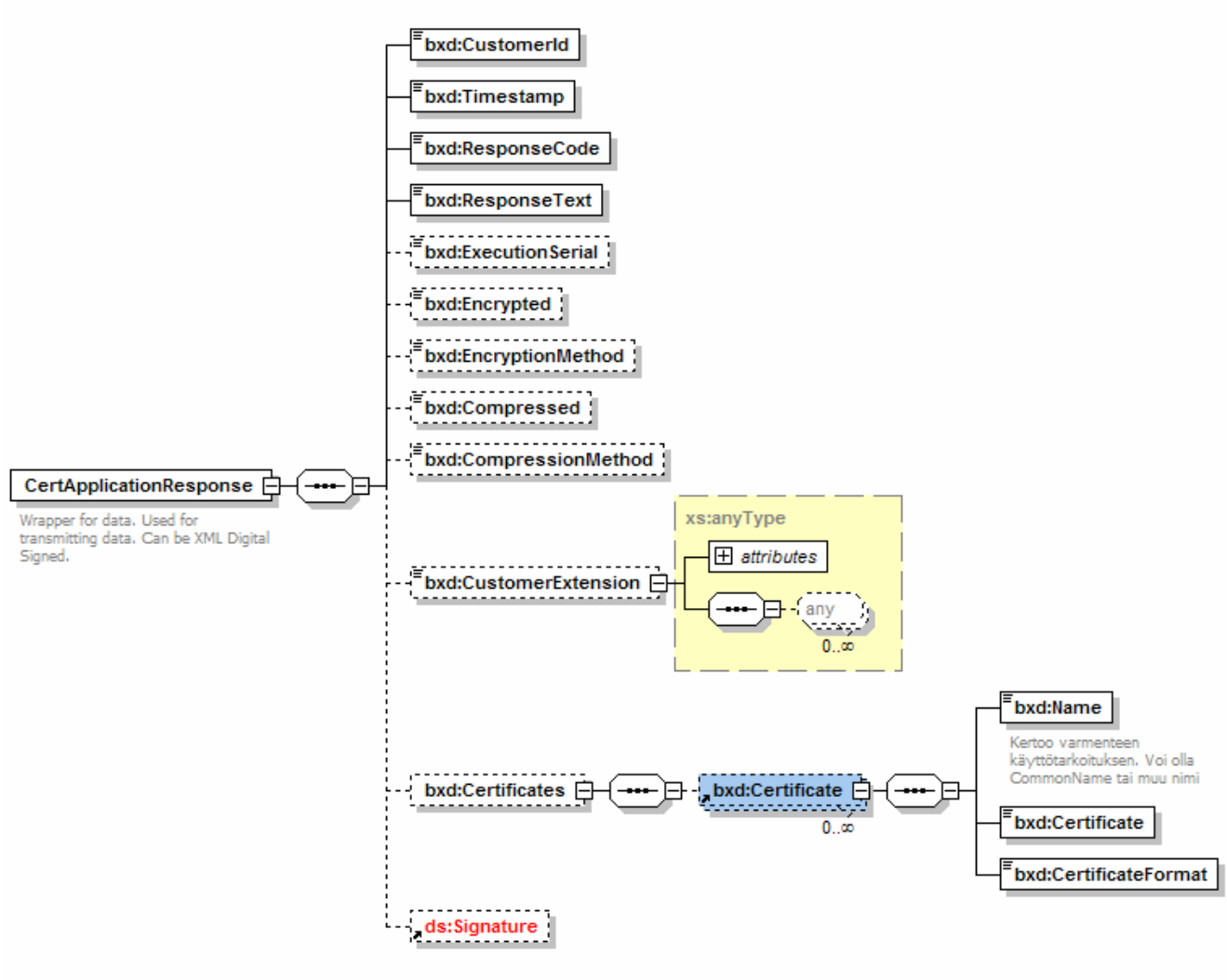
*Timestamp* – timestamp for the generation of the application request (in most cases, in support of sorting only)

*Environment* – in the production environment case, ‘PRODUCTION’ (otherwise, the request will be rejected)

*SoftwareId* – name of the software submitting the application request (in most cases, in support of sorting only)

*Service* – MATU

#### 4.2.2.2 CertApplicationResponse




**Example Content for the Certificate Service**

### 4.2.3 Request message

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
     <env:Header/>
     <env:Body>
          <opc:getCertificatein xmlns:opc="http://mlp.op.fi/OPCertificateService">
               <opc:RequestHeader>
                    <opc:SenderId>1000012222</opc:SenderId>
                    <opc:RequestId>123</opc:RequestId>
                    <opc:Timestamp>2010-01-26T14:32:43.800+02:00</opc:Timestamp>
               </opc:RequestHeader>
               <opc:ApplicationRequest>PD94bWwgdmVy...
               GlvblJlcXVlc3Q+</opc:ApplicationRequest>
          </opc:getCertificatein>
     </env:Body>
</env:Envelope>
```

### 4.2.4 Response message

```
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
     <env:Header/>
     <env:Body>
          <opc:getCertificateout xmlns:opc="http://mlp.op.fi/OPCertificateService">
               <opc:ResponseHeader>
                    <opc:SenderId>1000012222</opc:SenderId>
                    <opc:RequestId>123</opc:RequestId>
                    <opc:Timestamp>2010-01-26T14:32:45.909+02:00</opc:Timestamp>
                    <opc:ResponseCode>00</opc:ResponseCode>
                    <opc:ResponseText>OK.</opc:ResponseText>
               </opc:ResponseHeader>
               <opc:ApplicationResponse>PD94bWwgdmVyc2...
               W9uUmVzcG9uc2U+</opc:ApplicationResponse>
          </opc:getCertificateout>
     </env:Body>
</env:Envelope>
```

### 4.2.5 Application request for certificate renewal

In the example below, Username (CustomerID) 1000000047 submits an application request for certificate renewal. The application request is signed, because identification and authentication is based on a valid certificate held by the same username.

```
<?xml version="1.0" encoding="UTF-8"?>
<CertApplicationRequest xmlns="http://op.fi/mlp/xmldata/">
     <CustomerId>1000000047</CustomerId>
     <Timestamp>2010-01-26T14:32:44.191+02:00</Timestamp>
     <Environment>TEST</Environment>
     <SoftwareId>soft</SoftwareId>
     <Compression>false</Compression>
     <Service>MATU</Service>
     <Content>MIICZzCCAU8CA... 3slAmKGflLvw==</Content>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
               signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>i81y7OKgB8FBmOlv4gQWNtcCmLg=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>ZWSGuxU... gkZMGWA==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIDmjCCAoKg... Ct1jB0+UOw=</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</CertApplicationRequest>
```

### 4.2.6 Application response to certificate renewal request

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:CertApplicationResponse xmlns:xd="http://op.fi/mlp/xmldata/">
     <xd:CustomerId>1000000047</xd:CustomerId>
     <xd:Timestamp>2010-01-26T14:32:51.808+02:00</xd:Timestamp>
     <xd:ResponseCode>00</xd:ResponseCode>
     <xd:ResponseText>OK.</xd:ResponseText>
     <xd:Certificates>
          <xd:Certificate>
```

```
                <xd:Name>CN=1000000047,C=FI</xd:Name>
                <xd:Certificate>MIICvTCCAa... Ne+0U19z3z25nFb</xd:Certificate>
                <xd:CertificateFormat>X509v3</xd:CertificateFormat>
           </xd:Certificate>
     </xd:Certificates>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
           <SignedInfo>
                <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
                20010315#WithComments"/>
                <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                <Reference URI="#xpointer(/)">
                     <Transforms>
                          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                signature"/>
                     </Transforms>
                     <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                     <DigestValue>ZdaOhjgcjfFb5aRwgMeWtlR5Oj0=</DigestValue>
                </Reference>
           </SignedInfo>
           <SignatureValue>PXPPXC... +TLjnO2g==</SignatureValue>
           <KeyInfo>
                <X509Data>
                     <X509Certificate>MIIDnDCCAo... A7xVA==</X509Certificate>
                </X509Data>
           </KeyInfo>
     </Signature>
</xd:CertApplicationResponse>
```

### 4.2.7 Certificate application request with transfer key

```
<?xml version="1.0" encoding="UTF-8"?>
<CertApplicationRequest xmlns="http://op.fi/mlp/xmldata/">
     <CustomerId>1000010583</CustomerId>
     <Timestamp>2010-02-04T12:40:00.929+02:00</Timestamp>
     <Environment>TEST</Environment>
     <SoftwareId>software 1.01</SoftwareId>
     <Compression>false</Compression>
     <Service>MATU</Service>
     <Content>MIICZz... Vr5kiQ==</Content>
     <TransferKey>2251401483958635</TransferKey>
</CertApplicationRequest>
```

### 4.2.8 Application response to certificate application request with transfer key

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:CertApplicationResponse xmlns:xd="http://op.fi/mlp/xmldata/">
     <xd:CustomerId>1000010583</xd:CustomerId>
     <xd:Timestamp>2010-02-04T12:29:32.704+02:00</xd:Timestamp>
     <xd:ResponseCode>00</xd:ResponseCode>
     <xd:ResponseText>OK.</xd:ResponseText>
     <xd:Certificates>
          <xd:Certificate>
                <xd:Name>CN=1000010583,C=FI</xd:Name>
                <xd:Certificate>MIICvT... AssyGCD</xd:Certificate>
                <xd:CertificateFormat>X509v3</xd:CertificateFormat>
          </xd:Certificate>
     </xd:Certificates>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
                <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
                20010315#WithComments"/>
                <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                <Reference URI="#xpointer(/)">
                     <Transforms>
                          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                signature"/>
                     </Transforms>
                     <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
```

```
                    <DigestValue>pROjhxTaOs2FznVwOPhA7lbJYAE=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>Kv0oDf... 9BU3Iw==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIDn... xVA==</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</xd:CertApplicationResponse>
```

### 4.2.9 Application request for certificate retrieval with serial number

```
<?xml version="1.0" encoding="UTF-8"?>
<CertApplicationRequest xmlns="http://op.fi/mlp/xmldata/">
  <CustomerId>1000010583</CustomerId>
  <Timestamp>2010-02-04T12:53:55.325+02:00</Timestamp>
  <Environment>TEST</Environment>
  <SoftwareId>software 1.01</SoftwareId>
  <Compression>false</Compression>
  <Service>MATU</Service>
  <SerialNumber>442519889</SerialNumber>
</CertApplicationRequest>
```

### 4.2.10 Application response to certificate retrieval request with serial number

```
<?xml version="1.0" encoding="UTF-8"?>
CertApplicationResponseDocument::<xd:CertApplicationResponse
               xmlns:xd="http://op.fi/mlp/xmldata/">
  <xd:CustomerId>1000010583</xd:CustomerId>
  <xd:Timestamp>2010-02-04T12:54:02.370+02:00</xd:Timestamp>
  <xd:ResponseCode>00</xd:ResponseCode>
  <xd:ResponseText>OK.</xd:ResponseText>
  <xd:Certificates>
    <xd:Certificate>
      <xd:Name>CN=1000010583,C=FI</xd:Name>
      <xd:Certificate>MIICvTC... AssyGCD</xd:Certificate>
      <xd:CertificateFormat>X509v3</xd:CertificateFormat>
    </xd:Certificate>
  </xd:Certificates>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <Reference URI="#xpointer(/)">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <DigestValue>fYSxDgACYGnJyt3R0Vg9aOLkdyk=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>O4vxL... n/th4DA==</SignatureValue>
    <KeyInfo>
      <X509Data>
        <X509Certificate>MIIDnD... 7xVA==</X509Certificate>
      </X509Data>
    </KeyInfo>
  </Signature>
</xd:CertApplicationResponse>
```

### 4.2.11 Application request for retrieval of service certificates

```
<CertApplicationRequest xmlns="http://op.fi/mlp/xmldata/">
  <CustomerId>1000010522</CustomerId>
  <Timestamp>2010-02-04T12:59:35.727+02:00</Timestamp>
  <Environment>TEST</Environment>
  <SoftwareId>software 1.01</SoftwareId>
  <Service>MATU</Service>
</CertApplicationRequest>
```

### 4.2.12 Application response to retrieval request for service certificates

```
<?xml version="1.0" encoding="UTF-8"?>
<xd:CertApplicationResponse xmlns:xd="http://op.fi/mlp/xmldata/">
     <xd:CustomerId>1000010522</xd:CustomerId>
     <xd:Timestamp>2010-02-04T12:59:36.589+02:00</xd:Timestamp>
     <xd:ResponseCode>00</xd:ResponseCode>
     <xd:ResponseText>OK.</xd:ResponseText>
     <xd:Certificates>
          <xd:Certificate>
               <xd:Name>CN=OPK z/OS Certificate Authority, OU=TES3, O=OPK, L=Helsinki,
               C=FI</xd:Name>
               <xd:Certificate>MIIDz... mRFjA==</xd:Certificate>
               <xd:CertificateFormat>X509v3</xd:CertificateFormat>
          </xd:Certificate>
          <xd:Certificate>
               <xd:Name>CN=MATU-demo-CA, C=FI</xd:Name>
               <xd:Certificate>MIICxD... BpA==</xd:Certificate>
               <xd:CertificateFormat>X509v3</xd:CertificateFormat>
          </xd:Certificate>
     </xd:Certificates>
     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
               <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
               20010315#WithComments"/>
               <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <Reference URI="#xpointer(/)">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
               signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>k1wtlZ83yQmxQSAEdNHU4xIGfMo=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>Den... dCheQ==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIID...A7xVA==</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</xd:CertApplicationResponse>
```

# 5 The test environment

OP-Pohjola provides the developers of WSC client software with a test environment. The test environment is operated similarly to the production service but has a different address and different usernames and certificates. The functionality of the test service is not as comprehensive as that of the production service.

The use of the Customer test environment requires conclusion of a Corporate Bank Connection Agreement (i.e., an agreement for the WSC) and a C2B Agreement with OP-Pohjola Group. Use of the test environment requires that the Customer's bank connectivity software support C2B Content and the related feedback as well as the Corporate Bank Connection Service (i.e., the WSC).

## 5.1 Identity codes and username

Testing requires conclusion of the WSC and the C2B Content agreements. For transmission of Content to the test environment, the user must apply the CustomerId and paymentId and the payment accounts entered in the C2B Agreement. With respect to the WSC, the user must apply the username stated in the WS Agreement and also the transfer key for the test environment delivered separately and the certificate retrieved with the key.

## 5.2 The certificate

The use of the Customer test environment requires a separate dedicated WSC certificate. The production certificate is not valid in the Customer test environment. Retrieval of the test environment certificate requires a transfer key, which will be delivered to the Customer after conclusion of the agreement. The certificate can be retrieved with the transfer key as described in this user guide.

## 5.3 Contact addresses

The Customer can obtain the codes and keys by contacting us at the address **opk-sepa@op.fi**.

The address for the Customer test environment is https://wsk.asiakastesti.op.fi.

The WSDL description for the WSC services of the Customer test environment is available at:
https://wsk.asiakastesti.op.fi/wsdl/MaksuliikeWS.xml

The WDSL description for the Certificate Service of the Customer test environment is available at:
https://wsk.asiakastesti.op.fi/wsdl/MaksuliikeCertService.xml

The general service descriptions and the XML Schemas are available on the site of the Finnish Federation of Financial Services:
http://www.fkl.fi/www/page/fk_www_3830

## 5.4 Limitations of the Customer test environment

The maximum size of Content transmitted to the Customer test environment is 20 Kbit, calculated for the combined size of the Content payload and the SOAP wrapper in the WSC. The Customer is allowed to send Content to the Customer test environment 20 times per day at maximum.
The value of the parameter 'Environment' for an application request submitted to the Customer test environment must be 'TEST'.

Only C2B Content (pain.001.001.02) may be sent to the Customer test environment. A validation check will be performed on the submitted Content and the related validation and receipt feedback files generated (first- and second-level feedback). The Customer test environment will not generate a feedback file of the type ‘paid-up’ (third-level feedback).