User Guidelines for the Corporate Web Services Channel and the Related Certificate Service of OP FINANCIAL GROUP

User guide
May 2018

1 Introduction

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

The guide provides advice on how to obtain and use the certificates required by the WSC of OP Financial Group. The system is referred to as the WSC Certificate Service or, in brief, the Certificate Service.

These guidelines describe the operations and message descriptions of both the WSC and Certificate Service. In addition, the document includes instructions on software implementation, and example content and messages that can be utilised in such implementation. The guide does not describe the content of payment transfer data or account reporting, for example, for which there are separate detailed descriptions.

1.1 Web Services channel

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

The WSC allows customer systems to upload and download payment transfer content to and from the Bank. Such content may be C2B payment transfer files, bank statements, e-invoice files, and the related notification messages.

The WSC message specifications were jointly issued by various bank consortia and are available free of charge on the website of Finance Finland (FFI) at www.finanssiala.fi.

The WSC authenticates the integrity and authenticity of 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 recipient must verify their signature. A public key, or, in practice, a certificate, is required for verifying and authenticating the sender's signature. The certificates are managed through the Certificate Service.

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 certificates used for the 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 – from the end user's perspective this means through the customer's information system.

Due to the user rights conferred by the certificate, at the beginning of a certificate's life cycle the customer must visit the Bank for verification of their identity, thus allowing the certificate to be linked securely to the WSC username. This first identity verification cannot be performed digitally.

1.3 Division of responsibilities

This guide does not contain any information on the client software; instead, it confines itself to describing the functions and operations required in the software used by the customer. The user will 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 use of the keys and certificates. Legal specification of duties and responsibilities is presented in the terms and conditions of the Web Services Channel Agreement.

1.4 Source material

The WSC common guidelines, i.e. message specification, are publicly available on the website of Financial Finland at www.finanssiala.fi.

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
Public key methods use two related keys: one public, one private. Combined, the keys form a key pair. The purpose of use of the keys is defined in the Certificate, which in turn is governed by the Certificate Policy.

CA
An organisation that issues certificates and is responsible for the generation of certificates; draws up a Certificate Policy and Certification Practice Statement describing its operations.

CA certificate
Certificate of certificate authority

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 used in asymmetric encryption in the Public Key Infrastructure. Data encrypted with a Public Key (e.g. with the RSA algorithm) can only be decrypted using the key pair's Private Key. When the holder of the Public Key is known, the electronic signature performed with the corresponding Private key can be verified. The holder of a Public Key can be reliably identified with a Certificate.

Root certificate
A root certificate is a Certificate issued by a root certificate authority itself and the top-most level of the certificate chain (so-called trust anchor). The root

certificate is always distributed to users separately from other certificates and via a different route. In many cases, it is included in the customer's software installation package.

Root certificate authority

The first and most trusted level in a hierarchical PKI certificate chain. The root certificate authority defines the Certificate Policies as well as technical and operative norms.

Application Request

An XML document by the name of ApplicationRequest in the WSC, incorporating the command data for the service requested by the client system from the Bank, any content related to the application request, and the digital signature required for authenticating the request.

Bank's service certificate

The Bank signs its response messages using a service certificate. The Bank can use several service certificates depending on the information system that signs the response message. The service certificate is issued by a different sub-CA than the one issuing customer certificates. The Bank may renew its service certificate without notifying its customers.

PKCS10

The standardised format of a certificate application request.

PKI

Public Key Infrastructure. A set of technical and administrative solutions that is used to create, manage, distribute, use, store, and revoke Public Key Certificates. The system also defines the controls and standards that Certification Authorities must comply with in their activities so as to ensure the compatibility, identifiability and availability of electronic certificates. PKI is based on the Public Key encryption algorithm.

Registration

The event in which the identity of the holder of a new certificate to be generated is verified. Registration ensures certainty about the identity of the certificate holder and that authorisations can be linked to the certificate.

Revocation

The permanent invalidation of a certificate by adding it to the revocation list. The systems that trust the certificate must revoke the certificate. In particular, a certificate is revoked in cases where the customer suspects, or becomes aware of, unauthorised access to the private key.

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 as part of the request. A concept forming part of 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. Within the WSC Certificate Service, the most important information is the Common Name (CN) – i.e. the username for which the certificate was issued.

Revocation List

CRL, Certificate Revocation List. A list electronically signed by a Certificate Authority that contains the serial numbers of revoked Certificates and the revocation reason codes.

Revocation Service

A Certificate Authority’s service responsible for revoking and placing Certificates on hold (temporary revocation).

Authentication service

OP Financial Group's service which produces customer certificates required in the WS channel and the related support functions, such as deactivation.

Certificate

A digital document (e.g. an XML document) whose primary purpose is to link a public key and the information on its holder to 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 Chain

With the Certificate Chain one can, by trusting the root certificate, verify the chain’s certificates and make a trust decision regarding an end user’s certificate. The Certificate Chain begins from a root certificate and ends at an end-user’s certificate. In a Certificate Chain, a higher placed certification authority accredits a lower placed Certification Authority by signing their Certificate. The Certificate of the highest placed CA (Root CA) is self-signed.

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.

Certification Authority

see CA Certification Authority

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. A revoked certificate cannot be reinstated. The customer must register a new certificate and submit a new certificate application request. (See Revocation)

Web Services Channel

Based on Web Services and SOAP standards, this bank service enables corporate customers' information systems to send to the bank and retrieve from the bank electronic data. It also enables the use of real-time services.

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. This signature is issued with a private key and authenticated with a public key.

X.509v3

The most commonly used ITU (International Telecommunication Union) standard for Public Key Infrastructure (PKI). X.509 defines the standard format of Public Keys, certificates, revocation lists, attribute certificates and certificate’s certificate chains. X.509 is an ITU recommendation, which is why PKI vendors have implemented the standards in various ways.

Private Key

The private part of the key pair used in Public Key Infrastructure for asymmetric encryption. This key is designated unambiguously to a specific party, thereby enabling it to be used for, for example, creating an electronic signature. Data encrypted using the Key Pair’s Public Key can be decrypted using the Private Key. In addition, it can be used for establishing a shared secret. In certain PKI algorithms, data encrypted using the Private Key can be decrypted using the Key Pair’s Public Key. Such algorithms include RSA, which is used by the Certificate Authority used in the WS channel.

2 General security principles

Key security targets within the Certificate Service are as follows:

1. Safekeeping and use of the private key must be arranged in such a way 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 '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 in a secure and reliable fashion. Such a procedure will ensure that the certificate is 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 must be up-to-date at all times. This refers to the Bank in particular, since the Bank uses these 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 critical to data security, but the three issues 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. This 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 key pair's generation and safekeeping.

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 key's generation must be sufficiently random to avoid vulnerability to replication. When implementing the application and generating the key pair, it must be ensured that the quality of the algorithm used in 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 its use.

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

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

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

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

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

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

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

In the case of the renewal of a valid certificate, the element CertApplicationRequest must be signed using the key for the existing certificate already in use. The element CertApplicationRequest.customerId must contain the 10-character WSC username.

In the event that the Customer's system submits a certificate application request for 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 previously generated one.

2.4 Certificate revocation and use of revocation list information

The Customer can revoke its certificate by calling 010 252 8470.

The 10-character WSC username or the serial number of the certificate to be revoked is required for such a revocation.

After its revocation, the certificate will be invalid in the WSC and cannot be reinstated. If the Customer starts to use a new certificate after revoking a certificate, the Customer must re-register at a bank branch and submit a new certificate application request through the WSC together with a transfer key.

The Bank publishes a certificate revocation list. The revocation list addresses can be found in the CRL Distribution Points field of trusted certificates (e.g. http://crl.op-palvelut.fi/crl/rootca/ and http://crl.op-palvelut.fi/crl/subca/ directories). This revocation list is updated at least once a day and is in force for three days at a time. Accordingly, the Customer's system must retrieve the revocation list and be up-to-date at all times. From the revocation list, the system must check the validity of its trusted certificates (CA certificate and the Bank's service certificates).

The Bank does not consent to any use of WSC Certificates other than for WSC purposes. This being so, the publication or accuracy of the revocation list intended for the Bank's internal use places the Bank under no obligation or liability.

3 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.

Operation of the WSC is based on the Security and Message Specification issued through cooperation between banks operating in Finland.

The preferred mode of connection for the WSC is an SSL-protected HTTPS connection over the public Internet. A digitally signed SOAP message forms the channel's transport unit. This message contains the XML document ApplicationRequest, which is the actual service request. The ApplicationRequest (i.e. the service request) is also 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. Content sent is saved on the Bank side, pending processing. Processing may generate response data, which the Customer's system must download separately.

Real-time services are implemented using a file transfer mechanism, in which the Customer's software uploads the Content to the Bank and immediately receives the real-time service's final response in the response message.

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.asiakastesti.op.fi/wsdl/MaksuliikeWS.xml

As the Customer uses the production username in the test environment, it must sign a WSC agreement with the Bank. For security reasons, the key pair and certificate used in the test environment are only to be used for testing. The Customer can request the transfer keys for the test environment through OP's customer service (Corporate and Payment Services).

3.1 Operations of the Web Services Channel

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.

Upon transmission, the WSC performs a format validation check on the Content and rejects any corrupted Content. The WSC does not save 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 Content item 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 requested by the Customer and downloadable Content generated by the Bank from the WSC.

When downloading, the Customer must specify the Content for download, citing the Content reference (FileReference). Content references will be shown upon listing the Content. Then the Customer can download Content by the Content reference. Within the response message sent to Content transmission, the Customer also receives the reference for Content sent to the channel.

Only one Content item at a time can be downloaded.

The Content is 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 Compression of content

We recommend that Content sent to the Bank be compressed at all times. In compliance with RFC 1952, the compression algorithm is GZIP. 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 when downloading Content from the Bank. Setting the value of ApplicationRequest.compression = 'true' in the download request will compress the downloadable Content.

3.1.4 Real-time services

The WSC currently provides the real-time services listed below.

Content type	Description
camt.060.001.02	Account transaction and balance queries, XML format
ORDER TU	Request for account statement re-generation
pain.001.001.02 TP4 PS01	POP urgent payment, schema version V02. The feedback is pain.002.001.02 TP4 PS01.
pain.001.001.03 TP4 PS01	POP urgent payment, schema version V03. The feedback is pain.002.001.03 TP4 PS01.
TP1 ES	Credit transfer between Customer's own accounts (real-time payment).
TP1 1SS	Account balance query
TP1 1VA	Currency account balance summary
TP1 2ST	Account transaction query
TP1 2SY	Accounts' extended balance summary
TP1 3ST	Current day account transaction statement query
TP4 PS01	POPS urgent payment

The real-time services utilise the uploadFile operation. The client software sends a request to the WSC in the ApplicationRequest.content element, using the name of the real-time service as the ApplicationRequest.fileType, e.g. "TP1 1SS".

Descriptions of the XML real-time services are provided in a separate account reporting guide available via OP eServices.

3.1.4.1 Balance query

The client software can make an account balance query.

The name and FileType of the service is TP1 1SS.

In the element ApplicationRequest.content, the client software makes a Base64-encoded application request containing the following:

$$TP1 1SS BranchCode AccountNumber X

where:

- the length of the branch code is 6 characters
- the length of the account number is 8 characters
- X is the character X.

The response to the balance query is returned in the ApplicationResponse.content element and has the following structure.

Name	Length	Description
Record sequence number	1	=1
Response type	1	1=OK, other=error*
Reserved for future use	3	
Transaction branch code	6	
Payment terminal code	2	
Transaction number	4	
Account holder	15	
Branch code	6	
Account number	8	
Date	6	ddmmyy
Balance	11	2 decimals
Balance prefix	1	+/-
Credit limit	11	2 decimals
Credit limit prefix	1	+/-
Funds available for withdrawal	11	2 decimals
Available funds prefix	1	+/-
Currency code	1	1=euro

3.1.4.2 Current day transaction statement query

The client software can request a current day bank statement for transactions not yet downloaded.

The name and FileType of the service is TP1 3ST.

In the element ApplicationRequest.content, the client software makes a Base64-encoded application request containing the following:

$$TP1 3ST BranchCode AccountNumber X

where:

- the length of the branch code is 6 characters
- the length of the account number is 8 characters
- X is the character 1, if all transactions of the day, including those already downloaded, are requested; in all other cases, the service returns only the new transactions for the WSC username (CustomerID) not yet downloaded.

Response message record descriptions

Records are separated from one another using the record separators. Each record ends with the 'carriage return' and 'line feed' symbols.

The basic record of the current day transaction statement

Field	Name	Format	Description
1	File identifier	AN1	S
2	Record identifier	AN2	00
3	Record length	N3	322
4	Version number	AN3	001
5	Account number	AN14	
6	Current day transaction statement no.	AN3	Blank
7	Query date		
	.1 Start date	N6	YYMMDD
	.2 End date	N6	YYMMDD
8	Generation time		
	.1 Current date	N6	YYMMDD
	.2 Time	N4	HHMM
9	Customer code	AN17	
10	Not in use	N6	
11	Not in use	AN19	
12	Not in use	N6	
13	Account currency code	AN3	ISO code
14	Account name	AN30	
15	Account limit	AN18	16 integers + 2 decimals
16	Account holder	AN35	
17	Bank name	AN40	
18	Not in use	AN40	
19	Not in use	AN30	
20	Not in use	AN30	
	TOTAL	322	

Field 4 indicates the software version used to generate the current day account statement.

Field 7 The start date and end date are identical, i.e. the query date.

Field 9 indicates the Customer ID assigned by the bank to the account holder and the specifier, if applicable.

(In the initial phase, the country code or the standard code, and the specifier remain blank).

- country code X(4) or .1 standard code X(4)
- Customer ID X(8) .2 customer specifier X(10)
- customer specifier X(5) .3 customer specifier X(3)

Field 15 indicates the limit of a checking account with a credit line. No limit is associated with the account, if the field contains only zeros. The field indicates the limit of a sub-account under a cash pool account.

The basic record of a transaction

Field	Name	Format	Description
1	File identifier	AN1	S
2	Record identifier	AN2	10
3	Record length	N3	188
4	Transaction generation time	N6	HHMMSS
5	Original archiving code	AN18	
6	Entry date	N6	YYMMDD
7	Value date	N6	YYMMDD
8	Payment date	N6	YYMMDD
9	Transaction code	AN1	1, 2, 3, 4
10	Posting description .1 Code .2 Description	 AN3 AN35	
11	Transaction amount .1 Prefix .2 Amount	 AN1 N18	 16 integers + 2 decimals
12	Receipt code	AN1	E = itemisations to be excluded from the current day transaction statement
13	Transfer method	AN1	
14	Payee/Payer .1 Name .2 Source of name data	 AN35 AN1	 space character, A, J, or K
15	Payee's account .1 Account number .2 Account changed data	 AN14 AN1	 space character, *
16	Reference	AN20	
17	Form number	AN8	
18	Level code	AN1	0
	TOTAL	188	

Field 5 indicates the archiving code, assigned by the bank which generated the transaction, by
which the original payment order can be traced. The archiving code indicates the date on which the bank processed the payment order, and the bank branch or system involved.

YYMMDD XXXXXXXXXXXX

^____________________________identifier

^_____________________________________ date

The identifier of the archiving code is bank-specific. The first two characters indicate the code of the relevant bank group.

Field 9 contains the transaction code whose values are:

1	=	credit to account
2	=	debit to customer account
3	=	correction of credit to account
4	=	correction of debit to account

Note! Any correction of a correction must be either transaction type 1 (credit to account) or 2 (debit to account).

Field 10 contains a posting description that indicates the service through which, or how, the transaction is being posted in the account-holding bank. The primary purpose of the posting description code is to enable automated posting of the account transactions in the Customer's own bookkeeping. Identifying codes are assigned to the transactions to be posted automatically, while generic codes are applied to all other transactions. The code values are common to all banks. The description texts are bank-specific.

The values of the posting description code are:

700	=	payment transfer service credit to customer account / debit to customer account
701	=	recurring payment service credit to customer account / debit to customer account
702	=	bill payment service debit to customer account
703	=	payment terminal service credit to customer account
704	=	direct debiting service / automatic payment service credit to customer account / debit to customer account
705	=	reference payment service credit to customer account
706	=	payment service debit to customer account
710	=	credit to customer account credit to customer account
720	=	debit to customer account debit to customer account
721	=	card payment debit to customer account
722	=	checking account debit to customer account
723	=	taxi/bus voucher debit to customer account
730	=	collection of fee debit to customer account
740	=	collection of interest debit to customer account
750	=	payment of interest credit to customer account
760	=	loan (including repayment, interest, and fee) debit to customer account
761	=	loan repayment debit to customer account

In the corrections, the codes are used both for credit and debit transactions.

Field 12 contains a receipt code, which indicates whether receipt information is to be provided on the bank statement, as a separate paper document, or as an itemisation of the individual transactions in binary form.

The receipt code values are:

space character	=	the Bank does not issue a paper receipt to the Customer .
E	=	an itemisation is linked with the transaction.
P	=	the Bank issues a paper receipt to the Customer.

Field 13 contains a transfer method code, assigned by the bank which received the payment order, indicating how the payment order was transferred to the Bank and where the original instruction is stored.

In sorting situations, the transfer method is used to determine the party to be contacted for additional information on the transaction. Where the transfer method value is A, the sorting request is in all cases addressed to the initiator of the payment instruction. The account-holding bank is to be contacted in all other cases.

The transfer method code values are:

A	=	the Customer has transferred the payment instruction in
binary form or		as a self-service transaction. The original payment order is with the Customer.
J	=	The transaction is generated by the Bank's system The reason for its generation is available at the system sorting point indicated by the archiving code.
K	=	The transaction is executed at a bank branch and saved by the Bank's employee. The payment order can be retrieved using the archiving code.

Field 14 contains the name of the counterparty to the transaction, where available. This information is not available for batch transactions.

The name is either the name of the payee, in the case of an individual payer transaction, or the name of the payer in the case of an individual payee transaction. The source of the name information is included only in transactions where the ‘Payee/Payer' information is present, and indicates the source of the Payee/Payer name forwarded.

The values for the 'source of name' are:

A	=	The name data originates from binary content submitted by the Customer, or is saved by the Customer through self-service.
J	=	The name data is retrieved from the Bank's register code.
K	=	The name data is saved by a bank employee at branches.

In a payer transaction, **Field 15** contains the payee's account number, included by the payer's bank upon the transfer of the transaction. The payer can use this data to check into which account the payment was made. The 'account changed' data is linked only to the payee's account number and indicates that the account originally provided by the payer has changed in the Bank's system.

The values for the 'account changed' data are:

space character	=	not changed
*	=	changed

Additional record of a transaction

Field	Name	Format	Description
1	File identifier	AN1	S
2	Record identifier	AN2	11
3	Record length	N3	
4	Type of additional information	AN2	
5	Supplementary information	ANnnn	
	TOTAL	8+nnn	

The additional record of the transaction comprises the first part, common to all additional records, and the additional information, whose length varies according to the type of additional information.

Open-ended message, type = 00			
5.1	Message - 1	AN35	
5.2	Message - 2	AN35	
...	....		
5.12	Message - 12	AN35	
	TOTAL	Max. 420	

Number of transactions, type = 01			
5.1	Number of transactions	N8	
	TOTAL	8	

Billing transaction data, type = 02			
5.1	Customer number	AN10	
5.2	Blank	AN1	
5.3	Invoice number	AN15	
5.4	Blank	AN1	
5.5	Date of invoice	AN6	YYMMDD
	TOTAL	33	

Card transaction data, type of additional information = 03			
5.1	Card number	AN19	
5.2	Blank	AN1	
5.4	Merchant's archiving reference	AN14	
	TOTAL	34	

Correction event data, type = 04			
5.1	The original archiving code for the transaction being corrected	AN18	
	TOTAL	18	

Currency transaction data, type of additional information = 05			
5.1	Equivalent value		
	.1 Prefix	AN1	
	.2 Amount	N18	16 integers + 2 decimals
5.2	Blank	AN1	
5.3	ISO currency code	AN3	
5.4	Blank	AN1	
5.5	Exchange rate	N11	4 integers + 7 decimals
5.6	Exchange rate reference	AN6	
	TOTAL	41	

Originator information, type = 06			
5.1	Originator information-1	AN35	
5.2	Originator information -2	AN35	
	TOTAL	70	

Additional information provided by the bank, type = 07			
5.1	Additional information-1	AN35	
5.2	Additional information-2	AN35	
...	....		
5.12	Additional information-12	AN35	
	TOTAL	Max. 420	

Payment purpose data, type = 08			
5.1	Payment purpose code	N3	
5.2	Blank	AN1	
5.3	Description of the payment purpose	AN31	
	TOTAL	35	

Name specifier data, type = 09			
5.1	Specifier of the Payee/Payer name	AN35	
	TOTAL	35	

Balance record

Field	Name	Format	Description
1	File identifier	AN1	S
2	Record identifier	AN2	40
3	Record length	N3	50
4	Query date	N6	YYMMDD
5	Balance at the time of query .1 Prefix .2 Amount	 AN1 N18	 16 integers + 2 decimals
6	Available funds .1 Prefix .2 Amount	 AN1 N18	 16 integers + 2 decimals
	TOTAL	50	

The notice record is forwarded to the Customer only if the query fails or if the data is not up-to-date due to disruptions in the service.

Field	Name	Format	Description
1	File identifier	AN1	S
2	Record identifier	AN2	70
3	Record length	N3	
4	Bank Group code	AN3	
5	Notice .1 Row-1 (e.g. cause of disturbance) … .6 Row-6	 AN80 AN80	
	TOTAL	Max 489	

3.1.4.3 Urgent payment

Real-time payment made to another financial institution.

The technical name, i.e. dataset type, of the service is TP4 PS01.

In the element ApplicationRequest.content, the client software makes a Base64-encoded application request containing the following:

Name	Length	Description
Control command	11	”$$TP4 PS01 ”
Bank branch of the Payer	6	5nnnnn
Payer's account number	8	
Payer's name	30	
Bank branch of the Payee	6	
Payee’s account number	8	
Payee’s name	30	
Amount to be transferred	14	Presented in pennies or cents, see below
Currency unit code	1	1 euro
Due date	10	dd.mm.yyyy; blank until further notice

Reference	20	Starting zeros to be filled in
Message	140	
Paper receipt to the Payer	1	"E", no receipt until further notice
Notification to the Payee	1	0 no notification 1 phone 2 fax 9 other
Contact details of the Payee	70	The contact details of the Payee in connection with a notification; in all other cases blank
Time stamp	15	Yymmddmmssnnn, unique identifier
Message version	1	”1”
User key sequence number	1	0 .. 9
Check	16	not in use, leading zeros to be filled in

Sample request for urgent payment transfer. In the example, space characters to be used in an actual request are represented by dots, to illustrate their number and position.

```
$$TP4.PS01.57803820021333Saku.Eeroila..................13934600001181Simo.Sammila..
................00000000000001127.11.201100000000000000001245......................
...................................................................................
...................................E0..............................................
........................11072714570000010000000000000000
```

Received input acknowledgement of urgent payment request

An input acknowledgement of an urgent payment request comprises two files: the input acknowledgement file and the end-of-event file ($$EOF) for the OP transaction. Such an input acknowledgement can also be the $$ERROR error message returned by the OP service, e.g. PERMISSION ERROR or NO RESPONSE FROM HOST. The client software must allow for a response time that is longer than usual, of up to 120 seconds (the transaction may be processed in another financial institution). If the OP service does not return an input acknowledgement or returns the $$ERROR - NO RESPONSE FROM HOST error message, the client software must alert the Customer to contact the Bank or check the success status of the urgent payment, by making a current day account statement request, for example. If a transaction corresponding to the urgent payment order is displayed on the account, this means that the urgent payment has been successfully executed.

The system calculates a unique MAC (Message Authentication Code) CheckSum for the input acknowledgement file, in compliance with the PATU standard (see the PATU system description issued by the Finnish Federation of Financial Services). The CheckSum is calculated using the user key, from the start of the input acknowledgement file up to the 'CheckSum' field, in a similar manner as for all other PATU messages (ESI, SUO, VAR, and PTE).

Name	Length	Description
Success status code	2	“00” Succeeded All other numerical values are errors and the explanation text provides

		the error reason, e.g. "REJECTED, INSUFFICIENT FUNDS".
Explanation text	80	Explanation text, in the language of the Customer
Archiving code	22	Included if operation succeeds; in all other cases blank
Time stamp	15	Yymmddmmssnnn
Message version	1	”1”
User key sequence number	1	0 .. 9
Check	16	Not in use, zeros to be added

3.1.4.4 Real-time payment – credit transfer between Customer's own accounts

The client software can execute a credit transfer between the Customer's own accounts.

The name and FileType of the service is TP1 ES.

In the element ApplicationRequest.content, the client software makes a Base64-encoded application request containing the following:

$$TP1 ES X vknro vtnro hknro htnro euroAmount message

where:

- X is the character X
- vknro is the branch to be debited, presented using 6 characters
- vtnro is the account number to be debited, presented using 8 characters
- hknro is the branch to be credited, presented using 6 characters
- htnro is the account number to be credited, presented using 8 characters
- euro amount is the amount to be transferred, presented in cents without a decimal separator, using max. 11 characters
- message is the message to be forwarded, presented using max. 70 characters enclosed in double quotes

Sample credit transfer where EUR 1,500 is transferred from account 500015-118 to account 500015-22228 with the message, Sample credit transfer:

$$TP1 ES X 500015 10000018 500015 20002228 150000 "Sample credit transfer"

Response message to credit transfer application request

Name	Length	Description
Record sequence number	1	1
Response type	1	1=OK, other=error*
Reserved for future use	3	
Transaction branch	6	
Payment terminal code	2	
Transaction number	4	
Account holder	15	
Date	6	ddmmyy
Branch code debited	6	

Account number debited	8	
Balance of the account debited	11	presented in cents without decimal separator
Balance prefix	1	+/-
Branch code credited	6	
Account number credited	8	
Reserved for future use	12	
Amount in euros transferred	11	presented in cents without decimal separator
Prefix	1	+
Currency code	1	1=euro

3.1.4.5 Cash pool account current day account statement query

The client software can send a current day account statement query concerning the balance, and the debit and credit transactions, of a cash pool account.

The name and FileType of the service is TP1 2KS.

In the element ApplicationRequest.content, the client software makes a Base64-encoded application request containing the following:

$$TP1 2KS BranchCode AccountNumber X

where:

- the length of the branch code is 6 characters
- the length of the account number is 8 characters
- X is the character X.

Response message to cash pool account current day account statement query

Name	**Length**	**Description**
Record sequence number	1	1
Response type	1	1=OK, other=error*
Reserved for future use	3	
Transaction branch	6	
Payment terminal code	2	
Transaction number	4	
Name of the account holder	15	
Account-holding branch code of the cash pool account	6	
Account number of the cash pool account	8	
Date	6	ddmmyy
Balance	13	2 decimals
Prefix	1	+/-
Current day debit transactions	13	2 decimals
Prefix	1	+/-
Current date credit transactions	13	2 decimals
Prefix	1	+/-
Currency code	1	1=euro

3.1.4.6 Request for account statement re-generation

The client software can request bank statement re-generation from the OP WSC.

The name and FileType of the service is ORDER TU.

The request is sent in the following form:

$$ORDER TU StartDate EndDate BranchCode AccountNumber

where:

- the start date is the start date of the bank statement period presented as yyyymmdd
- the end date is the end date of the bank statement period presented as yyyymmdd

- the length of the branch code is 6 characters
- the length of the account number is 8 characters

If the request succeeds, the service returns the response code 00 OK. The bank statement is re-generated according to the bank statement generation schedule and can be downloaded on the following morning.

3.1.5 Listing of Content

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

- The moment of saving 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, using the deleteFile operation, will not be shown in the listing (see 'Deletion of Content').

When generating a listing, please note that both the Content sent to the Bank by the Customer and Content available for the Bank for download by the Customer are shown on the Content list. By applying appropriate filters to the getFileList operation, the Customer's software can select the Content to be displayed in the list.

3.1.6 Deletion of Content

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

Within the WSC, via the deleteFile operation, the Customer can also delete any Content the Customer has sent to the Bank. Deletion of Content simply changes the status of the Content from 'WFP' to 'DEL'. Such a status change only denies entry of the Content for processing, with 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, this time slot is 30 minutes at a maximum.

This means that Content must be deleted fairly quickly after transmission, since Content being processed (with the status 'FWD') cannot be deleted or cancelled in the WSC. In the event of such an attempt, the WSC will return an error message.

The time during which 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 business days from 7am to 6pm at 30-minute intervals.

3.1.7 Administrator and authorisations

Authorisation related to payment transfer content is based on the Generator role for the 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 location number which is a parameter for the username. This administrator identifier – i.e. the location – 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, the related usernames and the certificates linked to the usernames.

3.2 Example messages and application requests

3.2.1 Request message

A sample 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_ag0md1SPzDjcLWHg" 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">MIIC9TCCA...z2nIv3xpHPU=</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_87p1SixC35qs3Lpk">
                  <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>ztKnhKXLpBQM/r3we3D0BdVeibE=</dsig:DigestValue>
               </dsig:Reference>
               <dsig:Reference URI="#Timestamp_MpXSne5nUJot8ltt">
                  <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>NRvpjFck2OEDAcgy0WxxV1WTz3w=</dsig:DigestValue>
               </dsig:Reference>
            </dsig:SignedInfo>
            <dsig:SignatureValue>UPzp6yAQ...6Od5+GRI0w==</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_2u1tu89DgKYG7uPe">
                  <wsse:Reference URI="#bst_ag0md1SPzDjcLWHg" 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 wsu:Id="Timestamp_MpXSne5nUJot8ltt" xmlns:wsu="http://docs.oasis-
         open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
            <wsu:Created>2011-08-16T11:42:28Z</wsu:Created>
            <wsu:Expires>2011-08-16T13:22:28Z</wsu:Expires>
         </wsu:Timestamp>
      </wsse:Security>
   </env:Header>
   <env:Body wsu:Id="Body_87p1SixC35qs3Lpk" 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>1313494952760</mod:RequestId>
            <mod:Timestamp>2011-08-16T14:42:28.031+03:00</mod:Timestamp>
            <mod:Language>FI</mod:Language>
            <mod:UserAgent>OP Client</mod:UserAgent>
            <mod:ReceiverId>OKOYFIHH</mod:ReceiverId>
         </mod:RequestHeader>
         <mod:ApplicationRequest
         xmlns:mod="http://model.bxd.fi">PD94bWwg...ZXF1ZXN0Pg==</mod:ApplicationRequest>
      </cor:downloadFileListin>
   </env:Body>
</env:Envelope>
```

3.2.2 Response message

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<S:Envelope xmlns:exc14n="http://www.w3.org/2001/10/xml-exc-c14n#"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
1.0.xsd" xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
secext-1.0.xsd" xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Header>
      <wsse:Security S:mustUnderstand="1">
         <wsu:Timestamp wsu:Id="_3" xmlns:ns11="http://docs.oasis-open.org/ws-sx/ws-
         secureconversation/200512" xmlns:ns10="http://www.w3.org/2003/05/soap-envelope">
            <wsu:Created>2011-08-16T11:42:29Z</wsu:Created>
            <wsu:Expires>2011-08-16T11:47:29Z</wsu:Expires>
         </wsu:Timestamp>
         <wsse:BinarySecurityToken wsu:Id="uuid_5ac774c6-d670-4168-be0f-084dcb8d92ac"
         EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-
         security-1.0#Base64Binary" ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
         200401-wss-x509-token-profile-1.0#X509v3" xmlns:ns11="http://docs.oasis-open.org/ws-
         sx/ws-secureconversation/200512" xmlns:ns10="http://www.w3.org/2003/05/soap-
         envelope">MIID2DCC...iuycKgsL6euA==</wsse:BinarySecurityToken>
         <ds:Signature Id="_1" xmlns:ns11="http://docs.oasis-open.org/ws-sx/ws-
         secureconversation/200512" xmlns:ns10="http://www.w3.org/2003/05/soap-envelope">
            <ds:SignedInfo>
               <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
               c14n#"/>
               <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
               <ds:Reference URI="#_5002">
                  <ds:Transforms>
                     <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                  </ds:Transforms>
                  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <ds:DigestValue>lkuQU09sgqWIp02wRR1BDxCrxyk=</ds:DigestValue>
               </ds:Reference>
               <ds:Reference URI="#_3">
                  <ds:Transforms>
                     <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                  </ds:Transforms>
                  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <ds:DigestValue>dz7uPSeuk9tmjOU777o6/+GczFE=</ds:DigestValue>
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>BDV8Ctp...8rc0GX95w==</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
                  200401-wss-x509-token-profile-1.0#X509v3" URI="#uuid_5ac774c6-d670-4168-be0f-
                  084dcb8d92ac"/>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S:Header>
   <S:Body wsu:Id="_5002">
      <ns2:downloadFileListout xmlns="http://model.bxd.fi"
      xmlns:ns2="http://bxd.fi/CorporateFileService">
         <ResponseHeader>
            <SenderId>1000000000</SenderId>
            <RequestId>1313494952760</RequestId>
            <Timestamp>2011-08-16T14:42:29.672+03:00</Timestamp>
            <ResponseCode>00</ResponseCode>
            <ResponseText>OK.</ResponseText>
            <ReceiverId>OKOYFIHH</ReceiverId>
         </ResponseHeader>
         <ApplicationResponse>PD94bWwgd...BvbnNlPg==</ApplicationResponse>
      </ns2:downloadFileListout>
   </S:Body>
</S:Envelope>
```

3.2.3 Application request getFileList

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
   <CustomerId>1000000000</CustomerId>
   <Timestamp>2011-08-15T09:48:31.177+03:00</Timestamp>
   <Status>NEW</Status>
   <Environment>TEST</Environment>
   <SoftwareId>soft</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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>sPNzEb+Mf5dchY5MTGq7GL1grEg=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>aIqreFNkxuy...nM4SXE8g==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIIC9TCCA...Iv3xpHPU=</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationRequest>
```

3.2.4 Application response getFileList

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationResponse xmlns="http://bxd.fi/xmldata/"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
   <CustomerId>1000000000</CustomerId>
   <Timestamp>2011-08-15T09:48:33.668+03:00</Timestamp>
   <ResponseCode>00</ResponseCode>
   <ResponseText>OK.</ResponseText>
   <FileDescriptors>
      <FileDescriptor>
         <FileReference>5802</FileReference>
         <TargetId>MLP</TargetId>
         <ParentFileReference>5801</ParentFileReference>
         <FileType>pain.002.001.02</FileType>
         <FileTimestamp>2011-07-29T12:00:16.483+03:00</FileTimestamp>
         <Status>NEW</Status>
      </FileDescriptor>
      <FileDescriptor>
         <FileReference>5803</FileReference>
         <TargetId>MLP</TargetId>
         <ParentFileReference>5801</ParentFileReference>
         <FileType>pain.002.001.02</FileType>
         <FileTimestamp>2011-07-29T12:01:16.971+03:00</FileTimestamp>
         <Status>NEW</Status>
      </FileDescriptor>
   </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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>LlpU5jyiDd5kO5FjJDIL7AWZyBQ=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>WKtQ1t8V1...LkGV9DMz0cQ==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIID1zCCAr...JKaoOlc5gLu</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationResponse>
```

3.2.5 Application request getFile

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
   <CustomerId>1000000000</CustomerId>
   <Timestamp>2011-08-15T13:01:46.911+03:00</Timestamp>
   <StartDate>2011-08-15+03:00</StartDate>
   <Environment>TEST</Environment>
   <FileReferences>
      <FileReference>5803</FileReference>
   </FileReferences>
   <Compression>true</Compression>
   <SoftwareId>soft</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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>OQA4fiudfd6KJKR0KINTsE9Fyxc=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>c2RzFUa...9VBAnMQ==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIIC9TC....v3xpHPU=</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationRequest>
```

3.2.6 Application response getFile

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationResponse xmlns="http://bxd.fi/xmldata/"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
   <CustomerId>1000000000</CustomerId>
   <Timestamp>2011-08-15T13:01:49.591+03:00</Timestamp>
   <ResponseCode>00</ResponseCode>
   <ResponseText>OK.</ResponseText>
   <Compressed>true</Compressed>
   <CompressionMethod>RFC1952</CompressionMethod>
   <Content>H4sIAAAA...epSdAwAA</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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>gQf1Tmlhw7KdS7MT10L5yaTDmm4=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>bzS0Itu...U/y6jRg==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIID1zCCA...oOlc5gLu</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationResponse>
```

3.2.7 Application request uploadFile

```
<ApplicationRequest xmlns="http://bxd.fi/xmldata/">
   <CustomerId>1000000000</CustomerId>
   <Timestamp>2011-08-15T13:01:31.990+03:00</Timestamp>
   <Environment>TEST</Environment>
   <TargetId>target</TargetId>
   <Compression>true</Compression>
   <SoftwareId>soft</SoftwareId>
   <FileType>pain.001.001.02</FileType>
   <Content>H4sIAAA...KU0HAAA=</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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>o9/bmBaH58Phw01oiQS/ttrP/sY=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>NwNRa...dTtMMqvg==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIIC9TC...nIv3xpHPU=</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationRequest>
```

3.2.8 Application response uploadFile

In the following example, a validation error has been detected in the pain.001.001.02 Content sent by the Customer.

```
<ApplicationResponse xmlns="http://bxd.fi/xmldata/"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:date="http://exslt.org/dates-and-times">
<CustomerId/>
<Timestamp>2018-03-16T17:14:38+02:00</Timestamp>
<ResponseCode>12</ResponseCode>
<ResponseText>Schema validation failed. - Tranid = 661232927</ResponseText>
<Compressed>false</Compressed>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
   <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
         <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <Reference URI="">
         <Transforms>
            <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
         </Transforms>
         <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
         <DigestValue>TlA6ACHFI9HVswrPCi6jhA10G14=</DigestValue>
      </Reference>
   </SignedInfo>
<SignatureValue>o9F1TZvdEFTeb09aBSf6TzGmCE/F09jd...S5YAiEGZtxvfR/FqO3i6u5P9VfK0cCy6czYqJs9Ew
==</SignatureValue>
   <KeyInfo>
      <X509Data>
         <X509Certificate>MIIGLzCCBBegAwIBAgIDKCf...POM88+Y+luwn7HmqB</X509Certificate>
         <X509IssuerSerial>
            <X509IssuerName>C=FI, CN=CUSTOMER TEST OP Services CA V2</X509IssuerName>
            <X509SerialNumber>2631673</X509SerialNumber>
         </X509IssuerSerial>
      </X509Data>
   </KeyInfo>
</Signature>
</ApplicationResponse>
```

The service returns the following message in the event of another type of schema error.

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationResponse xmlns="http://bxd.fi/xmldata/"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
   <CustomerId>1000000000</CustomerId>
   <Timestamp>2011-08-15T13:01:34.851+03:00</Timestamp>
   <ResponseCode>12</ResponseCode>
   <ResponseText>Schemavalidation failed.</ResponseText>
   <FileType>pain.002.001.02</FileType>
   <Content>PD94bWw...dW1lbnQ+Cg==</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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>3GyOY2gXwgT7RFP8CIli4KQ5kcg=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>cBs4Lm...QvD1Q==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIID1zC...aoOlc5gLu</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationResponse>
```

In this second error example, the Application Request.content element contains the following pain.002.001.02 data in Base64-encoded form. Further information on the Content and usage of payment feedback of this kind is available in the separate customer instructions concerning C2B payments.

```
<?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>1313401940313</MsgId>
<CreDtTm>2011-08-15T12:52:20+03:00</CreDtTm>
</GrpHdr>
<OrgnlGrpInfAndSts>
<NtwkFileNm>1313401937067</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: 'A1001.00' is n</AddtlStsRsnInf>
<AddtlStsRsnInf>ot a valid value for 'decimal'.cvc-complex-type.2.2: Element 'InstdAmt' must
have no element [children],</AddtlStsRsnInf>
<AddtlStsRsnInf>and the value must be valid.</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>2011-08-15T09:53:53.778+03:00</Timestamp>
   <StartDate>2011-08-15+03:00</StartDate>
   <Environment>TEST</Environment>
   <FileReferences>
      <FileReference>6152</FileReference>
   </FileReferences>
   <SoftwareId>soft</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="">
            <Transforms>
               <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            </Transforms>
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <DigestValue>TsZYDgKXMO6/nfTlGGFGlHL43pI=</DigestValue>
         </Reference>
      </SignedInfo>
      <SignatureValue>dgUhp4b...qelFFvQ==</SignatureValue>
      <KeyInfo>
         <X509Data>
            <X509Certificate>MIIC9TCCAd2g...Iv3xpHPU=</X509Certificate>
         </X509Data>
      </KeyInfo>
   </Signature>
</ApplicationRequest>
```

3.2.10 Application response deleteFile

```
<?xml version="1.0" encoding="UTF-8"?>
<ApplicationResponse xmlns="http://bxd.fi/xmldata/"
xmlns:ns2="http://www.w3.org/2000/09/xmldsig#">
     <CustomerId>1000000000</CustomerId>
     <Timestamp>2011-08-15T09:53:56.147+03:00</Timestamp>
     <ResponseCode>00</ResponseCode>
     <ResponseText>OK.</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="">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                         signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>F4NXYMUcrwJ83p92msZ48Jga7+c=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>OUjhFKVG...qL5xb4MQ==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIID1zCC...aoOlc5gLu</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</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 PKI key pair and certificate issued by the Certificate Service of OP Financial 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 (agreement on the use of Web Services). Registration is performed by a Bank employee and always correlates to a specific WSC username.

In conjunction with 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, comprising eight characters.

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

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

4.1.2 Generation of the key pair

The key length must be 2048 bits, and the algorithm RSA; the message digest algorithm for the signature is sha256RSA.

The key pair must be generated with an algorithm and method that will ensure sufficient 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 used, the private key will be generated inside the security module and cannot be removed during the course of regular usage. If the private key is not contained in a security module, the key must at least be 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 its use and safekeeping, as well as for any unauthorised use.

In the event that the Customer suspects, or becomes aware of, unauthorised access to, or use of, the private key, the Customer must immediately revoke the certificate linked with the key, through the Bank's blocking service.

4.1.4 Submission of a certificate application request and certificate creation

The Customer's software submits a certificate application request to the Bank's WSC Certificate Service. For this operation, the software requires the 16-character transfer key entered by the Customer and the 10-character username activated for the WSC Agreement. The Customer's software sends the certificate application request to the Certificate Service and receives the customer certificate in the response message.

The public key is used for creating a certificate application request in PKCS 10 format.

The following two items, and only these items, should be entered in the subject field of the certificate application request:

```
C=FI

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

Certificate application requests vary widely, and identification and authentication by the Bank's Certificate Service is case-specific.

The CertApplicationRequest.content element must contain the binary certificate application request in PKCS 10 format (DER).

The first certificate application request made without a certificate is based on registration performed at the Bank, i.e. on a transfer key; the 16-character transfer key must also be included in the CertApplicationRequest.transferKey element. In such as case, the CertApplicationRequest and SOAP message do not require a signature.

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 using the key corresponding to the certificate issued for the username, for which a certificate is being sought in this case. No signature is required for the SOAP message.

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. No signature is required for the CertApplicationRequest and SOAP message.

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

It is imperative that, upon submitting a certificate application request, the Customer's software check the SSL certificate of the Bank's Certificate Service issued for the domain wsk.op.fi. Such a check would ensure 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 via the WSC.

Signature is performed with the private key. The signing system must include the certificate corresponding to the private key alongside the signature. The certificate contains the public key which the receiver uses 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 WSC certificates, the Common Name information in the subject field of the certificate bearing the WSC username is the identifier for the holder.

4.1.6 Certificate life cycle and renewal

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

The customer certificate is valid for a maximum of two years. To ensure uninterrupted traffic, the Customer must renew the certificate before the expiry of the prior certificate. The certificate can be renewed no sooner than 60 calendar days before the expiry of the valid certificate. In the event that the certificate lapses before a new certificate is obtained, the Customer must restart the registration process from the beginning, i.e. obtain new transfer keys from the Bank.

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 can easily detect 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 in the certificate renewal message.

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 using the private key for which the username holds a valid certificate. Thus, in the Certificate

Service, the verification of the renewal request's authenticity is based on the immediately preceding certificate for the username, valid at the time of the request's submission.

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

The revocation list address can be found in the trusted certificate's CRL Distribution Points field.

4.1.8 Premature expiry of the certificate

In the event that the Customer suspects, or becomes aware of, unauthorised access to the Customer's private key, the Customer must revoke the certificate without delay via the telephone number 010 252 8470 or by contacting the Customer's own bank branch.

Please see the revocation instructions referred to 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.asiakastesti.op.fi/wsdl/MaksuliikeWS.xml

The production environment username is used in the test environment, but, for security reasons, the key pair and certificate are issued for test purposes only.

4.2.1 SOAP Messages and WSDL

The WSDL file describes the SOAP message structure.

The SOAP message does not require a signature within the Certificate Service. Authenticity is verified by signature at 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 WSDL for the Certificate Service is available at:
https://wsk.op.fi/wsdl/MaksuliikeWS.xml

The Customer Test Environment WSDL for the Certificate Service is available at:
https://wsk.asiakastesti.op.fi/wsdl/MaksuliikeCertService.xml

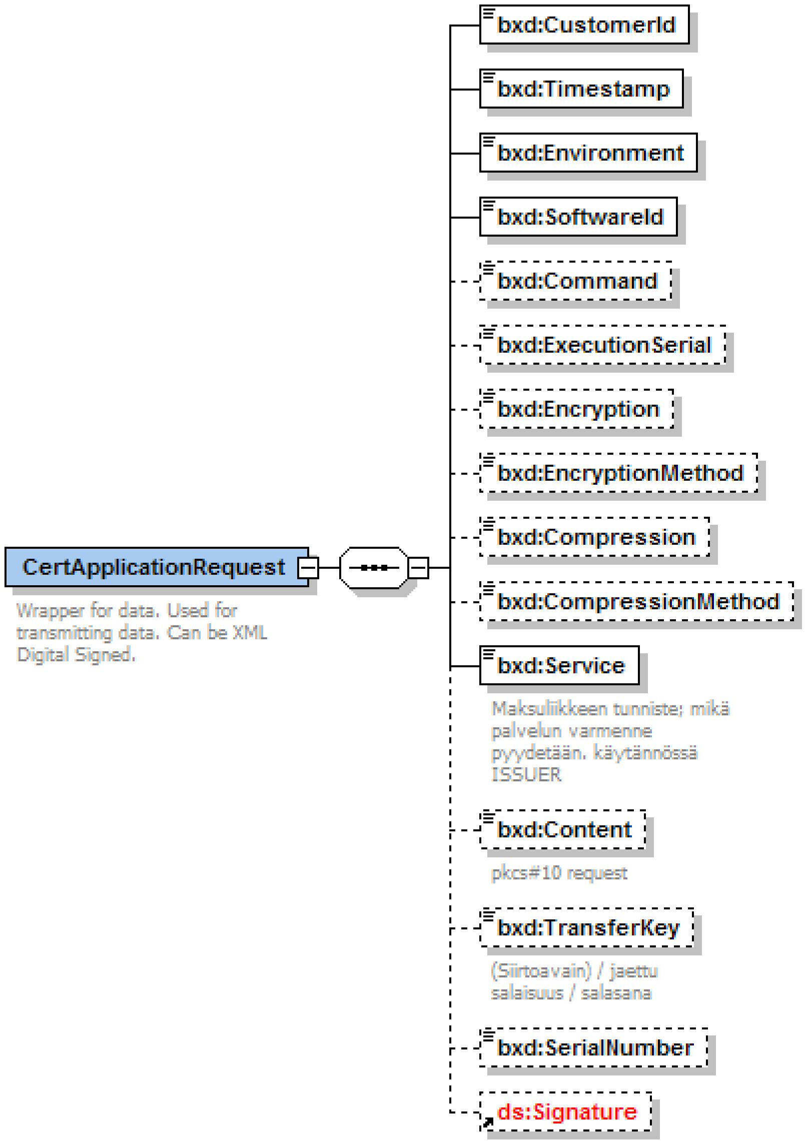
The schema files are available at:

https://media.op.fi/media/certapplication/CertApplicationRequest_200812.xsd

https://media.op.fi/media/certapplication/CertApplicationResponse_200812.xsd

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

4.2.2.1 CertApplicationRequest



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

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

Content – certificate application request in PKCS10 format, Base64-encoded

TransferKey – transfer key (16 characters) in the case of submission of the first certificate application request, under the username in question

Signature – XML signature in the case of certificate renewal

In addition, there are a few mandatory items:

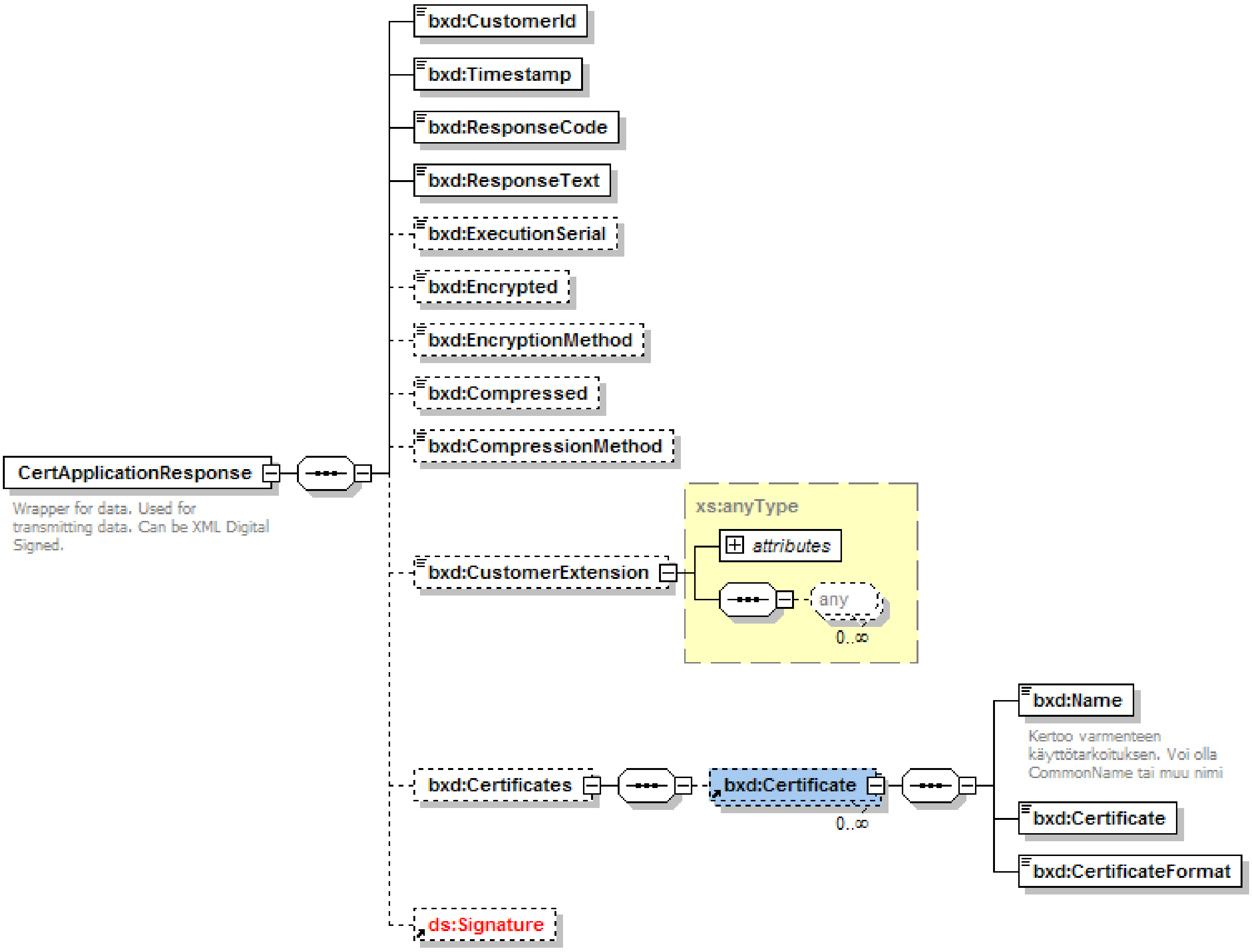
Timestamp – timestamp for the generation of the application request (in most cases, in support of sorting only)

Environment – in the production environment case, 'PRODUCTION' (otherwise, the request will be rejected) In the Customer test, the form 'TEST' is used in the field.

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

Service – MATU

4.2.2.2 CertApplicationResponse



4.3 Sample Content for the Certificate Service

4.3.1 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>
```

4.3.2 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>
```

4.3.3 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 are 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="">
                    <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>
```

4.3.4 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="">
                        <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>
```

4.3.5 Certificate application request with a 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>
```

4.3.6 Application response to certificate application request with a 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="">
                    <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>
```

4.3.7 Application request for certificate retrieval with a 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>
```

4.3.8 Application response to certificate retrieval request with a 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="">
        <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>
```

4.3.9 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>
```

4.3.10 Application response to retrieval request for service certificates

```
<?xml version="1.0" encoding="UTF-8"?>
<CertApplicationResponse xmlns="http://op.fi/mlp/xmldata/"xmlns:ns2=
"http://www.w3.org/2000/09/xmldsig#">
     <CustomerId>1000000047</CustomerId>
     <Timestamp>2018-0319T09:43:33.504+02:00</Timestamp>
     <ResponseCode>00</ResponseCode>
     ResponseText>OK.</ResponseText>
     <Certificates>
          <Certificate>
                <Name>CN=CUSTOMER TEST OP-Pohjola Services CA, C=FI</Name>
                <Certificate>MIIGIDCCBAigAwI...kVj8Sv1dNBrnd52LISFjx2wCXud</Certificate>
                <CertificateFormat>X509v3</CertificateFormat>
          </Certificate>
          <Certificate>
                <Name>CN=CUSTOMER TEST OP-Pohjola WS CA, C=FI</Name>
                <Certificate>MIIGGjCCBAKgAwIBAgIDAT5bMA0G...dMwP+ujyr/EoHCNOrGcpAs</Certi
                ficate>
                <CertificateFormat>X509v3</CertificateFormat>
          </Certificate>
          <Certificate>
                <Name>C=FI, CN=CUSTOMER TEST OP Services CA V2</Name>
                <Certificate>MIIGGzCCBAOgAwIBAgIDKCJGMA0GCSqGS...3U+YS9431RzBqGk48uE5KSxAcUZ
                vLnc6372j0a7WsISQ==</Certificate>
                <CertificateFormat>X509v3</CertificateFormat>
          </Certificate>
          <Certificate>
               <Name>C=FI, CN=CUSTOMER TEST OP WS CA V2</Name>
               <Certificate>MIIGFTCCA/2gAwIBAgIDKBo1M...tkoEmxWW1K8rootLAROAf+a
               2K13wgSwOA==</Certificate>
               <CertificateFormat>X509v3</CertificateFormat>
          </Certificate>
     </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="">
                    <Transforms>
                         <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
               signature"/>
                    </Transforms>
                    <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                    <DigestValue>VyXRntiU4/X/h1GOGj0Tjtt7wlc=</DigestValue>
               </Reference>
          </SignedInfo>
          <SignatureValue>RR5AfAz0Rt7NPUQnnTJA0IuRUtZ9cQUIZRq0DN....sp
          ViIxA==</SignatureValue>
          <KeyInfo>
               <X509Data>
                    <X509Certificate>MIIGKjCCBBKgA...HsHt8Os4G7ov7mhKYQ==</X509Certificate>
               </X509Data>
          </KeyInfo>
     </Signature>
</CertApplicationResponse>
```

5 Customer test environment and tests

OP provides a test environment for both client application developers and WSC payment services customers. Using the test environment requires that the software developer has read OP's instructions and, based on these, executed the data to be sent and retrieved through the WS channel. OP does not offer a productised service for supporting the construction of WS channel's bank connection software. The use of the test environment requires a production WSC agreement with the Customer's account-holding bank.

The test environment has functionalities which are similar to those of the actual service. On some occasions, it features a newer version of the service. Within the test environment, usernames and certificates are for testing purposes only. The functions performed in the test environment are technical validation of the Content payload files and generation of feedback as in the production environment.

The Customer's actual WSC usernames, payment identifiers and account numbers are used in the test environment, which means that the Customer also needs production agreements for the services. The only production identifier not used, for security reasons, is the actual key and associated certificates. The test environment is intended for validation of the bank connectivity application and other software, prior to the deployment of new services. Developer Test Environment wsk.asiakastesti.op.fi.

The test environment allows processing of the business payload in addition to providing WSC functionality. For further information on the payload processing available in this environment, please consult the separate Customer Guidelines.

5.1.1 Requesting test usernames and other codes

The usernames, SenderIDs and CustomerIDs, are the same both in the test and production environments. For security reasons, however, separately issued key pairs and certificates must be used in order to prevent access to the WSC from the customer test environment.

The Customer can request the transfer keys for using the test environment in conjunction with signing the Web Services Channel Agreement or at a later time of its choice. The transfer key can be requested later via OP's customer service (Corporate and Payment Services).

5.1.2 Address of the test environment and location of files

The test environment WSC WSDL file is available at:

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

The test environment Certificate Service WSDL file is available at:

https://wsk.asiakastesti.op.fi/wsdl/MaksuliikeCertService.xml

5.1.3 Obtaining the test certificate

The client software will retrieve the certificate from the WSC Customer Test Environment via the WSC Certificate Service interface. This interface is described above.

The client software generates the key pair as described in the section 'WSC Certificate Service' in this document. The client software generates a certificate application request based on the public key and executes the request, addressed and routed to the Customer Test Environment of the WSC Certificate Service.

In the application response, the Customer Test Environment of the WSC Certificate Service returns a certificate correlating to the Customer's test username.

6 FAQ

Where can I find the valid certificates used in the WSC? Certificate Service website www.op.fi/varmennepalvelu

How can I request user identifiers for the customer test environment? Use of the customer test environment requires a production WSC agreement, which is concluded with the bank. Once the agreement is in place, the test transfer keys can be requested from OP's customer service (Corporate and Payment Services, tel. 0100 05151)

Where can get information on upcoming changes to the WS channel? Software suppliers are advised to follow the Information to software suppliers page on op.fi (https://www.op.fi/corporate-customers/information-to-software-suppliers). The 'Online connections' section provides general WS channel information and 'Service notice on payment transactions' updates on any disturbances.