

XML FORMAT

PosConnexion

SWISH API

POS Application

Release	Date	Author	Modification
Rel. 1.23	2021104	Peter Andersson, Infrasec Sweden	New Version

Table of contents

TABLE OF CONTENTS	2
1 SCOPE	5
2 OVERVIEW	5
3 NORMATIVE REFERENCES	5
4 THE POSCONNEXION API	6
4.1 XML PosCONNEXION REQUEST EXAMPLE.....	6
4.2 XML PosCONNEXION RESPONSE EXAMPLE	6
4.3 GENERAL HEADING ATTRIBUTES.....	7
4.3.1 ApplicationID.....	7
4.3.2 RequestID	7
4.3.3 OrgNr.....	7
4.3.4 RegisterID	7
4.3.5 DateTime.....	7
4.4 GENERAL SERVICE ATTRIBUTES	7
4.4.1 ServiceID	7
4.4.2 ServiceVersion	7
4.4.3 ServiceFormat	7
4.4.4 ServiceEncoding	7
4.4.5 ServiceData	7
4.4.6 Internal: ServiceForward	7
4.5 GENERAL SERVICE RESPONSE ATTRIBUTES.....	8
4.5.1 ServiceID	8
4.5.2 ResponseCode	8
4.5.3 ResponseMessage	8
4.5.4 ResponseReason	8
4.5.5 ResponseAlert	8
4.5.6 ServiceData	8
4.5.7 Internal: ServiceForward	8
4.6 GENERAL SERVICE FORWARD ATTRIBUTES.....	8
4.6.1 Internal: ServiceForwardID	8
4.6.2 Internal: ServiceForwardData.....	8
5 POSCONNEXION SWISH CORPORATE	9
5.1 SERVICE PROVIDER	10
5.1.1 Swedbank Provider - “Swish I kassa”	10
5.1.2 Getswish Provider - “Swish Handel”	10
5.2 PosCONNEXION SERVICE REQUEST SWISH CORPORATE	11
5.2.1 ServiceID	11
5.2.2 ServiceVersion	11

5.2.3	ServiceFormat	11
5.2.4	ServiceEncoding	11
5.2.5	ServiceData	11
5.3	REFUNDPAYMENT NOTE	12
5.4	LISTPAYMENTS NOTE	12
5.5	SWEDBANK SWISHNUMBER LIST NOTE	12
5.6	GETSWISH CUSTOMER SWISH NUMBER NOTE	12
5.7	SWISHC API TAGS	13
6	SWISH CORPORATE XML EXAMPLES.....	14
6.1	SWISH CORPORATE XML REQUEST EXAMPLE	14
6.2	SWISH CORPORATE XML RESPONSE EXAMPLES	14
6.3	SERVICEDATA: REGISTER SWISH NUMBER.....	15
6.3.1	REGISTER: ServiceData Request	15
6.3.2	REGISTER: ServiceData Response	15
6.4	SERVICEDATA: UNREGISTER SWISH NUMBER	15
6.4.1	UNREGISTER: Request	15
6.4.2	UNREGISTER: Response	15
6.5	SERVICEDATA: QUERY REGISTERED INFORMATION	15
6.5.1	STATUS: Request	15
6.5.2	STATUS: Response	16
6.6	SERVICEDATA: LIST AVAILABLE ACTIONS.....	16
6.6.1	LISTACTIONS: Request	16
6.6.2	LISTACTIONS: Response	16
6.7	SERVICEDATA: INITIATE PAYMENT	17
6.7.1	INITIATEPAYMENT: Request.....	17
6.7.2	INITIATEPAYMENT: Response Ok	17
6.7.3	INITIATEPAYMENT: Response With Error	18
6.8	SERVICEDATA: WAIT FOR PAYMENT AND OTHER NOTIFICATIONS.....	18
6.8.1	WAITPAYMENT: Request.....	18
6.8.2	WAITPAYMENT: Response Ok	19
6.8.3	WAITPAYMENT: Response TimeOut	19
6.9	SERVICEDATA: LIST UNCONFIRMED PAYMENTS	20
6.9.1	LISTPAYMENTS: Request	20
6.9.2	LISTPAYMENTS: Response	20
6.10	SERVICEDATA: LIST CONFIRMED PAYMENTS.....	21
6.10.1	LISTCONFIRMED: Request	21
6.10.2	LISTCONFIRMED: Response	21
6.11	SERVICEDATA: EXECUTE REFUND REQUEST.....	22
6.11.1	REFUNDPAYMENT: Request.....	22
6.11.2	REFUNDPAYMENT: Response.....	22
6.12	SERVICEDATA: CONFIRME PAYMENT	23
6.12.1	CONFIRMPAYMENT: Request	23
6.12.2	CONFIRMPAYMENT: Response.....	23
6.13	SERVICEDATA: UNCONFIRM PAYMENT.....	23
6.13.1	UNCONFIRMPAYMENT: Request	23
6.13.2	UNCONFIRMPAYMENT: Response.....	23

6.14	EMULATION OF SWISH APP PAYMENT (PosCONNEXION TEST AND VERIFY)	24
6.14.1	SIMULATEMPAYMENT: Purchase	24
6.14.2	SIMULATEMPAYMENT: Refund	24
6.15	INVESTIGATING MISSING PAYMENTS	25
6.15.1	FETCHCONFIRMATION: Request	25
6.15.2	FETCHCONFIRMATION: Response	25

1 Scope

This document describes the PosConneXion SWISH API and is aimed for POS Application Vendor / Partner developers that enable provided SWISH services in their POS Application. The PosConneXion API supply Retailers/Merchants an easy access to enroll a specific POS in the Infrasec PosConneXion Service Cloud.

Current PosConneXion API services:

This release

- Central Tax Control Unit
- Swish Handel for GetSwish and Swedbank Swish Corporate “Swish I kassa”
- Distributed Order with Clearing Services (Pilot)
- Distributed Voucher Balance with Clearing Services (Pilot)

Pipeline

- Distributed Electronic Receipt Storage including distributed receipt garbage management

2 Overview

The POS uses a unique register identity when communicating with the Infrasec PosConneXion Services Cloud that connects the POS instance with the requested services.

The process of registration and management of a POS / Register to the PosConneXion services is called enrollment. The best suitable method for a specific POS Application to register its organization, administrators, stores and register is decided in cooperation with Infrasec.

Enrollment can be executed either through IDM system GUI or API by a POS Partner LRA Administrator/User or it can be delegated to trusted enrollment parties acting with credentials from the POS Partner.

Once a POS / Register is enrolled in can execute all services it is enrolled for using the credentials required from the enrollment. When a POS / Register includes a new service the PosConneXion enrolled credentials are used to access the new services in and through the Infrasec PCX cloud.

The various PosConneXion Services a POS / Register requests are bound through this common API. The different services require information relevant to the target service. This means that the POS Application supply different information depending on the service it requests.

The API supports multiple services in one request. This means that the POS Application may send one request that contains both a request to the Central Control Unit and one to the Electronic Receipt Service. Since the Electronic Receipt depends on the success of the transaction to the Central Control Unit the CCU ID and Control Code on completion then is forwarded to the Electronic Receipt Service.

3 Normative references

X.509

Infrasec PosConneXion API

Infrasec Enrollment API

Infrasec CCU API

Infrasec Voucher API

Infrasec Swish API

4 The PosConneXion API

The PosConneXion API is built to handle provided services through a generic list of extensions. A Request must always contain a header followed by a list of details for requested services. The Response always contains a header with general error information and a list of services holding details for concerned services.

4.1 XML PosConneXion Request Example

The Request contains a header section with general information and a list of extensions:

```
<posConneXionRequest>
  <ApplicationID>ACME POS v1.0</ApplicationID>
  <RequestID>806e589fc9506474</RequestID>
  <OrgNr>1212121212</OrgNr>
  <RegisterId>EX11212121212101</RegisterId>
  <DateTime>20170131142620</DateTime>
  <posConnexionServiceList>
    <posConnexionService>
      <ServiceID>CCU</ServiceID>
      <ServiceVersion>1.0</ServiceVersion>
      <ServiceFormat>application/xml</ServiceFormat>
      <ServiceEncoding>base64</ServiceEncoding>
      <ServiceData>BASE64XXX</ServiceData>
    </posConnexionService>
  </posConnexionServiceList>
</posConneXionRequest>
```

4.2 XML PosConneXion Response Example

The response contains a header section including eventual error information and a list of extensions:

```
[
<posConneXionResponse>
  <TransactionID>18198178230534683852</TransactionID>
  <ApplicationID>ACME POS v1.0</ApplicationID>
  <RequestID>806e589fc9506474</RequestID>
  <OrgNr>1212121212</OrgNr>
  <RegisterId>EX11212121212101</RegisterId>
  <ResponseCode>0</ResponseCode>
  <ResponseMessage>Success</ResponseMessage>
  <ResponseReason></ResponseReason>
  <posConnexionServiceList>
    <posConnexionService>
      <ServiceID>CCU</ServiceID>
      <ResponseCode>0</ResponseCode>
      <ResponseMessage>Success</ResponseMessage>
      <ResponseReason></ResponseReason>
      <ServiceData>BASE64XXX</ServiceData>
    </posConnexionService>
  </posConnexionServiceList>
</posConneXionResponse>
```

4.3 General Heading Attributes

The PosConneXion API heading attributes provides a general set of attributes that are common for all requests. The heading attributes are used to inform PosConneXion about the id of the calling application with a unique trace/debug request identification, the unique register identity with its organization number and the datetime the request is sent from the calling application.

4.3.1 ApplicationID

Application ID is an information that represents the calling application. This is normally the application identification and version info of the POS / Register issuing the request. Information is returned in response.

4.3.2 RequestID

Request Id is information that marks this request individually. This is normally a sequence number or a guid that uniquely identifies this request. Information is returned in response.

4.3.3 OrgNr

Organization number of the merchant / store responsible for the register issuing the request.

4.3.4 RegisterID

Registered register identity of the POS / Register issuing the request

4.3.5 DateTime

The date and time retrieved and sent by the POS / Register when the request is issued.

4.4 General Service Attributes

The PosConneXion API also provides a general set of attributes that are common for all services and identify the particular service that is requested and how the provided data is encoded. The Information that is specific to the particular service is passed as a base64 encoded object.

4.4.1 ServiceID

This field identifies the requested service that shall process this entry. The information is authorized that the POS / Register is allowed to communicate with the requested service, determines endpoint

4.4.2 ServiceVersion

This field defines the version the client executes, determines endpoint

4.4.3 ServiceFormat

This field is application/xml (Certain services supports application/json)

4.4.4 ServiceEncoding

This field is base64

4.4.5 ServiceData

This field contains a base64 encoding of the data to be processed by the requested service.

4.4.6 Internal: ServiceForward

This section is for PosConneXion Service Providers as described in General Service Forward Attributes

4.5 General Service Response Attributes

The PosConneXion API provides a general set of attributes that are common for all services. Information that is specific to the particular service must be passed as a base64 encoded object.

4.5.1 ServiceID

This field identifies the requested service that has processed this entry.

4.5.2 ResponseCode

This field identifies a numerical value for the success of this request. Value 0 means success

4.5.3 ResponseMessage

This field will contain a textual explanation corresponding to the response code above, Value Success is ok

4.5.4 ResponseReason

This may contain a textual explanation of the reason a failing request.

4.5.5 ResponseAlert

4.5.6 ServiceData

This field contains a base64 encoding of the data to be processed by the requested service.

4.5.7 Internal: ServiceForward

This section is for PosConneXion Service Providers as described in General Service Forward Attributes

4.6 General Service Forward Attributes

This section is for PosConneXion Service Providers as described in General Service Forward Attributes

4.6.1 Internal: ServiceForwardID

This identifies the target ServiceForwardID this information is allowed to be forwarded to. Information in this field is made available to PosConneXion Service Providers.

4.6.2 Internal: ServiceForwardData

This field contains base64 of data forwarded to target service identified by the ServiceForwardID above. Information in this field is made available to PosConneXion Service Providers.

5 PosConneXion Swish Corporate

The PosConneXion Swish Corporate API is designed to provide a common interface to the POS Application for Swish payments Independent that the end merchant is running the POS application using SWEDBANK (Swish I kassa) or GETSWISH (Swish Handel) Swish payment service. The provided functionality vary between the SWEDBANK implementation that it supports detection of uninitiated payments. The GETSWISH functionality requires that each payment is initiated with a Getswish payment token before it can be paid upon. This means that the POS Application must call the PosConneXion Swish Corporate API with a INITIATEPAYMENT request that is required when the service method is GETSWISH, optional with SWEDBANK.

These payment flows work without the requirement that the end customer must give away his or her mobile number to POS application cashier. The payment is initiated by the POS application and a QR code is provided to the end customer to be paid upon. The end customer mobile number is in this case not required to invoke the Swish payment. With GetSwishj a payment can be directed directly to a customer mobile and the QR code may be omitted.

When integrating this API the flow to manage individual Swish payments that they will work for both SWEDBANK (Swish I kassa) and GETSWISH (Swish Handel) implementation the POS Register Application should execute the action INITIATEPAYMENT before WAITPAYMENT. GETSWISH implementation requires a Getswish payment token that will be created by the INITIATEPAYMENT and provided in the QR Code to be provided to the Swish app to be paid upon. The SWEDBANK implementation will work even without the INITIATEPAYMENT but the call is normally used where to create a payment QR Code that provided to the Swish app to be paid upon.

The QR Code is returned by the API if the `<SwishQrCode></SwishQrCode>` tag is supplied in the INITIATEPAYMENT request. If the call is executed without the `<SwishQrCode>` tag no QR code is returned.

The INITIATEPAYMENT request returns the data string representing the content of the QR code in the `<SwishQrData>` tag. This data string can be used if the implementation wishes to create the QR Code outside the scope of this API. The `<SwishQrData>` contains the string that is the actual payment initiator for the payment.

To distinguish different payments, it is convenient and we recommend that the payment message is unique for every payment. Often a random or sequence number can be part of the user message. With Swedbank service it is possible to execute the payment flow from static QR Codes used in kiosk like shops or in Queue busting scenarios. To separate and select between different payments in this case the LISTPAYMENTS function is recommended.

When a payment is managed by the POS Application, we recommend that it is confirmed using the CONFIRMPAYMENT action that the payment will be marked as handled on not show up in succeeding LISTPAYMENTS. To list confirmed payments the LISTCONFIRMED is used.

The PosConneXion Swish Corporate API links a POS / Register to the Swish Corporate Services. The merchant store organization must register their organization with their Bank to Swish to acquire or link a corporate/merchant Swish number to their company. The acquired/linked Swish number must then be registered to a RegisterID in Infrasec IDM management system via IDM GUI, IDM Excel-sheet upload or through IDM Enrollment API.

For the Swedbank (Swish I kassa) Implementation also a dedicated register identity can group several Swish numbers after that they have been registered with Infrasec to a dedicated register identity. The cause for this is that a partner application that waits for asynchronous payments (in contrast to synchronous initiated payments) on several Swish numbers can execute just one call for the listed Swishnumbers in one API request and does not need to execute one request for every Swishnumber. Note: the grouped Swishnumbers that is bundled on one RegisterID must belong to the same organization number. The partner application may omit the Swishnumber/numbers when wait and list payments are executed. For these calls the in IDM RegisterID configured list of Swishnumbers are used. In the response, the received payments contain the receiving Swishnumber. For the calls of Confirm/Unconfirm and Refund this Swishnumber must be supplied together with the TransactionID. Note that the Status call returns the list of configured Swishnumber(s).



5.1 Service Provider

5.1.1 Swedbank Provider - "Swish I kassa"

When a merchant has signed up for the SWEDDBANK "Swish I kassa" service and received a Swish number Infrasec Enrollment / IDM support connection of the merchant's Swish number to Infrasec on Swedbank for the PosConneXion register.

The initial registration for a Swishnumber must be managed by a RegisterID that controls a particular Swishnumber. After this the registered Swishnumber can be linked to the group RegisterID together with other individually registered Swishnumbers.

The Swedbank "Swish I kassa" implementation supports payments that is paid without initiation. This means that payments can be processed without explicit initiation. This implementation supports both dynamic and static QR Code payments

With Swedbank "Swish I kassa" the INITIATEPAYMENT call is optional. Once a Swishnumber is successfully registered to a RegisterID, the Swish payments to this Swishnumber is available for listing independent of that the INITIATEPAYMENT is executed or not.

5.1.2 Getswish Provider - "Swish Handel"

When a merchant has signed up for a Swish Handel including Infrasec as a Technical Partner at the merchant's bank the Enrollment / IDM support connection of the merchant's Swish number to Infrasec for the PosConneXion register.

The Swish Handel implementation only supports payments that has been initiated with INITIATEPAYMENT before the payment can be processed. This means that all payments should be handled synchronously. Static QR Codes for payments is NOT supported. Each payment must be initiated and each payment QR Code will be unique.

With Getswish as provider the payment request may be directed to a specific customer mobile number. In this case the INITIATEPAYMENT must contain mobile nr tag `<SwishCustomerMobile>46705998877</SwishCustomerMobile>`. When the customer mobile is supplied the payment is invoked in the customers Swish app. In this case the tag for the QR code may be omitted `<SwishQrCode></SwishQrCode>`.

5.2 PosConneXion Service Request Swish Corporate

Through this API a POS / Register POS Application can manage and detect Swish payments. The request will be linked to the Swish service provider the register is bound to by the IDM Enrollment process.

5.2.1 ServiceID

This field will set to "SWISHC"

5.2.2 ServiceVersion

This field will set to "1.0"

5.2.3 ServiceFormat

This field default is "application/xml"

5.2.4 ServiceEncoding

This field will be set to "base64"

5.2.5 ServiceData

This base64 content must be formatted as in this specification

5.3 REFUNDPAYMENT note

Payments through the Swish framework is normally performed while executing a purchase. The framework support refunds on previous payments. A refund can be executed in whole or in parts up to the total amount of the initial amount. If refund is attempted above the executed payment amount a error is returned.

5.4 LISTPAYMENTS note

In exceptional events a refund that has been approved can by the bank later in the chain be returned to the store account. This can be the case if a bank at a later state decides that the payment (refund) must not be executed. If this happens the amount will be returned to the store account and a asynchronous notification will be available for listing.

The following may be returned when listing payments

- PURCHASE, The payment is a purchase
- REFUND, The payment is a refund
- CORRECTION, The payment is a correction on a refund, refund inhibited at a later stage by the bank
- UNKNOWN, Not implemented payment type

5.5 Swedbank SwishNumber list note

For the Swedbank implementation where Asynchronous payments can be detected and a partner uses several Swishnumbers on one organization number, these Swishnumbers can be linked to a Swishnumber group RegisterID.

Through this group RegisterID the partner can operate on the linked Swishnumbers and do not need to operate the all the individual RegisterID the Swishnumbers are originally registered with.

The Swishnumber must first be registered to a unique RegisterID. It is through this RegisterID the Swishnumber is linked to the Swedbank services. When the Swishnumber is successfully registered with Swedbank and the RegisterID, it can be linked to a group RegisterID.

As an example if RegisterID

ACME010001000001 is registered with Swishnumber 1230000001

ACME010001000002 is registered with Swishnumber 1230000002

ACME010001000003 is registered with Swishnumber 1230000003

Then we can link these Swishnumbers on a group RegisterID ex:

ACME0100010000CC with linked comma separated list of above Swishnumbers: 1230000001,1230000002,1230000003

In the PosConneXion API call the <SwishNumber> tag may then be set empty or contain part or the full registered Swishnumber list. If the <SwishNumber> tag is empty the

5.6 Getswish Customer Swish number note

For the Getswish implementation with INITIATEPAYMENT the tag <SwishCustomerMobile> can be used to direct the payment request to a specific customer mobile number. When this tag is used the customer with this mobilenummer Swish app will be populated with this unique payment information:

- Payee (Payment recipient) company name
- Amount to pay.

When the <SwishCustomerMobile> tag is used to address a particular customer the <SwishQrCode> tag may be omitted, since no QR code is required to initiate the payment.

5.7 SwishC API Tags

Attribute	Value	Comment
SwishAction	REGISTER UNREGISTER STATUS LISTACTIONS INITIATEPAYMENT WAITPAYMENT LISTPAYMENTS LISTCONFIRMED CONFIRMPAYMENT UNCONFIRMPAYMENT REFUNDPAYMENT FETCHCONFIRMATION SIMULATEPAYMENT	Optional - Register Swish Number to Registerid Optional - Unregister Swish Number from Registerid Get status of this Registerid link to Swish number(s) List Available Swish Actions for this Swish Nr Initiate payment (QRCode and current payment) Wait for Payment Confirmation (Duration limit) List Payments not flagged Confirmed (interval) List Payments flagged Confirmed (interval) Flag Payment as Confirmed Flag Confirmed Payment as Unconfirmed Execute Refund on previous Payment Execute query to the bank if payments missing Simulate Swish Payment App, PCX Verify
SwishProvider	SWEDBANK (syn and asyn) GETSWISH (only syn)	Swish Provider Identity. GETSWISH register must execute INITIATEPAYMENT. Optional on SWEDBANK
SwishPartner	Organization Number	Organization number of PosConneXion Partner
SwishOrgNumber	Organization Number	Organization number of the Store/Register
SwishNumber	Swish number => Swish number list =>	Swish number for this request. Optional on: WAITPAYMENT, LISTPAYMENTS LISTCONFIRMED
SwishSecurityCode	String	Option on Register, If set will be required on refund
SwishDate	DateTime YYYYMMDDHHMMSS	Optional: The time of the payment
SwishDelta	String Delta Seconds	Optional: Delta time for search backwards in time or wait forward in time
SwishCustomerName	String	Optional when filtering payments:
SwishCustomerMobile	String	Optional when filtering payments: Optional when Getswish INIRIATEPAYMENT: Direct payment to specific customer mobile Swish APP
SwishUserMessage	String	Optional when filtering payments: (Max 50 chars)
SwishAmount	String	Optional: Required: INITIATEPAYMENT
RefundAmount	String	Required: Refund
AccountNr	String	Optional: May be used to verify refund saccount
ClearingNr	String	Optional: May be used to verify refund clearing
SwishQrCode	String	Request a QR Code when initiating a payment Response QR Code Base64 encoded (png default)
SwishQrData	String	Response data content as within the QR Code
SwishQrLabel	String	QR Code label (Not supported, ie Getswish)
SwishAction2	String	Request LISTPAYMENTS when initiating a payment
SwishTransactionId	String	Required on Refund: Bank Transaction Id
SwishReference	String	PosConneXion Transaction Reference

6 Swish Corporate XML Examples

Examples of a Swish Corporate Main XML Structure.

6.1 Swish Corporate XML Request Example

A PosConneXion example for a request for the Swish Corporate Services.

```
<posConneXionRequest>
  <ApplicationID>ACME POS v1.0</ApplicationID>
  <RequestID>806e589fc9506474</RequestID>
  <OrgNr>1212121212</OrgNr>
  <RegisterId>EX11212121212101</RegisterId>
  <DateTime>20170131142620</DateTime>
  <posConnexionServiceList>
    <posConnexionService>
      <ServiceID>SWISHC</ServiceID>
      <ServiceVersion>1.0</ServiceVersion>
      <ServiceFormat>application/xml</ServiceFormat>
      <ServiceEncoding>base64</ServiceEncoding>
      <ServiceData>BASE64XXX</ServiceData>
    </posConnexionService>
  </posConnexionServiceList>
</posConneXionRequest>
```

6.2 Swish Corporate XML Response Examples

A PosConneXion example for a response from the Swish Corporate Services.

```
<posConneXionResponse>
  <TransactionID>16013029892193042480</TransactionID>
  <ApplicationID>ACME POS v1.0</ApplicationID>
  <RequestID>806e589fc9506474</RequestID>
  <OrgNr>1212121212</OrgNr>
  <RegisterId>EX11212121212101</RegisterId>
  <ResponseCode>0</ResponseCode>
  <ResponseMessage>Success</ResponseMessage>
  <ResponseReason></ResponseReason>
  <posConnexionServiceList>
    <posConnexionService>
      <ServiceID>SWISHC</ServiceID>
      <ServiceTransactionId>guid</ServiceTransactionId>
      <ResponseCode>0</ResponseCode>
      <ResponseMessage>Success</ResponseMessage>
      <ResponseReason></ResponseReason>
      <ServiceData>BASE64XXX</ServiceData>
    </posConnexionService>
  </posConnexionServiceList>
</posConneXionResponse>
```

6.3 ServiceData: Register Swish Number

This section illustrates an example registration when binding a Swish number to a POS Register.

Note that this function is optional though the function normally is executed when registering a register for Swish through PCX Administration IDM. Please check the Enrollment API specification

6.3.1 REGISTER: ServiceData Request

```
<SwishRequest>
  <SwishAction>REGISTER</SwishAction>
  <SwishProvider>SWEDBANK</SwishProvider>
  <SwishPartner>1010101010</SwishPartner>
  <SwishOrgNr>1212121212</SwishOrgNr>
  <SwishNumber>123444444</SwishNumber>
  <SwishOwnerMobile>+46705247248</SwishOwnerMobile>
  <SwishSecurityCode>abc123</SwishSecurityCode>
</SwishRequest>
```

6.3.2 REGISTER: ServiceData Response

```
<SwishResponse></SwishResponse>
```

6.4 ServiceData: Unregister Swish Number

This section illustrates an example of deregistration when unbinding a Swish number from a POS Register.

Note that this function is optional though the function also is executed when unregistering a register from Swish through PCX Administration IDM

6.4.1 UNREGISTER: Request

```
<SwishRequest>
  <SwishAction>UNREGISTER</SwishAction>
  <SwishProvider>SWEDBANK</SwishProvider>
  <SwishPartner>5568388200</SwishPartner>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
</SwishRequest>
```

6.4.2 UNREGISTER: Response

```
<SwishResponse></SwishResponse>
```

6.5 ServiceData: Query registered information

This section illustrates an example status request to check registered Swish information for the register

6.5.1 STATUS: Request

```
<SwishRequest>
  <SwishAction>STATUS</SwishAction>
  <SwishOrgNr>1212121212</SwishOrgNr>
</SwishRequest>
```

6.5.2 STATUS: Response

Example response for a register identity where multiple Swishnumbers are linked

```
<SwishResponse>
  <SwishProvider>SWEDBANK</SwishProvider>
  <SwishNumber>1234567891,1234567892,1234567893</SwishNumber>
  <SwishOwnerMobile>Not Available</SwishOwnerMobile>
  <SwishSecurityCode></SwishSecurityCode>
</SwishResponse>
```

Example response for a register identity with a single Swishnumber

```
<SwishResponse>
  <SwishProvider>GETSWISH</SwishProvider>
  <SwishNumber>1234567899</SwishNumber>
  <SwishOwnerMobile>Not Available</SwishOwnerMobile>
  <SwishSecurityCode></SwishSecurityCode>
</SwishResponse>
```

6.6 ServiceData: List Available Actions

This section illustrates an example when a POS Application determines what Actions this Swish number supports

6.6.1 LISTACTIONS: Request

```
<SwishRequest>
  <SwishAction>LISTACTIONS</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
</SwishRequest>
```

6.6.2 LISTACTIONS: Response

```
<SwishResponse>
  <SwishActionList>
    <SwishAction>REGISTER</SwishAction>
    <SwishAction>UNREGISTER</SwishAction>
    <SwishAction>LISTACTIONS</SwishAction>
    <SwishAction>INITIATEPAYMENT</SwishAction>
    <SwishAction>WAITPAYMENT</SwishAction>
    <SwishAction>LISTPAYMENTS</SwishAction>
    <SwishAction>LISTCONFIRMED</SwishAction>
    <SwishAction>CONFIRMPAYMENT</SwishAction>
    <SwishAction>UNCONFIRMPAYMENT</SwishAction>
    <SwishAction>REFUNDPAYMENT</SwishAction>
    <!-- Test / Verify environment -->
    <SwishAction>SIMULATEPAYMENT</SwishAction>
  </SwishActionList>
</SwishResponse>
```

6.7 ServiceData: Initiate Payment

This section illustrates an example when a POS Application call initiate payment to list already available transactions and to create a QR Code for the Swish App with payment amount, user message and Store (Payee) Swish number.

6.7.1 INITIATEPAYMENT: Request

```
<SwishRequest>
  <SwishAction>INITIATEPAYMENT</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
  <SwishDelta>120</SwishDelta>
  <SwishAmount>1.00</SwishAmount>
  <SwishUserMessage>
    Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
  </SwishUserMessage>
  <!-- GetSwish option: Request to mobilenumbers Swish app -->
  <SwishCustomerMobile>467123456789</SwishCustomerMobile>
  <!-- QR Code for Swish app payment initiation scanning -->
  <SwishQrCode></SwishQrCode>
  <!-- List any matching payments available when calling -->
  <SwishAction2>LISTPAYMENTS</SwishAction2>
</SwishRequest>
```

6.7.2 INITIATEPAYMENT: Response Ok

This response will be returned with a Swish payment and a QR Code that is in png format and Base64 encoded

```
<SwishResponse>
  <SwishPaymentList>
    <!-- List of unconfirmed payments withing history of delta
         time SwishDelta back from now to this Swishnumber -->
    <SwishPayment>
      <SwishAction>PURCHASE</SwishAction>
      <SwishDate>2017-09-27 18:05:06</SwishDate>
      <SwishCustomerName>Peter Andersson</SwishCustomerName>
      <SwishCustomerMobile>46705247248</SwishCustomerMobile>
      <SwishAmount>1.00</SwishAmount>
      <SwishTransactionId>5282635995992244</SwishTransactionId>
      <SwishReference>1580260CB5114EFE948F521CD7DD56DE</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
  </SwishPaymentList>
  <!--QR Code returned if init tag <SwishQrCode></SwishQrCode> -->
  <SwishQrCode>BASE64XXX</SwishQrCode>
  <SwishQrData>XXXXXXXXXX</SwishQrData>
</SwishResponse>
```

6.7.3 INITIATEPAYMENT: Response With Error

This response will be returned with error information in case of client Swish nr or the Swish solution including bank may be in error.

```
<SwishResponse>
  <ResponseCode>188</ResponseCode>
  <ResponseMessage>
    Target service identity not available
  </ResponseMessage>
  <ResponseReason>
    Swish corporate Swish service in problem
  </ResponseReason>
  <ResponseAlert>
    More details about the situation
  </ResponseAlert>
</SwishResponse>
```

6.8 ServiceData: Wait for payment and other notifications

This section illustrates an example when a POS Application wait for payment and other notifications for a specific duration. Wait duration times is accepted between 15s and 150s. If WAITPAYMENT request terminates before payment is received the POS client may call again to issue a new wait period. The Request has implemented a glitch period to support payment confirmations between repetitive calls. Filter data to only wait for payment from a given customer name, customer mobile or customer user message are optional.

6.8.1 WAITPAYMENT: Request

```
<SwishRequest>
  <SwishAction>WAITPAYMENT</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <!--If empty, IDM register configured Swishnumber is used -->
  <SwishNumber>1234950341</SwishNumber>
  <SwishDate></SwishDate>
  <SwishDelta>120</SwishDelta>
  <!-- Optional filter if waiting for specific payment -->
  <SwishCustomerName>Peter Andersson</SwishCustomerName>
  <SwishCustomerMobile>46705247248</SwishCustomerMobile>
  <SwishAmount>1.00</SwishAmount>
  <SwishUserMessage>
    Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
  </SwishUserMessage>
</SwishRequest>
```

6.8.2 WAITPAYMENT: Response Ok

```
<SwishResponse>
  <SwishPaymentList>
    <SwishPayment>
      <SwishAction>PURCHASE</SwishAction>
      <SwishDate>2017-09-27 18:05:06</SwishDate>
      <SwishCustomerName>Peter Andersson</SwishCustomerName>
      <SwishCustomerMobile>46705247248</SwishCustomerMobile>
      <SwishAmount>1.00</SwishAmount>
      <SwishTransactionId>5282635995992244</SwishTransactionId>
      <SwishReference>1580260CB5114EFE948F521CD7DD56DE</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
  </SwishPaymentList>
</SwishResponse>
```

6.8.3 WAITPAYMENT: Response TimeOut

This response will be returned if Swish payment confirmation not is returned within the wait delta duration time

```
<SwishResponse>
  <ResponseCode>23</ResponseCode>
  <ResponseMessage>
    Timer released on a waitable object
  </ResponseMessage>
  <ResponseReason>
    Swish corporate Swish transaction payment confirmation was not
    received within given time
  </ResponseReason>
</SwishResponse>
```

6.9 ServiceData: List Unconfirmed payments

This section illustrates an example when a POS Application lists payment and other notifications for a specific date and time interval. If the list should be backwards from current time the date shall not be set

6.9.1 LISTPAYMENTS: Request

```
<SwishRequest>
  <SwishAction>LISTPAYMENTS</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <!--If empty, IDM register configured Swishnumber is used -->
  <SwishNumber>1234950341</SwishNumber>
  <SwishDate>201709272200</SwishDate>
  <SwishDelta>1800</SwishDelta>
  <!-- Optional filter to narrow payment response list -->
  <!-- Not used in this API call example
  <SwishCustomerName>Peter Andersson</SwishCustomerName>
  <SwishCustomerMobile>46705247248</SwishCustomerMobile>
  <SwishAmount>1.00</SwishAmount>
  <SwishTransactionId>5282635995992244</SwishTransactionId>
  <SwishUserMessage>
    Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
  </SwishUserMessage>
  -->
</SwishRequest>
```

6.9.2 LISTPAYMENTS: Response

```
<SwishResponse>
  <SwishPaymentList>
    <SwishPayment>
      <SwishAction>PURCHASE</SwishAction>
      <SwishDate>2017-09-27 22:13:18</SwishDate>
      <SwishCustomerName>Peter Andersson</SwishCustomerName>
      <SwishCustomerMobile>46705247248</SwishCustomerMobile>
      <SwishAmount>5.00</SwishAmount>
      <SwishTransactionId>5282784911813166</SwishTransactionId>
      <SwishReference>2CAA45B798E4426992ED6886F79DE613</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
    <SwishPayment>
      <SwishAction>REFUND</SwishAction>
      <SwishDate>2017-09-27 22:22:16</SwishDate>
      <SwishCustomerName></SwishCustomerName>
      <SwishCustomerMobile>1234950341</SwishCustomerMobile>
      <SwishAmount>1.00</SwishAmount>
      <SwishTransactionId>5282790293943118</SwishTransactionId>
      <SwishReference>8D1CF3F0A3C111E79CF61FC11878C8F3</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
  </SwishPaymentList>
</SwishResponse>
```

6.10 ServiceData: List Confirmed payments

This section illustrates an example when a POS Application lists payment and other notifications that are Confirmed. If the list should be backwards from current time the date shall not be set

6.10.1 LISTCONFIRMED: Request

```
<SwishRequest>
  <SwishAction>LISTCONFIRMED</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <!--If empty, IDM register configured Swishnumber is used -->
  <SwishNumber>1234950341</SwishNumber>
  <SwishNumber>1234950341</SwishNumber>
  <SwishDate>201709272200</SwishDate>
  <SwishDelta>1800</SwishDelta>
</SwishRequest>
```

6.10.2 LISTCONFIRMED: Response

```
<SwishResponse>
  <SwishPaymentList>
    <SwishPayment>
      <SwishAction>PURCHASE</SwishAction>
      <SwishDate>2017-09-27 22:13:18</SwishDate>
      <SwishCustomerName>Peter Andersson</SwishCustomerName>
      <SwishCustomerMobile>46705247248</SwishCustomerMobile>
      <SwishAmount>5.00</SwishAmount>
      <SwishTransactionId>5282784911813166</SwishTransactionId>
      <SwishReference>2CAA45B798E4426992ED6886F79DE613</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
    <SwishPayment>
      <SwishAction>REFUND</SwishAction>
      <SwishDate>2017-09-27 22:22:16</SwishDate>
      <SwishCustomerName></SwishCustomerName>
      <SwishCustomerMobile>1234950341</SwishCustomerMobile>
      <SwishAmount>1.00</SwishAmount>
      <SwishTransactionId>5282790293943118</SwishTransactionId>
      <SwishReference>8D1CF3F0A3C111E79CF61FC11878C8F3</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
  </SwishPaymentList>
</SwishResponse>
```

6.11 ServiceData: Execute Refund request

This section illustrates an example when a POS Application executes a Refund request.
If the date is for another day, the date shall be supplied

6.11.1 REFUNDPAYMENT: Request

```
<ServiceData>
  <SwishRequest>
    <SwishAction>REFUNDPAYMENT</SwishAction>
    <SwishOrgNr>5566241674</SwishOrgNr>
    <SwishDate>20170927</SwishDate>
    <SwishNumber>1234950341</SwishNumber>
    <SwishSecurityCode>aqlsw2</SwishSecurityCode>
    <SwishTransactionId>5282682494972905</SwishTransactionId>
    <RefundAmount>1</RefundAmount>
    <!-- Optional, If set must match -->
    <AccountNr>1234567891</AccountNr>
    <ClearingNr>13579</ClearingNr>
  </SwishRequest>
```

6.11.2 REFUNDPAYMENT: Response

```
<SwishResponse>
  <SwishPaymentList>
    <SwishPayment>
      <SwishAction>REFUND</SwishAction>
      <SwishDate>2017-09-27 22:24:53</SwishDate>
      <SwishCustomerName></SwishCustomerName>
      <SwishCustomerMobile>1234950341</SwishCustomerMobile>
      <SwishAmount>1.00</SwishAmount>
      <SwishTransactionId>5282791871303116</SwishTransactionId>
      <SwishReference>EB2007D0A3C111E7B88DD729E4B58550</SwishReference>
      <SwishUserMessage>
        Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
      </SwishUserMessage>
    </SwishPayment>
  </SwishPaymentList>
</SwishResponse>
```

6.12 ServiceData: Confirme payment

This section illustrates an example when a POS Application confirms a payment to not list it in future LISTPAYMENT requests but in LISTCONFIRMED requests.

6.12.1 CONFIRMPAYMENT: Request

```
<SwishRequest>
  <SwishAction>CONFIRMPAYMENT</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
  <SwishTransactionId>5282784911813166</SwishTransactionId>
</SwishRequest>
```

6.12.2 CONFIRMPAYMENT: Response

```
<SwishResponse></SwishResponse>
```

6.13 ServiceData: Unconfirm payment

This section illustrates an example when a POS Application unconfirms a payment that has been notified as confirmed. After this call the payment is again will be listed using LISTPAYMENT requests and through LISTCONFIRMED.

6.13.1 UNCONFIRMPAYMENT: Request

```
<SwishRequest>
  <SwishAction>UNCONFIRMPAYMENT</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
  <SwishTransactionId>5282784911813166</SwishTransactionId>
</SwishRequest>
```

6.13.2 UNCONFIRMPAYMENT: Response

```
<SwishResponse></SwishResponse>
```

6.14 Emulation of Swish App payment (PosConneXion Test and Verify)

This section illustrates an emulation example of a payment as it has been payed using the Swish App. Normally this function is executed distinct and separate from the POS implementation. This function is only available in Infrasec test and verification environment where real Swish payments not is avaiabale. The Swish system does not contain any test environment.

6.14.1 SIMULATEPAYMENT: Purchase

When the Register is waiting for a payment the API wil await the notification on the payment operation. With this call the confirmation on the payment operation is simulated. If this call not is executed the wait call will timeout

```
<SwishRequest>
  <SwishAction>SIMULATEPAYMENT</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
  <SwishDate>20170927222453</SwishDate>
  <SwishAmount>1.00</SwishAmount>
  <SwishCustomerName>Peter Andersson</SwishCustomerName>
  <SwishCustomerMobile>1234950341</SwishCustomerMobile>
  <SwishUserMessage>
    Butik AB: Ka: 001 Nr: 12345 Id: 123456789123456789
  </SwishUserMessage>
</SwishRequest>
```

6.14.2 SIMULATEPAYMENT: Refund

When the Register is requesting a refund the API wil await the notification on the refund operation. With this call the confirmation on the refund operation is simulated. If this call not is executed the refund call will timeout

```
<SwishRequest>
  <SwishAction>SIMULATEPAYMENT</SwishAction>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
  <SwishDate>20170927222453</SwishDate>
  <SwishAmount>-1.00</SwishAmount>
  <SwishTransactionId>5282784911813166</SwishTransactionId>
</SwishRequest>
```

6.15 Investigating missing payments

This section illustrates an example of a request to query for payments that may have been unsuccessful to be distributed between the Bank and PosConneXion. This request investigates and returns transaction identities for such payments.

6.15.1 FETCHCONFIRMATION: Request

Example to query for missing payments.

The SwishTransactionId is optional but can be supplied when a specific payment is queried. When SwishTransactionId is supplied and also located the payment is returned, in the same way as when a LISTPAYMENTS is executed. If the SwishTransactionId was not located a error code is returned.

If SwishTransactionId is not supplied as in this example below a list of located payments SwishTransactionId is returned. If no missing payments was located an empty list is returned.

```
<SwishRequest>
  <SwishAction>FETCHCONFIRMATION</SwishAction>
  <SwishProvider>SWEDBANK</SwishProvider>
  <SwishPartner>5566241674</SwishPartner>
  <SwishOrgNr>5566241674</SwishOrgNr>
  <SwishNumber>1234950341</SwishNumber>
  <SwishDate>20181003</SwishDate>
  <SwishTransactionId></SwishTransactionId>
</SwishRequest>
```

6.15.2 FETCHCONFIRMATION: Response

Below is two two versions of responses when missing payments was located and when payments was not located

Example response when two payments was missing

```
<?xml version="1.0" encoding="UTF-8"?>
<SwishResponse>
  <SwishTransactionIdList>
    <SwishTransactionId>5602556301286918</SwishTransactionId>
    <SwishTransactionId>5602566885296518</SwishTransactionId>
  </SwishTransactionIdList>
</SwishResponse>
```

Example response when no payment was missing

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