THE Berlin GROUP

A EUROPEAN STANDARDS INITIATIVE

Joint Initiative on a PSD2 Compliant XS2A Interface

NextGenPSD2 XS2A Framework

Implementation Guidelines

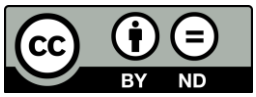
Version 1.0

08 February 2018

License Notice

This Specification has been prepared by the Participants of the Joint Initiative pan-European PSD2-Interface Interoperability* (hereafter: Joint Initiative). This Specification is published by the Berlin Group under the following license conditions:

- "Creative Commons Attribution-NoDerivatives 4.0 International Public License"

  This means that the Specification can be copied and redistributed in any medium or format for any purpose, even commercially, and when shared, that appropriate credit must be given, a link to the license must be provided, and indicated if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. In addition, if you remix, transform, or build upon the Specification, you may not distribute the modified Specification.

- Implementation of certain elements of this Specification may require licenses under third party intellectual property rights, including without limitation, patent rights. The Berlin Group or any contributor to the Specification is not, and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.
- The Specification, including technical data, may be subject to export or import regulations in different countries. Any user of the Specification agrees to comply strictly with all such regulations and acknowledges that it has the responsibility to obtain licenses to export, re-export, or import (parts of) the Specification.

---

* The 'Joint Initiative pan-European PSD2-Interface Interoperability' brings together participants of the Berlin Group with additional European banks (ASPSPs), banking associations, payment associations, payment schemes and interbank processors.

Contents

1 Introduction

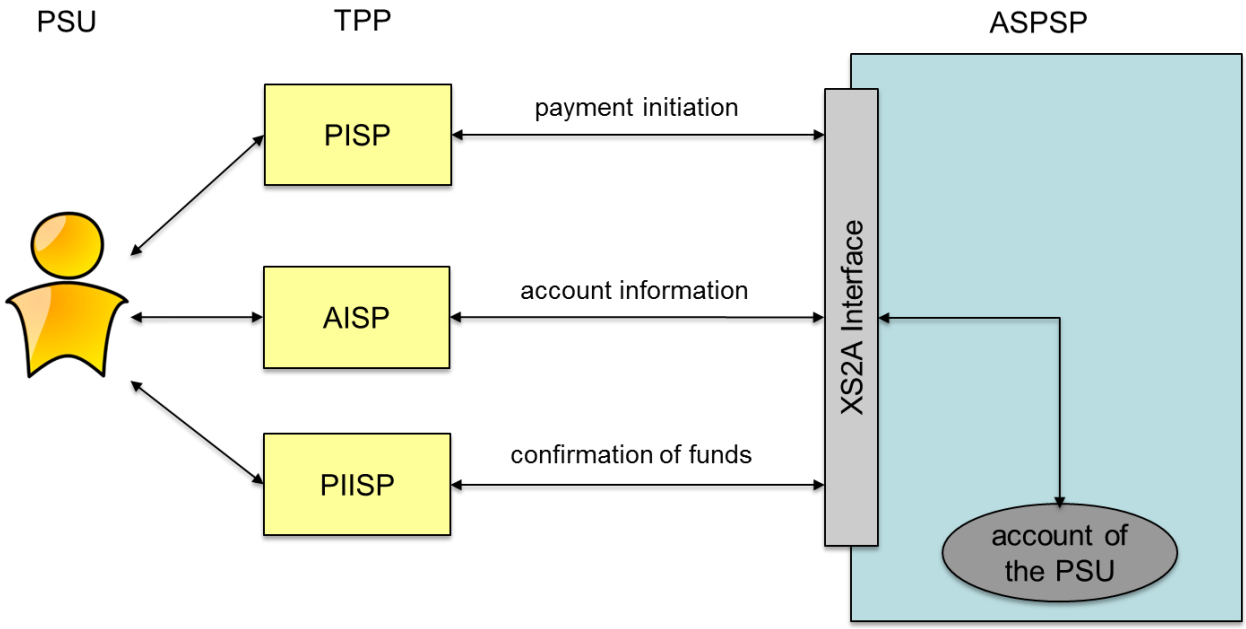
1.1 Background

With [PSD2] the European Union has published a new directive on payment services in the internal market. Member States have to adopt this directive into their national law until 13th of January 2018.

Among others [PSD2] contains regulations of new services to be operated by so called Third Party Payment Service Providers (TPP) on behalf of a Payment Service User (PSU). These new services are

- Payment Initiation Service (PIS) to be operated by a Payment Initiation Service Provider (PISP) TPP as defined by article 66 of [PSD2],
- Account Information Service (AIS) to be operated by an Account Information Service Provider (AISP) TPP as defined by article 67 of [PSD2], and
- Confirmation of the Availability of Funds service to be used by Payment Instrument Issuing Service Provider (PIISP) TPP as defined by article 65 of [PSD2].

For operating the new services a TPP needs to access the account of the PSU which is usually managed by another PSP called the Account Servicing Payment Service Provider (ASPSP). As shown in the following figure, an ASPSP has to provide an interface (called "PSD2 compliant Access to Account Interface" or short "XS2A Interface") to its systems to be used by a TPP for necessary accesses regulated by [PSD2]:



Further requirements on the implementation and usage of this interface are defined by a Regulatory Technical Standard (short RTS) from the European Banking Authority (short EBA) and to be published in the Official Journal of the European Commission.

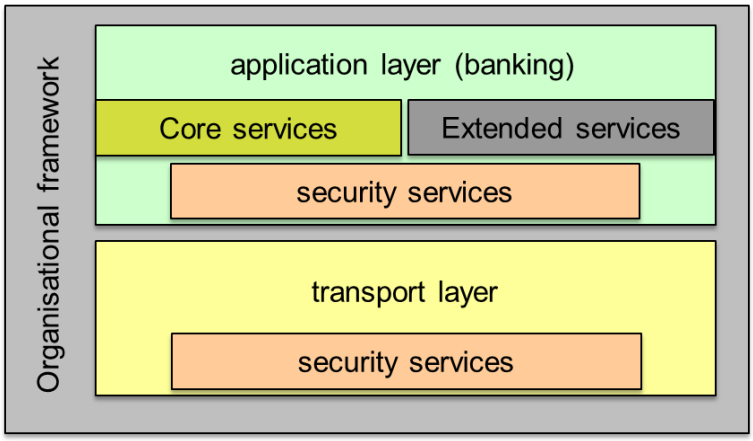
1.2 XS2A Interface Specification

This document is part of the NextGenPSD2 XS2A Specification which defines a standard for an XS2A Interface and by this reaching interoperability of the interfaces of ASPSPs at least for the core services defined by [PSD2]. An ASPSP may then use this standard as a basis for the implementation of its XS2A Interface to be compliant with PSD2.

The XS2A Interface is designed as a B2B interface between a TPP server and the ASPSP server. For time being, the protocol defined in this document is a pure client-server protocol, assuming the TPP being the client, i.e. all API calls are initiated by the TPP. In future steps, this protocol might be extended to a server-server protocol, where also the ASPSP initiates API calls.

The Interoperability Framework defines operational rules, requirements on the data model and a process description in [XS2A-OR].

This document details the standard in defining messages and detailed data structures for the XS2A Interface. For the specification the two layers shown in the following figure are distinguished:



At the application layer only the core services will be specified in the first version of the framework. In addition the framework will be prepared such that the interface of an ASPSP may be extended with its own additional corporate specific services (included in the figure as “Extended services”). In future versions of this framework, some extended services will also be part of the standard. This framework documentation will point out extended services where the market need is already identified.

Using defined parameters different versions and variants of this protocol can be distinguished and implemented.

1.3 Structure of the Document

This document first outlines notations in Chapter 2 and requirements on the transport layer in Chapter 3. In Chapter 4, guiding principles for the definition of the XS2A interface and the API structure with API endpoints and permitted access methods are described. Chapter 5 then specifies in detail how a Payment Initiation Service Provider (PISP) can initiate payments within the Berlin Group XS2A. Chapter 6 then repeats this for the access of Account Information Service Provider (AISP) to a Payment Service User (PSU) account. The AIS and the PIS service are sharing potentially some API calls, specifically for authorising transactions directly through the TPP / ASPSP interface. These methods used potentially within both services are specified in Chapter 7. Chapter 8 then shortly explains how AIS and PIS services might be technically combined within one TPP / ASPSP business session.

The Confirmation of Funds Service for Payment Instrument Initiation Service Provider (PIISP) is specified in detail in Chapter 9. Following these chapters with functional chapters, the Chapter 10 to Chapter 13 specify core payment structure, requirements on the optional integration of OAuth2, the usage of electronic seals for authentication on application level and general complex data structures.

1.4 Document History

Version	Change/Note	Approved
0.99	Market consultation draft of the Berlin Group XS2A Interface Framework	NextGenPSD2 Taskforce, 27 September 2017
1.0	Version 1.0 for publication. Takes into account the results of the market consultation and the final EBA-RTS on SCA and CSC.	NextGenPSD2 Taskforce, 08 February 2018

2 Character Sets and Notations

2.1 Character Set

The character set is UTF 8 encoded. This specification is only using the basic data elements “string”, “boolean”, "ISODateTime", "ISODate", "UUID" and “integer” and ISO based code lists (with a byte length of 32 bytes). For codes defined by ISO, a reference to the corresponding ISO standard is given in 13.21.

Max35Text, Max70Text, Max140Text and Max512Text are defining strings with a maximum length of 35, 70, 140 and 512 characters respectively.

ASPSPs will accept for strings at least the following character set:

a b c d e f g h i j k l m n o p q r s t u v w x y z

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

0 1 2 3 4 5 6 7 8 9

/ - ? : ( ) . , ' +

Space

ASPSPs may accept further character sets for text fields like names, addresses, text. Corresponding information will be contained in the ASPSP documentation of the XS2A interface. ASPSPs might convert certain special characters of these further character sets, before forwarding e.g. submitted payment data.

2.2 Notation

For API request calls query parameters, http header parameters and body content parameters are specified within this specification.

The definitions are given by tables like

Attribute	Type	Condition	Description
attribute tag	type of attribute	condition	description of the semantic of the attribute and further conditions.

The following conditions can be set:

- Optional: The attribute is supported by the ASPSP, usage is optional for the TPP.

- Conditional: The attribute is supported by the ASPSP and might be mandated by the ASPSP in its own documentation of the support of this XS2A interface.

- Mandatory: The attribute is supported by the ASPSP and shall be used by the TPP.

- [Optional]: It is optional for the ASPSP to support this attribute. If the ASPSP is supporting the attribute as indicated in its own documentation of this XS2A interface, it might be used by the TPP optionally. If the ASPSP is not supporting the attribute, then the request is rejected when it is contained.

- [Conditional]: It is optional for the ASPSP to support this attribute. If the ASPSP is supporting the attribute, it might even be mandated by the ASPSP as indicated in its own documentation of this XS2A interface. If the ASPSP is not supporting the attribute, then the request is rejected when it is contained.

Remark: Please note that the conditions [Optional] and [Conditional] are used rarely in this specification.

3 Transport Layer

The communication between the TPP and the ASPSP is always secured by using a TLS-connection using TLS version 1.2 or higher. For the choice of cipher suite selections, NIST recommendations on the cryptographical strength should be followed. For ASPSPs, further cipher suite requirements of their national IT security agency might apply.

This TLS-connection is set up and controlled by the TPP. It is not necessary to set up a new TLS-connection for each transaction, however the ASPSP might terminate an existing TLS-connection if required by its security setting.

The TLS-connection has to be established always including client (i.e. TPP) authentication. For this authentication the TPP has to use a qualified certificate for website authentication. This qualified certificate has to be issued by a qualified trust service provider according to the eIDAS regulation [eIDAS]. The content of the certificate has to be compliant with the requirements of [XS2A-OR] NextGenPSD2 XS2A Framework, Operational Rules, The Berlin Group Joint Initiative on a PSD2 Compliant XS2A Interface, version1.0, published 08 February 2018

[EBA-RTS]. The certificate of the TPP has to indicate all roles the TPP is authorised to use.

4 Application Layer: Guiding Principles

4.1 Location of Message Parameters

The XS2A Interface definition follows the REST service approach. This approach allows to transport message parameters at different levels:

- message parameters as part of the http level (http header)
- message parameters by defining the resource path (URL path information) with additional query parameters and
- message parameters as part of the http body.

The content parameters in the corresponding http body will be encoded either in JSON or in XML syntax. XML syntax is only used where

- an ISO20022 based payment initiation (pain.001 message) with the corresponding payment initiation report (pain.002 message) or
- ISO20022 based account information message (camt.052, camt.053 or camt.054 message)

is contained.

As an exception, response messages might contain plain text format in account information messages to support MT940, MT941 or MT942 message formats.

The parameters are encoded in

- in spinal-case (small letters) on path level,
- in Spinal-case (starting capital letters) on http header level and
- in lowerCamelCase for query parameters and JSON based content parameters.

The following principle is applied when defining the API:

Message parameters as part of the http header:

- Definition of the content syntax,
- Certificate and Signature Data where needed ,
- PSU identification data (the actual data from the online banking frontend or access token),
- Protocol level data like Request Timestamps or Request/Transaction Identifiers

Message parameters as part of the path level:

- All data addressing a resource:
  - Provider identification,
  - Service identification,
  - Payment product identification,
  - Account Information subtype identification,
  - Resource ID

Query Parameters:

- Additional information needed to process the request as process steering flags or filtering information,

Message parameters as part of the http body:

- Business data content,
- PSU authentication data,
- Messaging Information
- Hyperlinks to steer the full TPP – ASPSP process

4.2 Signing Messages at Application Layer

If requested by the ASPSP the TPP has to sign the request messages to be sent to the ASPSP.

The signature shall be included in the HTTP-Header as defined by [signHTTP], chapter 4.

The electronic signature of the TPP has to be based on a qualified certificate for electronic seals. This qualified certificate has to be issued by a qualified trust service provider according to the eIDAS regulation [eIDAS]. The content of the certificate has to be compliant with the requirements of [XS2A-OR] NextGenPSD2 XS2A Framework, Operational Rules, The Berlin Group Joint Initiative on a PSD2 Compliant XS2A Interface, version1.0, published 08 February 2018

[EBA-RTS]. The certificate of the TPP has to indicate all roles the TPP is authorised to use.

4.3 Optional Usage of OAuth2 for PSU Authentication or Authorisation

The XS2A API will allow an ASPSP to implement OAuth2 as a support for the consent management of the PSU towards the TPP for the payment initiation and/or account information service. In this case, the TPP will be the client, the PSU the resource owner and the ASPSP will be the resource server in the abstract OAuth2 model.

This specification supports two ways of integrating OAuth2. The first support is an authentication of a PSU in a pre-step, translating this authentication into an access token to be used at the XS2A interface afterwards. This pre-step shall use processing options of the OAuth2 protocol without redirection to the authentication server. This usage of OAuth2 will be referred to in this specification as "if OAuth2 has been used as PSU authentication". Further details will be defined in the documentation of the ASPSP of this XS2A interface.

The second option to integrate OAuth2 is an integration as an OAuth2 SCA Approach to be used of authorisation of payment initiations and consents. In both services, PIS and AIS, OAuth2 will in this option be used in an integrated way, by using the following steps:

1.) The payment data, resp. AIS consent data is posted to the corresponding payment initiation resp. consents endpoint of the XS2A API.
2.) The OAuth2 protocol is used with the "Authorisation Code Grant" flow to get the consent on the payment resp. the AIS access authorised by the PSU, while using the "scope" attribute in OAuth2 to refer to the data from Step 1.).
3.) In case of a PIS, the corresponding payment is then automatically initiated by the ASPSP after a successful authorisation by the PSU.
In case of an AIS, the TPP can use the access token resp. received during the OAuth2 protocol to access the /accounts endpoint for authorised account information for the validity period of the authorised consent or the technical access token.

For Step 2.), details are described in Section 12.

When using OAuth2, the XS2A API calls will work with an access token instead of using the PSU credentials.

4.4 XS2A Interface API Structure

The XS2A Interface is resource oriented. Resources can be addressed under the API endpoints

https://{provider}/v1/{service-endpoint}

using additional content parameters {parameters}

where

- {provider} is the host of the XS2A API, which is not further mentioned
- v1 is denoting the version one of the interface
- {service} has the values consents, payments, bulk-payments, standing-orders, accounts, card-accounts or funds-confirmations, eventually extended by more information on product types and request scope
- {parameters} are content attributes defined in JSON or XML encoding according to the following
    - XML encoding appears only when ISO20022 pain.001 messages are transported when demanded by the ASPSP for the corresponding payment product
    - all other request bodies are encoded in JSON

The structure of the request/response is described according to the following categories

- Path: Attributes encoded in the Path, e.g. “payments/sepa-credit-transfers” for {resource}
- Query Parameters: Attributes added to the path after the ? sign as process steering flags or filtering attributes for GET access methods
- Header: Attributes encoded in the http header of request or response
- Request: Attributes within the content parameter set of the request
- Response: Attributes within the content parameter set of the response, defined in XML, text or JSON:
    - XML encoding appears only, when camt.052, camt.053 or camt.054 messages (reports, notifications or account statements) or pain.002 payment status messages are transported. pain.002 messages will only be delivered for the GET Status Request, and only in cases where the payment initiation was performed by using pain.001 messages.

- Text encoding appears only, when MT940, MT941 or MT942 messages (reports, notifications or account statements) are transported.
- All other response bodies are encoded in JSON.

The HTTP response codes which might be used in this XS2A interface are specified in Section **13.10**. This is not repeated for every API call definition.

Remark: For JSON based responses, this specification defines body attributes which are responded from ASPSP to TPP following POST or PUT API calls and which have not been already used in the API call requests by the TPP. The ASPSP is free to return the whole addressed resource within the response, following usual REST methodologies.

4.5 API Access Methods

The following table gives an overview on the HTTP access methods supported by the API endpoints and by resources created through this API.

Conditions

It further defines, whether this method support is mandated for the ASPSP by this specification or whether it is an optional feature for the ASPSP. Please note that this condition is given relative to the parent node of the path, i.e. the condition e.g. on a method on /v1/consents/{consent-ID} applies only if the endpoint /v1/consents is supported at all.

Please note that all methods submitted by a TPP, which are addressing dynamically created resources in this API, may only apply to resources which have been created by the same TPP before.

Endpoints/Resources	Method	Condition	Description
payments/{payment-product}	POST	Mandatory	Create a payment initiation resource addressable under {paymentId} with all data relevant for the corresponding payment product. This is the first step in the API to initiate the related payment.
payments/{payment-product}/{paymentId}	PUT	Mandatory for Embedded SCA Approach, Conditional for other approaches	Update data on the payment resource if needed. It may authorise a payment within the Embedded SCA Approach where needed. Independently from the SCA Approach it supports e.g. the selection of the authentication method and a non-SCA PSU authentication.
payments/{payment-product}/{paymentId}	GET	Mandatory	Read the details of an initiated payment.
payments/{payment-product}/{paymentId}/status	GET	Mandatory	Read the transaction status of the payment
bulk-payments/{payment-product}	POST	Optional	Create a bulk payment initiation resource addressable under {paymentId} with all data relevant for the corresponding payment product. This is the first step in the API to

Endpoints/Resources	Method	Condition	Description
			initiate the related bulk payment.
bulk-payments/{payment-product}/{paymentId}	PUT	Mandatory for Embedded SCA Approach, Conditional in other SCA Approaches	Update data on the bulk payment resource if needed. It may authorise a bulk payment within the Embedded SCA Approach where needed. Update of other data independently of the SCA approach.
bulk-payments/{payment-product}/{paymentId}	GET	Mandatory	Read the details of an initiated bulk payment.
bulk-payments/{payment-product}/{paymentId}/status	GET	Mandatory	Read the transaction status of the bulk payment
periodic-payments/{payment-product}	POST	Optional	Create a standing order initiation resource for recurrent i.e. periodic payments addressable under {paymentId} with all data relevant for the corresponding payment product and the execution of the standing order. This is the first step in the API to initiate the related recurring/periodic payment.
periodic-payments/{payment-product}/{paymentId}	PUT	Mandatory for Embedded SCA Approach	Update data on the payment resource if needed. In one of these method, a standing order within the Embedded SCA Approach is authorised.
periodic-payments/{payment-product}/{paymentId}	GET	Mandatory	Read the details of an initiated standing order for recurring/periodic payments.
periodic-payments/{payment-product}/{paymentId}/status	GET	Mandatory	Read the transaction status of the standing order for recurring/periodic payments.
accounts	GET	Mandatory	Read all identifiers of the accounts, to which an account access has been granted to through the /consents endpoint by the PSU. In

Endpoints/Resources	Method	Condition	Description
			addition, relevant information about the accounts and hyperlinks to corresponding account information resources are provided if a related consent has been already granted. **Remark:** Note that the /consents endpoint optionally offers to grant an access on **all available** payment accounts of a PSU. In this case, this endpoint will deliver the information about all available payment accounts of the PSU at this ASPSP.
accounts/?withBalance	GET	Optional	Read the identifiers of the available payment account together with booking balance information, depending on the consent granted
accounts/{account-id}	GET	Mandatory	Give detailed information about the addressed account.
accounts/{account-id}/?withBalance	GET	Optional	Give detailed information about the addressed account together with balance information
accounts/{account-id}/balances	GET	Mandatory	Give detailed balance information about the addressed account
accounts/{account-id}/transactions	GET	Mandatory	Read transaction reports or transaction lists of a given account, depending on the steering parameter "bookingStatus" For a given account, additional parameters are e.g. the attributes “dateFrom” and “dateTo”. The ASPSP might add balance information, if transaction lists without balances are not supported.
accounts/{account-id}/transactions/?withBalance	GET	Optional	Read transaction reports or transaction lists of a given account, depending on the steering parameter

Endpoints/Resources	Method	Condition	Description
			"bookingStatus" together with balances.
accounts/{account-id}/transactions/{transaction-id}	GET	Optional	Read transaction details of an addressed transaction.
card-accounts	GET	Optional	This endpoint will deliver credit card account related account information. It will be sub-structured analogously to the /accounts interface with similar sub-endpoints. Further details will be published with the next version of the specification.
consents	POST	Mandatory	Create a consent resource, defining access rights to dedicated accounts of a given PSU-ID. These accounts are addressed explicitly in the method as parameters as a core function.
consents[/?withBalance]	POST	Optional	As an option, an ASPSP might optionally accept a specific access right on the access on all psd2 related services for all available accounts. As another option an ASPSP might optionally also accept a command, where only access rights are inserted without mentioning the addressed account. The relation to accounts is then handled afterwards between PSU and ASPSP. This option is supported only within the Decoupled, OAuth2 or Re-direct SCA Approach. As a last option, an ASPSP might in addition accept a command with no defined access rights at all. In this case, this command is defining only the access right to the list of all

Endpoints/Resources	Method	Condition	Description
			payment accounts of a given PSU-ID as such. If in this case the optional parameter “withBalance” is used, then the requested consent is on all accounts together with balances.
consents/{consentId}	GET	Mandatory	Reads the exact definition of the given consent resource {consentId} including the validity status
	PUT	Mandatory for Embedded SCA Approach, conditional for other SCA approaches	Update data on the consent resource, authorise a consent within the Embedded SCA Approach where needed. Might be needed also in other SCA approaches e.g. to select the authentication method.
	DELETE	Mandatory	Delete the addressed consent.
consents/{consentId}/status	GET	Mandatory	Read the authentication status of the addressed consent resource.
funds-confirmations	POST	Mandatory	Checks whether a specific amount is available at point of time of the request on an account linked to a given tuple card issuer(TPP)/card number, or addressed by IBAN and TPP respectively

Remark: Note that the {account-id} and {card-account-id} parameters can be tokenised by the ASPSP such that the actual account numbers like IBANs or PANs are not part of the path definitions of the API for data protection reasons. This tokenisation is managed by the ASPSP.

4.6 HTTP Response Codes and Additional Error Information

The http response code is communicating the success or failure of a TPP request message. The 4XX http response codes should only be given if the current request cannot be fulfilled, e.g. a payment initiation cannot be posted or account transactions cannot be retrieved. A request to get status of an existing payment or a consent usually returns http code 200 since the actual request to retrieve status succeeded, regardless if that payment or consent state is set to failure or not.

This specification supports the following http response codes:

Status Code	Description
200 OK	PUT, GET Response Codes This return code is permitted when the request is repeated due to a time-out. The response in that might be either a 200 or 201 code depending on the ASPSP implementation. The POST for a Funds request will also return 200 since it does not create a new resource.
201 Created	POST Response code where Payment Initiation or Consent Request was correctly performed.
204 No Content	DELETE Response code where a consent resource was successfully deleted. The code indicates that the request was performed, but no content is returned.
400 Bad Request	Validation error occurred. This code will cover malformed syntax in request or incorrect data in payload.
401 Unauthorised	The TPP or the PSU is not correctly authorised to perform the request. Retry the request with correct authentication information.
403 Forbidden	Returned if the resource that was referenced in the path exists but cannot be accessed by the TPP or the PSU. This code should only be used for non-sensitive id references as it will reveal that the resource exists even though it cannot be accessed.
404 Not found	Returned if the resource that was referenced in the path does not exist or cannot be referenced by the TPP or the PSU. When in doubt if a specific id in the path is sensitive or not, use http code 404 instead of 403.

Status Code	Description
405 Method Not Allowed	This code is only sent when the http method (PUT, POST, DELETE, GET etc) is not supported on a specific endpoint. It has nothing to do with the consent, payment or account information data model.
406 Not Acceptable	The ASPSP cannot generate the content that the TPP specified in the Accept header.
408 Request Timeout	The sever is still working correctly, but an individual request has timed out.
415 Unsupported Media Type	The TPP has supplied a media type which the ASPSP does not support.
429 Too Many Requests	The TPP has exceeded the number of requests allowed by the consent or by the RTS.
503 Service Unavailable	The ASPSP server is currently unavailable. Generally, this is a temporary state.

If necessary the ASPSP might communicate additional error information to the TPP within a request/response dialogue. The additional error information is sent to the TPP using the data element TPP Message Information with the attribute category set to “ERROR”. The attribute “code” indicates the error, cp. Section 13.10 and if applicable the path of the element of the request message which provoked this error message. It will further offer a free text field to describe the error context or actions to be taken to the TPP.

This error element can be embedded in all JSON based response messages of the Berlin Group XS2A Interface. This is not mentioned in the following API call definitions. If an error information is sent to the TPP the transaction status is always set to “Rejected” where applicable.

Example:

```
{
 “transactionStatus”: “Rejected”,
 “tppMessages”: [ {
        “category”: “ERROR”,
        “code”: “TOKEN_INVALID”,
        “text”: “additional text information of the ASPSP up to 512 characters”
   } ]
```

```
}
```

4.7 API Steering Process by Hyperlinks

The XS2A API requires for the payment initiation and account information service several requests from the TPP towards the ASPSP. With the Payment Initiation Request and the Account Information Consent Request, a resource presentation is generated by the ASPSP. The location header of the response will usually contain a link to the created resource.

In addition, the ASPSP can embed a hyperlink together with a “tag” for the semantics of this hyperlink into the response to these first requests and to all succeeding requests within the services,. This hyperlink then can be either a relative link, which is recommend to save space, for the host starting e.g. with “/v1/payments/sepa-credit-transfers” or it can be a global link like https://www.testbank.com/psd2/authentication/v1/payments/sepa-credit-transfers/transaction/asdf-asdf-asdf-1234.

The global links might be needed in some circumstances, e.g. a re-direct. The tag of the hyperlink transports the functionality of the resource addressed by the link, e.g. “ authorise-transaction”. This link indicates that results of a SCA method are to be posted to the resource addressed by this link to authorise e.g. a payment.

The steering hyperlinks are transported in the “_links” data element. It may contain one or several hyperlinks.

The following table gives an overview on the steering hyperlinks used in this specification as well as additional data elements, which are always transported in the context of the corresponding hyperlink. Further links might be added by ASPSP implementations.

Hyperlink	**Additional Link Related Data**	**Description**
redirect		Routing information for a redirect scenario.
oAuth		A link to the OAuth2 configuration of the ASPSP's authorisation server.
updatePsuIdentification		The link to the resource, which needs to be updated by a PSU identification.
updatePsuAuthenication	(challengeData)	The link to the payment initiation/consent resource, which needs to be updated by a PSU password and eventually the PSU identification if not delivered yet. **Remark**: In rare cases the ASPSP will ask only for some dedicated ciphers of the passwords. This information is then transported to the TPP by using the "challenge" data element, normally used only in SCA context.
updateProprietaryData		A link to the resource which needs to be updated by proprietary data. The TPP can find the scope of missing proprietary data in the ASPSP documentation. The usage of this hyperlink is not further specified in the specification but is used analogously to e.g. the updatePsuIdentification hyperlink.
selectAuthenticationMethod	authenticationMethods	This is a link to a resource, where the TPP can select the applicable strong customer authentication methods for the PSU, if there were

Hyperlink	Additional Link Related Data	Description
		several available authentication methods.
authoriseTransaction	challengeData, chosenScaMethod	A link to the resource, where a "Transaction Authorisation Request" can be sent to. This request transports the result of the SCA method performed by the customer, generating a response to the challenge data.
viewAccount		A link to an account, which can be directly used for retrieving account information from this dedicated account.
viewBalances		A link to the resource providing the balance of a dedicated account.
viewTransactions		A link to the resource providing the transaction history of a dediated amount.
self		The link to the resource created by the undergoing request. This link can be used to retrieve the resource data
status		The link to retrieve the transaction status of a resource.
first		Navigation link for account reports, if paginated.
previous		Navigation link for account reports, if paginated.
next		Navigation link for account reports, if paginated.
last		Navigation link for account reports, if paginated.

Hyperlink	Additional Link Related Data	Description
download		Download link for huge AIS data packages.

4.8 Data Extensions

The ASPSP might add more data attributes to response messages. Such extensions then shall be documented in the ASPSP's documentation of its XS2A interface. These data attributes can be either ignored by the TPP or can be interpreted as defined by the above mentioned documentation.

The ASPSP might add additional optional data attributes to be submitted, e.g. for setting up additional services. In addition, an ASPSP can ask the TPP for a submission of proprietary data in a second step via the "proprietaryData" hyperlink. This shall be published by the ASPSP in its documentation.

Remark: Before defining these additional proprietary data elements, the ASPSP is requested to submit the attribute description to the Berlin Group NextGen Taskforce, where it will be decided on a standardised approach for the related data attributes.

5 Payment Initiation Service

Remark: The API design differs across the various SCA approaches (Embedded, Redirect, OAuth2 or Decoupled, cp. [XS2A OR]), but most between the Embedded SCA Approach and the others, since the Embedded SCA Approach demands the support of the full SCA complexity within the API itself. For that reason, all data or processes, which are needed for the Embedded SCA Approach only, are shown with a light blue background, to increase the readability of the specification.

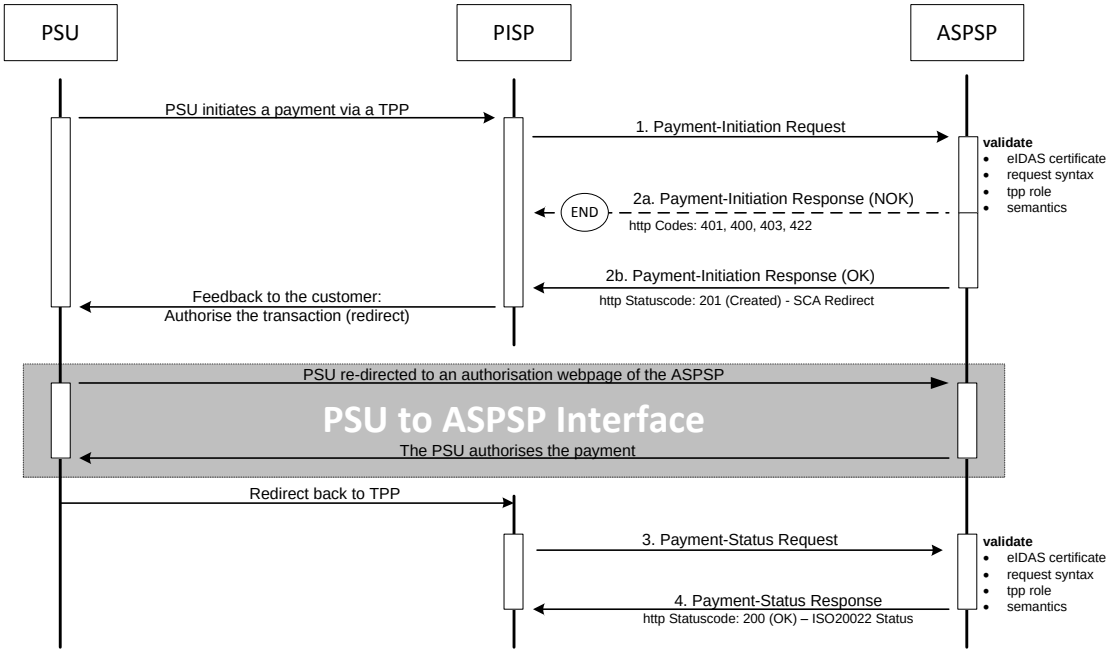
5.1 Payment Initiation Flows

The payment initiation flow depends heavily on the SCA approach implemented by the ASPSP. The most complex flow is the flow for the Embedded SCA Approach, which further differs on whether there are various authentication methods available for the PSU. In the following, the different API flows are provided as an overview for these different scenarios.

Remark: The flows do not always cover all variances or complexities of the implementation and are exemplary flows.

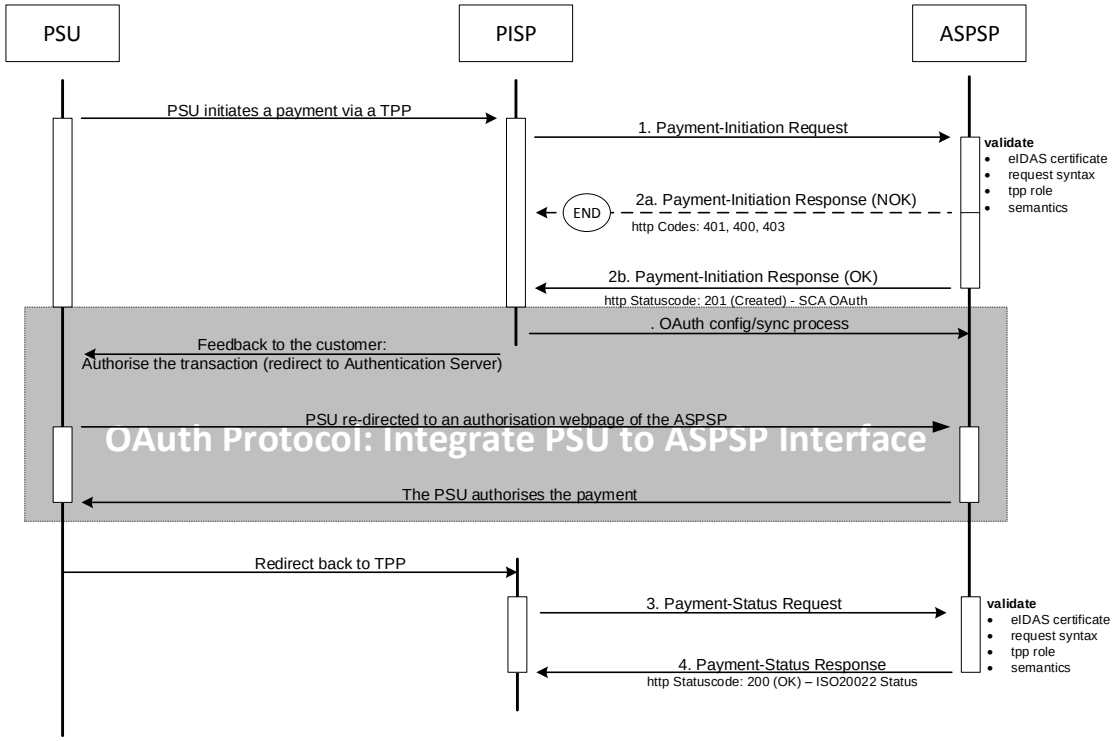
Redirect SCA Approach

If the ASPSP supports the Redirect SCA Approach, the message flow within the payment initiation service is simple. The Payment Initiation Request is followed by a redirection to the ASPSP SCA authorisation site. A status request might be requested by the TPP after the session is re-redirected to the TPP's system.



OAuth2 SCA Approach

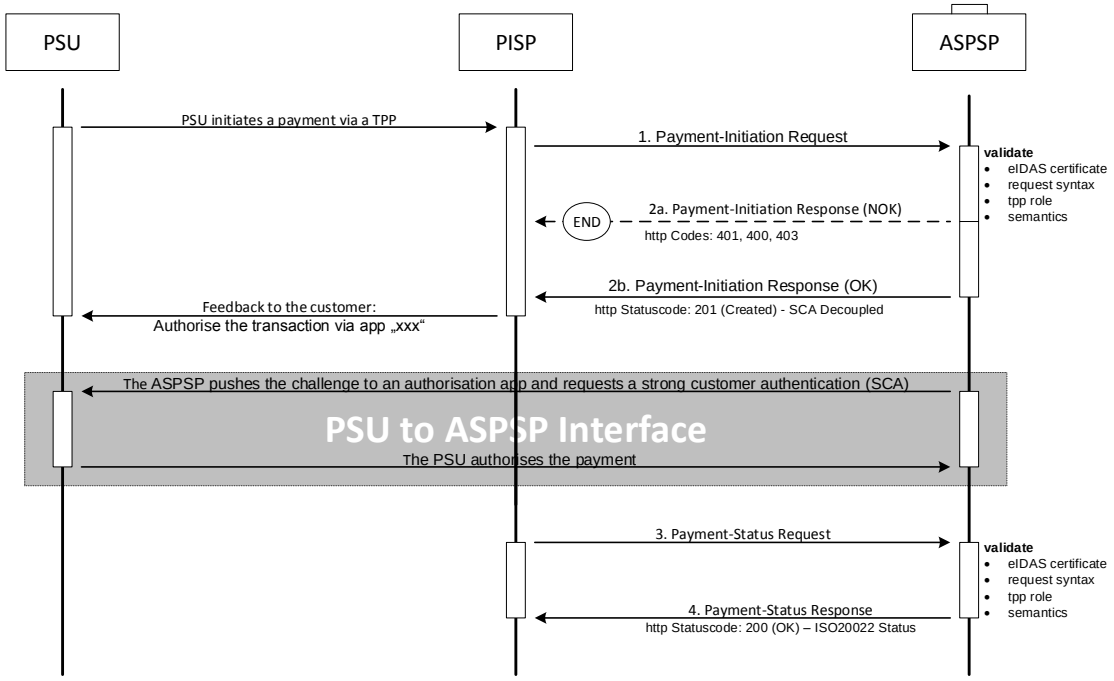
If the ASPSP supports the OAuth2 SCA Approach, the flow is very similar to the Redirect SCA Approach. Instead of redirecting the PSU directly on an authentication server, the OAuth2 protocol is used for the transaction authorisation process.



Decoupled SCA Approach

The transaction flow in the Decoupled SCA Approach is similar to the Redirect SCA Approach. The difference is that the ASPSP is asking the PSU to authorise the payment e.g. via a dedicated mobile app, or any other application or device which is independent from the online banking frontend. The ASPSP is asking the TPP to inform the PSU about this authentication by sending a corresponding PSU Message like “Please use your xxx App to authorise the payment”.

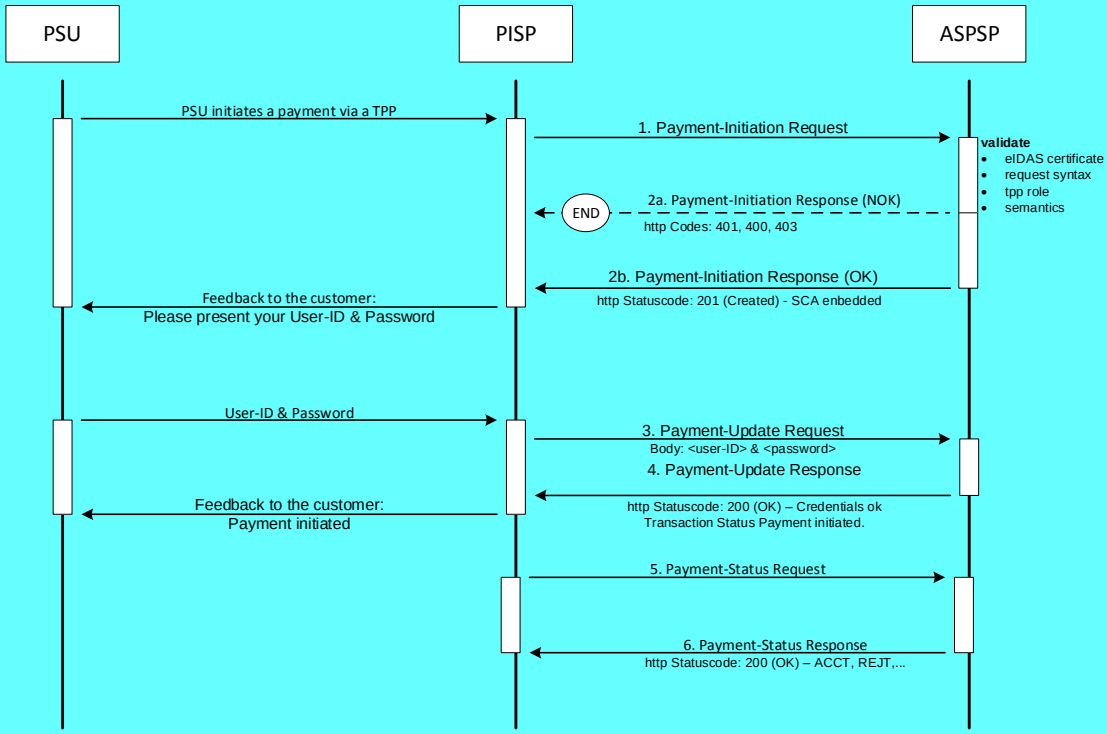
After the SCA having been processed between ASPSP and PSU, the TPP then needs to ask for the result of the transaction.



Embedded SCA Approach without SCA method (e.g. Creditor in Exemption List)

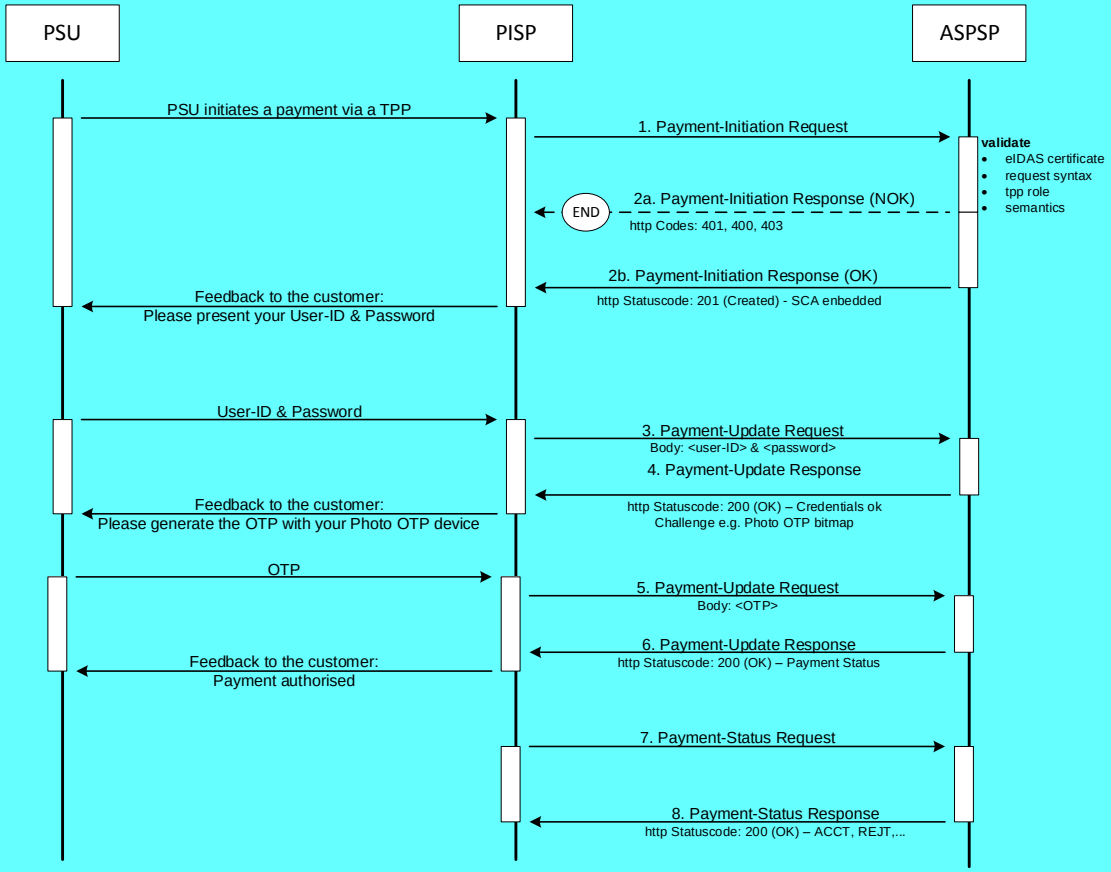
In the following, several exemplary flows are shown, where the ASPSP has chosen to process the SCA methods through the PISP – ASPSP interface. In any case, the PSU normally will need to authenticate himself with a first factor, before any account or SCA method details will be available to the PISP. So even in case where the Payment Initiation is accepted without an SCA method due e.g. to an exemption list, the PSU is asked via the PISP to provide the PSU Identification and e.g. a password or an OTP. The later exemplary flows then will show scenarios, where complexities like SCA processing and choosing an SCA method will be added.

Remark: In case where OAuth2 is requested by the ASPSP as a pre-step for PSU authentication, the sequence of the PSU authentication with the first authentication factor is omitted. This applies also for all examples for the Embedded SCA Approach.



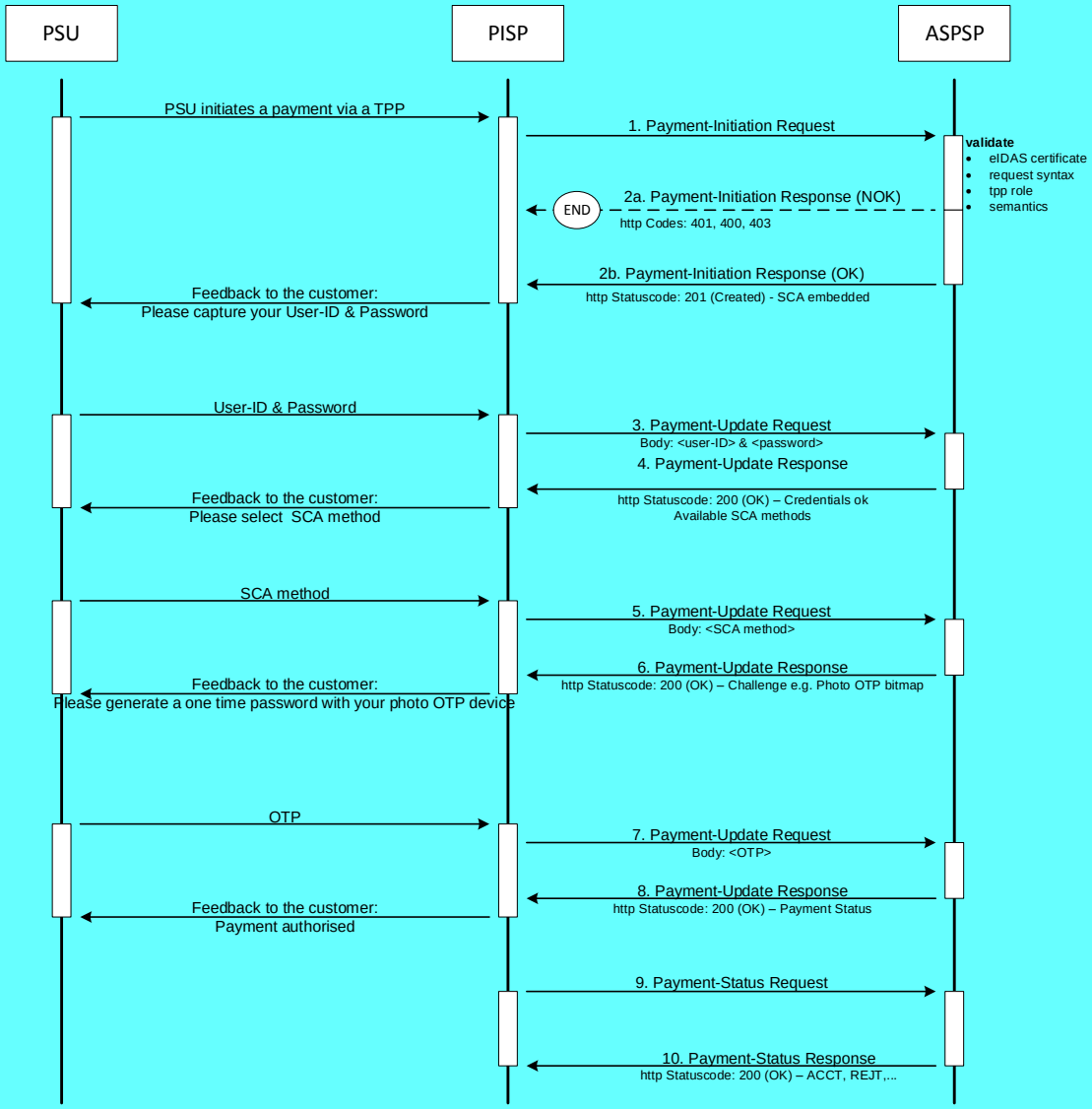
Embedded SCA Approach with only one SCA method available

In case where only one SCA method is available, the “Authorise Transaction Request” is added to the flow, where the TPP is transmitting the authentication data of the customer, e.g. an OTP with included dynamic linking to the transaction details.



Embedded SCA Approach with Selection of an SCA method

In the following flow, there is a selection of an SCA method added in case of the ASPSP supporting several SCA methods for the corresponding PSU. The ASPSP transmits first the available methods to the PISP. The PISP might filter them, if not all authentication methods can be technically supported. The available methods then are presented to the PSU for choice.



Combination of Flows due to mixed SCA Approaches

If an ASPSP supports for a PSU at least one decoupled SCA method and at the same time at least one SCA method that is not decoupled, then the above flows might be mixed as follows, since the ASPSP then needs to start the process with the assumption of one specific SCA approach to offer all available SCA methods to the PSU.

In case the ASPSP is starting the payment initiation flow with a redirect the PSU can choose on the authentication site of the ASPSP the decoupled authentication method. This is then transparent for the TPP and has no influence on the flows defined above.

In case the ASPSP is starting the payment initiation flow with the Embedded SCA Approach the ASPSP will provide a list of available SCA methods to the PSU via the TPP. If the PSU chooses an authentication method which requires the Decoupled SCA Approach, then the ASPSP is branching into the transaction flow for the Decoupled Approach as shown above: The ASPSP will return the current status of the payment initiation, e.g. “AcceptedTechnicalValidation” but will return no hyperlink for further action other than the “self” and “status” hyperlink. The next request of the TPP then needs to be the GET Status Request to get the final status of the transaction after having processed the SCA method.

In case the ASPSP needs to decide between the Decoupled and the Redirect SCA approach, the ASPSP also might first offer the SCA methods available to the PSU and then branch after the selection of the PSU into the Decoupled or Redirect SCA Approach.

5.2 Data Overview Payment Initiation Service

The following table defines the technical description of the abstract data model as defined in [XS2A OR] for the Payment Initiation service. The columns give an overview on the API protocols as follows:

- The “Data element” column is using the abstract data elements following [XS2A OR] to deliver the connection to rules and role definitions in this document.
- The “Attribute encoding” is giving the actual encoding definition within the XS2A API as defined in this document.
- The “Location” columns define, where the corresponding data elements are transported as http parameters on path, header or body level, resp. are taken from eIDAS certificates.

  Remark: Please note that website authentication certificate related data elements are not elements of the actual API call. They are indicated here, since they are mandated in the backend processing and might be transported from the API endpoint internally to the backend on the application layer. Please note, that in difference to this, the certificate data for the electronic seal can be transported within a dedicated http header field.
- The “Usage” column gives an overview on the usage of data elements in the different services and API Calls. Within [XS2A OR], the XS2A calls are described as abstract API calls. These calls will be technically realised as HTTPS POST, PUT and GET commands. The calls are divided into the following calls for Payment Initiation:
  - The Initiation Request which shall be the first API Call for every transaction within the corresponding XS2A service Payment Initiation. This call generates the corresponding resource within the Payment Initiation Service. The Payment Initiation can address a single payment, bulk payments and recurring payments. The latter are implemented as an initiation of a standing order.
  - The Update Data Call is a call, where the TPP needs to add PSU related data, which is requested in the return of the first call. This call might be repeated.
  - The Authorisation Request is only used in an Embedded SCA Approach to authorise the transaction in case a second factor authentication is needed.
  - The Status Request is used e.g. in cases, where the SCA control is taken over by the ASPSP and the TPP needs later information about the outcome.

The following usage of abbreviations in the Location and Usage columns is defined, cp. Also [XS2A OR] for details.

- x: This data element is transported on the corresponding level.
- m: Mandatory
- o : Optional for the TPP to use
- c: Conditional. The condition is described in the addressed API Calls, condition defined by the ASPSP

Data element	**Attribute encoding**	**Location**					**Usage**							
		Path	Query P.	Header	Body	Certificate[2]	Init Req.	Init Resp.	Upd. Req.	Upd. Resp	Auth. Req.	Auth Resp.	Stat. Req.	Stat. Resp
TPP Registration Number						x	m		m		m		m	
TPP Name						x	m		m		m		m	
TPP Roles						x	m		m		m		m	
TPP National Competent Authority						x	m		m		m		m	
Transaction Identification	TPP-Transaction-ID (unique ID of TPP regarding PSD2 article 46b, 47 and EBA RTS article 29)			x			m		m		m		m	
Request Identification	TPP-Request-ID			x			m		m		m		m	
Resource ID	paymentId				x			m						
Access Token (from optional OAuth2)	Authorization Bearer			x			c		c		c		c	
TPP Signing	TPP-Certificate			x			c		c		c		c	

[2] This refers to the certificate for website authentication.

Data element	Attribute encoding	Location					Usage							
		Path	Query P.	Header	Body	Certificate[2]	Init Req.	Init Resp.	Upd. Req.	Upd. Resp	Auth. Req.	Auth Resp.	Stat. Req.	Stat. Resp
Certificate														
TPP Electronic Signature	Signature			x			c		c		c		c	
Service Type		x					m		m		m		m	
Response Code				x				m		m		m		m
Transaction Status	transactionStatus				x			m		m		m		m
PSU Message Information	psuMessage				x			o		o		o		o
TPP Message Information	tppMessages				x			o		o		o		o
PSU Identification	PSU-ID			x			c		c					
PSU Identification Type	PSU-ID-Type			x			c		c					
Corporate Identification	PSU-Corporate-ID			x			c		c		c		c	
Corporate ID Type	PSU-Corporate-ID-Type			x			c		c		c		c	
PSU Password	psuData.password				x				c					
PSU Authentication Data	scaAuthenticationData				x						m			
SCA Challenge Data	challengeData				x			c		c				
IP Address PSU	PSU-IP-Address			x			m							
PSU User Agent	PSU-User-Agent[3]			x			o							
GEO Information	PSU-Geo-Location			x			o							
Redirect URL ASPSP	_links.redirect				x			c						

[3] This field transports key information for risk management like browser type or PSU device operating system. The forwarding of further http header fields might be supported in future versions of the specification to transport other device related information.

Data element	Attribute encoding	Location					Usage							
		Path	Query P.	Header	Body	Certificate[2]	Init Req.	Init Resp.	Upd. Req.	Upd. Resp	Auth. Req.	Auth Resp.	Stat. Req.	Stat. Resp
Redirect Preference	tppRedirectPreferred		x				o							
Redirect URI TPP[4]	TPP-Redirect-URI			x			c							
Payment Product	payment-product	x					m		m		m		m	

The XS2A Interface calls which represent the messages defined in [XS2A OR] will be defined in the following sections.

Remark: The request timestamp of every call is contained in the mandatory http header “Date”, cp. Section 13.21 for the formatting information. This timestamp is not contained in the data tables below because it is a mandatory http header field anyhow and because incompatibilities could appear otherwise with future more formalised specification procedures.

Remark: The AIS and PIS service is sharing some sub processes which are once described in Section 7. So, for all Update Data Request/Response Definitions as well as for Authorise Transaction Request/Response Definitions, cp. Section 7.

[4] This redirect link must be contained, if the tppRedirectPreferred flag is contained and equals "true" or if the tppRedirectPreferred flag is not used.

5.3 Payment Initiation Request

5.3.1 Payment Initiation with JSON encoding of the Payment Instruction

Call

POST /v1/payments/{payment-product}

Creates a payment initiation request at the ASPSP.

Path

Attribute	Type	Description
payment-product	String	The addressed payment product endpoint, e.g. for SEPA Credit Transfers (SCT). The default list of products supported in this standard is: • sepa-credit-transfers • instant-sepa-credit-transfers • target-2-payments • cross-border-credit-transfers The ASPSP will publish which of the payment products/endpoints will be supported. For definitions of basic non euro generic products see Annex A. Further products might be published by the ASPSP within its XS2A documentation. These new product types will end in further endpoints of the XS2A Interface.

Query Parameters

Attribute	Type	Description
tppRedirectPreferred	Boolean	If it equals “true”, the TPP prefers a redirect over an embedded SCA approach. If it equals “false”, the TPP prefers not to be redirected for SCA. The ASPSP will then choose between the Embedded or the Decoupled SCA approach, depending on the choice of the SCA procedure by the TPP/PSU.

Attribute	Type	Description
		If the parameter is not used, the ASPSP will choose the SCA approach to be applied depending on the SCA method chosen by the TPP/PSU.

Request Header

Attribute	Type	Condition	Description
Content-Type	String	Mandatory	application/json
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party.
TPP-Request-ID	UUID	Mandatory	ID of the request, unique to the call, as determined by the initiating party.
PSU-ID	String	Conditional	Client ID of the PSU in the ASPSP client interface. Might be mandated in the ASPSP's documentation. Is not contained if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in an preceeding AIS service in the same session. .
PSU-ID-Type	String	Conditional	Type of the PSU-ID, needed in scenarios where PSUs have several PSU-IDs as access possibility.
PSU-Corporate-ID	String	Conditional	Might be mandated in the ASPSP's documentation. Only used in a corporate context.
PSU-Corporate-ID-Type	String	Conditional	Might be mandated in the ASPSP's documentation. Only used in a corporate context.
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in an

Attribute	Type	Condition	Description
			preceeding AIS service in the same session.
PSU-Consent-ID	String	Optional	This data element may be contained, if the payment initiation transaction is part of a session, i.e. combined AIS/PIS service. This then contains the consentId of the related AIS consent, which was performed prior to this payment initiation.
PSU-Agent	string	Optional	The forwarded Agent header field of the http request between PSU and TPP.
PSU-IP-Address	string	Mandatory	The forwarded IP Address header field consists of the corresponding http request IP Address field between PSU and TPP.
PSU-Geo-Location	Geo Location	Optional	The forwarded Geo Location of the corresponding http request between PSU and TPP if available.
TPP-Redirect-URI	String	Conditional	URI of the TPP, where the transaction flow shall be redirected to after a Redirect
Signature	Cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Must be contained if a signature is contained, see above.

Request Body

The payment data to be transported in the request body is dependent on the chosen API endpoint. Some standard definitions related to the above mentioned standard products are defined in Section 10 of this document. Further definitions might be given community or ASPSP specific. In Annex A, a list of community specific payment product definitions and links regarding community/ASPSP specific payment product definitions are given. ASPSP or community definitions should reuse standard attribute names.

Response Header

The Location field is used as link to the created resource. No other specific requirements.

Response Body

Attribute	Type	Condition	Description
transactionStatus	Transaction Status	Mandatory	The values defined in Section 13.18 might be used.
paymentId	String	Mandatory	resource identification of the generated payment initiation resource.
transactionFees	Amount	Optional	Can be used by the ASPSP to transport transaction fees relevant for the underlying payments.
transactionFee Indicator	Boolean	Optional	If equals “true”, the transaction will involve specific transaction cost as shown by the ASPSP in their public price list or as agreed between ASPSP and PSU. If equals “false”, the transaction will not involve additional specific transaction costs to the PSU.
scaMethods	Array of authentication objects	Conditional	This data element might be contained, if SCA is required and if the PSU has a choice between different authentication methods. Depending on the risk management of the ASPSP this choice might be offered before or after the PSU has been identified with the first relevant factor, or if an access token is transported. If this data element is contained, then there is also an hyperlink of type “selectAuthenticationMethods” contained in the response body. These methods shall be presented towards the PSU for selection by the TPP.
chosenSca Method	Authentication object	Conditional	This data element is only contained in the response if the APSPS has chosen the Embedded SCA Approach, if the PSU is already identified e.g. with the first relevant factor or alternatively an access token, if SCA is required and if the authentication method is implicitly selected.

Attribute	Type	Condition	Description
challengeData	Challenge	Conditional	It is contained in addition to the data element chosenScaMethod if challenge data is needed for SCA. In rare cases this attribute is also used in the context of the psuAuthentication link.
_links	Links	Mandatory	A list of hyperlinks to be recognised by the TPP. The actual hyperlinks used in the response depend on the dynamical decisions of the ASPSP when processing the request. **Remark**: All links can be relative or full links, to be decided by the ASPSP. Type of links admitted in this response, (further links might be added for ASPSP defined extensions): “redirect”: In case of an SCA Redirect Approach, the ASPSP is transmitting the link to which to redirect the PSU browser. “oAuth”: In case of a SCA OAuth2 Approach, the ASPSP is transmitting the URI where the configuration of the Authorisation Server can be retrieved. The configuration follows the OAuth 2.0 Authorisation Server Metadata specification. “updatePsuIdentification”: The link to the payment initiation resource, which needs to be updated by the PSU identification. This might be used in an embedded, redirect or decoupled SCA Approach, where the PSU ID was missing in the first request. “updatePsuAuthenication”: The link to the payment initiation resource, which needs to be updated by a PSU password and eventually the PSU identification if not delivered yet. This is used in case of the

Attribute	Type	Condition	Description
			Embedded or Decoupled SCA approach. “selectAuthenticationMethod” : This is a link to a resource, where the TPP can select the applicable strong customer authentication methods for the PSU, if there were several available authentication methods. This link contained under exactly the same conditions as the data element “authenticationMethods”, see above. “authoriseTransaction” : The link to the payment initiation resource, where the “Payment Authorisation Request” is sent to. This is the link to the resource which will authorise the payment by checking the SCA authentication data within the Embedded SCA approach. “self” : The link to the payment initiation resource created by this request. This link can be used to retrieve the resource data. “status”: The link to retrieve the transaction status of the payment initiation.
psuMessage	String	Optional	Text to be displayed to the PSU
tppMessages	Array of Message	Optional	Messages to the TPP on operational issues.

Example

Request

```
POST https://api.testbank.com/v1/payments/sepa-credit-transfers
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7721
PSU-IP-Address:       192.168.8.78
PSU-GEO-Location:     GEO:52.506931,13.144558
PSU-Agent:            Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date:                 Sun, 06 Aug 2017 15:02:37 GMT

{
   "instructedAmount": {"currency": "EUR" , "content": "123.50"},
   "debtorAccount": { "iban" : "DE23100100101234567890"},
   "creditorName": "Merchant123" ,
   "creditorAccount": {"iban" : "DE23100120020123456789"},
   "remittanceInformationUnstructured": "Ref Number Merchant"
}
```

Response in case of a redirect

Response Code 201

Response Body

```
{
  "transactionStatus": "Received",
  "paymentId": "1234-wertiq-983",
  "_links": {
        "redirect": "https://www.testbank.com/asdfasdfasdf",
        "self": "/v1/payments/sepa-credit-transfers/1234-wertiq-983"
  }
}
```

Response in case of an OAuth2 response

Response Code 201

Response Body

```
{
  "transactionStatus" : "Received",
  "paymentId": "1234-wertiq-983",
  "_links": {
        "oAuth":   "https://www.testbank.com/oauth/.well-known/oauth-
authorization-server",
        "self": "/v1/payments/sepa-credit-transfers/1234-wertiq-983"
  }
}
```

Response in case of the decoupled approach

Response Code 201

Response Header

Response Body

```
{
  "transactionStatus" : "Received",
  "paymentId": "1234-wertiq-983",
  "_links": {
     "updatePsuIdentification": "/v1/payments/sepa-credit-
transfers/1234-wertiq-983",
     "self": "/v1/payments/sepa-credit-transfers/1234-wertiq-983"
    }
}
```

Response in case of the embedded approach

Response Code 201

```
{
    "transactionStatus": "Received",
    "paymentId": "1234-wertiq-983",
    "_links": {
        "updatePsuAuthenication":          "/v1/payments/sepa-credit-
transfers/1234-wertiq-983"
}
```

5.3.2 Payment Initiation with pain.001 XML message as Payment Instruction

Call

POST /v1/payments/{payment-product}

Creates a payment initiation request at the ASPSP.

Remark: The underlying pain.001 structure which is transported in the content body of this request may only contain one payment. In cases of the initiation of multiple payments, the endpoint defined in Section 5.3.3.2 shall be used.

Path

Attribute	Type	Description
payment-product	String	The addressed payment product, e.g. SCT. The default list of products supported in this standard is: • pain.001-sepa-credit-transfers • pain.001-instant-sepa-credit-transfers • pain.001-target-2-payments • pain.001-cross-border-credit-transfers Further products might be published by the ASPSP within its XS2A documentation. **Remark**: For all SEPA Credit Transfer based endpoints which accept XML encoding, the XML pain.001 schemes provided by EPC are supported by the ASPSP as a minimum for the body content. Further XML schemes might be supported by some communities. **Remark**: For cross-border and target-2-payments only community wide pain.001 schemes do exist, cp. Annex A, Section 15

Query Parameters

The same query parameter definition as in Section 5.3.1 applies.

Request Header

The same header as in Section 5.3.1, only the content type indicates XML encoding (“application/xml”).

Request Body

A pain.001 structure corresponding to the chosen payment product, see above on XML schema support.

Response

The same response as in Section 5.3.1.

Example

Request

```
POST https://api.testbank.com/v1/payments/pain.001-sepa-credit-transfers

Content-Encoding:     gzip
Content-Type:         application/xml
TPP-Transaction-ID:   "PI-123456789"
PSU-IP-Address:       "192.168.8.78"
PSU-Agent:            "Chrome_v12"

<Document xmlns="urn:iso:std:iso:20022:tech:xsd:pain.001.001.03">
  <CstmrCdtTrfInitn>
    <GrpHdr>
      <MsgId>MIPI-123456789RI-123456789</MsgId>
      <CreDtTm>2017-02-14T20:23:34.000Z</CreDtTm>
      <NbOfTxs>1</NbOfTxs>
      <CtrlSum>123</CtrlSum>
      <InitgPty>
        <Nm>PaymentInitiator</Nm>
        <Id><OrgId><Othr><Id>DE10000000012</Id>
          <SchmeNm><Prptry>PISP</Prptry></SchmeNm></Othr></OrgId></Id>
      </InitgPty>
    </GrpHdr>
    <PmtInf>
      <PmtInfId>BIPI-123456789RI-123456789</PmtInfId>
      <PmtMtd>TRF</PmtMtd>
      <NbOfTxs>1</NbOfTxs>
      <CtrlSum>123</CtrlSum>
      <PmtTpInf><SvcLvl><Cd>SEPA</Cd></SvcLvl></PmtTpInf>
      <ReqdExctnDt>2017-02-15</ReqdExctnDt>
      <Dbtr><Nm>PSU Name</Nm></Dbtr>
      <DbtrAcct><Id><IBAN>DE87200500001234567890</IBAN></Id></DbtrAcct>
      <ChrgBr>SLEV</ChrgBr>
      <CdtTrfTxInf>
```

```
        <PmtId><EndToEndId>RI-123456789</EndToEndId></PmtId>
        <Amt><InstdAmt Ccy="EUR">123</InstdAmt></Amt>
        <Cdtr><Nm>Merchant123</Nm></Cdtr>
        <CdtrAcct><Id><IBAN> DE23100120020123456789</IBAN></Id></CdtrAcct>
        <RmtInf><Ustrd>Ref Number Merchant-123456</Ustrd></RmtInf>
      </CdtTrfTxInf>
    </PmtInf>
  </CstmrCdtTrfInitn>
</Document>
```

Response Body

See above (JSON encoding)

5.3.3 Payment Initiation for Bulk Payments and Multiple Payments

The Online Banking frontends might support the

- upload of bulks or
- multiple payment functions, where a PSU can enter multiple payment data before authorisation all the payments with one SCA method.

Both functions are modelled as bulk payment in the XS2A interface. The multiple payment function can be offered by the TPP towards the PSU in its own GUI. In the XS2A interface this function is always supported as bulk payment. This function is an **optional** function of the ASPSP in the XS2A interface. It can be offered by the ASPSP in JSON or XML modelling of the payment data, i.e. the body content.

5.3.3.1 Bulk Payment Initiation with JSON encoding of the Payment Instruction

Call

POST /v1/bulk-payments/{payment-product}

Creates a bulk payment initiation request at the ASPSP.

Path

Attribute	Type	Description
payment-product	String	The addressed payment product endpoint for bulk payments e.g. for a bulk SEPA Credit Transfers (SCT). These endpoints are optional. Some default names are:

Attribute	Type	Description
		• sepa-credit-transfers • instant-sepa-credit-transfers • target-2-payments • cross-border-credit-transfers The ASPSP will publish which of the payment products/endpoints will be supported. For definitions of basic non euro generic products see Annex A. Further products might be published by the ASPSP within its XS2A documentation. These new product types will end in further endpoints of the XS2A Interface.

The same query parameter and http header definition as in Section 5.3.1 applies.

The body definition with the JSON based SEPA bulk payments is contained in Section 10, further definitions for non SEPA payments in Section 15 (Annex A). The response definition is analogous to the initiation of single payments, cp. Section 5.3.1.

5.3.3.2 Bulk Payment Initiation with XML encoding of the Payment Instruction

Call

POST /v1/bulk-payments/{payment-product}

Creates a bulk payment initiation request at the ASPSP.

Path

Attribute	Type	Description
payment-product	String	The addressed payment product endpoint for bulk payments e.g. for a bulk SEPA Credit Transfers (SCT). These endpoints are optional. Some default names are: • pain.001-sepa-credit-transfers • pain.001-instant-sepa-credit-transfers • pain.001-proprietary-credit-transfers The ASPSP will publish which of the payment

Attribute	Type	Description
		products/endpoints will be supported. **Remark**: For all SEPA Credit Transfer based endpoints which accept XML encoding, the XML pain.001 schemes provided by EPC are supported by the ASPSP as a minimum for the body content. Further XML schemes might be supported by some communities. **Remark**: Payment Initiations might be further restricted by the ASPSP on size or on multiplicity of entries. This could be e.g. a restriction on the usage of one ordering party or/and one debtor account. **Remark**: For proprietary payments only community wide pain.001 schemes do exist, cp. Annex A, Section 15.

The same query parameter and http header definition as in Section 5.3.2 applies.

The response definition is analogous to the initiation of single XML based payments, cp. Section 5.3.2.

5.3.4 Initiation for Standing Orders for Recurring/Periodic Payments

The recurring payments initiation function will be covered in this specification as a specific standing order initiation: The TPP can submit a recurring payment initiation where the starting date, frequency and conditionally an end date is provided. Once authorised by the PSU, the payment then will be executed by the ASPSP, if possible, following this “standing order” as submitted by the TPP. No further TPP action is needed. This payment is called a periodic payment in this context to differentiate the payment from recurring payment types, where third parties are initiating the same amount of money e.g. payees for using credit card transactions or direct debits for reccuring payments of goods or services. These latter types of payment initiations are not part of this interface.

5.3.4.1 Standing Orders for Recurring/Periodic Payments in JSON encoding

Call

POST /v1/periodic-payments/{payment-product}

Path Parameters

The same path parameter to determine the underlying payment type of the recurring payment as in Section 5.3.1 applies.

Query Parameters

The same query parameter definition as in Section 5.3.1 applies.

Request Header

For this initiation the same header as in Section 5.3.1 is used.

Request Body

First, any tag of the underlying payment as defined in Section 10.1 can be used. In addition the following tags are used:

Tag	Type	Usage	Description
startDate	ISODate	Mandatory	The first applicable day of execution starting from this date is the first payment.
executionRule	String	Optional	“following” or “preceeding” supported as values. This data attribute defines the behavior when recurring payment dates falls on a weekend or bank holiday. The payment is then executed either the “preceeding” or “following” working day. ASPSP might reject the request due to the communicated value, if rules in Online-Banking are not supporting this execution rule.
endDate	ISODate	Optional	The last applicable day of execution If not given, it is an infinite standing order.
frequency	Frequency Code	Mandatory	The frequency of the recurring payment resulting from this standing order.
dayOfExecution	DD	Conditional	"31" is ultimo

Response

The formats of the Payment Initiation Response resp. the subsequent transaction authorisation process for standing orders with JSON based payment data equals the corresponding Payment Initiation Response resp. the subsequent transaction authorisation process for a single payment containing JSON based payment data.

Remark: Please note that for the payment initiation of standing orders, the ASPSP will always mandate an SCA with dynamic linking, exemptions are not permitted.

Example for Variant 1 with full JSON encoding

```
POST https:/v1/periodic-payments/sepa-credit-transfers
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7721
PSU-IP-Address:       192.168.8.78
PSU-Agent:            Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date:                 Sun, 06 Aug 2017 15:02:37 GMT
{
 "instructedAmount": {"currency" : "EUR" , "content" : "123"},
 "debtorAccount": {"iban" : "DE23100100101234567890"},
 "creditorName": "Merchant123",
 "creditorAccount": {"iban" : "DE23100120020123456789"},
 "remittanceInformationUnstructured": "Ref Number Abonnement",
 "startDate": "2018-03-01",
 "executionRule": "latest",
 "frequency": "monthly",
 "dayOfExecution" : "01"
}
```

5.3.4.2 Payment Initiation for Standing Orders with XML based payment data

The standing order management data will be JSON based in the XS2A API also if the related payment data is based on XML syntax. For this reason, the Payment Initiation Request for standing orders is defined as a http multipart message in this case.

Call

POST /v1/periodic-payments/{product-name}

Path Parameters

The same path parameter to determine the underlying payment type of the recurring payment as in Section 5.3.2 applies.

Query Parameters

The same query parameter and http header definition as in Section 5.3.1 applies.

Request Header

The same header definitions as in Section 5.3.1 are used with the exception of the Content-Type Header. Here the following requirement applies:

Attribute	Type	Condition	Description
Content-Type	String	Mandatory	multipart/form-data; boundary=AaaBbbCcc

Request Body, Part 1

The first part of the body contains first a sub-header section as defined by the following table:

Attribute	Type	Condition	Description
Content-Disposition	String	Mandatory	form-data; name=”xml_sct”
Content-Type	String	Mandatory	application/xml

The first part content of the body is defined as for the Payment Initiation Request for a single request in an XML (pain.001) based format, cp. Section 5.3.2.

Request Body, Part 2

The second part of the body contains first a sub-header section as defined by the following table:

Attribute	Type	Condition	Description
Content-Disposition	String	Mandatory	form-data; name=”json_standingorderType”
Content-Type	String	Mandatory	application/json

The second part content of the body is defined as follows:

Tag	Type	Usage	Description
startDate	ISODate	Mandatory	The first applicable day of execution starting from this date is the first payment.
executionRule	String	Optional	"latest" or "earliest" as values. ASPSP might ignore this flag, depending on their rules in Online-Banking
endDate	ISODate	Optional	The last applicable day of execution If not given, it is an infinite standing order.
frequency	Frequency Code	Mandatory	Frequency of the recurring payment resulting from this standing order.
dayOfExecution	DD	Conditional	"31" is ultimo

Response

The formats of the Payment Initiation Response resp. the subsequent transaction authorisation process for standing orders with XML based payment data equals the corresponding Payment Initiation Response resp. the subsequent transaction authorisation process for a single payment containing XML based payment data.

Example with JSON Management Information and XML Payment Information

```
POST https:/v1/periodic-payments/sepa-credit-transfers
Content-Encoding: gzip
Content-Type: multipart/form-data; boundary=AaaBbbCcc
--AaaBbbCcc
Content-Disposition: form-data; name=”xml_sct”
Content-Type: application/xml
<Document xmlns=”urn:iso:std:iso:20022:tech:xsd:pain.001.001.03”>
  <CstmrCdtTrfInitn>
    <GrpHdr>
      <MsgId>MIPI-123456789RI-123456789</MsgId>
      <CreDtTm>2017-02-14T20:23:34.000Z</CreDtTm>
      <NbOfTxs>1</NbOfTxs>
      <CtrlSum>123</CtrlSum>
      <InitgPty>
        <Nm>PaymentInitiator</Nm>
        <Id><OrgId><Othr><Id>DE10000000012</Id>
              <SchmeNm><Prptry>PISP</Prptry></SchmeNm></Othr></OrgId></Id>
      </InitgPty>
    </GrpHdr>
    <PmtInf>
      <PmtInfId>BIPI-123456789RI-123456789</PmtInfId>
      <PmtMtd>TRF</PmtMtd>
      <NbOfTxs>1</NbOfTxs>
      <CtrlSum>123</CtrlSum>
      <PmtTpInf><SvcLvl><Cd>SEPA</Cd></SvcLvl></PmtTpInf>
      <ReqdExctnDt>2017-02-15</ReqdExctnDt>
      <Dbtr><Nm>PSU Name</Nm></Dbtr>
      <DbtrAcct><Id><IBAN>DE87200500001234567890</IBAN></Id></DbtrAcct>
      <ChrgBr>SLEV</ChrgBr>
      <CdtTrfTxInf>
        <PmtId><EndToEndId>RI-123456789</EndToEndId></PmtId>
        <Amt><InstdAmt Ccy=“EUR“>123</InstdAmt></Amt>
        <Cdtr><Nm>Merchant123</Nm></Cdtr>
        <CdtrAcct><Id><IBAN> DE23100120020123456789</IBAN></Id></CdtrAcct>
        <RmtInf><Ustrd>Ref Number Merchant-123456</Ustrd></RmtInf>
      </CdtTrfTxInf>
    </PmtInf>
  </CstmrCdtTrfInitn>
</Document>
--AaaBbbCcc
Content-Disposition: form-data; name=”json_standingordermanagement”
Content-Type: application/json
{“startDate” : “2018-03-01”,
 “frequency” : “monthly”,
```

```
 "executionRule": "latest",
 "dayOfExecution" : "01"
}
--AaaBbbCcc—
```

5.4 Get Status Request

Call

GET /v1/payments/{payment-product}/{paymentId}/status

Can check the status of a payment initiation.

Path

Attribute	Type	Description
payment-product	String	Payment product of the related payment.
paymentId	String	Resource Identification of the related payment.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	
TPP-Request-ID	UUID	Mandatory	
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in the current PIS transaction or in a preceeding AIS service in the same session, if no such OAuth2 SCA approach was chosen in the current PIS transaction.
Signature	cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.

Attribute	Type	Condition	Description
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Must be contained if a signature is contained, see above.

Query Parameters

No specific query parameters defined.

Request Body

No body.

Response Body in Case of JSON based endpoint

Attribute	Type	Condition	Description
transactionStatus	Transaction Status	Mandatory	In case where the Payment Initiation Request was JSON encoded as defined in Section 5.3.1, the status is returned in this JSON based encoding.

Response Body in Case of (SEPA-)XML based endpoint

If the Payment Initiation Request is encoded in XML, cp. Section 5.3.2, then the status is returned as a pain.002 structure. The chosen XML schema of the Status Request is following the XML schema definitions of the original pain.001 schema.

Example for JSON based endpoint

Request

```
GET https://api.testbank.com/v1/payments/sepa-credit-
transfers/qwer3456tzui7890/status
Accept                  application/json
TPP-Transaction-ID      3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID          99391c7e-ad88-49ec-a2ad-99ddcb1f7721
Date                    Sun, 06 Aug 2017 15:04:07 GMT
```

Response Code 200

```
Content-Type            application/json
{
  "transactionStatus" : "AcceptedCustomerProfile"
}
```

Example for XML based endpoint

```
GET       https://api.testbank.com/v1/payments/pain.001-sepa-credit-
transfers/qwer3456tzui7890/status

Accept                 application/xml
TPP-Transaction-ID     3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID         99391c7e-ad88-49ec-a2ad-99ddcb1f7721
Date                   Sun, 06 Aug 2017 15:04:07 GMT

Response Code 200

Content-Type:          application/xml
<Document xmlns=”urn:iso:std:iso:20022:tech:xsd:pain.002.001.03”>
..<CstmrPmtStsRpt>
….<GrpHdr>
……<MsgId>4572457256725689726906</MsgId>
......<CreDtTm>2017-02-14T20:24:56.021Z</CreDtTm>
......<DbtrAgt><FinInstnId><BIC>ABCDDEFF</BIC></FinInstnId></DbtrAgt
>
......<CdtrAgt><FinInstnId><BIC>DCBADEFF</BIC></FinInstnId></CdtrAgt
>
....</GrpHdr>
....<OrgnlGrpInfAndSts>
......<OrgnlMsgId>MIPI-123456789RI-123456789</OrgnlMsgId>
......<OrgnlMsgNmId>pain.001.001.03</OrgnlMsgNmId>
......<OrgnlCreDtTm>2017-02-14T20:23:34.000Z</OrgnlCreDtTm>
......<OrgnlNbOfTxs>1</OrgnlNbOfTxs>
......<OrgnlCtrlSum>123</OrgnlCtrlSum>
......<GrpSts>ACCT</GrpSts>
....</OrgnlGrpInfAndSts>
....<OrgnlPmtInfAndSts>
......<OrgnlPmtInfId>BIPI-123456789RI-123456789</OrgnlPmtInfId>
......<OrgnlNbOfTxs>1</OrgnlNbOfTxs>
......<OrgnlCtrlSum>123</OrgnlCtrlSum>
......<PmtInfSts>ACCT</PmtInfSts>
....</OrgnlPmtInfAndSts>
..</CstmrPmtStsRpt>
</Document>
```

6 Account Information Service

This specification foresees different types of account information services:

- Transaction reports for a given account including balances if applicable.
- Balances of a given account,
- A list of available accounts,
- Account details of a given account or of the list of all accessible accounts relative to a granted consent

Hereby the definition of the list of available and accessible accounts is as follows:

Definition: The list of **available** accounts of an ASPSP related to a PSU is the list of accounts of a PSU which are open for access through the XS2A interface according to the definition of payment accounts provided by [PSD2].

Definition: The list of **accessible** accounts of an ASPSP related to a PSU's consent is the list of accounts, where the consent of the PSU has been granted to at least one of the defined account information types.

Note: The Read Data Request for the list of available accounts and for account details of a given account is syntactically identical. The difference is only in the underlying consent resource, referred to through the http header parameter "Consent-id".

Example: An ASPSP is providing IBAN1 and IBAN2 to a PSU. The PSU has granted the TPP the consent to access transactions and balances of IBAN1. In this case, the available accounts are IBAN1 and IBAN2, the list of accessible accounts consists only of IBAN1.

Within this specification, the Account Information Service is separated in two phases:

- Establish Account Information Consent

  Within this phase of the Account Information Service, the PSU is giving the consent to the AISP on

  - the type of Account Information Service to grant an access to (see list at the beginning of this section),
  - the multiplicity of the Account Information Service, i.e. a one-off or recurring access, and
  - in the latter case on the duration of the consent in days or the maximum offeredby the ASPSP and optionally the frequency of a recurring request.

The result of this process is a consent resource. A link to this resource is returned to the AISP within this process. The TPP can retrieve the consent object by submitting a GET method on this resource. This object contains the detailed access rights, a consent-ID token and delivers aliases to the accounts used during the account access for addressing the accounts.

Note: The establishing of the AIS consent is one transaction following [XS2A-OR], thus all corresponding messages will use the same transaction identification in TPP-Transaction-ID.

- Read Account Data

  Within this phase, the AISP gets access to the account data as defined by the PSU's consent, see above. The Read Account Data Request is addressing the corresponding consent resource by using the above mentioned link to this resource.

  The Read Account Data Request will indicate

  - the type of account data to be accessed,
  - the identification of the addressed account, where applicable,
  - whether a PSU has directly initiated the request real-time,
  - whether balances should be delivered in addition where applicable,
  - in case of transaction reports as Account Information type additionally
    - the addressed account identification and
    - the period of the transaction report
    - in addition optionally a delta-flag indicating the request for a delta-report relative to the last request with additional data.
    - the preferred formats of the transaction reports.

  For the account access, the usual bank accounts and (credit) card accounts are separated on end-points, since the data is usually separated in the ASPSP backend.

  In case of a one-off consent, the access might be denied if the AISP is requesting the data more than once or if the validity of the consent has been timed out, e.g. after 20 minutes of the finalisation of the consent mechanism, depending on the ASPSP implementation.

The read data access will be further denied in case where the type of Account Information Service does not comply with the consented service, or if the actual access is not matching the consented duration or frequency.

If the PSU's consent is given to access a list of accounts, the frequency of the access is checked by the ASPSP per account that has been accessed and per PSU that has given consent for the access.

Note: The several Read account data transactions are own transactions following [XS2A-OR], thus a transaction identification will only be used several times in case of pagination while reading transaction lists/account statements.

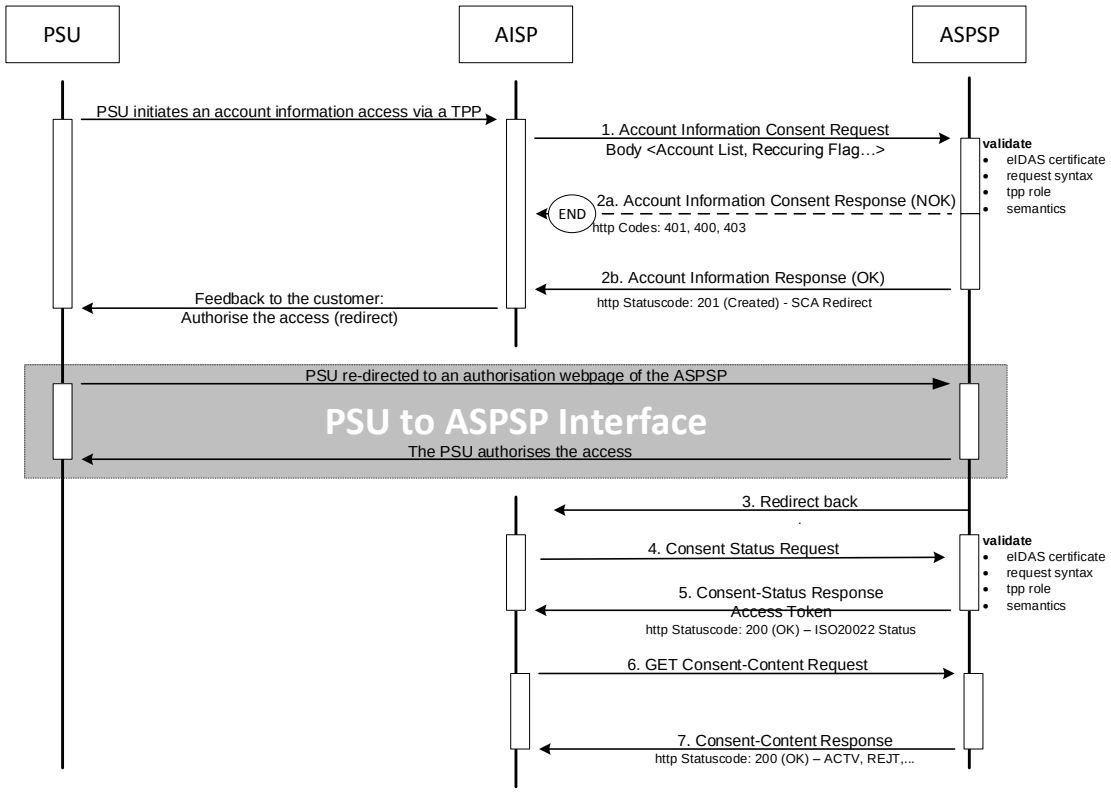
6.1 Account Information Service Flows

As for the payment initiation, please note that the following flows do not cover all possible variances and are exemplary flows.

6.1.1 Account Information Consent Flow

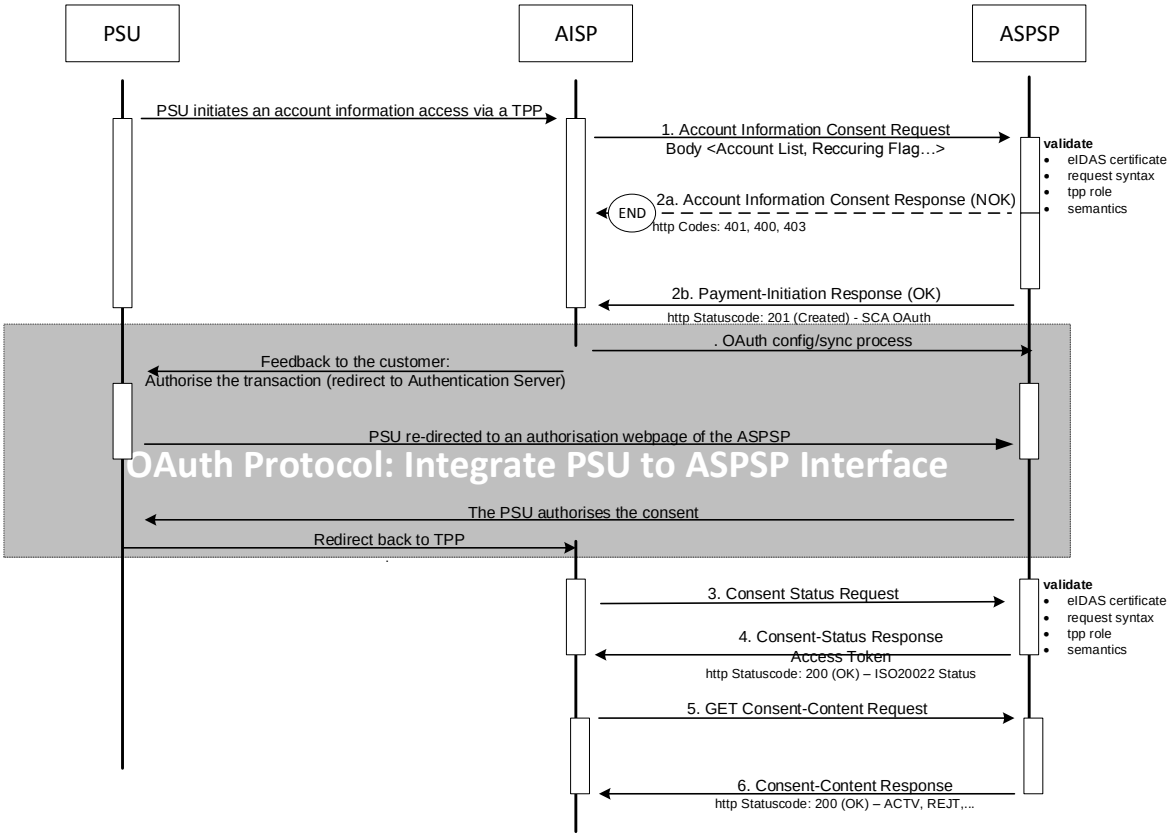
Redirect SCA Approach

If the ASPSP supports the Redirect SCA Approach, the message flow within the Account Information Consent sub-service is simple. The Account Information Consent Request is followed by a redirection to the ASPSP SCA authorisation site. A status or content request on the created consent resource might be requested by the TPP after the session is re-redirected to the TPP's system.



OAuth2 SCA Approach

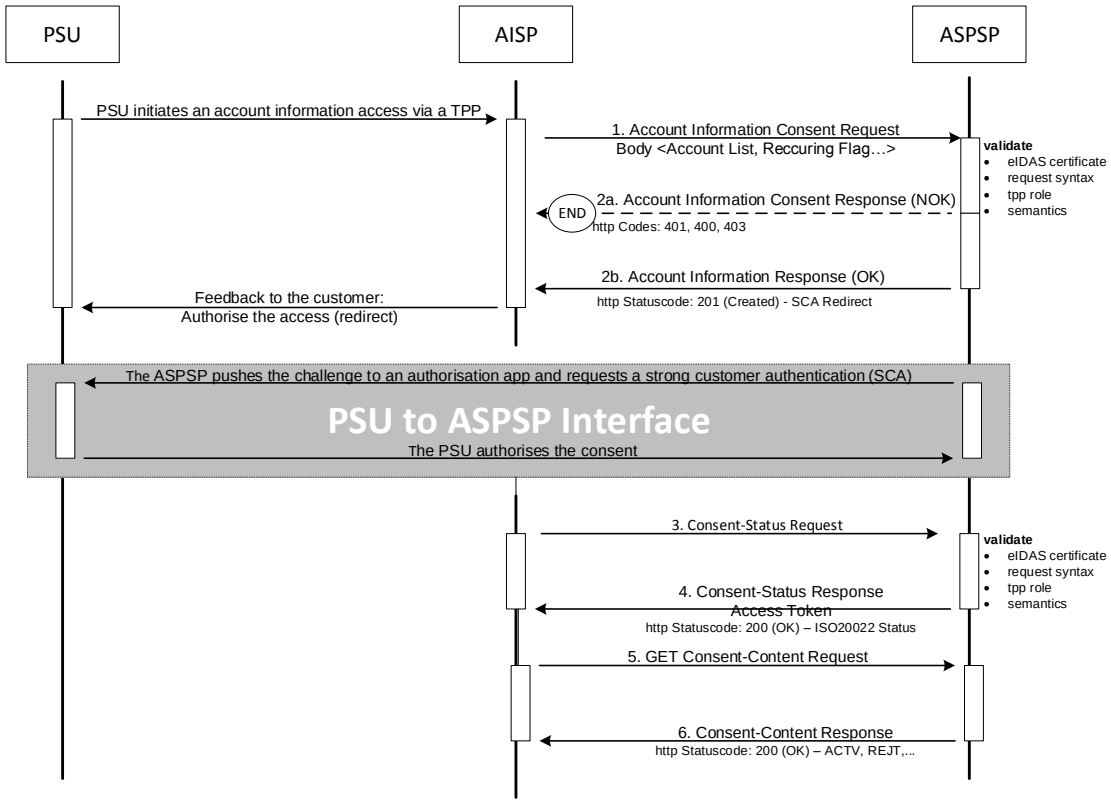
If the ASPSP supports the OAuth2 SCA Approach, the flow is very similar to the Redirect SCA Approach. Instead of redirecting the PSU directly on an authentication server, the OAuth2 protocol is used for the transaction authorisation process.



Decoupled SCA Approach

The transaction flow in the Decoupled SCA Approach is similar to the Redirect SCA Approach. The difference is that the ASPSP is asking the PSU to authorise the account access consent e.g. via a dedicated mobile app. The ASPSP is asking the TPP to inform the PSU about this authentication by sending a corresponding PSU Message like “Please use your xxx App to authorise the account access”.

After the SCA between ASPSP and PSU, the TPP then needs to ask for the result of the transaction.

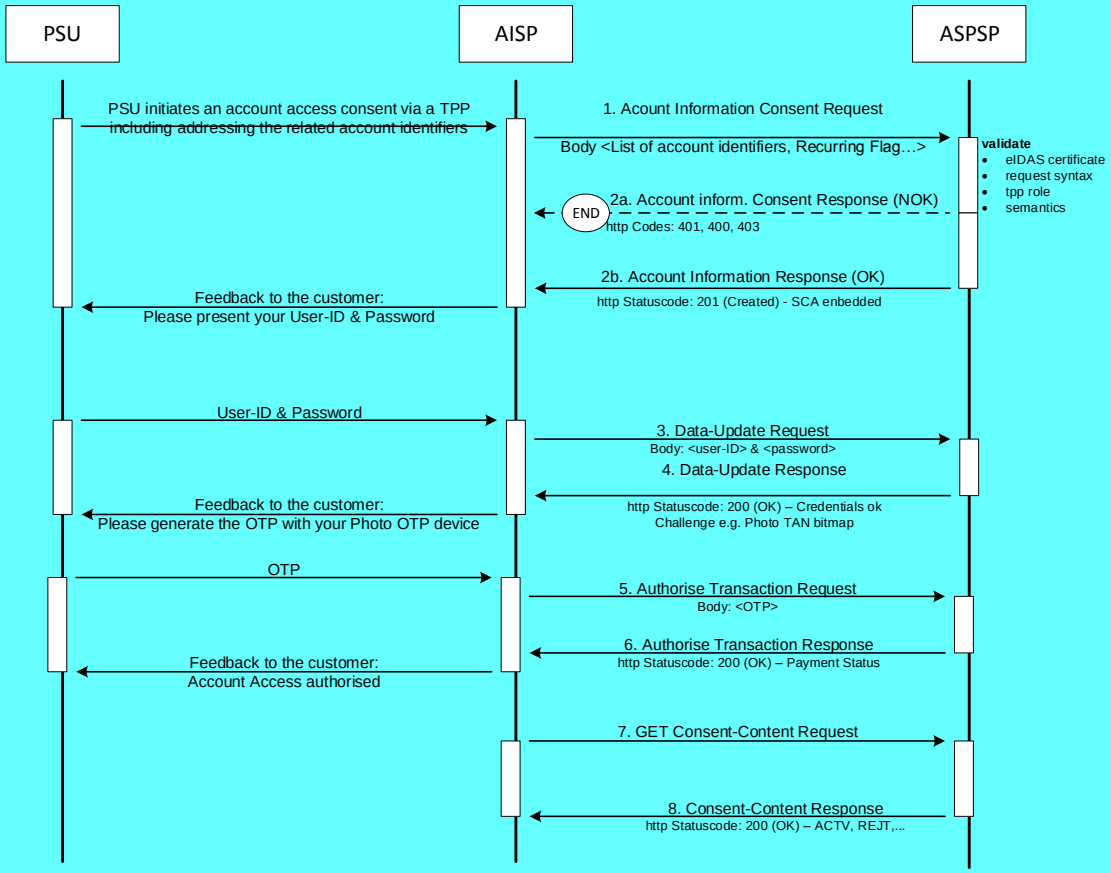


Embedded SCA Approach with only one SCA method available

In the following, several exemplary flows are shown, where the ASPSP has chosen to process the SCA methods for the consent approval through the PISP – ASPSP interface. In any case, the PSU normally will need to authenticate himself with a first factor, before any account or SCA method details will be available to the PISP.

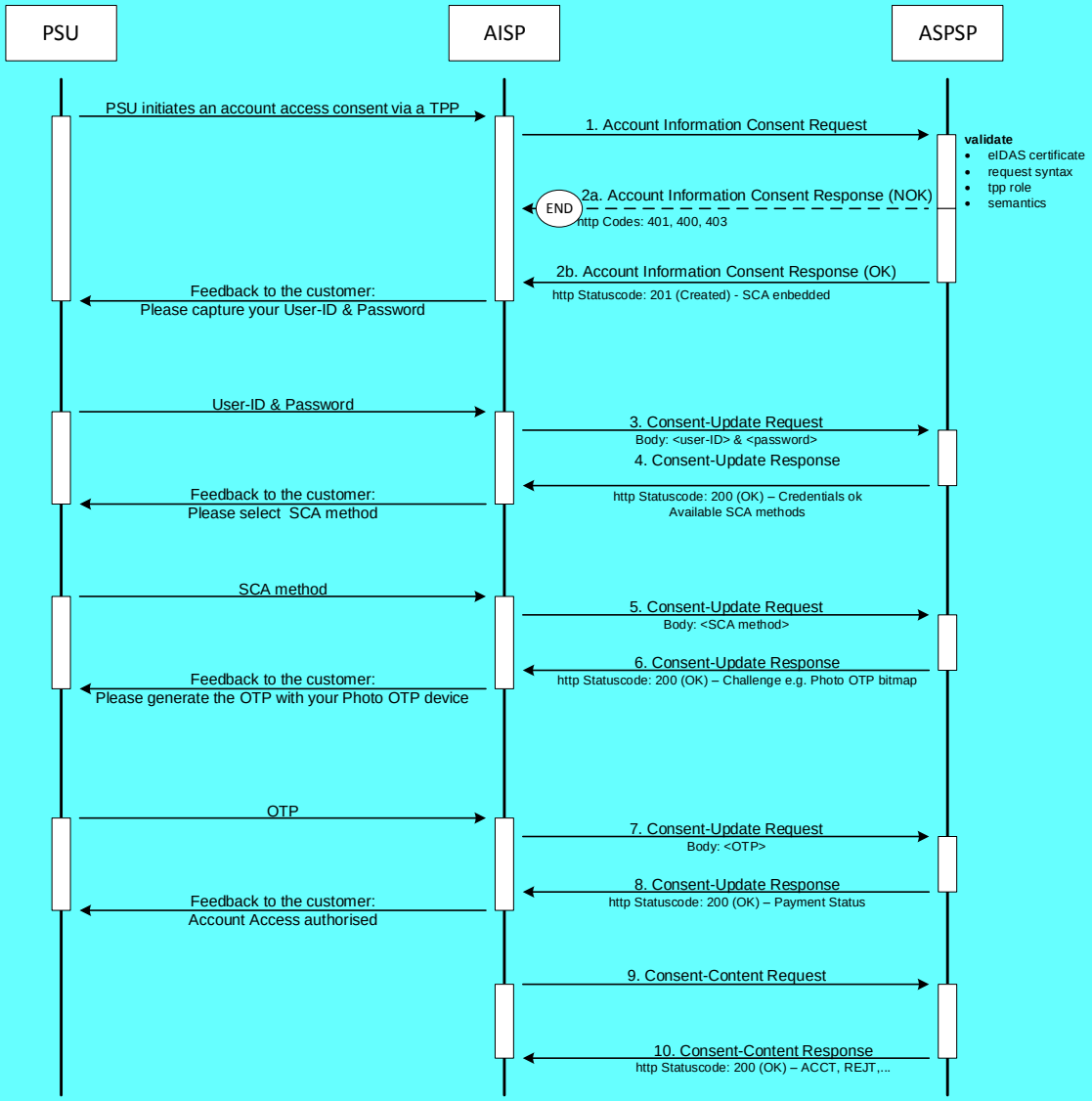
Remark: In case where OAuth2 is requested by the ASPSP as a pre-step to replace the PSU- and password by an access token, the sequence of the PSU authentication with the first authentication factor is omitted. This applies for all examples for the Embedded SCA Approach.

In case where only one SCA method is available, the “Authorise Transaction Request” is added to the flow, where the TPP is transmitting the authentication data of the customer, e.g. an OTP with included dynamic linking to the transaction details.



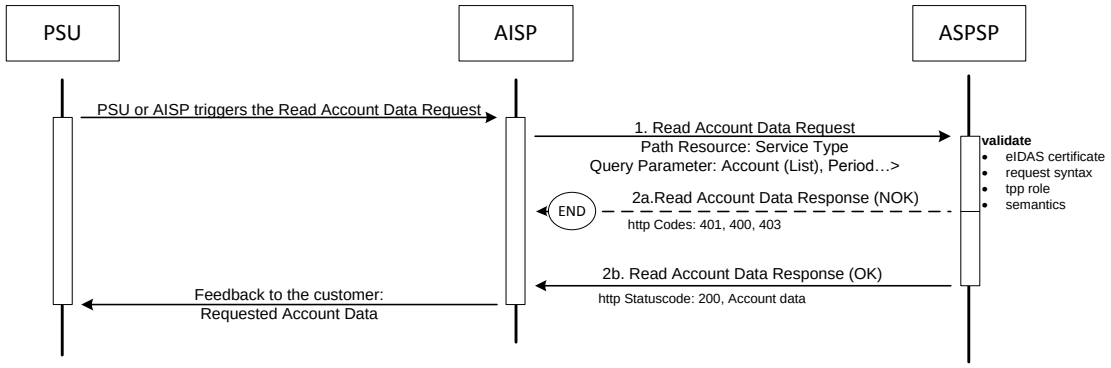
Embedded SCA Approach with Selection of an SCA method

In the following flow, there is a selection of an SCA method added in case of the ASPSP supporting several SCA methods for the corresponding PSU. The ASPSP transmits first the available methods to the PISP. The PISP might filter them, if not all authentication methods can be technically supported. The available methods then are presented to the PSU for choice.



6.1.2 Read Account Data Flow

The Read Account Data flow is independent from the corresponding Consent Management flow. It is a simple Request/Response process as follows:



6.2 Data Overview Account Information Service

The following table defines the technical description of the abstract data model as defined [XS2A OR] for the account information service. The columns give an overview on the API protocols as follows:

- The "Data element" column is using the abstract data elements following [XS2A OR] to deliver the connection to rules and role definitions in this document.
- The "Attribute encoding" is giving the actual encoding definition within the XS2A API as defined in this document.
- The "Location" columns define, where the corresponding data elements are transported as http parameters, resp. are taken from eIDAS certificates.
- The "Usage" column gives an overview on the usage of data elements in the different API Calls. Within [XS2A OR], the XS2A calls are described as abstract API calls. These calls will be technically realised as http POST, PUT, DELETE and GET commands. The calls are divided into the following calls:
    - Establish Consent Request, which shall be the first API Call for every transaction within XS2A Account Information service.
    - The Update Data Call is a call, where the TPP needs to add PSU related data, which is requested in the return of the first call. This call might be repeated.
    - The Authorisation Request is only used in an Embedded SCA Approach to authorise the transaction in case of a second factor is needed.
    - The Read Data Request is the request to retrieve Account Information data, which is addressed to different endpoints with different parameters.
    - The Status Request is used in cases, where the SCA control is taken over by the ASPSP and the TPP needs later information about the outcome.

The following usage of abbreviations in the Location and Usage columns is defined, cp. Also [XS2A OR] for details.

- x: This data element is transported on the corresponding level.
- m: Mandatory
- o : Optional for the TPP to use

- c: Conditional. The Condition is described in the API Calls, condition defined by the ASPSP

Data element	**Attribute encoding**	**Location**					**Usage**									
		Path	Query Param.	Header	Body	Certificate	Establ.. Cons.Req.	Establ. Cons. Resp.	Upd. Data Req.	Upd. Data Resp	Auth. Req.	Auth Resp.	Status Req.	Status Resp.	Read Data Req.	Read Data Resp
Provider Identification		x					m		m		m		m		m	
TPP Registration Number						x	m		m		m		m		m	
TPP Name						x	m		m		m		m		m	
TPP Role						x	m		m		m		m		m	
TPP National Competent Authority						x	m		m		m		m		m	
Transaction Identification	TPP-Transaction-ID			x			m		m		m		m		m	
Request Identification	TPP-Request-ID			x			m		m		m		m		m	
Resource ID	consentId				x			m								
Resource-ID[5]	Consent-ID			x											c	
Access Token (from optional OAuth2)	Authorization Bearer			x			c		c		c		c		c	
TPP Signing Certificate Data	TPP-Certificate			x			c		c		c		c		c	
TPP Signing Electronic Signature	Signature			x			c		c		c		c		c	

[5] Please note that the consent identification is addressed by different syntax depending of where it is transported.

Data element	**Attribute encoding**	**Location**					**Usage**									
		Path	Query Param.	Header	Body	Certificate	Establ.. Cons.Req.	Establ. Cons. Resp.	Upd. Data Req.	Upd. Data Resp	Auth. Req.	Auth Resp.	Status Req.	Status Resp.	Read Data Req.	Read Data Resp
Further signature related data	Digest			x			c		c		c		c		c	
Service Type		x					m		m		m		m		m	
Response Code				x				m		m		m		m		m
Transaction Status	transactionStatus				x			m		m		m		m		
PSU Message Information	psuMessage				x			o		o		o		o		o
TPP Message Information	tppMessages				x			o		o		o		o		o
PSU Identification	PSU-ID			x			c		c							
PSU Identification Type	PSU-ID-Type			x			c		c							
Corporate Identification	PSU-Corporate-ID			x			c		c		c		c			
Corporate Type	PSU-Corporate-ID-Type															
PSU Password	psuData.password				x				c							
PSU Authentication Data	psuData.authentication				x						m					
SCA Challenge Data	challengeData				x			c		c						
IP Address PSU	PSU-IP-Address			x			m									
PSU Agent	PSU-Agent			x			o									
GEO Information	PSU-Geo-Location			x			o									
Redirect URL ASPSP	_links.redirect				x			c								
Redirect Preference	tppRedirectPreferred		x				o									

Data element	**Attribute encoding**	**Location**					**Usage**									
		Path	Query Param.	Header	Body	Certificate	Establ.. Cons.Req.	Establ. Cons. Resp.	Upd. Data Req.	Upd. Data Resp	Auth. Req.	Auth Resp.	Status Req.	Status Resp.	Read Data Req.	Read Data Resp
Redirect URL TPP	TPP-Redirect-URI			x			c									
PSU Account	psuAccount				x										c	
PSU Account List	accessAccounts				x		m									
Date From	dateFrom		x												c	
Date To	dateTo		x												c	
Booking Status	bookingStatus		x												c	
Delta Indicator	deltaList		x												c	
With Balance Flag	withBalance		x												c	
PSU Involment Flag	psuInvolved		x												c	
Validity Period	validUntil				x		m									
Frequency	frequencyPerDay				x		m									
Recurring Indicator	recurringIndicator				x		m								c	
Combined service	combinedService Indicator				x		m									

Remark: The upper table refers to the "Acccount Information Consent Request" referring dedicated accounts, cp. Section 6.4.1.1.

The XS2A Interface calls which represent the messages defined in [XS2A OR] for the Payment Consent Request will be defined in the following sections.

Remark: The AIS and PIS services are sharing some sub processes which are once described in Section 7. So, for all Update Data Request/Response Definitions as well as for Authorise Transaction Request/Response Definitions, cp. Section 7.

6.3 Multicurrency Accounts

Definition: A multicurrency account is an account which is a collection of different sub-accounts which are all addressed by the same account identifier like an IBAN by e.g. payment initiating parties. The sub-accounts are legally different accounts and all differ in their currency, balances and transactions. An account identifier like an IBAN together with a currency always addresses uniquely a sub-account of a multicurrency account.

This specification supports to address multicurrency accounts either on collection or on sub-account level. The currency data attribute in the corresponding data structure "Account Reference" allows to build structures like

```
{"iban": "DE87123456781234567890"}
```

or

```
{"iban": "DE87123456781234567890",
 "currency": "EUR"}
```

If the underlying account is a multicurrency account, then

- the first reference is referring to the collection of all sub-accounts addressable by this IBAN, and
- the second reference is referring to the euro sub-account only.

This interface specification is acting on sub-accounts of multicurrency accounts in exactly the same way as on regular accounts.

The methods on multicurrency accounts differ in the inter-face due to the fact, that a collection of accounts is addressed. In the following the differences are described on abstract level.

Multicurrency Accounts in Submission of Consents

Multicurrency accounts are addressed by just using the external account identifier in the submission of a consent on dedicated accounts, without specifying a currency. Asking for the consent to retrieve account information data of a multicurrency accounts implies getting it for all sub-accounts.

Multicurrency Accounts in Reading Accounts or Account Details

The ASPSP will decide in its implementation whether to grant data access to a multicurrency account on aggregation level, on aggregation and sub-account level, or only on sub-account level.

Multicurrency Accounts in Reading Balances

The consequence for this function is that an array of balances of all sub-accounts are returned, if a multicurrency account is addressed on aggregation level.

Multicurrency Accounts in Reading Transactions

The consequence for this function is that the list of transactions will contain all transactions of all sub-accounts, if a multicurrency account is addressed on aggregation level. In this case the payment transactions contained in the report may have different transaction currencies.

6.4 Establish Account Information Consent

In this section, the Establish Account Information Consent process is defined for the XS2A Interface.

6.4.1 Account Information Consent Request

6.4.1.1 Consent Request on Dedicated Accounts

Call

POST /v1/consents

Creates an account information consent resource at the ASPSP regarding access to accounts specified in this request.

Side Effects

When this Consent Request is a request where the "recurringIndicator" equals “true”, and if it exists already a former consent for recurring access on account information for the addressed PSU, then the former consent automatically expires as soon as the new consent request is authorised by the PSU.

Query Parameters

Attribute	Type	Condition	Description
tppRedirectPreferred	Boolean	Optional	If it equals “true”, the TPP prefers a redirect over an embedded SCA approach. If it equals “false”, the TPP prefers not to be redirected for SCA. The ASPSP will then choose between the Embedded or the Decoupled SCA approach, depending on the choice of the SCA procedure by the TPP/PSU. If the parameter is not used, the ASPSP will choose the SCA approach to be applied depending on the SCA method chosen by the TPP/PSU.
withBalance	Boolean	[Optional]	This parameter may only be used together with the access sub attribute “available-accounts” in the request body. The request is rejected if the ASPSP is not

Attribute	Type	Condition	Description
			supporting this parameter. If the ASPSP accepts this parameter in the /consents endpoint, he shall also accept it for the GET access method on the /accounts endpoint.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party.
TPP-Request-ID	UUID	Mandatory	
PSU-ID	String	Conditional	Might be mandated in the ASPSP's documentation, if OAuth is not chosen as Pre-Step.
PSU-ID-Type	String	Conditional	Type of the PSU-ID, needed in scenarios where PSUs have several PSU-IDs as access possibility.
PSU-Corporate-ID	String	Conditional	Might be mandated in the ASPSP's documentation. Only used in a corporate context.
PSU-Corporate-ID-Type	String	Conditional	Might be mandated in the ASPSPs documentation. Only used in a corporate context.
Authorization Bearer	String	Conditional	If OAuth2 has been chosen as pre-step to authenticate the PSU.
TPP-Redirect-URI	String	Conditional	URI of the TPP, where the transaction flow shall be redirected to after a Redirect. Shall be contained at least if the tppRedirectPreferred parameter is set to true or is missing.
Signature	cp.	Conditional	A signature of the request by the TPP on application level. This might be mandated

Attribute	Type	Condition	Description
	Section 11		by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Shall be contained if the signature is used.

Request Body

Attribute	Type	Condition	Description
access	Account Access	Mandatory	Requested access services. Only the sub attributes with the tags "accounts", "balances" and "transactions" are accepted for this request.
recurringIndicator	Boolean	Mandatory	"true", if the consent is for recurring access to the account data "false", if the consent is for one access to the account data
validUntil	ISODate	Mandatory	This parameter is requesting a valid until date for the requested consent. The content is the local ASPSP date in ISODate Format, e.g. 2017-10-30. If a maximal available date is requested, a date in far future is to be used: “9999-12-31”. The consent object to be retrieved by the GET Consent Request will contain the adjusted date.
frequencyPerDay	Integer	Mandatory	This field indicates the requested maximum frequency for an access per day. For a one-off access, this attribute is set to “1”.
combinedService	Boolean	Mandatory	If "true" indicates that a payment initiation service will be

Attribute	Type	Condition	Description
Indicator			addressed in the same "session", cp. Section 8..

Note: All permitted "access" attributes ("accounts", "balances" and "transactions") used in this message shall carry a non-empty array of account references, indicating the accounts where the type of access is requested. Please note that a "transactions", "balances" or "accounts" access right also gives access to the generic /accounts endpoints, i.e. is implicitly supporting also the "accounts" access.

This specification mandates the ASPSP to support all POST consent requests with dedicated accounts, i.e. POST requests with the above mentioned sub-attributes, where at least one sub-attribute is contained, and where all contained sub-attributes carry a non-empty array of account references. This results in a consent on dedicated accounts. For this Consent Request on Dedicated Accounts, no assumptions are made for the SCA Approach by this specification.

Optionally, the ASPSP can support also Consent Requests, where the above mentioned sub-attributes "accounts", "balances" and "transactions" only carry an empty array or where the sub-attributes "available-accounts" or "allPsd2" are used – both with the value "all-accounts", cp. 6.4.1.2,

Response Header

Location hyperlink for the status of the resource.

Response Body

Attribute	Type	Condition	Description
transactionStatus	Transaction Status	Mandatory	authentication status of the consent
consentId	String	Conditional	Identification of the consent resource as it is used in the API structure Shall be contained, if a consent resource was generated.
scaMethods	Array of Authentication	Conditional	This data element might be contained, if SCA is required and if the PSU has a choice between different authentication

Attribute	Type	Condition	Description
	Objects		methods. Depending on the risk management of the ASPSP this choice might be offered before or after the PSU has been identified with the first relevant factor, or if an access token is transported. If this data element is contained, then there is also an hyperlink of type "selectAuthenticationMethods" contained in the response body. These methods shall be presented towards the PSU for selection by the TPP.
chosenScaMethod	AuthenticationObject	Conditional	This data element is only contained in the response if the APSPS has chosen the Embedded SCA Approach, if the PSU is already identified with the first relevant factor or alternatively an access token, if SCA is required and if the authentication method is implicitly selected.
challengeData	Challenge	Conditional	It is containded in addition to the data element chosenScaMethod if challenge data is needed for SCA.
			In rare cases this attribute is also used in the context of the psuAuthentication link.
_links	Links	Mandatory	A list of hyperlinks to be recognised by the TPP. Type of links admitted in this response (which might be extended by single ASPSPs as indicated in its XS2A documentation): "redirect" : In case of an SCA Redirect Approach, the ASPSP is transmitting the link to which to redirect the PSU browser. “oAuth”: In case of an OAuth2 based Redirect Approach, the ASPSP is transmitting the link where the configuration of the OAuth2 Server is

Attribute	Type	Condition	Description
			defined. "updatePsuIdentification" : The link to the payment initiation resource, which needs to be updated by the PSU identification. This might be used in a redirect or decoupled approach, where the PSU ID was missing in the first request. "updatePsuAuthenication" : The link to the account information resource, which needs to be updated by a PSU password and eventually the PSU identification if not delivered yet. This is used in a case of the Embedded SCA approach. "selectAuthenticationMethod" : This is a link to a resource, where the TPP can select the applicable SCA for the PSU, if there were several available authentication methods. This link is only contained under exactly the same conditions as the data element “authenticationMethods”, see above.,. "authoriseTransaction" : The link to the resource, where the "Transaction Authorisation Request" is sent to. This is the link to the resource which will authorise the transaction by checking the SCA authentication data within the Embedded SCA approach. “status”: The link to retrieve the transaction status of the account information consent..
psuMessage	String	Optional	Text to be displayed to the PSU, e.g. in a Decoupled SCA Approach

Example

Request

```
POST https://api.testbank.com/v1/consents
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80g
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7756
PSU-IP-Address:       192.168.8.78
PSU-Agent:            Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date:                 Sun, 06 Aug 2017 15:05:37 GMT

{
 "access":
     {"balances" :
         [{"iban": "DE2310010010123456789}},
          {"iban": "DE2310010010123456790”,
           "currency": "USD"},
          {"iban": "DE2310010010123456788"}],
      "transactions" :
         [{"iban": "DE2310010010123456789"},
          {"maskedPan": "123456xxxxxx1234"}
          ]
     }
 "recurringIndicator": "true",
 "validUntil": "2017-11-01",
 "frequencyPerDay" : "4"
}
```

Response in case of a redirect

Response Code 200

Response Header:

Location "v1/consents/1234-wertiq-983"

Response Body

```
{
  "transactionStatus": "Received",
  “consentId”: “1234-wertiq-983”,
  “_links": {
  "redirect":   "https://www.testbank.com/authentication/1234-wertiq-
983"
}
```


Response in case of the OAuth2 approach

Response Code 201

Response Header:

Location "v1/consents/1234-wertiq-983"

Response Body


```
{
  "transactionStatus" : "Received",
  “consentId”: “1234-wertiq-983”,
  "_links": {
    "self": "/v1/consents/1234-wertiq-983",
    “consentId”: “1234-wertiq-983”
  }
}
```


Response in case of the decoupled approach

Response Code 201

Response Header:

Location "v1/consents/1234-wertiq-983"

Response Body


```
{
  “transactionStatus” : “Received”,
  “consentId”: “1234-wertiq-983”,
  "_links": {
    "updatePsuIdentification": "/v1/consents/1234-wertiq-983"
  }
}
```

Response in case of the embedded approach

Response Code 201

```
{
  “transactionStatus”: “Received”,
  “consentId”: “1234-wertiq-983”,
  “_links": {
    "updatePsuAuthenication": "/v1/consents/1234-wertiq-983"
  }
}
```

6.4.1.2 Consent Request on Account List or without Indication of Accounts

Consent Request on Account List of Available Accounts

This function is supported by the same call as the Consent Request on Dedicated Accounts. The only difference is that the call only contains the "available-accounts" sub attribute within the "access" attribute with value "all-accounts".

In this case the call creates an account information consent resource at the ASPSP to return a list of all available accounts. For this specific Consent Request, no assumptions are made for the SCA Approach by this specification.

Consent Request without Indication of Accounts

This function is supported by the same call as the Consent Request on Dedicated Accounts. The only difference is that the call contains the "accounts", "balances" and/or "transactions" sub attribute within the "access" attribute all with an empty array.

The ASPSP will then agree bilaterally directly with the PSU on which accounts the requested access consent should be supported. The result can be retrieved by the TPP by using the GET Consent Request method, cp. 6.4.3. For this function the Embedded SCA Approach is not supported.

Consent Request for Access to all Accounts for all PSD2 defined AIS

This function is supported by the same call as the Consent Request on Dedicated Accounts. The only difference is that the call contains the "allPsd2" sub attribute within the "access" attribute with the value "all-accounts".

If this function is supported, it will imply a consent on all available accounts of the PSU on all PSD2 related account information services. For this specific Consent Request, no assumptions are made for the SCA Approach by this specification.

Example Consent on Account List of Available Accounts

Request

```
POST https://api.testbank.com/v1/consents
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80g
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7756
PSU-IP-Address:       192.168.8.78
PSU-Agent:            Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date:                 Sun, 06 Aug 2017 15:05:37 GMT

{"access":
     {"available-accounts": "all-accounts"},
 "recurringIndicator": "false",
 "validUntil": "2017-08-06",
 "frequencyPerDay": "1"
}
```

Example Consent without dedicated Account

Request

```
POST https://api.testbank.com/v1/consents
Content-Encoding      gzip
Content-Type          application/json
TPP-Transaction-ID    3dc3d5b3-7023-4848-9853-f5400a64e80g
TPP-Request-ID        99391c7e-ad88-49ec-a2ad-99ddcb1f7756
PSU-IP-Address        192.168.8.78
PSU-Agent             Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date                  Sun, 06 Aug 2017 15:05:37 GMT

{"access":
     {"balances": [],
      "transactions": []},
 "recurringIndicator": "true",
 "validUntil": "2017-11-01",
 "frequencyPerDay": "4"
}
```

6.4.2 Get Status Request

Call

GET /v1/consents/{consentId}/status

Can check the status of an account information consent resource.

Path

Attribute	Type	Description
consentId	String	The consent identification assigned to the created resource.

Query Parameters

No specific query parameters.

Request Header

See above.

Request Body

No body.

Response Body

Attribute	Type	Condition	Description
transactionStatus	Transaction Status	Mandatory	This is the “authentication status” of the consent. Remark: To check the validity, the call “GET Consent Request” is used, cp. Section 6.4.3

Example

Request

```
GET https://api.testbank.com/v1/consents/qwer3456tzui7890/status
```

Response

Response Code 200

```
{
  “transactionStatus” : "AcceptedTechnicalValidation",
}
```

6.4.3 Get Consent Request

Call

GET /v1/consents/{consentId}

Returns the content of an account information consent object. This is returning the data for the TPP especially in cases, where the consent was directly managed between ASPSP and PSU e.g. in a re-direct SCA Approach.

Path

Attribute	Type	Description
consentId	String	ID of the corresponding consent object as returned by an Account Information Consent Request

Query Parameters

No specific query parameter.

Request Header

See above for Get Status Request.

Request Body

No body.

Response Body

Attribute	Type	Condition	Description
access	Account Access	Mandatory	
recurringIndicator	Boolean	Mandatory	
validUntil	ISODate	Mandatory	

Attribute	Type	Condition	Description
frequencyPerDay	Integer	Mandatory	
lastActionDate	ISODate	Mandatory	This date is containing the date of the last action on the consent object either through the XS2A interface or the PSU/ASPSP interface having an impact on the status.
transactionStatus	Transaction Status	Mandatory	
consentStatus	String	Mandatory	The following code values are permitted "empty", "valid", "blocked", "expired", "deleted". These values might be extended by ASPSP by more values.

Example

Request

```
GET https://api.testbank.com/v1/consents/qwer3456tzui7890?
```

Response

```
{
  "access":
     {"balances" :
         [{"iban": "DE23100100101234567890"}]
     {"transactions" :
         [{"iban": "DE23100100101234567890"},
          {"pan": "123456xxxxxx3457"}]

 "recurringIndicator": "true",
 "validUntil": "2017-11-01",
 "frequencyPerDay" : "4",
 "transactionStatus": "AcceptedTechnicalValidation",
 "consentStatus": "valid",
 "_links": {"viewAccounts": "/v1/accounts"}
}
```

Remark: This specification supports no detailed links to AIS service endpoints corresponding to this account. This is due to the fact, that the /accounts endpoint will deliver all detailed information, including the hyperlinks e.g. to the balances or transactions of certain accounts. Still due to the guiding principles, the ASPSP may deliver more links in addition, which then will be documented in the ASPSPs XS2A API documentation. .

6.5 Delete an Account Information Consent Object

The TPP can delete an account information consent object if needed with the following call:

Call

DELETE /v1/consents/{consentId}

Deletes a given consent.

Path

Attribute	Type	Description
consentId	String	Contains the resource-ID of the consent to be deleted.

Query Parameters

No specific query parameters.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party. .
TPP-Request-ID	UUID	Mandatory	
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based SCA was performed in the corresponding consent transaction.

No Request Body.

No Response Body

Example

Request

```
DELETE https://api.testbank.com/v1/consents/qwer3456tzui7890

Content-Encoding      gzip
Content-Type          application/json
TPP-Transaction-ID    3dc3d5b3-7023-4848-9853-f5400a64e812
TPP-Request-ID        99391c7e-ad88-49ec-a2ad-99ddcb1f7757
Date                  Sun, 13 Aug 2017 17:05:37 GMT
```

Response

Response Code: 204

6.6 Read Account Data Requests

6.6.1 Read Account List

Call

GET /v1/accounts/ {query-parameters}

Reads a list of bank accounts, with balances where required. It is assumed that a consent of the PSU to this access is already given and stored on the ASPSP system. The addressed list of accounts depends then on the PSU ID and the stored consent addressed by consentId, respectively the OAuth2 access token.

Note: If the consent is granted only to show the list of available accounts (“available-accounts” access rights, cp. Section 6.4.1.2), much less details are displayed about the accounts. Specifically hyperlinks to balances or transaction endpoint should not delivered then.

Note: If the details returned in this call with the access rights “accounts”, “balances”, “transactions” or “allPsd2” are not sufficient, then more details can be retrieved by addressing the /accounts/{account-id} endpoint, cp. Section 6.6.2.

Query Parameters

Attribute	Type	Condition	Description
withBalance	Boolean	[Optional]	If contained, this function reads the list of accessible payment accounts including the booking balance. This call will be rejected if the withBalance parameter is used in a case, where the access right on balances is not granted in the related consent or if the ASPSP does not support the withBalance parameter.
psuInvolved	Boolean	Conditional	It must be contained if the PSU has asked for this account access in real-time. This flag is then set to "true". The PSU then might be involved in an additional consent process, if the given consent is not any more sufficient.

Request Header

Attribute	Type	Condition	Description
TPP-	UUID	Mandatory	ID of the transaction as determined by the

Attribute	Type	Condition	Description
Transaction-ID			initiating party.
TPP-Request-ID	UUID	Mandatory	
Consent-ID	String	Mandatory	Shall be contained since “Establish Consent Transaction” was performed via this API before.
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in the related consent authorisation.
Signature	cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. It shall be contained if a signature is used, see above.

Response Body

Attribute	Type	Condition	Description
accountList	Array of Account Details	Mandatory	

Example

Response in case of an example, where the consent has been given on tow different IBANs

```
{“account-list”:
   [
      {“id”:  “3dc3d5b3-7023-4848-9853-f5400a64e80f”,
       "iban": "DE23100100101234567 89",
       "currency": "EUR",
       “accountType”: “Girokonto”,
       “cashAccountType”: “CurrentAccount”,
       “name”: “Main Account”,
       “_links” : {
          “balances”:         “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/balances”,
          “transactions”   :   “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/transactions”}
      },
      {“id” : “3dc3d5b3-7023-4848-9853-f5400a64e81g”,
       "iban": "DE23100100101234567 88",
       “currency” : “USD”,
       “accountType”: “Fremdwährungskonto”,
       “cashAccountType” : “CurrentAccount”,
       “name” : “US Dollar Account”,
       “_links” : {
          “balances”     :     “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e81g/balances” }
       }
]}
```

Response in case of an example where consent on transactions and balances has been given to a multicurrency account which has two sub-accounts with currencies EUR and USD, and where the ASPSP is giving the data access only on sub-account level:

```
{“account-list”:
   [
      {“id”:  “3dc3d5b3-7023-4848-9853-f5400a64e80f”,
       "iban": "DE23100100101234567 88",
       "currency": "EUR",
       “accountType”: “Girokonto”,
       “cashAccountType”: “CurrentAccount”,
       “name”: “Main Account”,
       “_links”: {
          “balances”:         “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/balances”,
          “transactions”:     “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/transactions”}
```

```
      },
      {“id” : “3dc3d5b3-7023-4848-9853-f5400a64e81g”,
       "iban": "DE23100100101234567 88",
       “currency” : “USD”,
       “accountType”: “Fremdwährungskonto”,
       “cashAccountType” : “CurrentAccount”,
       “name” : “US Dollar Account”,
       “_links” : {
          “balances”    :    “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e81g/balances”,
          “transactions”:    “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e81g/transactions” }
       }
]}
```

Response in case of an example where consent on balances and transactions has been given to a multicurrency account which has two sub-accounts with currencies EUR and USD and where the ASPSP is giving the data access on aggregation level and on sub-account level:

```
{“account-list”:
   [
      {“id”:  “3dc3d5b3-7023-4848-9853-f5400a64e80f”,
       "iban": "DE23100100101234567 88",
       "currency": "XXX",
       “accountType”: “Multi currency account”,
       “cashAccountType”: “CurrentAccount”,
       “name”: “Aggregation Account”,
       “_links”: {
          “balances”:        “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e333/balances”,
          “transactions”:    “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e333/transactions”}
      },
{“id”:  “3dc3d5b3-7023-4848-9853-f5400a64e80f”,
       "iban": "DE23100100101234567 88",
       "currency": "EUR",
       “accountType”: “Girokonto”,
       “cashAccountType”: “CurrentAccount”,
       “name”: “Main Account”,
       “_links”: {
          “balances”:        “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/balances”,
          “transactions”:    “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/transactions”}
      },
```

```
      {“id” : “3dc3d5b3-7023-4848-9853-f5400a64e81g”,
       "iban": "DE23100100101234567 88",
       “currency” : “USD”,
       “accountType”: “Fremdwährungskonto”,
       “cashAccountType” : “CurrentAccount”,
       “name” : “US Dollar Account”,
       “_links” : {
          “balances”:         “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e81g/balances”,
          “transactions”:     “/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e81g/transactions” }
       }
]}
```

6.6.2 Read Account Details

Call

GET /v1/accounts/{account-id} {query-parameters}

Reads details about an account, with balances where required. It is assumed that a consent of the PSU to this access is already given and stored on the ASPSP system. The addressed details of this account depends then on the stored consent addressed by consentId, respectively the OAuth2 access token.**NOTE:** The account-id can represent a multicurrency account. In this case the currency code is set to "XXX".

Query Parameters

Attribute	Type	Condition	Description
withBalance	Boolean	[Optional]	If contained, this function reads the list of accessible payment accounts including the balance. This call will be rejected if the withBalance parameter is used in a case, where the access right on balances is not granted in the related consent or if the ASPSP does not support the withBalance parameter.
psuInvolved	Boolean	Conditional	It must be contained if the PSU has asked for this account access in real-time. This flag is then set to "true". The PSU then might be involved in an additional consent process, if the given consent is not any more sufficient.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party.
TPP-Request-ID	UUID	Mandatory	
Consent-ID	String	Mandatory	
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in the related consent authorisation.
Signature	cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. It shall be contained if a signature is used, see above.

Response Body

Attribute	Type	Condition	Description
account	Account Details	Mandatory	

Example

Response for a regular account

```
{"account :
      {"id" :  "3dc3d5b3-7023-4848-9853-f5400a64e80f",
       {"iban" :  "DE23100100101234567 89",
       "currency" :  "EUR"},
       "accountType" : "Girokonto",
       "cashAccountType" : "CurrentAccount",
       "name" : "Main Account",
       "_links" : {
          "balances"    :    "/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/balances",
          "transactions"  :  "/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/transactions"}
        }
}
```

Response for a multi-currency account

```
 {"account" :
   {"id":  "3dc3d5b3-7023-4848-9853-f5400a64e80f",
     "iban":  "DE23100100101234567 89",
     "currency":  "XXX",
     "accountType": "Multicurrency Account",
     "cashAccountType": "CurrentAccount",
     "name" : "Aggregation Account",
     "_links" : {
          "balances"    :    "/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/balances",
          "transactions"  :  "/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f/transactions"}
      }
}
```

6.6.3 Read Balance

Call

GET /v1/accounts/{account-id}/balances

Reads account data from a given account addressed by "account-id".

Remark: This account-id can be a tokenised identification due to data protection reason since the path information might be logged on intermediary servers within the ASPSP sphere. This account-id then can be retrieved by the “GET Account List” call, cp. Section 6.6.1.

The account-id is constant at least throughout the lifecycle of a given consent.

Path

Attribute	Type	Description
account-id	String	This identification is denoting the addressed account. The account-id is retrieved by using a “Read Account List” call. The account-id is the “id” attribute of the account structure. Its value is constant at least throughout the lifecycle of a given consent.

Query Parameters

Attribute	Type	Condition	Description
psuInvolved	Boolean	Conditional	It must be contained if the PSU has asked for this account access in real-time. This flag is then set to "true". The PSU then might be involved in an additional consent process, if the given consent is not any more sufficient.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party. .
TPP-Request-ID	UUID	Mandatory	
Consent-ID	String	Mandatory	
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in the related consent authorisation.

Attribute	Type	Condition	Description
Signature	cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. It shall be contained if a signature is used, see above.

Response

Attribute	Type	Condition	Description
balances	Balances	Mandatory	A list of balances regarding this account, e.g. the current balance, the last booked balance.

Example

Response in case of a regular account:


```
{
  "balances" :
      [{"closingBooked":
           {
            "amount": {"currency" : "EUR", "content": "500.00"},
            "date" : "2017-10-25"
           },
      "expected":
           {
            "amount": {"currency" : "EUR" ,"content" : "900.00"},
            "lastActionDateTime" : "2017-10-25T15:30:35.035Z"
           }
      }]
}
```


Response in case of a multicurrency account with one account in EUR, one in USD, where the ASPSP has delivered a link to the balance endpoint relative to the aggregated multicurrency account (aggregation level)


```
{
  "balances" :
      [{"closingBooked":
           {
            "amount": {"currency" : "EUR", "content": "500.00"},
            "date" : "2017-10-25"
           },
      "expected":
           {
            "amount": {"currency" : "EUR" ,"content" : "900.00"},
            "lastActionDateTime" : "2017-10-25T15:30:35.035Z"
           }
      },
       {"closingBooked":
           {
            "amount": {"currency" : "USD", "content": "350.00"},
            "date" : "2017-10-25"
           },
      "expected":
           {
            "amount": {"currency" : "USD" ,"content" : "350.00"},
            "lastActionDateTime" : "2017-10-24T14:30:21Z"
           }]
}
```

6.6.4 Read Transaction List

Call

GET /v1/accounts/{account-id}/transactions {query-parameters}

Reads account data from a given account addressed by “account-id”.

Remark: This account-id can be a tokenised identification due to data protection reason since the path information might be logged on intermediary servers within the ASPSP sphere. This account-id then can be retrieved by the “GET Account List” call, cp. Section 6.6.1.

Remark: Please note that the PATH might be already given in detail by the response of the “Read Account List” call within the _links subfield.

Path

Attribute	Type	Description
account-id	String	This identification is denoting the addressed account. The account-id is retrieved by using a “Read Account List” call. The account-id is the “id” attribute of the account structure. Its value is constant at least throughout the lifecycle of a given consent.

Query Parameters

Attribute	Type	Condition	Description
dateFrom	ISODate	Conditional	Starting date of the transaction list, mandated if no delta access is required
dateTo	ISODate	Optional	End date of the transaction list, default is now if not given..
transactionId	String	[Optional]	This data attribute is indicating that the AISP is in favour to get all transactions after the transaction with identification transactionId alternatively to the above defined period. This is a implementation of a delta access.

Attribute	Type	Condition	Description
			If this data element is contained, the entries “dateFrom” and “dateTo” might be ignored by the ASPSP if a delta report is supported.
psuInvolved	Boolean	Conditional	It must be contained if the PSU has asked for this account access in real-time. This flag is then set to "true". The PSU then might be involved in an additional consent process, if the given consent is not any more sufficient.
bookingStatus	String	Mandatory	Permitted codes are “booked”, “pending” and “both” “booked” and “both” are to be supported mandatorily by the ASPSP. To support the “pending” feature is optional for the ASPSP, Error code if not supported in the online banking frontend
withBalance	Boolean	[Optional]	If contained, the TPP is requiring to add the balance to the transaction list. This call will be rejected if the withBalance parameter is used in a case, where the access right on balances is not granted in the corresponding consen or if the ASPSP does not support the withBalance parameter .
deltaList	Boolean	[Optional]	This data attribute is indicating that the AISP is in favour to get all transactions after the last report access for this PSU on the addressed account. This is another implementation of a delta access-report. This delta indicator might be rejected by the ASPSP if this function is not supported.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party. In case of a once off read data request, this TPP-Transaction-ID equals the TPP-Transaction-ID of the corresponding Account Information Consent Request, cp. Section 6.4.1.
TPP-Request-ID	UUID	Mandatory	
Consent-ID	String	Mandatory	
Authorization Bearer	String	Conditional	Is contained only, if an OAuth2 based authentication was performed in a pre-step or an OAuth2 based SCA was performed in the related consent authorisation.
Accept	String	Conditional	The TPP can indicate the formats of account reports supported together with a priorisation following the http header definition. The formats supported by this specification are • xml • JSON • text Further definition of content by ASPSP/ communities cp. Annex B.
Signature	cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. It shall be contained if a signature is used, see above.

Remark: The Berlin Group intends to apply for vnd-entries within the “accept” attribute for camt.05x and MT94x formats. These values will be added to this specification as soon as available.

Response Header

Content-Type : application/json or application/xml or application/text

Response Body

In case the ASPSP returns a camt.05x XML structure, the response body consists of either a camt.052 or camt.053 format. The camt.052 may include pending payments which are not yet finally booked. The ASPSP will decide on the format due to the chosen parameters, specifically on the chosen dates relative to the time of the request.

In case the ASPSP returns a MT94x content, the response body consists of an MT940 or MT942 format in a text structure. The camt.052 may include pending payments which are not yet finally booked. The ASPSP will decide on the format due to the chosen parameters, specifically on the chosen dates relative to the time of the request.

A JSON response is defined as follows:

Attribute	Type	Condition	Description
_links	Links	Optional	A list of hyperlinks to be recognised by the TPP. Type of links admitted in this response: “download”: a link to a resource, where the transaction report might be downloaded from in case where transaction reports have a huge size. **Remark:** This feature shall only be used where camt-data is requested which has a huge size. Pagination links where transaction reports have a huge size and the ASPSP is only providing one page, where page size is defined by the ASPSP. "first", "next", "previous", "last"
transactions	Account Report	Optional	JSON based account report.

Example

Request

```
GET
https://api.testbank.com/v1/accounts/qwer3456tzui7890/transactions?d
ateFrom=2017-07-01&dateTo= 2017-07-30&psuInvolved=true

Accept: application/json, application/text;q=0.9,
application/xml;q=0.8
```

Response in JSON format for an access on a regular account

Response Code 200


```
{"transactions" :
   {"booked" :
     [{
       "transactionId" : "1234567" ,
       "creditorName" : "John Miles" ,
       "creditorAccount" : {"iban" : "DE43533700240123456900"},
       "amount" : {"currency" : "EUR", "content": "-256.67"} ,
       "bookingDate" : "2017-10-25" ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 1"
      },{
       "transactionId" : "1234568",
       "debtorName" : "Paul Simpson" ,
       "debtorAccount" : {"iban" : "NL354543123456900"} ,
       "amount" : {"currency" : "EUR", "content": "343.01"} ,
       "bookingDate" : "2017-10-25" ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 2"
      }],
   },
   {"pending" :
     [{
       "transactionId" : "1234569" ,
       "creditorName" : "Claude Renault" ,
       "creditorAccount" : {"iban" : "FR33554543123456900"},
       "amount" : {"currency" : "EUR", "content" : "-100.03"} ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 3"
      }]
   },
   {"_links":
     {"viewAccount" : "/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f"}
   }
}
```


Response in case of huge data amount as a download.


```
{
  "_links : {"download" : www.test-
api.com/xs2a/v1/accounts/12345678999/transactions/download/}
}
```

Response in JSON format for an access on a multicurrency account on aggregation level:


```
{"transactions" :
   {"booked" :
     [{
       "transactionId" : "1234567" ,
       "creditorName" : "John Miles" ,
       "creditorAccount" : {"iban" : "DE43533700240123456900"},
       "amount" : {"currency" : "EUR", "content": "-256.67"} ,
       "bookingDate" : "2017-10-25" ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 1"
      },{
       "transactionId" : "1234568",
       "debtorName" : "Paul Simpson" ,
       "debtorAccount" : {"iban" : "NL354543123456900"} ,
       "amount" : {"currency" : "EUR", "content": "343.01"} ,
       "bookingDate" : "2017-10-25" ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 2"
      },{
       "transactionId" : "1234569",
       "debtorName" : "Pepe Martin" ,
       "debtorAccount" : {"iban" : "SE1234567891234"} ,
       "amount" : {"currency" : "USD", "content": "100"} ,
       "bookingDate" : "2017-10-25" ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 3"
      }],
   },
   {"pending" :
     [{
       "transactionId" : "1234570" ,
       "creditorName" : "Claude Renault" ,
       "creditorAccount" : {"iban" : "FR33554543123456900"},
       "amount" : {"currency" : "EUR", "content" : "-100.03"} ,
       "valueDate" : "2017-10-26" ,
       "remittanceInformationUnstructured" : "Example 4"
      }]
   },
   {"_links":
     {"viewAccount" : "/v1/accounts/3dc3d5b3-7023-4848-9853-
f5400a64e80f"}
   }
}
```

7 Processes used commonly in AIS and PIS Services

Processes on PSU identification, PSU authentication and explicit authorisation of transactions by using SCA are very similar in PIS and AIS services. The API calls supporting these processes are described in the following independently from the service/endpoint. For reasons of clarity, the endpoints are defined always for the Payment Initiation Service and the Account Information Service separately. The structure of all parameters in the request header/body and the response header/body are equal.

7.1 Update PSU Data

There are several possible Update PSU Data requests needed, which depends on the SCA Approach:

- Redirect SCA Approach: A specific Update PSU Data Request is applicable for
  - the selection of authentication methods, before choosing the actual SCA approach.
- Decoupled SCA Approach: A specific Update PSU Data Request is only applicable for
  - adding the PSU Identification, if not provided yet in the Payment Initiation Request or the Account Information Consent Request, or if no OAuth2 access token is used, or
  - the selection of authentication methods.
- Embedded SCA Approach: The Update PSU Data Request might be used
  - to add credentials as a first factor authentication data of the PSU and
  - to select the authentication method.

The SCA Approach might depend on the chosen SCA method. For that reason, the following possible Update PSU Data request can apply to all SCA approaches:

- Select an SCA method in case of several SCA methods are available for the customer.

These different Update PSU Data Requests are differentiated in the following sub sections.

7.1.1 Update PSU Data (Identification) in the Decoupled Approach

This call is used, when in the preceding call the hyperlink of type “updatePsuIdentification” was contained in the response and is now followed by the TPP.

Call in case of a Payment Initiation Request

PUT /v1/payments/{payment-product}/{paymentId}

Updates the payment initiation data on the server by PSU data, if requested by the ASPSP.

Call in case of an Account Information Consent Request

PUT /v1/consents/{consentId}

Updates the account information consent data on the server by PSU data, if requested by the ASPSP.

Path

Attribute	Type	Description
payment-product	String	Only in case of an Update Data Request in a Payment Initiation context.
paymentId or consentId	String	Resource identification of the related payment initiation or consent resource.

Query Parameters

No specific query parameters.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	This is the same TPP-Transaction-ID as in the related payment initiation or establish consent request message.
TPP-Request-ID	UUID	Mandatory	
PSU-ID	String	Conditional	Contained if not yet contained in the first request, and mandated by the ASPSP in the related response
PSU-ID-Type	String	Conditional	Type of the PSU-ID, needed in scenarios where PSUs have several PSU-IDs as access possibility.

Attribute	Type	Condition	Description
PSU-Corporate-ID	String	Conditional	Contained if not yet contained in the first request, and mandated by the ASPSP in the related response. This field is relevant only in a corporate context.
PSU-Corporate-ID-Type	String	Conditional	Might be mandated by the ASPSP in addition if the PSU-Corporate-ID is contained.
Signature	Cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Shall be contained if a signature is used, see above.

Request Body

No Body.

Response Body

Attribute	Type	Condition	Description
transactionStatus	Transaction Status	Mandatory	
psuMessage	String	Optional	

Example

Request

```
PUT https://api.testbank.com/v1/payments/sepa-credit-transfers/qwer3456tzui7890
TPP-Transaction-ID:   asdfoeljkasdfoelkjasdf-1234ß089
TPP-Request-ID:       3dc3d5b3-7023-4848-9853-f5400a64e80f
PSU-ID:               PSU-1234
```

Response

```
Response Code 201
```

Response Body

```
{
 "transactionStatus": "AcceptedTechnicalValidation",
 "psuMessage": "Please use your BankApp for transaction Authorisation."
}
```

7.1.2 Update PSU Data (Authentication) in the Decoupled or Embedded Approach

This call is used, when in the preceding call the hyperlink of type "updatePsuAuthenication" was contained in the response and is followed by the TPP.[6]

Call in case of a Payment Initiation

PUT /v1/payments/{payment-product}/{paymentId}

Updates the payment initiation data on the server by PSU data, if requested by the ASPSP

Call in case of an Account Information Consent Request

PUT /v1/consents/{consentId}

Updates the account information consent data on the server by PSU data, if requested by the ASPSP

Path

Attribute	Type	Description
payment-product	String	Only in case of an Update Data Request in a Payment Initiation context.
paymentId or consentId	String	Resource identification of the related payment initiation or consent resource.

Query Parameters

No specific query parameters.

[6] The next release of this specification might support encryption methods for transmission of the PSU password between TPP and ASPSP on application level.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	This is the same TPP-Transaction-ID as in the related payment initiation or establish consent request message.
TPP-Request-ID	UUID	Mandatory	
PSU-ID	String	Conditional	Contained if not yet contained in the first request, and mandated by the ASPSP in the related response
Signature	Cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Shall be contained if a signature of the request is contained, see above.

Request Body

Attribute	Type	Condition	Description
psuData	PSU Data	Conditional	

Response Body

Attribute	Type	Condition	Description
chosenScaMethod	Authentication object	Conditional	A definition of the provided SCA method is contained, if only one authentication method is available, and if the Embedded SCA approach is chosen by the ASPSP.
challengeData	Challenge	Conditional	Challenge data might be contained, if only one authentication method is available.

Attribute	Type	Condition	Description
scaMethods	Array of authentication objects	Conditional	Might be contained, if several authentication methods are available. (name, type)
_links	Links	Conditional	A list of hyperlinks to be recognised by the TPP. Might be contained, if several authentication methods are available for the PSU. Type of links admitted in this response: “selectAuthenticationMethod” : This is a link to a resource, where the TPP can select the applicable second factor authentication methods for the PSU, if there were several available authentication methods. This link is only contained, if the PSU is already identified or authenticated with the first relevant factor or alternatively an access token, if SCA is required and if the PSU has a choice between different authentication methods. If this link is contained, then there is also the data element “scaMethods” contained in the response body “authoriseTransaction” : The link to the resource, where the “Transaction Authorisation Request” is sent to. This is the link to the resource which will authorise the transaction by checking the SCA authentication data within the Embedded SCA approach. “self” : The link to the resource itself. “status”: The link where the transaction status of the resource can be retrieved.
transactionStatus	Transaction Status	Mandatory	
psuMessage	String	Optional	

Example

Request in case of Embedded Approach

```
PUT                  https://api.testbank.com/v1/payments/sepa-credit-
transfers/qwer3456tzui7890
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID:       asdfoeljkasdfoelkjasdf-1234ß091
PSU-ID:               PSU-1234
{
"psuData": {
  "password": "start12"
  }
}
```

Response in case of the embedded approach

Response Code 200

```
{
  "transactionStatus": "AcceptedTechnicalValidation",
   links:{
   "authoriseTransaction":  "/v1/payments/sepa-credit-transfers/1234-
wertiq-983"
  }
}
```

7.1.3 Update PSU Data (Select Authentication Method)

This call is used, when in the preceding call the hyperlink of type "selectAuthenticationMethod" was contained in the response and was followed by the TPP.

Call in case of a Payment Initiation Request

PUT /v1/payments/{payment-product}/{paymentId}

Updates the payment initiation data on the server by PSU data, if requested by the ASPSP

Call in case of an Account Information Consent Request

PUT /v1/consents/{consentId}

Updates the account information consent data on the server by PSU data, if requested by the ASPSP

Path

Attribute	Type	Description
payment -product	String	Only in case of an Update Data Request in a Payment Initiation context.
paymentId or consentId	String	Resource identification of the related payment initiation or consent resource.

Query Parameters

No specific query parameter.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	This is the same TPP-Transaction-ID as in the related payment initiation or establish consent request message.
TPP-Request-ID	UUID	Mandatory	
Signature	Cp. Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Shall be contained if a signature is used for this request, see above.

Request Body

Attribute	Type	Condition	Description
authentication MethodId	String	Mandatory	The authentication method ID as provided by the ASPSP.

Response Body

Attribute	Type	Condition	Description

Attribute	Type	Condition	Description
chosenSca Method	Authentication object	Conditional	A definition of the provided SCA method is contained, if only one authentication method is available, and if the Embedded SCA approach is chosen by the ASPSP.
challengeData	Challenge	Conditional	Challenge data might be contained, if only one authentication method is available.
_links	Links	Conditional	A list of hyperlinks to be recognised by the TPP. Might be contained, if several authentication methods are available for the PSU. Type of links admitted in this response: “authoriseTransaction” : The link to the resource, where the “Transaction Authorisation Request” is sent to. This is the resource which will check the SCA authentication data within the Embedded SCA approach. All other types of hyperlinks like defined in the Payment Initiation Response or Establish Account Information Consent Response besides the selectAuthenticationMethods link could be contained, depending on the processing status within the ASPSP, cp. Section 6.4.1.1
transactionStatus	Transaction Status	Mandatory	
psuMessage	String	Optional	

Example

Request in case of Embedded Approach

```
PUT https://api.testbank.com/v1/payments/sepa-credit-transfers/qwer3456tzui7890
```

```
TPP-Request-ID:       asdfoeljkasdfoelkjasdf-1234ß093
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80f
{
authenticationMethodId: "myAuthenticationID"
}
```

Response in case of the embedded approach

```
{
  "transactionStatus" : "AcceptedTechnicalValidation",
  "chosenScaMethod" : {
     "authenticationType" : "SMS_OTP",
     "authenticationMethodId" : "myAuthenticationID"}
  "challengeData" : {
     "otpMaxLength" : "6"
     "otpFormat" : "integer"}
  _links{
     "authoriseTransaction":"/v1/payments/sepa-credit-
transfers/1234-wertiq-983"
  }
}
```

7.2 Transaction Authorisation

This call is only used in case of an Embedded SCA Approach.

Call in case of an Payment Initiation Request

PUT /v1/payments/{payment-product}/{paymentId}

Transfers data for SCA checks by the ASPSP.

Call in case of an Account Information Consent Request

PUT /v1/consents/{consentId}

Transfers data for SCA checks by the ASPSP.

Path

Attribute	Type	Description
payment	String	The related payment product of the payment initiation to be

Attribute	Type	Description
-product		authorized.
paymentId or consentId	String	Resource identification of the related payment initiation or consent resource.

Query Parameter

No specific query parameter.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	This is the same TPP-Transaction-ID as in the related payment initiation or establish consent request message.
TPP-Request-ID	UUID	Mandatory	
Authorization Bearer	String	Conditional	Is contained only, if the optional Oauth Pre-Step was performed.
Signature	String	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, in base64 encoding. Shall be contained if a signature was used in this request, see above.

Request Body

Attribute	Type	Condition	Description
scaAuthenticationData	String	Mandatory	SCA authentication data, depending on the chosen authentication method. If the data is binary, then it is base64 encoded.

Response Body

Attribute	Type	Condition	Description
transactionStatus	Transaction Status		

Example

Request

```
PUT       https://api.testbank.com/v1/payments/sepa-credit-transfers/
qwer3456tzui7890

{

 "scaAuthenticationData" : "123456"

}
```

Response in case of the embedded approach

Response Code 200

Response Body

```
{

  "transactionStatus" : "AcceptedCustomerProfile"

}
```

8 Sessions: Combination of AIS and PIS Services

The implementation of sessions in the sense of [XS2A-OR], i.e. the combination of AIS and PIS services is an optional feature of this interface. The ASPSP will inform about the support by its PSD2 documentation.

This feature might be relevant where account information services are needed within a payment initiation, especially for batch booking banks. In this case, a consent to access the corresponding account information is needed, cp. Section 6.4. The corresponding GET method to read the account data is using there the header parameter “Consent-ID”. The TPP then can use this consent-ID parameter also in the POST method when applying the Payment Initiation Request, cp. Section 5.3. A pre-requisite to use the Consent-ID in the subsequent Payment Initiation Request is that the flag “combinedServiceIndicator” in the Account Information Consent Request was set, cp. Section 6.4.1.

In a context, where the consent management for account access is fully provided by the OAuth2 model, the corresponding access tokens will support this feature analogously.

9 Confirmation of Funds Service

9.1 Overview Confirmation of Funds Service

The following table defines the technical description of the abstract data model as defined [XS2A OR] for the three PSD2 services. The columns give an overview on the API protocols as follows:

- The “Data element” column is using the abstract data elements following [XS2A OR] to deliver the connection to rules and role definitions in this document.
- The “Attribute encoding” is giving the actual encoding definition within the XS2A API as defined in this document.
- The “Location” columns define, where the corresponding data elements are transported as http parameters, resp. are taken from eIDAS certificates.
- The “Usage” column gives an overview on the usage of data elements in the different services and API Calls. Within [XS2A OR], the XS2A calls are described as abstract API calls. These calls will be technically realised as HTTP POST command. The calls are divided into the following calls:
    - Confirmation Request, which is the only API Call for every transaction within the Confirmation of Funds service.

The following usage of abbreviations in the Location and Usage columns is defined, cp. Also [XS2A OR] for details.

- x: This data element is transported on the corresponding level.
- m: Mandatory
- o : Optional for the TPP to use
- c: Conditional. The Condition is described in the API Calls, condition defined by the ASPSP

Data element	**Attribute encoding**	**Location**				**Usage**	
		Path	Header	Body	Certificate	Conf. Req.	Conf Resp.
Provider Identification		x				m	
TPP Registration Number					x	m	
TPP Name					x	m	
TPP Role					x	m	
Transaction Identification	TPP-Transaction-ID		x				
Request Identification	TPP-Request-ID						
TPP Certificate Data	TPP-Certificate		x			c	
TPP Electronic Signature	Signature		x			c	
Service Type		x				m	
Response Code			x				m
TPP Message Information	tppMessages			x			o
Card Number	cardNumber			x		c	
Account Number	psuAccount			x		m	
Name Payee	payee			x		o	
Transaction Amount	instructedAmount			x		m	

9.2 Confirmation of Funds Request

Call

POST /v1/funds-confirmations

Creates a confirmation of funds request at the ASPSP.

Query ParametersNo specific Query Parameters.

Query Parameter

No specific query parameter.

Request Header

Attribute	Type	Condition	Description
TPP-Transaction-ID	UUID	Mandatory	ID of the transaction as determined by the initiating party.
TPP-Request-ID	UUID	Mandatory	ID of the request, unique to the call, as determined by the initiating party.
Signature	cp Section 11	Conditional	A signature of the request by the TPP on application level. This might be mandated by ASPSP.
TPP-Certificate	String	Conditional	The certificate used for signing the request, In base64 encoding.

Request Body

Attribute	Type	Condition	Description
cardNumber	String	Conditional	Card Number of the card issued by the PIISP. Must be delivered if available.
psuAccount	Account Reference	Mandatory	PSU's account number.
payee	String	Optional	The merchant where the card is accepted as an information to the PSU.

Attribute	Type	Condition	Description
instructedAmount	Amount	Mandatory	Transaction amount to be checked within the funds check mechanism.

Response Body

Attribute	Type	Condition	Description
fundsAvailable	Boolean	Mandatory	Equals “true” if sufficient funds are available at the time of the request, “false” otherwise.

The following rules will apply in interpreting the Confirmation of Funds Request for multicurrency accounts:

The additional card number might support the choice of the sub-account.

If no card number, but the PSU account identifier is contained: check on default account registered by customer.

If no card number but the PSU and the account identifier with currency is contained: check the availability of funds on the corresponding sub-account.

If card number and the PSU account identifier is contained:: check on sub-account addressed by card, if the addressed card is registered with one of the sub-accounts.

If the card number is not registered for any of the accounts, the card number is ignored.

Example

```
POST https://api.testbank.com/v1/payments/sepa-credit-transfers
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e879
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7721
Date:                 Sun, 06 Aug 2017 15:02:37 GMT

{  "cardNumber": "12345678901234",
   "psuAccount":{"iban":"DE23100120020123456789"},
   "instructedAmount": {"currency" : "EUR" , "content" : "123"}
}
```

Response Body

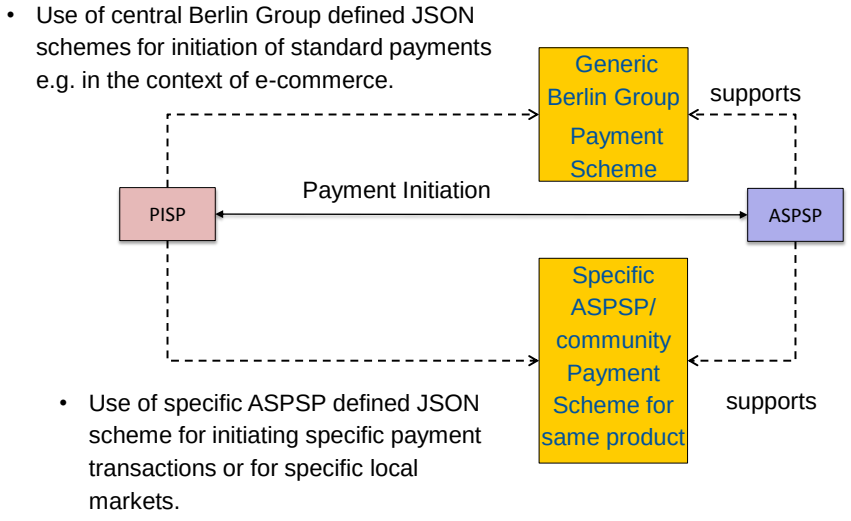
{“fundsAvailable”: ”true”}

10 Core Payment Structures

For core payment products in the European market, this document is defining JSON structures, which will be supported by all ASPSPs

- offering the corresponding payment products to their customers and
- providing JSON based payment endpoints, cp Section 5.3.1 and 5.3.3.1.

At the same time, the ASPSP may offer in addition more extensive JSON structures for the same payment products since they might offer these extensions also in their online banking system.



10.1 Single Payments

The following table first gives an overview on the generic Berlin Group defined JSON structures of standard SEPA payment products for single payments.

Data Element	Type	SCT EU Core	SCT_INST EU Core	Target2 Paym. Core	Cross Curr CT Core
endToEnd Identification	Max35Text	optional	optional	optional	n.a.
debtorAccount (incl. type)	Account Reference	mandatory	mandatory	mandatory	mandatory
ultimateDebtor	Max70Text	n.a.	n.a.	n.a.	n.a.
instructedAmount (inc. Curr.)	Amount	mandatory	mandatory	mandatory	mandatory
creditorAccount	Account Reference	mandatory	mandatory	mandatory	mandatory
creditorAgent	BICFI	optional	optional	optional	optional
creditorName	Max70Text	mandatory	mandatory	mandatory	mandatory
creditorAddress	Address	optional	optional	optional	mandatory
ultimateCreditor	Max70Text	n.a.	n.a.	n.a.	n.a.
purposeCode	Purpose Code	n.a.	n.a.	n.a.	n.a.
remittance Information Unstructured	Max140Text	optional	optional	optional	optional
remittance Information Structured	Remittance	n.a.	n.a.	n.a.	n.a.
requestedExecution Date	ISODate	n.a.	n.a.	n.a.	n.a.
requestedExecution Time	ISODateTime	n.a.	n.a.	n.a.	n.a.

Extensions of these tables are permitted by this specification

- if they are less restrictive (e.g. set the debtor account to optional) or
- if they open up for more data elements (e.g. open up the structured remittance information, or ultimate data fields.)

Only fields defined in Section 13 for payment structures may be added.

Remark: The ASPSP may reject a payment initiation request where additional data elements are used which are not specified.

10.2 Future Dated Payments

One example of an extension of the above defined JSON structure is the requested execution date e.g. for SEPA Credit Transfers. This field is n.a. since not all banks or banking communities might support this as a PSD2 core service.

The ASPSP will indicate the acceptance of future dated payments by issuing an ASPSP specific or community specific JSON scheme, where the attribute “requestedExecutionDate” is an optional field.

10.3 Bulk Payments

This specification offers the bulk payment function in JSON encoding as optional endpoints. The format of the bulk payment is an array of single payments, as offered by the ASPSP.

Example

[{JSON based payment initiation 1}, {JSON based payment initiation 2}]

11 Signatures

When an ASPSP requires the TPP to send a digital signature as defined in [signHTTP], chapter 4 in his HTTP-Requests, the signature must obey the following requirements according or additional to [signHTTP], chapter 4.

11.1 "Digest" Header mandatory

When a TPP includes a signature as defined in [signHTTP], chapter 4, he also must include a "Digest" header as defined in [RFC3230]. The "Digest" Header contains a Hash of the message body. The only hash algorithms that may be used to calculate the Digest within the context of this specification are SHA-256 and SHA-512 as defined in [RFC5843].

11.2 Requirements on the "Signature" Header

As defined in [signHTTP], chapter 4, a "Signature" header must be present. The structure of a "Signature" header is defined in [signHTTP], chapter 4.1, the following table lists the requirements on the "Signature" header from [signHTTP] and additional requirements specific to the PSD2-Interface.

Elements of the "Signature" Header				
Element	Type	Condition	Requirement [signHTTP]	Additional Requirement
keyId	String	Mandatory	The 'keyId' field is an opaque string that the server can use to look up the component they need to validate the signature. It could be an SSH key fingerprint, a URL to machine-readable key data, an LDAP DN, etc. Management of keys and assignment of 'keyId' is out of scope for this document.	Serial Number of the TPP's certificate included in the "Certificate" header of this request.
Algorithm-ID	String	Mandatory	The `algorithm` parameter is used to specify the digital signature algorithm to use when generating the signature. Valid values for this parameter can be found in the Signature Algorithms registry located at http://www.iana.org/assignments/signature-algorithms and MUST NOT be marked "deprecated".	The algorithm must identify the same algorithm for the signature as presented in the certificate (Element "TPP-Certificate") of this Request. It must identify SHA-256 or SHA-512 as Hash algorithm.

Elements of the “Signature” Header				
Element	Type	Condition	Requirement [signHTTP]	Additional Requirement
Headers	String	Optional	The `headers` parameter is used to specify the list of HTTP headers included when generating the signature for the message. If specified, it should be a lowercased, quoted list of HTTP header fields, separated by a single space character. If not specified, implementations MUST operate as if the field were specified with a single value, the `Date` header, in the list of HTTP headers. Note that the list order is important, and MUST be specified in the order the HTTP header field-value pairs are concatenated together during signing.	Mandatory. Must include • “Digest”, • “TPP-Transaction-ID”, • “TPP-Request-ID”, • “PSU-ID” (if and only if “PSU-ID” is included as a header of the HTTP-Request). • “PSU-Corporate-ID” (if and only if “PSU-Corporate-ID” is included as a header of the HTTP-Request). • “Date” **No other entries may be included.**
Signature	String	Mandatory	The `signature` parameter is a base 64 encoded digital signature, as described in RFC 4648 [RFC4648], Section 4. The client uses the `algorithm` and `headers` signature parameters to form a canonicalised `signing string`. This `signing string` is then signed with the key associated with `keyId` and the algorithm corresponding to `algorithm`. The `signature` parameter is then set to the base 64 encoding of the signature.	[No additional Requirements]

Example

Assume a TPP needs to include a signature in the following Request (example in Section 5.3.1)

```
POST https://api.testbank.com/v1/payments/sepa-credit-transfers
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:        3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7721
PSU-IP-Address:       192.168.8.78
PSU-ID:               PSU-1234
PSU-Agent:             Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date:                 Sun, 06 Aug 2017 15:02:37 GMT

{
   "instructedAmount": {"currency" : "EUR" , "amount" : "123"},
   "debtorAccount": { "iban" : "DE23100100101234567 89"},
   "creditor": { "name" : "Merchant123"} ,
   "creditorAccount": {"iban" : "DE23100120020123456789"},
   "remittanceInformationUnstructured": "Ref Number Merchant"
}
```

SHA-256 of the request body is (hex)

'367e0b0eacb5d1ec81f0b97fdfab42d3357ac13e4a67cf94229e11db94f8774c',

which encodes to

Nn4Ldqy10eyB8Ll/36tC0zV6wT5KZ8+Uip4R25T4d0w="

in Base64.

So using signature algorithm rsa-sha256 the signed request of the TPP will be

```
POST https://api.testbank.com/v1/payments/sepa-credit-transfers
Content-Encoding:     gzip
Content-Type:         application/json
TPP-Transaction-ID:   3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID:       99391c7e-ad88-49ec-a2ad-99ddcb1f7721
PSU-IP-Address:       192.168.8.78
PSU-ID:               PSU-1234
PSU-Agent:            Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0)
Gecko/20100101 Firefox/54.0
Date:                 Sun, 06 Aug 2017 15:02:37 GMT
Digest:               SHA-
256=+QuIK8Q57UdqQJeMYDM3WWLyrQmpOPc7bJ+zQgUWHLE=
Signature:
     keyId="Serial_Number_Of_The_TPP's_certificate",algorithm="rsa-
sha256",
   headers="Digest TPP-Transaction-ID TPP-Request-ID PSU-ID Date",
   signature="Base64(RSA-SHA256(signing string))"
TPP-Certificate: TPP's_eIDAS_Certificate

{
   "instructedAmount": {"currency" : "EUR" , "amount" : "123"},
   "debtorAccount": { "iban" : "DE23100100101234567890"},
   "creditor": { "name" : "Merchant123"} ,
   "creditorAccount": {"iban" : "DE23100120020123456789"},
   "remittanceInformationUnstructured": "Ref Number Merchant"
}
```

Where *signing string* is

```
Digest: SHA-256=Nn4Ldqy10eyB8Ll/36tC0zV6wT5KZ8+Uip4R25T4d0w=
TPP-Transaction-ID: 3dc3d5b3-7023-4848-9853-f5400a64e80f
TPP-Request-ID: 99391c7e-ad88-49ec-a2ad-99ddcb1f7721
PSU-ID: PSU-1234
Date: Sun, 06 Aug 2017 15:02:37 GMT
```

12 Requirements on the OAuth2 Protocol

The OAuth2 protocol as used optionally for this API is defined in [RFC 6749]. In this section, additional requirements on the protocol are defined.

The requirements on the data exchange between the TPP and the Oauth Server of the ASPSP are identical to the data exchange requirements between TPP and the XS2A Interface, cp. Section 3.

The grant types “authorisation code” and “refresh token” are recommended by this specification.

The ASPSP is required to provide TPPs with configuration data conforming to the “OAuth 2.0 Authorisation Server Metadata” specification.

12.1 Authorisation Request

For the “Authorisation Request” of the TPP to the /authorisation endpoint the following parameters are defined:

Query Parameters

Attribute	Condition	Description
responseType	Mandatory	“code” is recommended as response type.
clientId	Mandatory	NN concatenated with the registration number of the TPP at the national FSA, where NN is the two character country identifier of the corresponding FSA
scope	Mandatory	PIS: The scope is the reference to the payment resource in the form “PIS:<paymentId>”. AIS: The scope is the reference to the consent resource for account access in the form “AIS:<consentId>"
state	Mandatory	A dynamical value set by the TPP and used to prevent XSRF attacks.
redirectUri	Mandatory	the URI of the TPP where the OAuth2 server is redirecting the PSU after the authentication.
codeChallenge	Mandatory	PKCE challenge according to cryptographic RFC 7636 (https://tools.ietf.org/html/rfc7636) used to prevent code injection attacks.

Attribute	Condition	Description
codeChallengeMethod	Optional	Code verifier transformation method, is “S256” or “plain”. “S265” is recommended by this specification.

Example

```
GET /authorise?responseType=code&clientId=DE123456789&
scope=ais%3A1234-wertiq-983+offline_access&
state= S8NJ7uqk5fY4EjNvP_G_FtyJu6pUsvH9jsYni9dMAJw&
redirectUri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb&
codeChallengeMethod=”S256”
codeChallenge=5c305578f8f19b2dcdb6c3c955c0aa709782590b4642eb890b97e4
3917cd0f36 HTTP/1.1
    Host: api.testbank.com
```

12.2 Authorisation Response

The Authorisation Response of the ASPSP to the TPP will deliver the following data:

HTTP Response Code

302

Query Parameters

Attribute	Condition	Description
Location:	Mandatory	redirect uri of the TPP
code	Mandatory	Authorisation code
state	Mandatory	Same value as for the request.

Example

HTTP/1.1 302 Found

```
Location: https://client.example.com/cb
?code=SplxlOBeZQQYbYS6WxSbIA
&state=S8NJ7uqk5fY4EjNvP_G_FtyJu6pUsvH9jsYni9dMAJw
```

12.3 Token Request

The TPP sends a POST request to the token endpoint in order to exchange the authorisation code provided in the authorisation response for an access token and, optionally, a refresh token. The following parameters are used:

Request Parameters

Attribute	Condition	Description
grantType	Mandatory	“authorisationCode” is recommended as response type.
clientId	Mandatory	NN concatenated with the registration number of the TPP at the national FSA, where NN is the two character country identifier of the corresponding FSA.
Code	Mandatory	Authorisation code from the authorisation response
redirectUri	Mandatory	the exact uri of the TPP where the OAuth2 server redirected the user agent to for this particular transaction
codeVerifier	Mandatory	PKCE verifier according to cryptographic RFC 7636 (https://tools.ietf.org/html/rfc7636) used to prevent code injection attacks.

Example

POST /token HTTP/1.1

Host: https://api.testbank.com

```
Content-Type: application/x-www-form-urlencoded
clientId=DE123456789
&grantType=authorisationCode
&code=SplxlOBeZQQYbYS6WxSbIA
&redirectUri= https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
&codeVerifier=7814hj4hjai87qqhjz9hahdeu9qu771367647864676787878
```

The TPP is authenticated during this request by utilising “Oauth 2.0 Mutual TLS Client Authentication and Certificate Bound Access Tokens” in conjunction with the TPP’s eIDAS certificate.

12.4 Token Response

The ASPSPS responds with the following parameters:

Response Parameters

Attribute	Condition	Description
accessToken	Mandatory	Access Token bound to the scope as requested in the authorisation request and confirmed by the PSU.
tokenType	Mandatory	Set to “Bearer”
expiresIn	Optional	The lifetime of the access token in seconds
refreshToken	Optional	Refresh Token, which can be utilised to obtain a fresh access tokens in case the previous access token expired or was revoked. Especially useful in the context of AIS.

Example

HTTP/1.1 200 OK

```
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
{
 “accessToken”: “SlAV32hkKG”,
 “tokenType”: “Bearer”,
 “expiresIn”: 3600,
 “refreshToken”: “tGzv3JokF0XG5Qx2TlKWIA”
}
```

12.5 Refresh Token Grant Type

The ASPSP may issue refresh tokens at its discretion, e.g. if an AISP uses the standard scope value “offline_access” or if the recurringIndicator in is set to “true”.

13 Complex Data Types and Code Lists

In the following constructed data types are defined as used within parameter sections throughout this document.

13.1 PSU Data

Attribute	Type	Condition	Description
password	String	Optional	

13.2 TPP Message Information

Attribute	Type	Condition	Description
category	String	Mandatory	Only ”ERROR” or "WARNING" permitted
code	Message Code	Mandatory	
path	String	Conditional	
text	Max512Text	Optional	Additional explaining text.

13.3 Amount

Attribute	Type	Condition	Description
currency	Currency Code	Mandatory	ISO 4217 code
content	Floating Point Number	Mandatory	The amount given with fractional digits, where fractions must be compliant to the currency definition. The decimal separator is a dot.

13.4 Address

Attribute	Type	Condition	Description
street	Max70Text	Optional	
buildingNumber	String	Optional	
city	String	Optional	
postalCode	String	Optional	
country	Country Code	Mandatory	

13.5 Remittance

Attribute	Type	Condition	Description
reference	Max35Text	Mandatory	The actual reference.
referenceType	Max35Text	Optional	
referenceIssuer	Max35Text	Optional	

13.6 Links

Attribute	Type	Condition	Description
redirect	String	Optional	A link to an ASPSP site where SCA is performed within the Redirect SCA approach.
oAuth	String	Optional	The link refers to a JSON document specifying the OAuth details of the ASPSP's authorisation server. JSON document follows the definition given in https://tools.ietf.org/html/draft-ietf-oauth-discovery.
updatePsuIdentification	String	Optional	The link to the payment initiation or account information resource, which needs to be updated by the PSU

Attribute	Type	Condition	Description
			identification if not delivered yet.
updateProprietaryData	String	Optional	The link to the payment initiation or account information resource, which needs to be updated by the proprietary data.
updatePsuAuthentication	String	Optional	The link to the payment initiation or account information resource, which needs to be updated by a PSU password and eventually the PSU identification if not delivered yet.
selectAuthenticationMethod	String	Optional	This is a link to a resource, where the TPP can select the applicable second factor authentication methods for the PSU, if there were several available authentication methods.
authoriseTransaction	String	Optional	The link to the payment initiation or consent resource, where the "Transaction Authorisation"Request" is sent to. This is the link to the resource which will authorise the payment or the consent by checking the SCA authentication data within the Embedded SCA approach.
self	String	Optional	The link to the payment initiation resource created by the request itself. This link can be used later to retrieve the transaction status of the payment initiation.
status	String	Optional	
viewAccount	String	Optional	
viewBalances	String	Optional	A link to the resource providing the balance of a dedicated account.
viewTransactions	String	Optional	A link to the resource providing the transaction history of a dediated amount.

Attribute	Type	Condition	Description
first	String	Optional	Navigation link for paginated account reports.
next	String	Optional	Navigation link for paginated account reports.
previous	String	Optional	Navigation link for paginated account reports.
last	String	Optional	Navigation link for paginated account reports.
download	String	Optional	Download link for huge AIS data packages.

13.7 Authentication Object

Attribute	Type	Condition	Description
authenticationType	Authentication Type	Mandatory	Type of the authentication method.
authenticationVersion	String	Conditional	Depending on the authenticationType. This version can be used by differentiating authentication tools used within performing OTP generation in the same authentication type. This version can be referred to in the ASPSP's documentation.
authenticationMethodId	Max35Text	Mandatory	An identification provided by the ASPSP for the later identification of the authentication method selection.
name	String	Optional	This is the name of the authentication method defined by the PSU in the Online Banking frontend of the ASPSP. Alternatively this

Attribute	Type	Condition	Description
			could be a description provided by the ASPSP like “SMS OTP on phone +49160 xxxxx 28”. This name shall be used by the TPP when presenting a list of authentication methods to the PSU, if available.
explanation	String	Optional	detailed information about the sca method for the PSU

13.8 Authentication Type

More authentication types might be added during implementation projects and documented in the ASPSP documentation.

Name	**Description**
SMS_OTP	An SCA method, where an OTP linked to the transaction to be authorised is sent to the PSU through a SMS channel.
CHIP_OTP	An SCA method, where an OTP is generated by a chip card, e.g. an TOP derived from an EMV cryptogram. To contact the card, the PSU normally needs a (handheld) device. With this device, the PSU either reads the challenging data through a visual interface like flickering or the PSU types in the challenge through the device key pad. The device then derives an OTP from the challenge data and displays the OTP to the PSU.
PHOTO_OTP	An SCA method, where the challenge is a QR code or similar encoded visual data which can be read in by a consumer device or specific mobile app. The device resp. the specific app than derives an OTP from the visual challenge data and displays the OTP to the PSU.
PUSH_OTP	An OTP is pushed to a dedicated authentication APP and displayed to the PSU.

13.9 Challenge

Attribute	Type	Condition	Description
image	String	Optional	PNG data (max. 512 kilobyte) to be displayed to the PSU, Base64 encoding , cp. [RFC 4648]. This attribute is used only, when PHOTO_OTP or CHIP_OTP is the selected SCA method.
data	String	Optional	String challenge data
imageLink	String	Optional	A link where the ASPSP will provides the challenge image for the TPP.
otpMaxLength	Integer	Optional	The maximal length for the OTP to be typed in by the PSU.
otpFormat	String	Optional	The format type of the OTP to be typed in. The admitted values are “characters” or “integer”.
additional Information	String	Optional	Additional explanation for the PSU to explain e.g. fallback mechanism for the chosen SCA method. The TPP is obliged to show this to the PSU.

13.10 Message Code

The permitted message error codes and related http response codes are listed below.

Service unspecific Error Codes

Message Code	http response code	Description
CERTIFICATE_INVALID	401	The contents of the signature/corporate seal certificate are not matching PSD2 general PSD2 or attribute requirements.
CERTIFICATE_EXPIRED	401	Signature/corporate seal certificate is expired.
CERTIFICATE_BLOCKED	401	Signature/corporate seal certificate has been blocked by the ASPSP.
CERTIFICATE_REVOKED	401	Signature/corporate seal certificate has been revoked by QSTP.
CERTIFICATE_MISSING	401	Signature/corporate seal certificate was not available in the request but is mandated for the corresponding.
SIGNATURE_INVALID	401	Application layer eIDAS Signature for TPP authentication is not correct.
SIGNATURE_MISSING	401	Apllication layer eIDAS Signature for TPP authentication is mandated by the ASPSP but is missing.
FORMAT_ERROR	400	Format of certain request fields are not matching the XS2A requirements. An explicit path to the corresponding field might be added in the return message.
PSU_CREDENTIALS_INVALID	401	The PSU-ID cannot be matched by the addressed ASPSP or is blocked, or a password resp. OTP was not correct. Additional information might be added.
SERVICE_INVALID	400 (if payload) 405 (if http method)	The addressed service is not valid for the addressed resources or the submitted data.

Message Code	http response code	Description
SERVICE_BLOCKED	403	This service is not reachable for the addressed PSU due to a channel independent blocking by the ASPSP. Additional information might be given by the ASPSP.
CORPORATE_ID_INVALID	401	The PSU-Corporate-ID cannot be matched by the addressed ASPSP.
CONSENT_UNKNOWN	403 (if path) 400 (if payload)	The consent-ID cannot be matched by the ASPSP relative to the TPP.
CONSENT_INVALID	401	The consent was created by this TPP but is not valid for the addressed service/resource.
CONSENT_EXPIRED	401	The consent was created by this TPP but has expired and needs to be renewed.
TOKEN_UNKNOWN	401	The OAuth2 token cannot be matched by the ASPSP relative to the TPP.
TOKEN_INVALID	401	The OAuth2 token is associated to the TPP but is not valid for the addressed service/resource.
TOKEN_EXPIRED	401	The OAuth2 token is associated to the TPP but has expired and needs to be renewed.
RESOURCE_UNKNOWN	404 (if account-id in path) 403 (if other resource in path) 400 (if payload)	The addressed resource is unknown relative to the TPP.
RESOURCE_EXPIRED	403 (if path) 400 (if payload)	The addressed resource is associated with the TPP but has expired, not addressable anymore.
TIMESTAMP_INVALID	400	Timestamp not in accepted time period.
PERIOD_INVALID	400	Requested time period out of bound.

Message Code	http response code	Description
SCA_METHOD_UNKNOWN	400	Addressed SCA method in the Authentication Mehtod Select Request is unknown or cannot be matched by the ASPSP with the PSU.
TRANSACTION_ID_INVALID	400	The TPP-Transaction-ID is not matching the temporary resource.

PIS specific error codes

Message Code	http response code	Description
PRODUCT_INVALID	403	The addressed payment product is not available for the PSU .
PRODUCT_UNKNOWN	404	The addressed payment product is not supported by the ASPSP.
PAYMENT_FAILED	400	The payment initiation POST request failed during the initial process.. Additional information may be provided by the ASPSP.
REQUIRED_KID_MISSING	401	The payment initiation has failed due to a missing KID. This is a specific message code for the Norwegian market, where ASPSP can require the payer to transmit the KID.

AIS specific error code

Message Code	http response code	Description
CONSENT_INVALID	401	The consent definition is not complete or invalid. In case of being not complete, the bank is not supporting a completion of the consent towards the PSU. Additional information will be provided.

Message Code	http response code	Description
SESSIONS_NOT_SUPPORTED	400	The combined service flag may not be used with this ASPSP.
ACCESS_EXCEEDED	429	The access on the account has been exceeding the consented multiplicity per day.
REQUESTED_FORMATS _INVALID	406	The requested formats in the Accept header entry are not matching the formats offered by the ASPSP.

PIIS specific error code

Message Code	http response code	Description
CARD_INVALID	400	Addressed card number is unknown to the ASPSP or not associated to the PSU.
NO_PIIS_ACTIVATION	400	The PSU has not activated the addressed account for the usage of the PIIS associated with the TPP.

13.11 Transaction Status

The transaction status is filled with value of the ISO20022 data table.

Code	Name	ISO 20022 Definition
ACCP	AcceptedCustomerProfile	Preceding check of technical validation was successful. Customer profile check was also successful.
ACSC	AcceptedSettlementCompleted	Settlement on the debtor's account has been completed. Usage : this can be used by the first agent to report to the debtor that the transaction has been completed. Warning : this status is provided for transaction status reasons, not for financial information. It can only be used after bilateral agreement
ACSP	AcceptedSettlementInProcess	All preceding checks such as technical validation and customer profile were successful and therefore the payment initiation has been accepted for execution.

Code	Name	ISO 20022 Definition
ACTC	AcceptedTechnicalValidation	Authentication and syntactical and semantical validation are successful
ACWC	AcceptedWithChange	Instruction is accepted but a change will be made, such as date or remittance not sent.
ACWP	AcceptedWithoutPosting	Payment instruction included in the credit transfer is accepted without being posted to the creditor customer's account.
RCVD	Received	Payment initiation has been received by the receiving agent.
PDNG	Pending	Payment initiation or individual transaction included in the payment initiation is pending. Further checks and status update will be performed.
RJCT	Rejected	Payment initiation or individual transaction included in the payment initiation has been rejected.

If the response is XML based, then the Code entry is used, as required by the pain.002 schema.

If the response is JSON based, then the Name entry is used, to get a better readability.

13.12 Account Access

Attribute	Type	Condition	Description
accounts	Array of Account Reference	Optional	Is asking for detailed account information. If the array is empty, the TPP is asking for an accessible account list. This may be restricted in a PSU/ASPSP authorization dialogue.
balances	Array of Account Reference	Optional	Is asking for balances of the addressed accounts. If the array is empty, the TPP is asking for the balances of all accessible account lists. This may be restricted in a PSU/ASPSP authorization dialogue
transactions	Array of Account Reference	Optional	Is asking for transactions of the addressed accounts. If the array is empty, the TPP is asking for the transactions of all accessible account lists. This may be restricted in a PSU/ASPSP authorization dialogue
availableAccounts	String	[Optional]	Only the value "all-accounts" is admitted.
allPsd2	String	[Optional]	Only the value "all-accounts" is admitted.

13.13 Account Reference

This type is containing any account identification which can be used on payload-level to address specific accounts. The ASPSP will document which account reference type it will support. Exactly one of the attributes defined as "conditional" shall be used.

Remark: The currency of the account is needed, where the currency is an account characteristic identifying certain sub-accounts under one external identifier like an IBAN. These sub-accounts are separated accounts from a legal point of view and have separated balances, transactions etc.

Attribute	Type	Condition	Description
iban	IBAN	Conditional	

Attribute	Type	Condition	Description
bban	BBAN	Conditional	This data elements is used for payment accounts which have no IBAN.
pan	String	Conditional	Primary Account Number (PAN) of a card, can be tokenised by the ASPSP due to PCI DSS requirements.
maskedPan	String	Conditional	Primary Account Number (PAN) of a card in a masked form.
msisdn	String	Conditional	An alias to access a payment account via a registered mobile phone number.
currency	Currency Code	Optional	

13.14 Account Details

Remark: The ASPSP shall give at least one of the account reference identifiers listed as optional below.

Attribute	Type	Condition	Description
id	Max35Text	Conditional	This is the data element to be used in the path when retrieving data from a dedicated account, cp. Section 6.6.3 or Section 6.6.4 below. This shall be filled, if addressable resource are created by the ASPSP on the /accounts or /card-accounts endpoint.
iban	IBAN	Optional	This data element can be used in the body of the Consent Request Message for retrieving account access consent from this payment account, cp. Section 6.3.1.1.
bban	BBAN	Optional	This data element can be used in the body of the Consent Request Message for retrieving account access consent from this account, cp. Section 6.3.1.1. This data elements is used for payment accounts which have no IBAN.

Attribute	Type	Condition	Description
pan	Max35Text	Optional	Primary Account Number (PAN) of a card, can be tokenized by the ASPSP due to PCI DSS requirements. This data element can be used in the body of the Consent Request Message for retrieving account access consent from this card, cp. Section 6.4.1.1.
maskedPan	Max35Text	optional	Primary Account Number (PAN) of a card in masked form. This data element can be used in the body of the Consent Request Message for retrieving account access consent from this card, cp. Section 6.4.1.1.
msisdn	Max35Text	optional	An alias to access a payment account via a registered mobile phone number. This alias might be needed e.g. in the payment initiation service, cp. Section 5.3.1. The support of this alias must be explicitly documented by the ASPSP for the corresponding API Calls.
currency	Currency Code	Mandatory	Account currency
name	Max35Text	Optional	Name of the account given by the bank or the PSU in Online-Banking
accountType	Max35Text	Optional	Product Name of the Bank for this account, proprietary definition
cashAccountType	Cash Account Type	Optional	ExternalCashAccountType1Code from ISO20022
bic	BICFI	Optional	The BIC associated to the account.
balances	Array of Balances	Conditional	
_links	Links	Optional	Links to the account, which can be directly used for retrieving account information from this dedicated account. Links to “balances” and/or “transactions”

Attribute	Type	Condition	Description
			These links are only supported, when the corresponding consent has been already granted.

13.15 Balances

Attribute	Type	Condition	Description
closingBooked	Single Balance	Optional	Balance of the account at the end of the pre-agreed account reporting period. It is the sum of the opening booked balance at the beginning of the period and all entries booked to the account during the pre-agreed account reporting period.
expected	Single Balance	Optional	Balance composed of booked entries and pending items known at the time of calculation, which projects the end of day balance if everything is booked on the account and no other entry is posted.
authorised	Single Balance	Optional	The expected balance together with the value of a pre-approved credit line the ASPSP makes permanently available to the user.
openingBooked	Single Balance	Optional	Book balance of the account at the beginning of the account reporting period. It always equals the closing book balance from the previous report.
interimAvailable	Single Balance	Optional	Available balance calculated in the course of the account 'servicer's business day, at the time specified, and subject to further changes during the business day. The interim balance is calculated on the basis of booked credit and debit items during the calculation time/period specified.

13.16 Single Balance

Attribute	Type	Condition	Description
amount	Amount	Mandatory	
lastActionDateTime	ISODateTime	Optional	This data element might be used to indicate e.g. with the expected or booked balance that no action is known on the account, which is not yet booked.
date	ISODate	Optional	

13.17 Account Report

Attribute	Type	Condition	Description
booked	Array of transactions	Mandatory	
pending	Array of transactions	Optional	
_links	Links	Mandatory	The following links might be used within this context:
			viewAccount (mandatory)
			frst (optional)
			next (optional)
			previous (optional)
			last (optional)

13.18 Transactions

Attribute	Type	Condition	Description
transactionId	Max35Text	Optional	Can be used as access-ID in the API, where more details on an transaction is offered.

Attribute	Type	Condition	Description
endToEndId	Max35Text	Optional	
mandateId	Max35Text	Optional	Identification of Mandates, e.g. a SEPA Mandate ID
creditorId	Max35Text	Optional	Identification of Creditors, e.g. a SEPA Creditor ID
bookingDate	ISODate	Optional	
valueDate	ISODate	Optional	
amount	Amount	Mandatory	
creditorName	Max70Text	Optional	Name of the creditor if a “Debited” transaction
creditor Account	Account Reference	Conditional	
ultimate Creditor	Max70Text	Optional	
debtorName	Max70Text	Optional	Name of the debtor if a “Credited” transaction
debtorAccount	Account Reference	Conditional	
ultimateDebtor	Max70Text	Optional	
remittance Information Unstructured	Max140Text	Optional	
remittance Information Structured	Max140Text	Optional	Reference to be transported in the field.
purposeCode	Purpose Code	Optional	
bank TransactionCode	Bank Transaction	Optional	Bank transaction code as used by the ASPSP in ISO20022 related formats.

Attribute	Type	Condition	Description
	Code		

13.19 Geo Location

Format using [RFC 2426], i.e. “GEO:”<latitude>,< longitude >.

13.20 Frequency Code

The following codes from the EventFrequency7Code of ISO20022 are supported:

Daily, Weekly, EveryTwoWeeks, Monthly, EveryTwoMonths, Quarterly, SemiAnnual, Annual

13.21 Other ISO-related basic Types

The following codes and definitions are used from ISO20022

- Purpose Code: ExternalPurpose1Code
- Cash Account Type: ExternalCashAccountType1Code
- Bank Transaction Code: ExternalBankTransactionDomain1Code
- BICFI: BICFIIdentifier
- IBAN: IBAN2007Identifier
- BBAN: BBANIdentifier
- Floating Point Number: Same definitions as done by the ActiveOrHistoricCurrencyAndAmount definition on the amount

For all codes used in JSON structures, not the abbreviation defined for XML encoding, but the name of the code is used as value.

The following Codes are used from other ISO standards:

- Currency Code: Codes following ISO 4217

Further basic ISO data types:

- ISODateTime: A particular point in the progression of time defined by a mandatory date and a mandatory time component, expressed in either UTC time

format (YYYY-MM-DDThh:mm:ss.sssZ), local time with UTC offset format (YYYY-MM-DDThh:mm:ss.sss+/-hh:mm), or local time format (YYYY-MMDDThh:mm:ss.sss). These representations are defined in "XML Schema Part 2: Datatypes Second Edition - W3C Recommendation 28 October 2004" which is aligned with ISO 8601.

- ISODate: A particular point in the progression of time in a calendar year expressed in the YYYY-MM-DD format.

14 References

[XS2A-OR] NextGenPSD2 XS2A Framework, Operational Rules, The Berlin Group Joint Initiative on a PSD2 Compliant XS2A Interface, version1.0, published 08 February 2018

[EBA-RTS] Commission Delegated Regulation (EU) No .../.. of XXX supplementing Directive 2015/2366 of the European Parliament and of the Council with regard to Regulatory Technical Standards for Strong Customer Authentication and Common and Secure Open Standards of Communication, C(2017) 7782 final, published 27 November 2017

[eIDAS] Regulation (EU) No 910/2014 of the European Parliament and of the Council on Electronic Identification and Trust Services for Electronic Transactions in the Internal Market, 23 July 2014, published 28 August 2014

[PSD2] Directive (EU) 2015/2366 of the European Parliament and of the Council on payment services in the internal market, published 25 November 2016

[signHTTP] Signing HTTP messages, Network Working Group, Internet Draft, https://datatracker.ietf.org/doc/draft-cavage-http-signatures/

[RFC3230] Mogul, J. and A. Van Hoff, "Instance Digests in HTTP", RFC 3230, DOI 10.17487/RFC3230, January 2002,<https://www.rfc-editor.org/info/rfc3230>

[RFC5843] Bryan, A, "Additional Hash Algorithms for HTTP Instance Digests", RFC 5843, DOI 10.17487/RFC5843, April 2010,<https://www.rfc-editor.org/info/rfc5843>

15 Appendix A: Additional Payment Products

Remark: Additional payment products might in future also be published on the Berlin Group website to be able to react in a shorter time.

15.1 JSON based Payment Products

Norway

Data Element	Type	Norway CT non-Euro
endToEndIdentification	Max35Text	n.a.
debtorAccount (incl. type)	Account Reference	mandatory
ultimateDebtor	Max70Text	optional
instructedAmount (inc. Curr.)	Amount	mandatory
creditorAccount	Account Reference	mandatory
creditorAgent	BICFI	optional
creditorName	Max70Text	mandatory
creditorAddress	Address	mandatory
ultimateCreditor	Max70Text	optional
purposeCode	Purpose Code	optional
remittance Information Unstructured	String	optional
remittance Information-structured	Remittance	optional
requestedExecution Time	ISODateTime	optional

Sweden

It is planned to publish common JSON based payment initiation definitions for Swedish payment types in a next version of the specification.