STET PSD2 API

Documentation Part 1: Framework

Author: Robache Hervé

Date: 2019-01-15

Version: 1.4.1.3 (English)

Table of content

1. Introduction

1.1. Context

The revised Payment Service Directive (PSD2) points out some new roles providing services to a Payment Service User (PSU):

- Third Party Providers (TPP) which can be subdivided into three categories
  - Account Information Service Providers (AISP)
  - Payment Initiation Service Providers (PISP)
  - Card Based Payment Instrument Issuers (CBPII)
- Account Servicing Payment Service Providers (ASPSP).

Each Member Country has to transpose the PSD2, within its own national law.

The PSD2 is completed by a set of documents provided by the European Banking Authority (EBA). Among these documents, the Regulatory Technical Standards (RTS) for Strong Customer Authentication (SCA) details some requirements, for instance on security principles: traceability, strong customer authentication…

1.2. Mission

STET has been mandated by its shareholders in order to design and provide an open API (Aka STET PSD2 API) that would specify the different interactions between TPPs and ASPSPs for carrying out the different use cases of PSD2. This API could be extended to other (non-PSD2) use cases in the future but this extension is not part of the mandate.

As the RTS for SCA are now finalised, this version of the API and its documentation takes into account the new constraints and rules that have been introduced.

This version also includes

- Items that have been identified and studied in common with the BERLIN GROUP, in a strategy of convergence of the different European API initiatives.
- Evolvements linked to the change requests that have been received after first public releases of STET PSD2 API.

The STET PSD2 API does not cover:

- Interactions between PSUs and TPP
- Interactions between PSUs and ASPSP
- Registration information management

The technical characteristics of this API are provided within a SWAGGER 2.0 file. The present document purpose is to provide extra-information on this API and to give some interaction samples.

1.3. Legal framework

PSD2:

- http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32015L2366

EBA RTS on SCA and CSC:

- https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.L_.2018.069.01.0023.01.ENG&toc=OJ:L:2018:069:TOC

EBA Opinion on the implementation of the RTS on SCA and CSC:

- https://www.eba.europa.eu/documents/10180/2137845/Opinion+on+the+implementation+of+the+RTS+on+SCA+and+CSC+%28EBA-2018-Op-04%29.pdf

EIDAS:

- http://eur-lex.europa.eu/legal-content/FR/TXT/?uri=celex%3A32014R0910

1.4. Licence

This specification is published under the following licence

"Creative Commons – Attribution 3.0 France (CC BY 3.0 FR)"

This work has been coordinated by STET with the following contributors:

- BNP Paribas
- Le Groupe BPCE
- Le Groupe Crédit Agricole
- La Banque Fédérative du Crédit Mutuel – CIC
- La Banque Postale
- La Société Générale

- La Caisse des Dépôts et Consignations
- Le Crédit Mutuel - ARKEA
- HSBC France
- L'OCBF
- La Fédération Bancaire Française
- LUXHUB
- RAIFFEISEN LU

This release also takes into accounts the work of the Working Group of the French CNPS (Comité National des Paiements Scripturaux), co-chaired by:

- La Banque de France
- La Direction Générale du Trésor

Other attendees than banks to this Working Group were:

- L'ACPR (Autorité de Contrôle Prudentiel et de Résolution)
- La DINSIC (Direction Interministérielle des Systèmes d'Information et de Communication)
- L'AFEPAME (Association des Établissements de Paiement et de Monnaie Électronique)
- MERCATEL
- La FEVAD (Fédération du e-commerce et de la vente à distance)
- L'ASF (Association française des Sociétés Financières)
- WORLDLINE
- BANKIN'
- LINXO
- BUDGET INSIGHT
- LYDIA
- LYRA NETWORK
- AMERICAN EXPRESS

2. Business Model

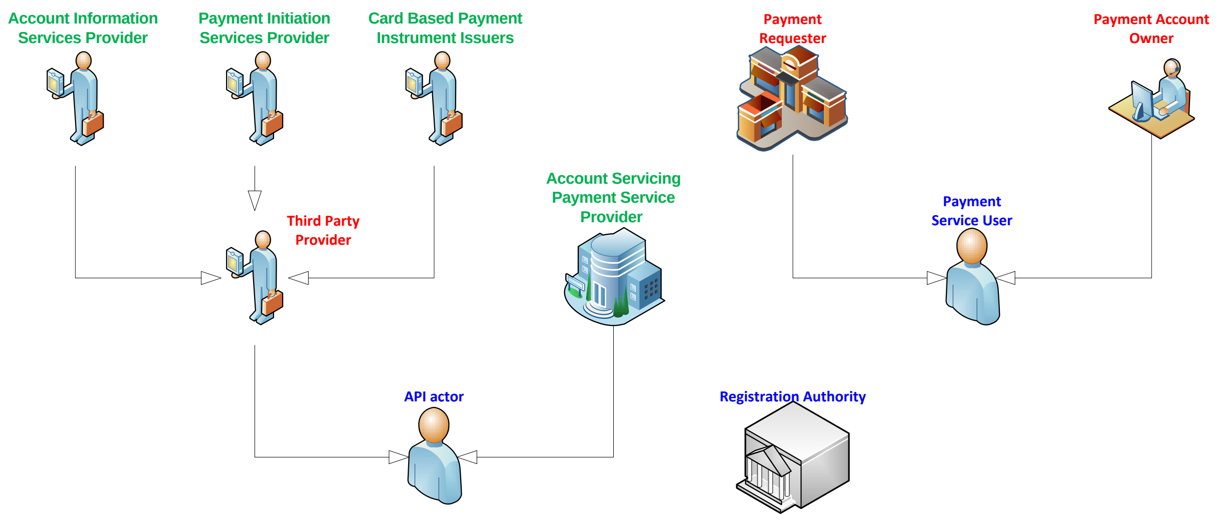
2.1. Actors and Roles

A PSD2 actor is either an entity or a physical person which can endorse one or several roles.

Most of the roles are defined in PSD2. However some extra-roles have been specified for the purpose of the STET PSD2 API during the analysis phase of the project.

Within the following diagram:

- Actors have cyan-coloured labels
- Pure PSD2 roles have green-coloured labels
- Specific STET PSD2 API roles have red-coloured labels


Account Information Services Provider
Payment Initiation Services Provider
Card Based Payment Instrument Issuers
Payment Requester
Payment Account Owner
Account Servicing Payment Service Provider
Third Party Provider
Payment Service User
API actor
Registration Authority

2.1.1. Payment Service User (PSU)

PSUs are the end-users of the services provided by TPPs and ASPSPs.

They are either physical persons or entities (organisations, companies, administrations…).

They do not interact directly with the STET PSD2 API.

A given PSU endorses at least one of the following roles:

- Payment Account Owner (PAO) for one or several accounts held by one or several ASPSPs.
- Payment Requester (PR) asking either for a payment or a coverage check.

2.1.2. API actors

2.1.2.1. Account Servicing Payment Service Provider (ASPSP)

These are Payment Service Providers (PSPs) which are in charge of holding payment accounts for their customers (PSU).

2.1.2.2. Third Party Provider (TPP)

These actors can intermediate between PSUs and ASPSPs, acting on behalf of a PAO or a PR.

On one hand, a given PAO may contract with a TPP in order to use the services provided by this TPP:

- Account Information Services (AISP role) will allow the PAO to get information, through a single interface, about all of his/her accounts, whatever the ASPSP holding this account.
- Card Based Payment Instrument Issuers (CBPII role) that will check the coverage of a given payment amount by the PSU’s account.

On the other hand, a PR may also contract with a TPP that will provide the following services:

- Payment Initiation Services for requesting a Payment Request approval by the PSU and requesting the subsequent execution through a Credit Transfer (PISP role).

2.1.3. Registration Authorities (RA)

RAs are in charge of registering and overviewing the PSD2 actors.

The registration information is the foundation on which each actor can rely in order to know:

- Who is a given actor?
    - Identity
    - Contacts (business, legal, operational…)
    - Insurance coverage
    - Authentication media
        - X.509 eIDAS certificates (https://eur-lex.europa.eu/eli/reg/2014/910/oj)
            - QWAC for TLS mutual authentication
            - QSEALC for content signature
        - Certification chain and services (revocation list, OCSP)
- For which roles this actor has been registered
    - AISP
    - PISP
    - CBPII
    - ASPSP
- Technical characteristics
    - APIs that are provided
    - URLs that are to be used, for test or live processing.

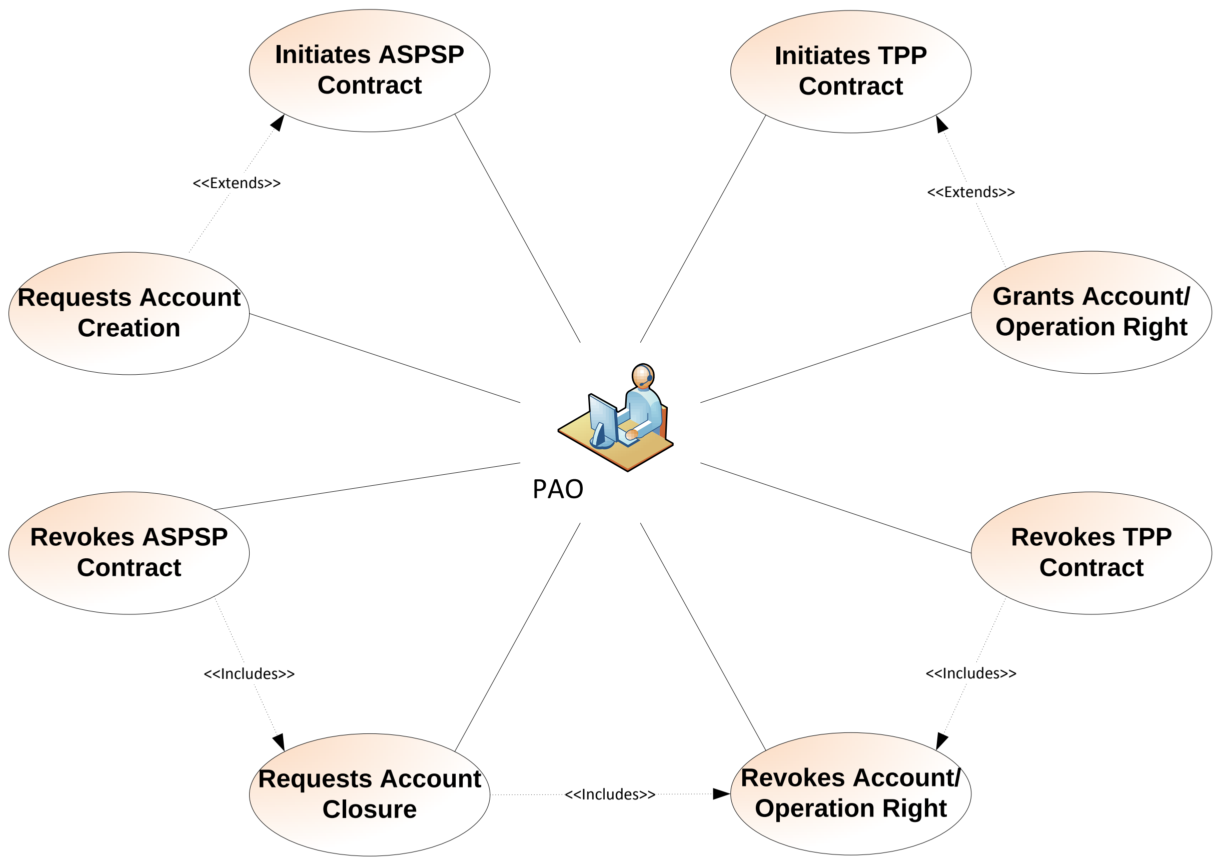
Registration Authorities must keep track of changes for each actor in order to recover the full history of the actor.

2.2. Use cases

Some of the use cases that are listed below are directly implemented by the STET PSD2 API, for they rely on interactions between TPPs and ASPSPs.

Other uses cases are tagged as "NON-API" and are only described for global understanding purpose.

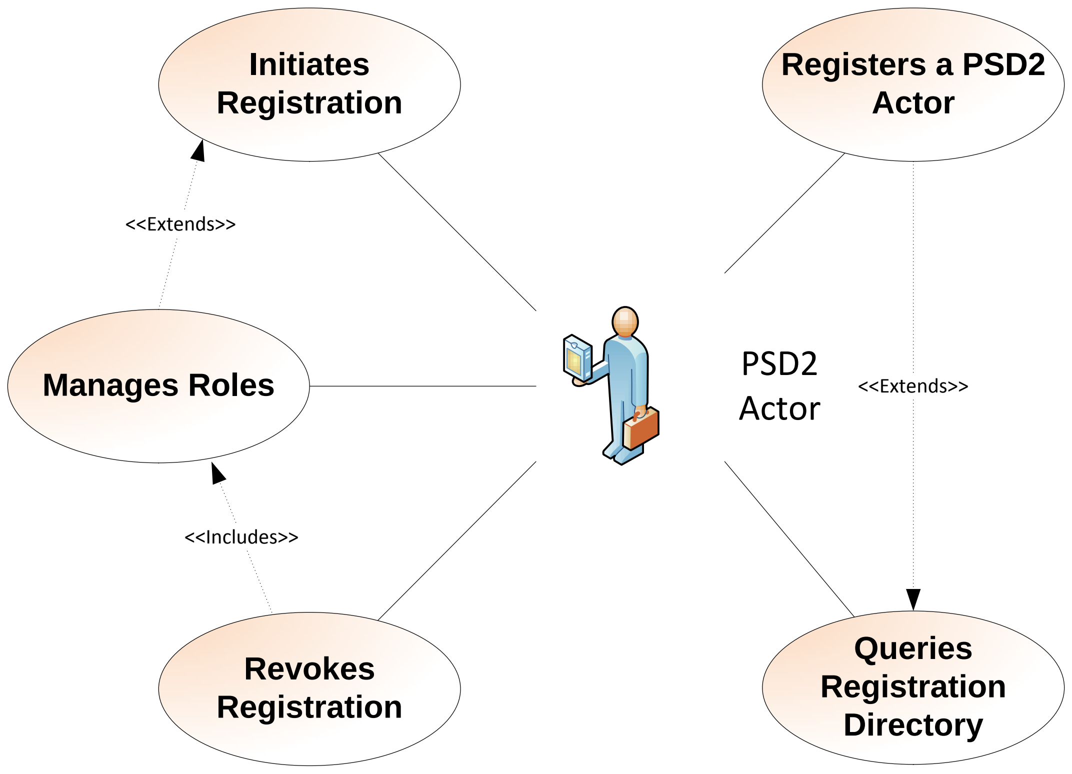
2.2.1. PAO uses cases (NON-API)



USE CASE (PAO)	DESCRIPTION	INTERACTIONS
Initiates ASPSP Contract	The user contracts with an ASPSP in order to use its services. This use case is likely extended by one or more occurrences of the "Requests Account Creation" use case	ASPSP
Requests Account Creation	The user asks the ASPSP to open a new payment account Requires a contract between the PAO and the ASPSP	ASPSP
Requests Account Closure	The user asks the ASPSP to close an existing payment account This use case includes the "revokes Account/Operation Accreditation" use case for all operations on this account and for all granted TPP.	ASPSP TPP (indirectly)

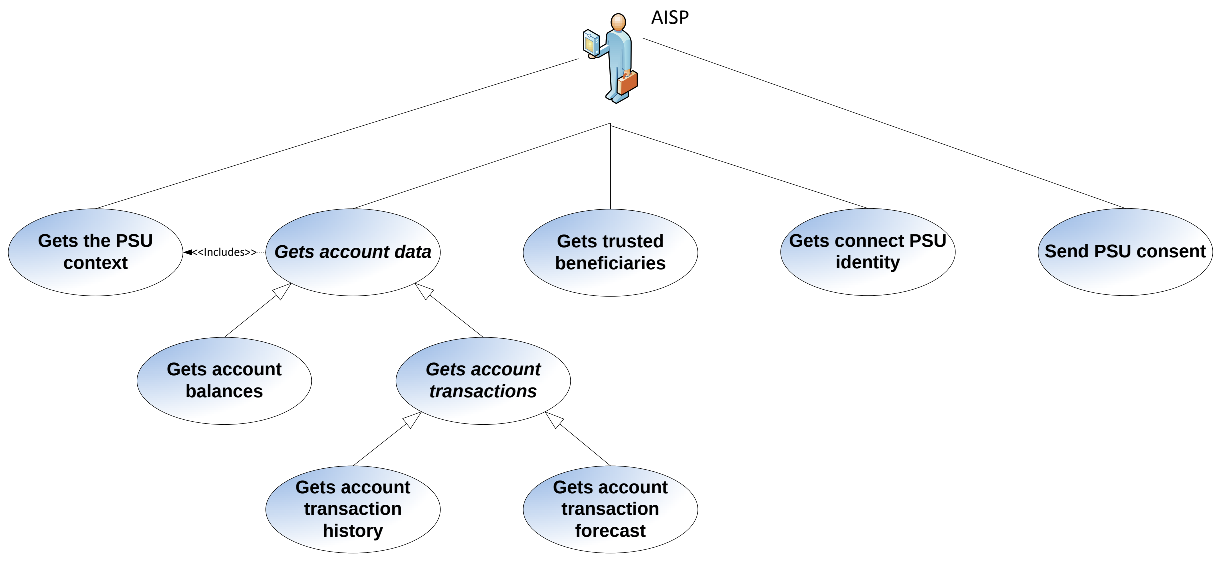
Revokes ASPSP Contract	The user revokes the contract with the ASPSP This use case includes the “Requests Account Closure” use case for each account that is held by the ASPSP. This use case includes the “Revokes Account/Operation Accreditation” use case for all operations on each of these accounts and for all granted TPP.	ASPSP TPP (indirectly)
Initiates TPP Contract	The user contracts with a TPP having AISP and/or CBPII roles in order to use its service This use case is likely extended by one or more occurrences of the “Grants Account/Operation Accreditation” use case	TPP
Grants Account/Operation accreditation	The user allows the TPP to access a given set of operations on one of his/her payment accounts. Requires a contract between the PAO and the ASPSP, a contract between the PAO and the TPP and the registration of this PAO-TPP relationship by the ASPSP to enable the OAUTH2 token management (cf. 3.4.3). Requires also that the capture and the execution of the accreditations are handled by the TPP (the further forwarding of these accreditations is an aISP use case and so out of scope of this use case: cf. 2.2.3).	ASPSP TPP
Revokes Account/Operation accreditation	The user asks the ASPSP to revoke the TPP access for a given set of operations on a given PAO account Requires that the capture and the execution of the revocation are handled by the TPP.	ASPSP TPP
Revokes TPP Contract	The user revokes the contract with the TPP. This use case includes the “Revokes Account/Operation Accreditation” for all grants given to the TPP, whatever the ASPSP. Since this cannot be automated, it is the PAO’s duty to initiate all the relevant revocations with each ASPSP.	TPP ASPSP

2.2.2. Registration use cases (NON-API)



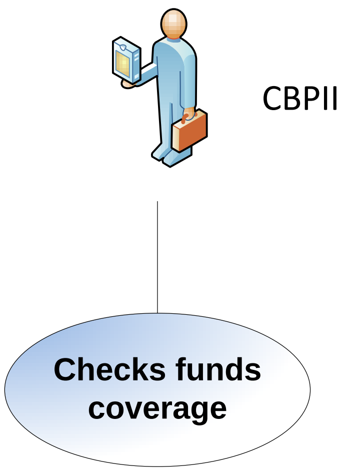
USE CASE (PSD2 ACTOR)	DESCRIPTION	INTERACTIONS
Initiates Registration	The user asks the RA for registration. This use case is likely extended by one or more occurrences of the "Manages Roles" use cases	RA other actors (indirectly)
Manages Roles	The user asks the RA to be referenced for a given set of roles. This use case can be replayed in order to reference or dereference any role.	RA other actors (indirectly)
Revokes registration	The user informs the RA that its registration is to be cancelled	RA other actors (indirectly)
Queries Registration Directory	The user queries the RA directory in order to get data on other PSD2 actors: roles, certificates…	RA other actors (indirectly)
Registers a PSD2 actor	The user registers a given PSD2 actor into its own Directory	None

2.2.3. AISP use cases


AISP
Gets the PSU context
<<Includes>>
Gets account data
Gets trusted beneficiaries
Gets connect PSU identity
Send PSU consent
Gets account balances
Gets account transactions
Gets account transaction history
Gets account transaction forecast

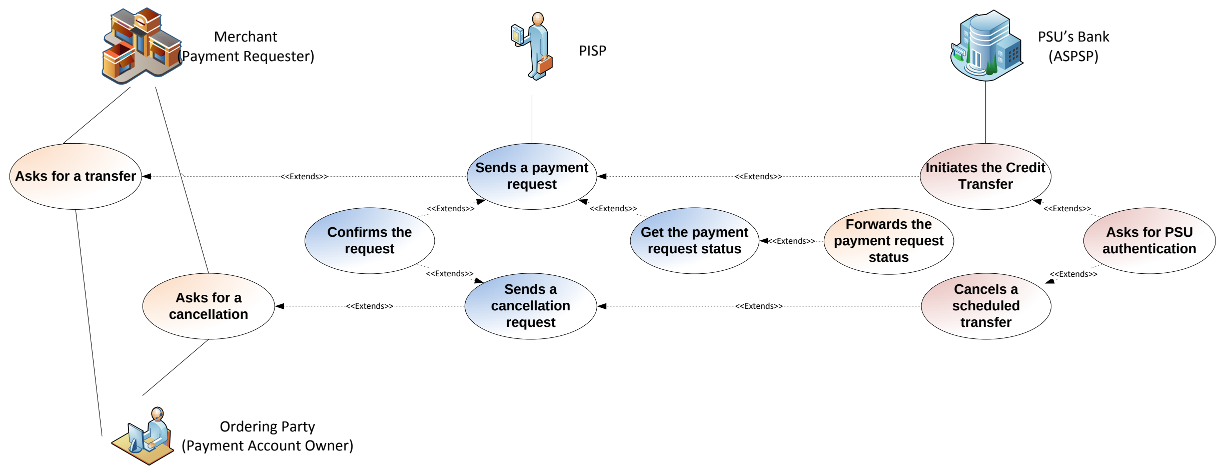
USE CASE (AISP)	DESCRIPTION	INTERACTIONS
Gets the PSU Context	The user queries the ASPSP in order to get the payment accounts that are eligible for the relevant PSU.	ASPSP
Sends the PSU consent to the ASPSP	Having captured the consent choices from the PSU, the user sends them to the ASPSP	ASPSP
Gets Account Data	*This use case is abstract. Its purpose is to stress that the "Gets the PSU Context" is a prerequisite for all other use cases on a given account*	*none*
Gets Account Balance	The user queries the ASPSP in order to get the balance on one given account. The ASPSP can provide several balance computing's (Instant Balance, Accounting Balance…), each balance type being specified with an explicit label.	ASPSP
Gets List of Transactions	This use case is abstract and can be seen as the common interface for the two following uses-cases.	ASPSP
Gets Account Transaction History	The user queries the ASPSP in order to get all the transactions that have been committed to one given PSU account within a given range of value dates.	ASPSP
Gets Account Transaction Forecast	The user queries the ASPSP in order to get all the transactions that are known by the ASPSP to be committed to a given PSU account	ASPSP
Gets connected PSU identity	The user queries the ASPSP in order to get the identity of the PSU on behalf of whom the AISP is connected	ASPSP
Gets trusted beneficiaries	The user queries the ASPSP in order to get all the beneficiaries that were registered as "trusted" par the PSU.	ASPSP

2.2.4. CBPII use cases



USE CASE (CBPII)	DESCRIPTION	INTERACTIONS
Checks Funds Coverage	The user queries the ASPSP in order to check if a given transaction amount can be covered by one given PSU account	ASPSP

2.2.5. PISP uses cases


Merchant (Payment Requester)
PISP
PSU's Bank (ASPSP)
Asks for a transfer
Sends a payment request
Initiates the Credit Transfer
Confirms the request
Get the payment request status
Forwards the payment request status
Asks for PSU authentication
Asks for a cancellation
Sends a cancellation request
Cancels a scheduled transfer
<<Extends>>
Ordering Party (Payment Account Owner)

USE CASE (PSU)	DESCRIPTION	INTERACTIONS
Asks for a transfer (Non-API)	Either the Merchant or the Payment Account Owner asks the PISP to initiate a transfer with one or several payment instructions or to set a standing order.	PISP
Asks for a cancellation (Non-API)	Either the Merchant or the Payment Account Owner asks the PISP to cancel: - all or part of the payment instructions that have been initiated - or a previously set standing order	PISP

USE CASE (PISP)	DESCRIPTION	INTERACTIONS
Sends a Payment Request	The user sends to the ASPSP all the information needed to initiate a Payment. The payment might have been requested either by the beneficiary (e.g. Merchant) or by the account owner him/herself. The payment may include one or several instructions, the maximum number of instructions can be specified by each ASPSP. Those instructions might have - Either a same requested execution date but multiple beneficiaries - Or a same beneficiary but different requested executions dates, those being either explicitly specified or scheduled through a given periodicity (standing orders)	ASPSP
Sends a cancellation request	The user sends to the ASPSP a request to cancel, from a previously posted payment request, one, several or all instructions provided that they have not yet been executed. Cancellations can be performed by sending the Payment request with modifications of the status and reason code at payment level and/or at instruction level.	ASPSP

Confirms the Request	The user confirms the Payment Request or the Cancellation Request to the ASPSP and might forward, through an EMBEDDED workflow, a PAO authentication factor so that the ASPSP can complete the PAO authentication and proceed the request	ASPSP
Gets the Payment Request status	The user gets the status of the Payment Request from the ASPSP. This status embeds: - Information about the payment request and the execution of the subsequent Credit Transfers - Information about the effective booking of the payment instructions that are about to be executed - Information about the availability of funds for payment instructions that are about to be executed but are not effectively booked - Information about the trust given by the PSU to the beneficiary of the payment	ASPSP
Forwards the Payment Request status to the Creditor (Non-API)	The user informs the PR of the status of the Payment Request	PR (Creditor)

USE CASE (ASPSP)	DESCRIPTION	INTERACTIONS
Asks for PSU authentication (Non-API)	Provided the Payment Request is valid, the user asks the PAO in order to authenticate before execution of the relevant Payment Request or Cancellation Request	PSU(PAO)
Initiates the Credit Transfer (Non-API)	Provided the PAO has authenticated, the ASPSP initiates the relevant Credit Transfer.	Beneficiary's ASPSP (Creditor Agent)
Cancels a scheduled transfer (Non-API)	Provided the PAO has authenticated and the relevant transfers have not yet been executed, the ASPSP cancels the execution of the instructions that were specified by the PISP	None

3. Prerequisites and technical details

3.1. Actors registration

PSD2 actors must be registered by a registration authority. The information that has been collected must be accessible to other actors in order to provide trust and interoperability.

A non-registered actor cannot interact with another actor.

Each actor must be provided with at least one eIDAS certificate (QWAC), for TLS 1.2 purpose, delivered by a registered Qualified Trust Service Provider (QTSP).

The European Commission list of QTSPs can be retrieved at the following URL:

https://webgate.ec.europa.eu/tl-browser/

3.2. Cross-Authentication and Data Encryption

The STET PSD2 API relies on TLS 1.2 protocol in order to get cross-authentication between actors. Moreover, this protocol also ensures data confidentiality during their transport on the network.

Whenever a TPP connects as a client to an ASPSP API service, it will check the ASPSP server certificate (QWAC) and present its own eIDAS certificate (QWAC) respecting the ETSI/TS119495 Technical Specification.

The Organisational Identification within the Subject Distinguished Name of the certificate should actually be regarded as an Authorization Number that will respect the following format rules:

- "PSD" as 3 character legal person identity type reference;
- 2 character ISO 3166 [7] country code representing the NCA country;
- hyphen-minus "-" (0x2D (ASCII), U+002D (UTF-8)); and
- 2-8 character NCA identifier (A-Z uppercase only, no separator);
- hyphen-minus "-" (0x2D (ASCII), U+002D (UTF-8)); and
- PSP identifier (authorization number as specified by the NCA).

In case of authentication failure, on one side or the other, the connection must be closed.

No additional encrypting or authenticating feature is required.

3.3. Customer Authentication

Three different approaches can be used by a TPP to allow the authentication by the ASPSP. These three approaches rely on a PSU identification that must be relevant to the ASPSP (National identifier or Bank customer identifier).

These three approaches are implemented in different ways, depending on the relevant use case:

- either during the authorisation process (cf. § 3.4.3), mostly for AISP and CBPII use cases
- or during the consent management process, for instance in case of Payment Request (cf. § 3.4.4)

3.3.1. Redirect Approach

Through the Redirect approach, the PSU authentication process is fully processed by the ASPSP.

In order to allow this, the TPP has to redirect the PSU to the ASPSP authentication service, meaning the PSU will leave temporarily the TPP interface for authenticating towards the ASPSP interface.

The TPP might have already captured a PSU identifier that can be handled by the ASPSP for unambiguously recognizing the PSU. In this case this identifier might be forwarded through the redirection, when the redirect protocol allows the forwarding of this identifier.

After finalisation of the authentication, the ASPSP redirects the PSU back to the TPP interface.

3.3.2. Decoupled approach

Through the Decoupled approach, the PSU authentication process is fully processed by the ASPSP.

In order to allow this the TPP has to capture a PSU identifier that can be handled by the ASPSP for unambiguously recognizing the PSU, and to forward this identifier to the ASPSP.

Based on this identifier, the ASPSP will trigger a Strong Customer Authentication through a decoupled device or application, meaning that the PSU will not leave the TPP interface during the authentication process.

3.3.3. Embedded approach

Through the Embedded approach, the PSU authentication process involves the TPP that will forward one or two authentication factor, these factors being:

- One “Knowledge” factor, e.g. an unlock PIN known by the PSU
- One “Possession” factor, e.g.
    - a One-Time Password sent by the ASPSP on a separate device or application owned by the PSU
    - a response to a challenge sent by the ASPSP on a separate device or application owned by the PSU

3.3.4. Exemptions to Strong Customer Authentication

Exemptions to Strong Customer Authentication are specified by the EBA RTS on SCA, especially for Payment Initiation Services.

In this context, the API allows the PISP to forward to the ASPSP any useful information. Moreover, the PISP may also hint the ASPSP on whether or not the relevant payment request could be subject to an exemption.

Eventually, the ASPSP keeps the final decision to apply or not this exemption.

3.4. Authorization

3.4.1. Technical basis

The TPP is authorized to access the ASPSP’s API through an access token that can be retrieved through the OAUTH2 Authorisation Framework (https://tools.ietf.org/html/rfc6749).

Different authorisation grants can be used, depending on the TPP’s role and use case to be applied.

The OAUTH2 protocol is enforced by checking the identity of the TPP during the OAUTH2 procedures through the TPP’s eIDAS certificate, based on MTLS draft (https://datatracker.ietf.org/doc/draft-ietf-oauth-mtls/).

This enforcement is done by the mandatory provisioning by the TPP of a [client_id] field within all OAUTH2 request. This [client_id] must be linked to the Authorisation Number located within the TPP’s eIDAS certificate.

- This link can be obviously direct when the [client_id] is equal to the Authorisation Number. In this case the ASPSP's API MANAGER might be able to register "on the fly" the OAUTH2 request.
- However in most cases, especially when the API MANAGER is unable to process an "on the fly" registration, this registration should occur prior to the OAUTH2 token request and will result by the provisioning of a [client_id] value by the ASPSP to the TPP.
    - This pre-registration allows the provisioning of multiple [client_id] values that could be used for different use cases by the TPP
    - Moreover this pre-registration may allow the exchange of operational data between the TPP and the ASPSP for further use: logos, phone numbers, email addresses…

In all cases, the ASPSP will have to check the match between the Authorisation Number located within the TPP's eIDAS certificate and the [client_id] value for each OAUTH2 request, this match being indirect or direct, depending on a pre-registration process.

3.4.2. Levels of authorization

The following levels of authorization may be checked and combined in order to compute the effective rights granted to the TPP:

AUTHORIZATION LEVEL	DESCRIPTION
Authorization by TPP role	Once the TPP has been registered for a given role, it can call any of the PSD2 features provided by an ASPSP through the STET PSD2 API for this role.
Authorization by TPP-ASPSP agreement	The TPP can call any of the additional (non PSD2) features provided by an ASPSP through the STET PSD2 API, provided there is a bilateral agreement to use these features.
Authorization by TPP-PSU agreement	If the PSU has contracted with a TPP, he/she must - Give a list of the ASPSPs that it allows the TPP to access - Process an authentication against each of those relevant ASPSPs that will further allow the TPP to access the PSU data.
Authorization by PSU context	The PSU is able to specify his/her PSU context detailing, for each of its relevant accounts: - If this account will be accessible or not by the TPP - Which features can be used by the TPP The PSU can modify at any time his/her PSU context.

3.4.3. AISP and CBPII authorization levels

Since a TPP is acting on behalf of a PSU being a PAO, the PSD2 use cases that are linked with AISP and CBPII roles require the following authorization levels:

- Authorization by Role
- Authorization by TPP-PSU agreement
- Authorization by PSU context

However, in some cases, the CBPII might have been previously enrolled by the PSU to the relevant ASPSP (cf. § 3.4.5).

3.4.3.1. List of the relevant ASPSPs

When contracting with a TPP, the PSU will provide a list of the ASPSPs that it allows the TPP to access. This list may not be exhaustive and so may not include some of the PSU's ASPSPs.

3.4.3.2. Registration of the TPP-PSU agreement by each ASPSP

This registration is due to enable the further access of the TPP to the PSU's data that is hosted by a given ASPSP by providing the TPP with an OAUTH2 access token.

It is requested that AISP and CBPII roles will not be mixed within a single scope definition OAUTH2 access token request.

AISP scope

The OAUTH2 scope requested by an AISP can be one of the following values:

- "aisp"
- "aisp extended_transaction_history"

The first scope value allows the AISP accessing all accessible accounts and data allowed by the PSU until expiration of the by-law specified delay between two SCAs. However, the value does not allow requesting an extended transaction history, i.e. history including transactions older than 90 days.

The second scope value allows the AISP accessing all accessible accounts and data allowed by the PSU until expiration of the by-law specified delay between two SCAs. It also allows requesting an extended transaction history.

However this "aisp extended_transaction_history" scope will be restricted to "aisp" by the ASPSP during the first token refresh. Thus:

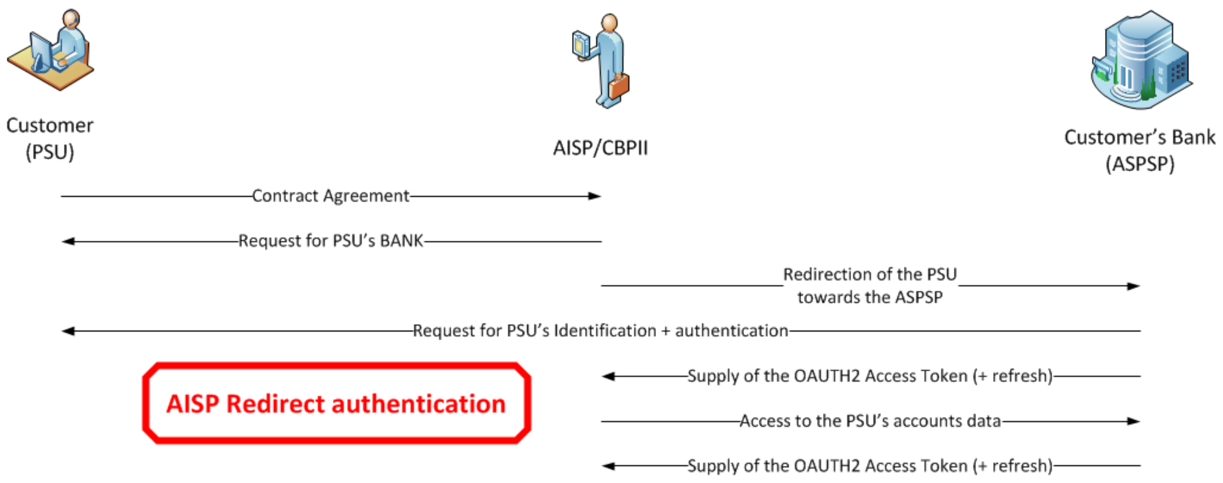
- The AISP will be able to ask for an extended transaction history with the very first access token retrieved after a token request. So, In this case a single SCA will be required and used to get the token and to ask for an extended transaction history.
- Any further extended transaction history request will be considered as out of scope (cf. § 3.4.3.3)

CBPII scope

The OAUTH2 scope requested by a CBPII can only be “cbpii”.

Redirect Approach for AISP and CBPII

The registration process relies on an OAUTH2 sequence for obtaining an Authorization Code Grant (cf. https://tools.ietf.org/html/rfc6749#section-4.1) and can be summarized through the following steps.


Customer (PSU)
AISP/CBPII
Customer’s Bank (ASPSP)
Contract Agreement
Request for PSU’s BANK
Redirection of the PSU towards the ASPSP
Request for PSU’s Identification + authentication
AISP Redirect authentication
Supply of the OAUTH2 Access Token (+ refresh)
Access to the PSU’s accounts data
Supply of the OAUTH2 Access Token (+ refresh)

- The PSU specifies, to the TPP, the identity of one of its ASPSPs
- The TPP initiates the OAUTH2 sequence by redirecting the PSU to the relevant ASPSP’s authorization infrastructure, through the following URL pattern and parameters
- Notice: The RFC 6749 does not specify the Authorization Code Grant to support the forwarding of the Resource Owner (PSU) user name (field “username”). However, this parameter can be valued by the TPP only if, on the ASPSP side, The API MANAGER is able to take this field into account in order to speed up the authentication process.

```
GET /authorize?response_type=code&client_id={clientId}&redirect_uri={redirectUrl}&scope={scope}[&state={state}]
```

NAME		DATA	TYPE AND CONSTRAINS
response_type	[1..1]	Expected type of token	String[10] Must be valued with “code”
client_id	[1..1]	TPP identification	String[34] must be equal or linked to the OrganizationIdentifier part of the Distinguished Name of the eIDAS certificate, according to ETSI specification
redirect_uri	[0..1]	Call-back URL of the TPP	String[140]

scope	[0..1]	Specifies the generic accreditations that both the PSU and the TPP agreed on: - For AISP ○ aisp ○ extended_transaction_history - for CBPII ○ cbpii.	String[140] Space delimited roles list.
state	[0..1]	Internal state that can be used by the TPP for context management.	String[34]

- The ASPSP
  - Identifies and authenticates the PSU
  - Computes the relevant TPP checks (roles, validity, non-revocation…)

- Afterwards, the ASPSP redirects the PSU to the TPP, using the previously given call-back URL (redirect_uri) and the following parameters:

NAME		DATA	TYPE AND CONSTRAINS
code	[1..1]	Short-time code to use in order to get the access token	String[34]
state	[0..1]	Internal state if provided by the TPP	String[34]

- In order to get the access token, the TPP is now able to call, through a POST request, the ASPSP's authorization infrastructure with the following parameters.

```
POST /token HTTP/1.1
    Host: server.example.com

            grant_type=authorization_code
            &code={code}
            &redirect_uri={redirectUrl}
            &client_id={clientId}
```

NAME		DATA	TYPE AND CONSTRAINS
grant_type	[1..1]	Requested authorization type	String[34] Must be valued with "authorization_code"
code	[1..1]	Short-time code previously provided by the ASPSP	String[34]
redirect_uri	[1..1]	Call-back URL of the TPP	String[140] Must be equal to the one provided during the authorization code request
client_id	[1..1]	TPP identification.	String[34] must be equal or linked to the OrganizationIdentifier part of the Distinguished Name of the eIDAS certificate, according to ETSI specification

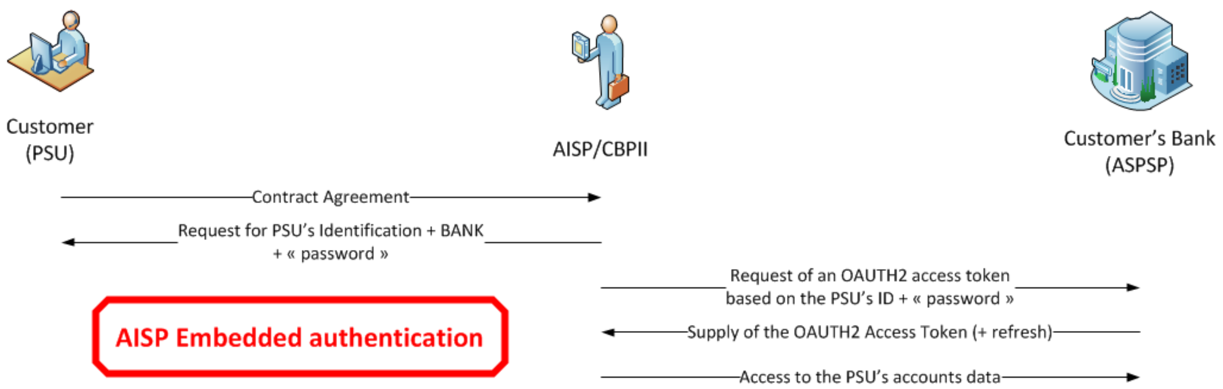
- The ASPSP
  - Identifies and authenticates the TPP through the presented eIDAS certificate (QWAC)

    - Checks the direct or indirect matching between the Authorization Number within the eIDAS certificate and the [client_id] value.
    - Computes the relevant TPP checks (roles, validity, non-revocation…)
- The ASPSP answers through a HTTP200 (OK) response that embeds the following data.

NAME		DATA	TYPE AND CONSTRAINS
access_token	[1..1]	Access token provided by the ASPSP to the TPP.	String[140]
token_type	[1..1]	Type of the provided access token (“Bearer” or “MAC”)	String[10] Must be valued with “Bearer”
expires_in	[0..1]	Token lifetime, in seconds. The token can be used several times as far as it is not expired.	Numeric
refresh_token	[0..1]	Refresh token that can be used for a future token renewal request.	String[140]

Embedded Approach

The registration process relies on an OAUTH2 sequence for obtaining a Resource Owner Password Grant (cf. https://tools.ietf.org/html/rfc6749#section-4.3) and can be summarized through the following steps.


Customer (PSU)
AISP/CBPII
Customer’s Bank (ASPSP)
Contract Agreement
Request for PSU’s Identification + BANK + « password »
AISP Embedded authentication
Request of an OAUTH2 access token based on the PSU’s ID + « password »
Supply of the OAUTH2 Access Token (+ refresh)
Access to the PSU’s accounts data

- The PSU specifies, to the TPP, the identity of one of his/her ASPSPs and provides him with
    - His/her identifier against the ASPSP services
    - A “password” that is the result of a Strong Customer Authentication applied to the PSU by the ASPSP.
- The TPP initiates the OAUTH2 sequence by sending the following request directly to the ASPSP’s Authorisation Service.

```
POST /token HTTP/1.1
    Host: server.example.com


     grant_type=password
     &username=johndoe
     &password=A3ddj3w
     &client_id={clientId}
     &scope={scope}
```

NAME		DATA	TYPE AND CONSTRAINS
grant_type	[1..1]	type of requested grant	String[10] Must be valued with “password”
username	[1..1]	PSU identification	String[34]
password	[1..1]	PSU “password”	String[20] Result of the concatenation of a “knowledge factor” and a “possession” factor
client_id	[1..1]	TPP identification	String[34] must be equal or linked to the OrganizationIdentifier part of the Distinguished Name of the eIDAS certificate, according to ETSI specification
scope	[0..1]	Specifies the generic accreditations that both the PSU and the TPP agreed on: - For AISP ○ aisp ○ extended_transaction_history - for CBPII ○ cbpii.	String[140] Space delimited roles list.

- The ASPSP
  - Identifies and authenticates the TPP through the presented eIDAS certificate (QWAC)
  - Checks the direct or indirect matching between the Authorization Number within the eIDAS certificate and the [client_id] value.
  - Computes the relevant TPP checks (roles, validity, non-revocation…)
- The ASPSP checks the identifier of the PSU and parse the “password” in order to retrieve and check the “Knowledge” factor and the “Possession” factor, thus processing the SCA.
- In case of successful SCA, the ASPSP answers through a HTTP200 (OK) response that embeds the following data.

NAME		DATA	TYPE AND CONSTRAINS
access_token	[1..1]	Access token provided by the ASPSP to the TPP.	String[140]
token_type	[1..1]	Type of the provided access token (“Bearer” or “MAC”)	String[10] Must be valued with “Bearer”
expires_in	[0..1]	Token lifetime, in seconds. The token can be used several times as far as it is not expired.	Numeric
refresh_token	[0..1]	Refresh token that can be used for a future token renewal request.	String[140]

3.4.3.3. Use of the access token

The access token must be used within each request within the “Authorization” header, prefixed by the token type “Bearer”.

The [client_id] that is linked to the access token must directly or indirectly match with the Authorisation Number that is located within the TPP’s eIDAS certificate (QWAC).

If the access token is expired, the request will be rejected with HTTP401 with an error equal to “invalid_token” and the request can be replayed once the access token has been refreshed.

If the access token scope cannot cover the request (case of extended transaction history request for instance):

- The request will be rejected with HTTP403 with an error equal to “insufficient_scope”
- The refresh token will be revoked so the request could be replayed once a new token, having the right scope, would have been requested and provided.
- The new refresh token will be valid up to 90 days.

3.4.3.4. Refreshing the Access Token

According to the RFC 6749 (cf. https://tools.ietf.org/html/rfc6749#section-6), the Refresh Token can be used by the TPP in order to get a refreshed Access Token by the following request.

```
POST /token HTTP/1.1
    Host: server.example.com

    grant_type=refresh_token
    &client_id={clientId}
    &refresh_token=tGzv3JOkF0XG5Qx2TlKWIA
    &scope={scope}
```

NAME		DATA	TYPE AND CONSTRAINS
grant_type	[1..1]		Must be valued with “refresh_token”
refresh_token	[1..1]	Value of the provided refresh token	
client_id	[1..1]	TPP identification	String[34] must be equal or linked to the OrganizationIdentifier part of the Distinguished Name of the eIDAS certificate, according to ETSI specification
scope	[0..1]	Specifies the generic accreditations that both the PSU and the TPP agreed on: “aisp” or “cbpii”. “extended_transaction_history” is not allowed in this case.	String[140] Space delimited roles list.

- The ASPSP

- Identifies and authenticates the TPP through the presented eIDAS certificate (QWAC)
    - Checks the direct or indirect matching between the Authorization Number within the eIDAS certificate and the [client_id] value.
- The ASPSP answers through a HTTP200 (OK) response that embeds the following data.

NAME		DATA	TYPE AND CONSTRAINS
access_token	[1..1]	Access token provided by the ASPSP to the TPP.	String[140]
token_type	[1..1]	Type of the provided access token ("Bearer" or "MAC")	String[10] Must be valued with "Bearer"
expires_in	[0..1]	Token lifetime, in seconds. The token can be used several times as far as it is not expired.	Numeric
refresh_token	[0..1]	Refresh token that can be replace the previous refresh token.	String[140]

If the refresh token has been revoked, the request will be rejected with HTTP400 and an error equal to "invalid grant".

3.4.3.5. Refresh Token Revocation

The refresh token provided to an AISP is de facto revoked by the ASPSP

- After timeout of the by-law specified delay between two SCAs.
- After timeout of the ASPSP specified delay based on internal rules if any.
- After reject of a request for insufficient scope in order to allow the AISP to request another token with the desired scope.
- On request of a PSU wanting to revoke the TPP access on his/her account data.

The TPP is also able to ask for the revocation of the refresh token, according to RFC 7009 (cf. https://tools.ietf.org/html/rfc7009) through the following request.

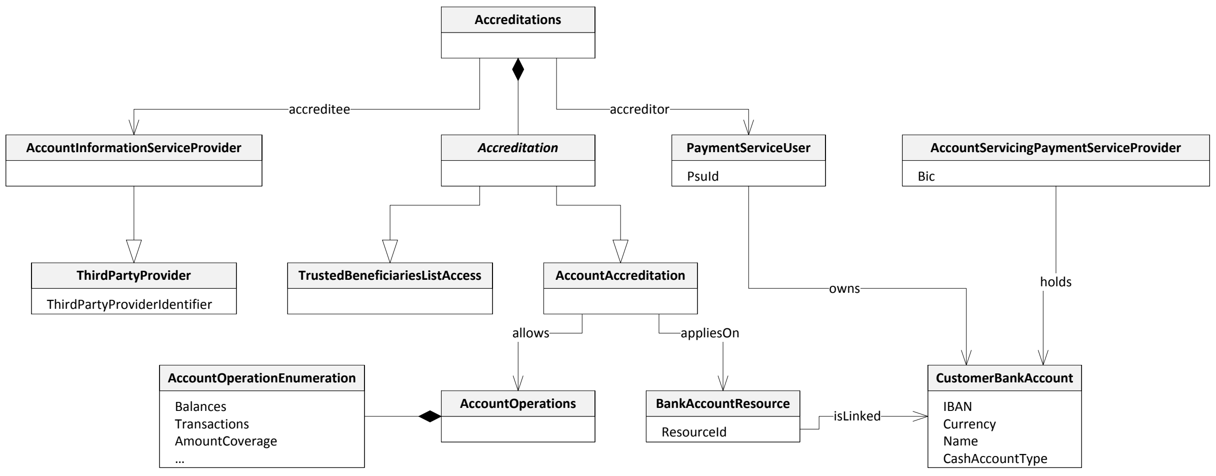
```
POST /revoke HTTP/1.1
    Host: server.example.com

     token=45ghiukldjahdnhzdauz
     &token_type_hint=refresh_token
     &client_id={clientId}
```

NAME		DATA	TYPE AND CONSTRAINS
token	[1..1]	Token to be revoked by the ASPSP.	String[140]
token_type_hint	[0..1]	Information about the type of token to be revoked	Must be valued with "refresh_token"
client_id	[1..1]	TPP identification	String[34] must be equal or linked to the OrganizationIdentifier part of the Distinguished Name of the eIDAS certificate, according to ETSI specification

- The ASPSP
  - Identifies and authenticates the TPP through the presented eIDAS certificate (QWAC)
  - Checks the direct or indirect matching of the [client_id] value with the Authorisation Number that is located within the TPP's eIDAS certificate (QWAC).
  - Revokes the refresh token

3.4.3.6. PSU context model

The PSU context can be seen as a collection of individual accreditations.



This collection is specific to a given PSU, a given TPP and a given ASPSP.

Each single accreditation relies on a specific account that is owned by the PSU and is held by the ASPSP. It specifies which pieces of data (transactions, balances) the TPP is allowed to carry out on this account.

The PSU manages this context with the AISP which is responsible of:

- The capture of the PSU choices:

    - The PSU specifies to the AISP which account and feature should be accessed or not.
- The execution of the PSU choices:
    - The AISP has the responsibility to respect the PSU choices and not to access any feature that it has not been granted for.

At any time, the PSU can edit his/her consent choices but this can only be done with the AISP.

Furthermore, the PSU consent may or may not be forwarded by the AISP to the ASPSP, according to one of the two following consent management models.

Full-AISP model (A1)

In this model, the ASPSP does not require to be informed of the details of the PSU consent.

Whatever the AISP request, the ASPSP will respond, being unable to check the compliance of the request against the PSU choices.

Actually, when getting the PSU context from the ASPSP (through the call [get /accounts]), the AISP will get all relevant HAL links for each eligible account. These HAL links will help the AISP to request the needed features on those accounts: balances and/or transactions.

Mixed model (A2)

In this model, the ASPSP does require to be informed of the details of the PSU consent. Therefore the ASPSP has implemented an ad-hoc API entry-point that can be called by the AISP in order to forward the PSU choices.

It is the charge of the ASPSP to implement or not the mixed model (A2). However, if this model has been implemented by the ASPSP, it is the charge of the AISP to forward the details of the PSU consent to the ASPSP whenever the PSU gives or edits this consent.

Once the details of the PSU consent has been received and saved by the ASPSP, the AISP, when getting the PSU context from the ASPSP (through the call [get /accounts]), will only get HAL links for authorized accounts and features.

3.4.4. PISP authorization levels

3.4.4.1. General rules

For Payment Request on behalf of a Merchant and Transfer Request on behalf of an Ordering Party, the law requires a SCA, unless exemption cases. This SCA will embed the PSU's consent to the subsequent Credit Transfer.

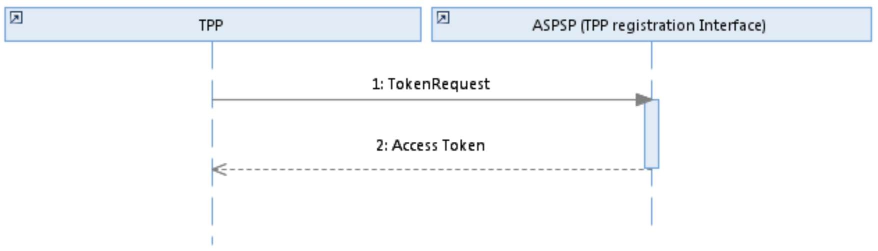
That for, the PSD2 use cases that are linked with the PISP role only require an "Authorization by Role" authorization level for accessing the ASPSP API services.

However, it must be noticed that a PAO may ask to be placed under an OPT-OUT statement by its ASPSPs, avoiding any incoming Payment Request to be processed on its accounts.

3.4.4.2. Registration of the TPP access

The registration of the TPP by the ASPSP relies on an OAUTH2 sequence for obtaining a Client Credential (cf. https://tools.ietf.org/html/rfc6749#section-4.4).

This procedure can be summarized through the following steps.



- The TPP sends directly, through a POST request, its access token request to the ASPSP authorization infrastructure with the following URL pattern and parameters

```
POST /token
        Host: authorization-server.com
        grant_type=client_credentials
        &scope={scope}
        &client_id={clientId}
```

NAME		DATA	TYPE AND CONSTRAINS
grant_type	[1..1]	Requested authorization type	String[34] Must be valued with "client_credentials"

scope	[0..1]	Specifies the generic accreditations that both the PSU and the TPP agreed on: PISP.	String[140] Space delimited roles list. Default value is "pisp"
client_id	[1..1]	TPP identification	String[34] must be equal or linked to the OrganizationIdentifier part of the Distinguished Name of the eIDAS certificate, according to ETSI specification

- The ASPSP
  - Identifies and authenticates the TPP through the presented eIDAS certificate (QWAC)
  - Checks the matching, direct or indirect, between the Authorization Number within the eIDAS certificate and the [client_id] value.
  - Computes the relevant TPP checks (roles, validity, non-revocation…)
- The ASPSP answers through a HTTP200 (OK) response that embeds the following data.

NAME		DATA	TYPE AND CONSTRAINS
access_token	[1..1]	Access token provided by the ASPSP to the TPP.	String[140]
token_type	[1..1]	Type of the provided access token ("Bearer" or "MAC")	String[10] Must be valued with "Bearer"
expires_in	[0..1]	Token lifetime, in seconds. The token can be used several times as far as it is not expired.	Numeric

3.4.4.3. Use of the access token

The access token must be used within each request within the "Authorization" header, prefixed by the token type "Bearer".

The [client_id] that is linked to the access token must directly or indirectly match with the Authorisation Number that is located within the TPP's eIDAS certificate (QWAC).

If the access token is expired, the request will be rejected with HTTP401 with an error equal to "invalid_token" and the request can be replayed once a new client credentials token has been requested and provided.

3.4.5. Pre-enrolled CBPII authorization level

When the PSU has previously enrolled the CBPII to his/her relevant ASPSP, the latest may prefer to apply a simpler authorization scheme.

Instead of using a REDIRECT approach by providing an "'Authorization Code" OAUTH2 token, or an EMBEDDED approach by providing a "Resource Owner Password" OAUTH2 token, the ASPSP can actually prefer to give access through a "Client Credential" OAUTH2 token, aiming that PSU authentication is useless since the PSU consent was previously captured.

3.5. Applicative authentication

Each request sent by the TPP has to be signed using http-signature mechanism which is specified by the following IETF draft-paper:

- https://datatracker.ietf.org/doc/draft-cavage-http-signatures/

ASPSP might also apply http-signature to their responses.

The way it should be implemented is the following

- Computing a SHA256 digest of the HTTP body and adding this digest as an extra HTTP header.
- Using a specific Qualified Certificate (QSealC), respecting the ETSI/TS119495 Technical Specification, in order to apply a RSA-SHA256 signature on
    - all the following headers that are present within the HTTP request sent by the TPP, including the previously computed digest
        - Date (if available)
        - Content-Type
        - Content-Length (when there is a payload)
        - X-Request-Id
        - All available "PSU"-prefixed Headers (cf. § 3.6)

    - all the following headers that are present within the HTTP response given by the ASPSP, including the previously computed digest
        - Date (if available)
        - Content-Type
        - Content-Length (when there is a payload)
        - X-Request-Id

    - on the specific "(request-target)" field which is specified by the IETF draft-paper

- Adding this signature within an extra HTTP header embedding
    - The key identifier which must specify the way to get the relevant qualified certificate. It is requested that this identifier is an URL aiming to provide the relevant Qualified Certificate.
    - The algorithm that has been used
    - The list of headers that have been signed
    - The signature itself.

If the ASPSP notes that the signature is either absent or invalid, it shall reject the request with HTTP400.

EXTRA HTTP HEADER	DATA	COMMENT
Digest	Digest of the body	
Signature	http-signature of the request (cf. https://datatracker.ietf.org/doc/draft-cavage-http-signatures/)	The keyId must specify the way to get the relevant qualified certificate. It is requested that this identifier is an URL aiming to provide the relevant Qualified Certificate.

In order to assure an easy discrimination of the certificate among others, it is requested that the last part of the URL to the certificate be suffixed by an underscore followed by the fingerprint of the certificate. E.g.:

https://path.to/myQsealCertificate_612b4c7d103074b29e4c1ece1ef40bc575c0a87e

3.6. Fraud detection oriented information

The following extra HTTP-headers must be used within the HTTP request sent by the TPP, provided the relevant pieces of data are available within the connection between the PSU and the TPP. This forwarding allows the ASPSP to integrate this information into its own fraud detection process.

Moreover these headers can be considered as proof of the PSU being connected.

EXTRA HTTP HEADER	DATA	COMMENT
PSU-IP-Address	IP Address of the PSU terminal when connecting to the TPP	In regards with GDPR rules, this must be subject to PSU's consent
PSU-IP-Port	IP Port of the PSU terminal when connecting to the TPP	
PSU-HTTP-Method	HTTP Method used for the most relevant PSU's terminal request to the TTP	
PSU-Date	Timestamp of the most relevant PSU's terminal request to the TTP	
PSU-User-Agent	"User-Agent" header field sent by the PSU terminal when connecting to the TPP	
PSU-Referer	"Referer" header field sent by the PSU terminal when connecting to the TPP	
PSU-Accept	"Accept" header field sent by the PSU terminal when connecting to the TPP	
PSU-Accept-Charset	"Accept-Charset" header field sent by the PSU terminal when connecting to the TPP	
PSU-Accept-Encoding	"Accept-Encoding" header field sent by the PSU terminal when connecting to the TPP	
PSU-Accept-Language	"Accept-Language" header field sent by the PSU terminal when connecting to the TPP	
PSU-GEO-Location	The forwarded Geo Location of the corresponding HTTP request between PSU and TPP if available.	In regards with GDPR rules, this must be subject to PSU's consent
PSU-Device-ID	UUID (Universally Unique Identifier) for a device, which is used by the PSU, if available. UUID identifies either a device or a device dependant application installation. In case of installation identification this ID need to be unaltered until removal from device.	In regards with GDPR rules, this must be subject to PSU's consent

3.7. Other specific HTTP headers to be used

EXTRA HTTP HEADER	DATA	COMMENT
X-Request-ID	Correlation header to be set in a request and retrieved in the relevant response.	

3.8. Specific HTTP return codes and messages to be used

MESSAGE	HTTP CODE	SIGNIFIANCE
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.
RESOURCE_UNKNOWN	404	If resourceId in path
PERIOD_INVALID	400	Requested time period out of bound.
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.

3.9. STET PSD2 API technical summary

TOPIC	CHOICE	COMMENT
Access network	Internet	
Network protocol	HTTP 1.1 (Minimum)	
Data encryption **Cross-authentication**	TLS 1.2	Could be enforced through STS and/or PFS
Authorization protocol	OAUTH2	In respect of RFC 6749, 7009 One of the following token modes - Authorization Code Grant (AISP, CBPII) - Resource Owner Password (AISP, CBPII) - Client credential (PISP, CBPII) Based on MTLS, the identity of the TPP is provided by its eIDAS certificate during OAUTH2 procedures. https://datatracker.ietf.org/doc/draft-ietf-oauth-mtls/
Applicative protocol	REST	In respect of the Richardson Maturity Model, on level three in order to provide HYPERMEDIA links.
Applicative authentication	http-signature	Notice this is actually an IETF draft, waiting for approval and so subject to some modifications. https://datatracker.ietf.org/doc/draft-cavage-http-signatures/
PSU Strong Customer Authentication approaches	REDIRECT, DECOUPLED or EMBEDDED	
Data format	JSON/UTF8	With use of ISO20022 based data structures
Technical documentation	SWAGGER 2.0	