



# Integration kit for CVCo by API

Technical documentation

<b>1</b>	<b>Introduction.....</b>	<b>3</b>
1.1	Version's history.....	3
1.2	Applicable Documents .....	4
1.3	Context.....	4
1.4	Purpose of the document .....	5
1.5	Kit Recipients .....	5
<b>2</b>	<b>CVCO Acceptance Modalities per channel.....</b>	<b>6</b>
2.1	For Online Acceptance .....	6
2.2	For Face-to-face Acceptance .....	6
2.3	For MOTO Acceptance.....	6
2.4	Reservation or Ticketing Kiosks .....	6
<b>3</b>	<b>Payment processing .....</b>	<b>8</b>
3.1	Immediate Payment acceptance .....	8
3.2	Request for Payment.....	10
3.2.1	Merchant process to obtain the request for payment .....	10
3.2.2	Customer process to accept the request for payment .....	11
<b>4</b>	<b>Open API messages.....</b>	<b>13</b>
4.1	Input & Output encoding types .....	13
4.2	Environments.....	13
4.2.1	Testing .....	13
4.2.2	Live .....	13
4.3	API operations .....	13
4.3.1	Request payment preamble .....	13
4.3.2	API for point of sale configuration .....	14
4.3.3	API for immediate acceptance transactions .....	14
4.3.4	API for delayed acceptance pre-transactions .....	14
4.4	Point of sale configuration .....	15
4.4.1	Retrieve info from point of sale .....	15
4.5	Immediate acceptance transactions .....	17
4.5.1	Create Transaction.....	17
4.5.2	Request Payment on Transaction by a payer .....	20
4.5.3	Retrieve Transaction.....	23
4.5.4	Execute Deferred Payment.....	26
4.5.5	Cancel Transaction.....	28
4.6	Delayed beneficiary acceptance pre-transactions .....	32
4.6.1	Create pre-transaction .....	32
4.6.2	Request Merchant Presented QR-Code .....	35
4.6.3	Retrieve pre-transaction .....	37
4.6.4	Send pre-transaction to contact .....	39
4.6.5	Abort pre-transaction.....	41
<b>5</b>	<b>Specific services .....</b>	<b>44</b>
5.1	Webhooks .....	44
5.2	Transactions daily reports .....	44
5.2.1	Daily log of Operations.....	46
5.2.2	Bank Repayments Journal.....	47
5.2.3	Specific Bank Repayment Journals and Daily Logs of Operations.....	48
5.2.4	SFTP Space.....	49
5.3	Security for payment processes.....	50
5.3.1	Rules.....	50
5.3.2	Sealing Algorithm .....	51
5.4	Reading Consumer ANCV ID from presented Bar Codes .....	51
5.4.1	Bar Code .....	52
5.4.2	QR Code .....	52
<b>6</b>	<b>Documentation appendix .....</b>	<b>53</b>
6.1	Glossary .....	53
6.2	Data definition .....	53
6.2.1	transaction .....	53

6.2.2	payer.....	61
6.2.3	pre-transaction.....	61
6.2.4	applicationContext.....	66
6.3	Reject codes and root cause .....	66
6.4	Graphic charter payment pages.....	68
6.4.1	Case #1 - Several formats are accepted.....	69
6.4.2	Case #2 - Only the Chèque-Vacances Connect is accepted.....	70
6.4.3	Next step is common to both cases - access to the payment page .....	70
6.4.4	Specific behavior for some reject causes .....	71
6.5	Route and graphic charter requested for the implementation of the Connect Holiday Vouchers on terminals and payment terminals .....	72
6.5.1	Integration of the payment QR Code in the receipt.....	72
6.5.2	Integration of the payment QR Code on an interactive kiosk.....	73
6.5.3	Integration of the payment QR Code on a payment terminal.....	75
6.6	SlipId number .....	77
6.7	Seal calculation .....	77
6.8	Timeline.....	78
6.8.1	Immediate payment without QR Code .....	78
6.8.2	Impact of transaction cancellation by a Partner.....	79
6.8.3	Impact of transaction abortion by a beneficiary .....	80
6.8.4	From a Pre-Transaction with QR Code .....	81
6.8.5	Impact of a pre-transaction cancellation by a Partner .....	82
6.9	Acceptance test plan .....	82
6.10	Application installation .....	82

# 1 Introduction

## 1.1 Version's history

Version	Date	Content	Available in staging	Available in production
0.1 to 0.13	12/12/2018	Internal work versions by ANCV	N/A	N/A
0.14	18/01/2019	First publication	N/A	N/A
0.15	17/04/2019	Description of security method and data	N/A	N/A
0.18	14/05/2019	Add new API methods for Merchant presented payment QR-Code	N/A	N/A
0.19	27/08/2020	<ul style="list-style-type: none"><li>• Specifications of Accept header for pre-transaction QR-Code retrieval</li><li>• DELAYED and NO_SLIP_FOUND transaction status added to the documentation</li></ul>	N/A	N/A
0.19_2	16/11/2020	<ul style="list-style-type: none"><li>• OrderId length augmentation (20 to 64 characters)</li><li>• More detailed documentation on acceptance API</li><li>• More detailed documentation on reject causes and associated codes</li><li>• Fix cancel transaction HTTP returned code</li></ul>	11/26/2020	12/19/2020
0.19_3	16/11/2020	<ul style="list-style-type: none"><li>• SlipId added to Bank Repayment Journals</li><li>• Specific Bank Repayment Journals and Daily Logs of Operations specified</li></ul>	12/22/2020	05/03/2021
0.19.4	14/04/2021	<ul style="list-style-type: none"><li>• Pre-transaction documentation enrichment</li><li>• Add a preamble to payment APIs</li><li>• Add slipId description</li><li>• More detailed documentation about ancv-security header calculation</li><li>• Add examples about DLO and BRJ</li><li>• Add multiples timelines</li></ul>	12/22/2020	05/03/2021
0.20	14/04/2021	Store status management	04/22/2020	05/03/2021
0.21	09/11/2021	<ul style="list-style-type: none"><li>• Redirect URLs become optional</li><li>• Platform URL changed</li></ul>	09/01/2021	10/22/2022
1.0	27/12/2021	<ul style="list-style-type: none"><li>• Adding information on SFTP spaces</li><li>• Refactoring document layout</li></ul>	N/A	N/A
1.01	2 févr. 2023	Documentation enhancement: <ul style="list-style-type: none"><li>• Undocumented existing transaction status added to the document</li><li>• Preconsitions on the expirationDate filed usage when</li></ul>	N/A	N/A



		associated with a QR Code presentation directly to a customer		
1.02	3 nov. 2023	Documentation enhancement: <ul style="list-style-type: none"> <li>Complementary information about balance calculation when <b>INSUFFICIENT_BALANCE</b> error has been encountered</li> <li>Recommended usage of Retrieve Transaction API added</li> <li>Minor corrections on deferred capture mode description</li> </ul>	N/A	N/A
1.03	15 avr. 2024	Documentation enhancement: <ul style="list-style-type: none"> <li>Deletion of obsolete content (only additional information or unimplemented functionalities)</li> <li>Add of information on cancellation transaction / pre-transaction</li> <li>Correction of misleading examples or descriptions</li> <li>Correction of captureTerm possible values</li> </ul>	N/A	N/A
1.04	30 avr. 2024	Control API added for IS : retrieval of point of sale info	07/10/2024	08/07/2024
1.05	10 juil. 2024	Documentary enhancement ; Recovery procedures in case of technical error for: <ul style="list-style-type: none"> <li>The <b>payment request</b></li> <li>The cancellation of a transaction</li> </ul>	N/A	N/A
1.06	16 déc. 2024	Updating minimum of payment to 1ct	16 déc. 2024	16 déc. 2024

## 1.2 Applicable Documents

Number	Date	Document Ref.	Type
<b>DA1</b>	06/21/2019	Solutions for Acceptance of CVCo Service Providers	Functional
<b>DA2</b>	06/21/2019	Solutions for Acceptance of CVCO Leisure and Travel Professionals	Functional
<b>DR3</b>	08/11/2021	App Mobile - EnvRecette_Procédure Mobile_Fr	Installation guide for Chèque-Vacances Application (French)

## 1.3 Context

This document is part of the “Chèque-Vacances Connect - an Online Holiday Voucher” Acceptance Solution integration package.

We assume that at least one of the Applicable Documents DA1 and DA2 is well known in order to apprehend the CVCo Acquiring Platform from a functional point of view and understand how to use this document for an appropriate integration of the open API as described hereafter.

## 1.4 Purpose of the document

This document has been created to be used as a **technical description** of the **implementation** to be done in order to integrate the **online holiday voucher Acceptance** (CVCo) into the existing environments.

This document details the technical modalities to respect for interacting with the open API for CVCo Acceptance on any channel:

- Online
- MOTO
- Face to face
- Interactive Kiosks

These modalities are immediately usable in order to perform the implementation of the open API in any system, according to the provided description of:

- The security method (Algorithm and Key management)
- The API messages
- The data exchanged
- The flow format

## 1.5 Kit Recipients

The recipients for this document are any **Integrator** or **Service Provider** (Payment Service Provider, Marketplace, Software or Hardware editors...) providing Leisure and Travel Professionals (Merchants) with interfaces and technical connection to the platform for acquiring payment transactions in online holiday voucher (CVCO).

The proposed acceptance channels are, mainly:

- Online (Sales in eCommerce),
- MOTO (Mail Order / Telephone Order),
- Face to face equipped with cash desk,
- Reservation or Ticketing Kiosks.

## 2 CVC0 Acceptance Modalities per channel

① Les pages de cet espace sont soumises à une workflow de validation. Pour consulter la version validée de cette page par ANCV, consulter le lien *afficher "Validé"* ci-dessus si l'état de la page n'est pas déjà **"Validé"**.

Pour plus d'information, consulter la page [Comprendre le workflow de validation](#)

### En rédaction

The chapter is a summary of the Modalities described in DA1 and DA2 applied to the specific use of the CVC0 open API only.

### 2.1 For Online Acceptance

The Merchant's eCommerce shop must use a specific PayPage (provided by its integrator, hosting site or PSP) using the Open API messages described in chapter "[Open API Messages](#)" in order to send transaction details to the CVC0 Acquiring Platform

### 2.2 For Face-to-face Acceptance

The face-to-face Acceptance device (Cashier, Reservation, and Ticketing Kiosks) must use at least one of the two available processes:

- First process is by collecting some Beneficiary Id, which will be used to transfer the Payment directly to her/his smartphone. The ID can be collected by several means :
  - Using some keyboard (only digits or full ones with special characters for email address compliance)
  - Reading the code directly from the Beneficiary's smartphone in one of two displayed formats :
    - Bar Code (Code 128)
    - QR-Code
  - Second process is the Merchant Presented Payment QR-Code one, which will be scanned by the Beneficiary to finalize the payment.

Then the device will transfer the payment request to the CVC0 Acquiring Platform using the messages of the open API according to the selected process. The detail of the use of the API is presented in chapter "[Open API Messages](#)"

The Merchant presented QR-Code will be generated server-side and provided as a picture using PNG format (size 300x300px).

### 2.3 For MOTO Acceptance

The reservation software used by the MOTO Agents must allow the input of either the Beneficiary account identifier (for immediate payment acceptance) or the Consumer email (for delayed authorization) and for both cases the amount in CVC0 to pay.

The actual connection of the reservation software to the CVC0 Acquiring Platform for payment in CVC0 is carried out by the integration of an open API as described in chapter "[Open API Messages](#)"

### 2.4 Reservation or Ticketing Kiosks

The device chosen to operate the transaction must use at least one of the two available processes:

- First process is by collecting some Beneficiary Id, which will be used to transfer the Payment directly to her/his smartphone. The ID can be collected by several means :
  - Using some keyboard (only digits or full ones with special characters for email address compliance)
  - Reading the code directly from the Beneficiary's smartphone in one of two displayed formats :
    - Bar Code (Code 128)
    - QR-Code
  - Second process is the Merchant Presented Payment QR-Code one, which will be scanned by the Beneficiary to finalize the payment.

The Interactive Kiosk is connected to the CVCO Acquiring Platform by integrating an open API according to the selected process as described in chapter "[Open API Messages](#)"

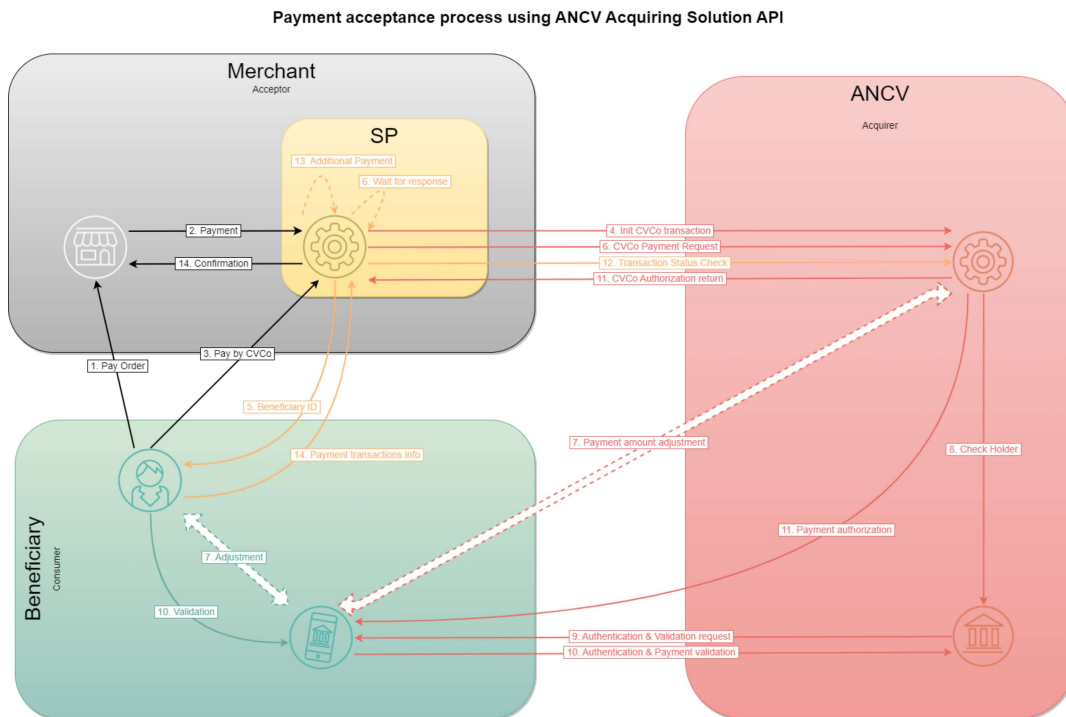
The Merchant presented QR-Code will be generated server-side and provided as a picture using PNG format (size 300x300px).

### 3 Payment processing

The payment process is done using Machine-to-Machine exchanges with Web Service (REST/JSON). The **SP** must implement the Web Service required for the exchanges with the CVCo Acquiring Platform as described in chapter 6. The Web Service is available on the internet with a secured connection on HTTPS using an authentication certificate delivered by an external authority.

According to its needs and the constraints from the channels, the **SP** can adapt some stages.

#### 3.1 Immediate Payment acceptance



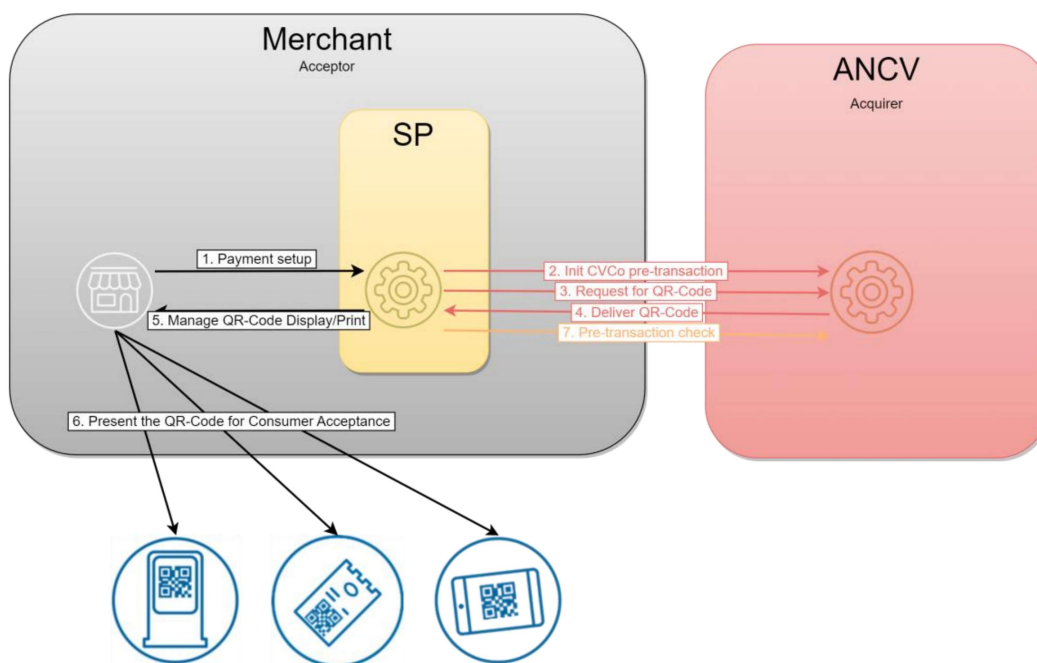
Step	Actor	Description
1	Consumer	The Consumer places an order to a Merchant and proceeds to the payment.
2	Merchant	Makes use of the service provided by its <b>Service Provider (SP)</b> to accept the payment from the Consumer
3	Consumer	Choose to pay all or part of the order using CVCo ( <i>merchant or SP gives access to an interface where the user can make his choice</i> ) NB : until this step, no communication with ANCV platform is done, the goal of these steps is only to redirect beneficiary to the CVCo).
4	SP	Connects to the CVCo Acquiring Platform and initialize a CVCo Payment transaction with appropriate context (Merchant, Order amount, Order ID ...) : <a href="#">/payment-transactions</a>
5	Consumer	Communicates its Beneficiary identifier to the <b>SP</b> (either by self-input or by the Merchant's representative or presenting her/his ANCV ID encoded in Bar Code and QR-Code).
6	SP	Transfers the Payment request towards the ANCV Acquiring Solution for the ongoing transaction using the Beneficiary account identifier and the CVCo amount : <a href="#">/payment-transactions/{id}/payer</a>

<b>6-bis</b>	SP	<i>The <b>SP</b> puts on hold the transaction process and waits for the ANCV server to push the transaction status (step 11-bis) and can check the transaction status while waiting using <a href="#">/payment-transactions/{id}</a>. Expected answer when retrieving the transaction status : « Payment processing ongoing »</i>
<b>7</b>	ANCV	<i>(If adjustment available) Asks the confirmation from the Beneficiary of the amount to pay using CVCo</i>
<b>7-bis</b>	Consumer	<i>(If adjustment available) Adjust and validate the amount to pay from her/his CVCo account</i>
<b>8</b>	ANCV	Checks the consumer's solvability.
<b>9</b>	ANCV	Requires the Beneficiary authentication process to validate the payment request
<b>10</b>	Consumer	Validates the transaction amount and performs a strong authentication of the CVCo Account's consumer by the input of her/his M-PIN on the ANCV Mobile App
<b>10-bis</b>	ANCV Mobile App	Makes exchanges with the ANCV Authorization Server in order to authenticate the consumer and validate the payment authorization
<b>11</b>	ANCV	Sends the CVCo Payment Authorization towards the Beneficiary through its Mobile App
<b>11-bis</b>	ANCV	Notify the <b>SP</b> of the transaction status with confirmation of the authorized CVCo amount using an automatic response service provided by the <b>SP</b> ( <a href="#">webhook</a> )
<b>12</b>	SP	<i>Checks the transaction status : <a href="#">/payment-transactions/{id}</a> « Transaction authorized »</i>
<b>13</b>	SP	<i>Manage any additional payment method in order to complete the Order amount (if required)</i>
<b>14</b>	SP	Confirms the payment to the Merchant
<b>14-bis</b>	SP	Confirms the payment to the Consumer with the transaction's details (including CVCo)

## 3.2 Request for Payment

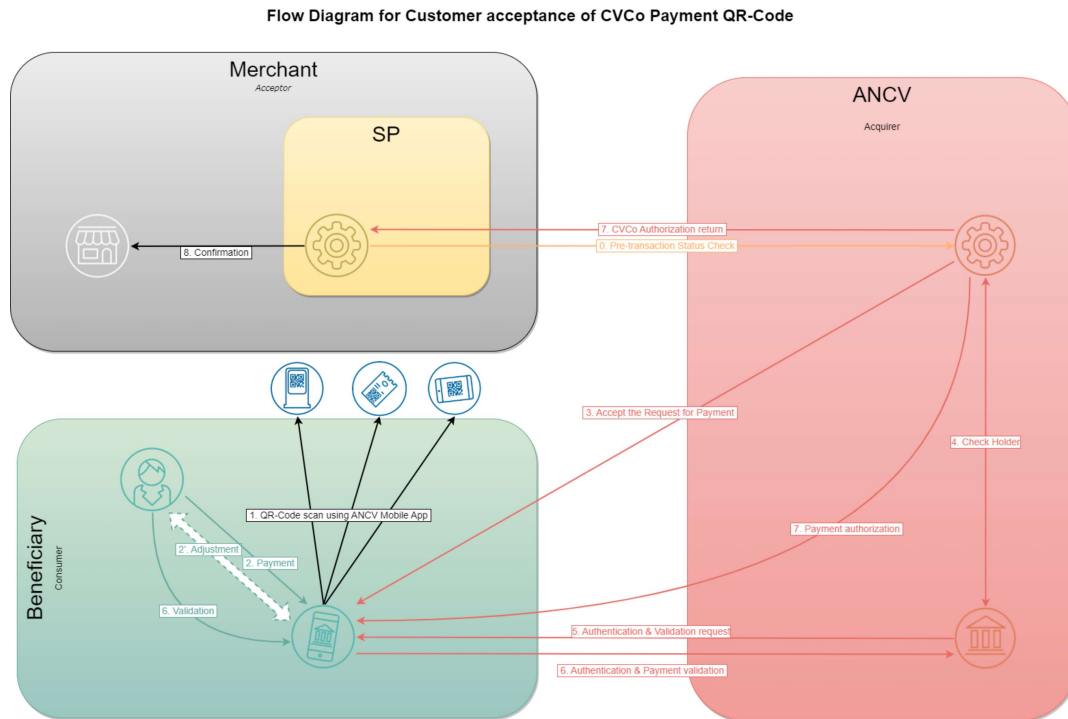
### 3.2.1 Merchant process to obtain the request for payment

Flow Diagram for Merchant Presented Payment QR-Code using ANCV Acquiring Solution API



Step	Actor	Description
1	Merchant	Uses the Payment service provided by its SP to present a request for payment on a specific order and using a CVCo Merchant Presented QR-Code
2	SP	Connects to the ANCV Acquiring Solution and initialize a CVCo Payment pre-transaction with appropriate context (Merchant, Order amount, Order ID ...)
3	SP	Requests for a Merchant Presented QR-Code on the ongoing pre-transaction
4	ANCV	Replies with the picture of the generated Payment QR-Code
5	SP	Deliver the QR-Code to the Merchant on the appropriate media (receipt, screen, ...)
6	Merchant	Presents the Payment QR-Code to the Consumer in order to obtain the payment acceptance.
7	SP	The <b>SP</b> puts on hold the pre-transaction process and waits for the ANCV server to push data on its usage, including a payment transaction if accepted. The timeout for this step will be defined between the ANCV and SP. If needed, the <b>SP</b> can check the pre-transaction status using the provided method Expected answer when retrieving the pre-transaction status : « Processing »

### 3.2.2 Customer process to accept the request for payment



Step	Actor	Description
0	SP	The <b>SP</b> puts on hold the pre-transaction process and waits for the ANCV server to push data on its usage, including a payment transaction if accepted. The timeout for this step will be defined between the ANCV and <b>SP</b> . If needed, the <b>SP</b> can check the pre-transaction status using the provided method Expected answer when retrieving the pre-transaction status : « Processing »
1	Consumer	Uses her/his Smartphone (Android or iOS) with the ANCV App to scan the Merchant Presented QR-Code and access the request for payment
2	Consumer	Check the request for payment and validate the will to accept it
2-bis	Consumer	(If adjustment available) Adjust and validate the amount to pay from her/his CVCo account
3	ANCV Mobile App	Makes exchanges with the ANCV server in order to accept the request for payment by the connected Beneficiary on the Mobile App
4	ANCV	Checks the consumer's solvability.
5	ANCV	Requires the Beneficiary authentication process to validate the payment request
6	Consumer	Validates the transaction amount and performs a strong authentication of the CVCo Account's consumer by the input of its M-PIN on the ANCV Mobile App
6-bis	ANCV Mobile App	Makes exchanges with the ANCV Authorization Server in order to authenticate the consumer and validate the payment authorization
7	ANCV	Confirms the CVCo Payment Authorization towards the Beneficiary through its Mobile App



		NB : at this step, the payment transaction is created from the pre-transaction information. Authorized amount appears in the transaction (cf. <a href="#">Retrieve transaction</a> )
<b>7-bis</b>	ANCV	Notify the <b>SP</b> of the transaction status with confirmation of the authorized CVCo amount using an automatic response service provided by the <b>SP</b>
<b>8</b>	SP	Confirms the payment to the Merchant

## 4 Open API messages

N.B.: Details for all the Objects and fields used in the following operations are provided in the Appendix on chapter [Data definition](#)

### 4.1 Input & Output encoding types

The accepted format for the open API Input and Output messages are:

Format	MIME Type
JSON	application/json

The encoding type must be:

Encoding
UTF-8

### 4.2 Environments

Two distinct environments are available for accessing the CVCo Acquiring Platform. One is to be used for testing during the implementation process of the open API; the other is to be used for live connection using real payment transactions

Each environment is accessed using a dedicated address. This access will be related to under the code name **{base\_url}** from now on.

#### 4.2.1 Testing

This environment is to be used by and Service Provider when testing its technical implementation towards the CVCo Acquiring Platform by use of the open API.

{testing\_url}: <https://recette.connect.ancv.com/acquisition/api/public/V1>

#### 4.2.2 Live

This is the effective live environment on which any transaction commits for an effective payment.

Only proper transactions between a Customer and a Professional can be processed on this environment, as no refund can be done on CVCo payments.

This environment **must not be used** for implementation testing.

{live\_url}: <https://connect.ancv.com/acquisition/api/public/v1>

### 4.3 API operations

#### 4.3.1 Request payment preamble

In order to succeed to use a payment API, some conditions must be complied:

- Request header must contain at least:
  - Content-Type : application/json
  - Ancv-Security : [{algo.version.seal}](#)

A sealing calculation help can be found in appendix [seal calculation](#).

- Request body must be like the payment API example, specified in **Post Data**

- API URL can contain some parameters specified in **Request Data**
- Every call must be executed with **HTTPS protocol**
- Calls to return URLs, done by CVCo platform to integrators domains, must be executed through, at least, TLS 1.2 protocol

An approved server authentication certificate, by an external authority, is required on the return URLs target domain.

### 4.3.2 API for point of sale configuration

The CVCo API for accessing the configuration of point of sale on any environment using its name: point-of-sale

The operations that can be requested on this API are:

Opération	Méthode	Description	URL
<i>Retrieve info from point of sale</i>	GET	Retrieves point of sale information from CVCo platform	{base_url}/point-of-sale/{id}

### 4.3.3 API for immediate acceptance transactions

The CVCo payment API for immediate acceptance is accessible on any environment using its name: payment-transactions

The operations that can be requested on this API are:

Operation	Method	Description	URL
<i>Create transaction</i>	POST	Initialize a payment transaction with order details	{base_url}/ <b>payment-transactions</b>
<i>Request Payment</i>	POST	Request for the payment of a transaction by a payer (CVCo Holder)	{base_url}/payment-transactions/{id}/ <b>payer</b>
<i>Retrieve transaction</i>	GET	Get a CVCo Payment Transaction details, including its payment status	{base_url}/payment-transactions/{id}
<i>Execute Deferred Payment</i>	POST	Execute the payment for an authorized deferred transaction	{base_url}/payment-transactions/{id}/ <b>execute</b>
<i>Cancel Transaction</i>	POST	To be used for the cancellation of a payment transaction	{base_url}/payment-transactions/{id}/ <b>cancellation</b>

Access to any of these operations is secured and includes authorization check for the Service Provider and Merchant performing (cf. [Security for Payment process](#))

### 4.3.4 API for delayed acceptance pre-transactions

The CVCo payment API using delayed acceptance by Consumer is accessible on any environment using its name: pre-transactions

The operations that can be requested on this API are:

Operation	Method	Description	URL
<i>Create pre-transaction</i>	POST	Initialize a request for payment pre-transaction with order details	{base_url}/ <b>pre-transactions</b>
<i>Request Payment QR-Code</i>	GET	Requests for a Merchant Presented QR-Code on the ongoing pre-transaction	{base_url}/pre-transactions/{id}/ <b>qr-code</b>

<i>Retrieve pre-transaction</i>	GET	Get a CVCo request for payment pre-transaction details, including its payment status	{base_url}/pre-transactions/{id}
<i>Send to contact</i>	POST	Requests for sending by email the ongoing pre-transaction towards a contact	{base_url}/pre-transactions/{id}/contact
<i>Abort pre-transaction</i>	POST	To be used for the abortion of an unused request for payment pre-transaction	{base_url}/pre-transactions/{id}/abort

Access to any of these operations is secured and includes authorization check for the Service Provider and Merchant performing (cf. [Security for Payment process](#))

## 4.4 Point of sale configuration

### 4.4.1 Retrieve info from point of sale

An optional step performed prior to the initial setup.

This call allows the Service Intermediary or integrator to verify information about the Point of Sale for which they are configuring their services (e.g. configuring a POS terminal).

This call is made using a dedicated WebService (REST/JSON) for retrieving information about the point of sale.

Name	Type	Visibility	Version API
<b>/point-of-sales/{shopId}?serviceProviderId={serviceProviderId}</b>	GET	Public	V1

#### 4.4.1.1 Precondition

- The Point of Sale must exist : if the provided shopID does not correspond to a known point of sale, the request will be denied and an HTTP error with status code 404 will be returned with the error code POINT\_OF\_SALE\_NOT\_FOUND (see. [Security for Payment process](#))
- The Point of Sale must be active : if it's inactive, the request will be denied and an HTTP error with status code 403 will be returned with the error code MERCHANT\_NOT\_ALLOWED (see. [Security for Payment process](#))

#### 4.4.1.2 Request Data (path)

The Service Intermediary must provide the identifier (shopId) of the Point of Sale as a parameter in the request.

Field	Type	Condition
shopId	Long	It must correspond to the identifier of the Point of Sale.

#### 4.4.1.3 Query Parameters

Field	Type	Condition
serviceProviderId	Long	Not specified for a Point of Sale Mandatory for a Service Intermediary

#### 4.4.1.4 Post Data

N/A

#### 4.4.1.5 Ordered list of Fields for sealing

Ordre	Champ
1	shopId
2	serviceProviderId <i>(only if transmitted)</i>

#### 4.4.1.6 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send a created (201) HTTP code. Details on the created object are provided in the response body.

Field	Type	Condition
pointOfSale	Objet	Mandatory
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of failure to create the transaction, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Champ	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject caus

#### 4.4.1.7 Call example for a point of sale

```
curl -v -X GET {base_url}/point-of-sales/13235554 \  
-H "Content-Type: application/json" \  
-H "ANCV-Security: {algo.version.seal}"
```

#### 4.4.1.8 Call example for a Service Intermediary

```
curl -v -X GET {base_url}/point-of-sales/13235554?serviceProviderId=123456 \  
-H "Content-Type: application/json" \  
-H "ANCV-Security: {algo.version.seal}"
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 200 OK
- Body:

```

{
  "pointOfSale": {
    "name": "Krusty Burger",
    "ptlBusinessId": 157387,
    "acquirerBusinessId": 001,
    "businessId": 001,
    "shopId": 13235554,
    "status": "ACTIVE"
  },
  "responseDate": "2018-08-28T12:22:03Z"
}

```

## 4.5 Immediate acceptance transactions

### 4.5.1 Create Transaction

The first step of any Payment with immediate acceptance using the CVC Co Acquiring Platform is to use a dedicated Web Service method (REST/JSON) for a CVC Co transaction initialization with some data on the Merchant and the Order.

Name	Type	Visibility	Version API
/payment-transactions	POST	Public	V1

#### 4.5.1.1 Precondition

The store must be active.

If it's inactive, request will be refused and a 403 HTTP error with MERCHANT\_NOT\_ALLOWED error code will be returned

#### 4.5.1.2 Request Data

N/A

#### 4.5.1.3 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
merchant	Object	Mandatory The merchant's identity in ANCV CVC Co repository
order	Object	Mandatory The order to be paid Minimum transaction amount in CVC Co is <b>1ct</b> . The data set (shopId, id, paymentId) must be unique for the day
paymentMethod	Object	Mandatory When using deferred payment mode, the maximum allowed delay is of <b>6 days</b>
redirectUrls	Object	Optional If filled, URLs are called to notify the service provider of the result of the transaction (cf. <a href="#">Webhooks</a> )
applicationContext	Object	Optional Technical data used by the connected device
requestDate	String	Optional Sending date and time of the request by the SP yyyy-MM-dd'THH:mm:ss.SSS'Z'

#### 4.5.1.4 Ordered list of Fields for sealing

Ordre	Field
1	Merchant → shopId
2	Merchant → serviceProviderId (only if provided)
3	Order → id
4	Order → paymentId
5	Order → Amount → total

#### 4.5.1.5 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send a created (201) HTTP code. Details on the created object are provided in the response body.

Field	Type	Constraint
transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

In case of request duplication on a shop for the day (same "order → id" and "order → paymentId"), the response will be an OK (200) HTTP code still with the same response body as for the previous response.

In case of failure to create the transaction, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.5.1.6 Call example

```
curl -v -X POST {base_url}/payment-transactions \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-d '{
  "merchant": {
    "shopId": 13235554,
    "serviceProviderId": 98232552
  },
  "order": {
    "id": "panier-33455",
    "paymentId": "42556",
    "amount": {
      "total": 4000,
      "currency": "978"
    }
  },
  "paymentMethod": {
    "captureMode": "NORMAL",
    "tspdMode": "001"
  },
  "redirectUrls": {
    "returnUrl": "http://ptl.com/response-url",
    "cancelUrl": "http://ptl.com/cancel-url"
  },
  "applicationContext": {
    "returnContext": "1st command from customer",
    "customerId": "customer1236555"
  },
  "requestDate": "2018-08-28T10:18:00Z"
}'
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status HTTP 201 Created
- Body:



```
{
  "transaction": {
    "id": "14fddh1256",
    "creationDate": "2018-08-28T11:18:00Z",
    "updateDate": "2018-08-28T11:18:00Z",
    "expirationDate": "2018-08-28T12:18:00Z",
    "state": "INITIALIZED",
    "merchant": {
      "shopId": 13235554,
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "paymentId": "42556",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "NORMAL",
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    }
  },
  "applicationContext": {
    "returnContext": "1st command from customer",
    "customerId": "customer1236555"
  },
  "responseDate": "2018-08-28T10:18:00Z"
}
```

RedirectUrls object is optional, if it's present it can be filled according to these examples (URLs must be replaced with correct values):

```
"redirectUrls": {
  "returnUrl": "http://ptl.com/response-url",
  "cancelUrl": "http://ptl.com/cancel-url"
}

"redirectUrls": {
  "returnUrl": "http://ptl.com/response-url"
}

"redirectUrls": {
  "cancelUrl": "http://ptl.com/cancel-url"
}

"redirectUrls": {}
```

## 4.5.2 Request Payment on Transaction by a payer

The payment request is done with the use of a dedicated Web Service method (REST/JSON) provided by the CVCo Acquiring Platform.

Name	Type	Visibility	Version API
/payment-transactions/{id}/payer	POST	Public	V1

The **SP** must provide all the data required for the CVCo Acquiring Platform to process the payment transaction in CVCO, including the Authorization request.

The main steps of this process are:

- Holder verification:
  - Beneficiary ID
- Payment Authorization
  - Beneficiary balance
  - Amount validation by the Beneficiary
  - Beneficiary authentication

If the **SP** has any doubt on the transaction status, then it is in its responsibility to check the real status with a call to the dedicated method (retrieve transaction).

Collecting the Beneficiary Id, which will be used to transfer the Payment directly to her/his smartphone, can be done by the Merchant and/or **SP** by several means:

- With some keyboard that the Consumer can use (either an only digits keyboard or full ones with special characters for email address compliance)
- Reading the code directly from the Beneficiary's smartphone in one of two displayed formats :
  - Bar Code (Code 128)
  - QR-Code

Details for this reading method are proposed on chapter [Reading Consumer ANCV ID from presented Bar Codes](#).

#### 4.5.2.1 Precondition

The store must be active.

If it's inactive, request will be refused and a 403 HTTP error with MERCHANT\_NOT\_ALLOWED error code will be returned

#### 4.5.2.2 Request Data

Field	Type	Constraint
id	String	The transaction ID ( <i>transaction</i> → <i>id</i> ) obtained in the response data of the create Transaction method

#### 4.5.2.3 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
payer	Object	Mandatory The CVCo holder that will pay all or part of the transaction with its prepaid account. If no specific amount to pay in CVCo is set, then the full order amount of the transaction is used.
requestDate	String	Optional yyyy-MM-dd'THH:mm:ss.SSS'Z'

#### 4.5.2.4 Ordered list of Fields for sealing

Ordre	Field
1	Transaction → id
2	Payer → beneficiaryId
3	Payer → Amount → total

#### 4.5.2.5 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an accepted (202) HTTP code. Details on the created object are provided in the response body.

Field	Type	Constraint
-------	------	------------

transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of request duplication (same "payer"), the response will be an OK (200) HTTP code still with the same response body. This request cannot be used to change the payer or add a new one.

In case of failure to create the transaction, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

After the request for payment is accepted, the ANCV Acquiring Solution will make use of one of the automatic response services (return or cancel) provided by the SP to notify of the transaction status change in the situations of:

- Authorized payment, which can be done for an adjusted CVCo amount
- Rejected payment (Consumer refusal, cancellation, timeout...)

### Recovery procedures in case of technical error

If a technical error is received (with an HTTP status 500 for example), the proper procedure is to call the API for [transaction status check](#) to verify whether the payment request has been made or not.

If not, it is possible that the beneficiary completes the payment process without the SP being notified (except through automated response services) that the transaction has been authorized by the beneficiary.

#### Call Example

```
curl -v -X POST {base_url}/payment-transactions/14fddh1256/payer \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-d '{
  "payer": {
    "beneficiaryId": 1536923388807,
    "amount": {
      "total": 3500,
      "currency": "978"
    }
  },
  "requestDate": "2018-09-10T18:00:00.000+02:00"
}'
```

A sealing calculation help can be found in appendix [Seal calculation](#)

#### Response:

- Status: HTTP 202 Accepted
- Body:

```

{
  "transaction": {
    "id": "14fddh1256",
    "creationDate": "2018-08-28T11:18:00Z",
    "updateDate": "2018-08-28T11:25:00Z",
    "expirationDate": "",
    "state": "PROCESSING",
    "subState": "IN_ADJUSTMENT",
    "merchant": {
      "shopId": 13235554,
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "paymentId": "42556",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "NORMAL",
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    },
    "payers": [
      {
        "beneficiaryId": 1536923388807,
        "amount": {
          "total": 3500,
          "currency": "978"
        }
      }
    ],
    "applicationContext": {
      "returnContext": "1st command from customer",
      "customerId": "customer1236555"
    },
    "responseDate": "2018-08-28T10:18:00Z"
  }
}

```

### 4.5.3 Retrieve Transaction

A CVCO transaction status check is to be done at any time when the **SP** needs to check if an existing CVCO transaction has evolved.

If the **SP** has any doubt on the transaction status; either due to some timeout, or in case of data check; then it is in its responsibility to check the real status with a call to this method.

It is achieved using a dedicated Web Service method (REST/JSON) for transaction status check.

Name	Type	Visibility	Version API
/payment-transactions/{id}	GET	Public	V1

The **SP** must provide the CVCO transaction identifier for the CVCo Acquiring Platform to deliver data about it.

When the transaction status gets to "authorized", the **SP** will receive additional data regarding the authorization request (number, date...) including the real amount paid in

CVCO. This data must be used by the **SP** to determine if the Order amount is fully paid or if an additional payment must be done.

#### 4.5.3.1 Request Data

Field	Type	Constraint
id	String	The transaction ID ( <i>transaction</i> → <i>id</i> ) obtained in the response data of the create Transaction method

#### 4.5.3.2 Post Data

N/A

#### 4.5.3.3 Ordered list of Fields for sealing

Ordre	Field
1	transaction → id

#### 4.5.3.4 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an OK (200) HTTP code. Details on the object are provided in the response body.

Field	Type	Constraint
transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

In case of failure to retrieve the transaction, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.5.3.5 Call Example

```
curl -v -X GET {base_url}/payment-transactions/14fddh1256 \  
-H "Content-Type: application/json" \  
-H "ANCV-Security: {algo.version.seal}"
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 200 OK
- Body:

```

{
  "transaction": {
    "id": "14fddh1256",
    "creationDate": "2018-08-28T11:18:00Z",
    "updateDate": "2018-08-28T11:25:00Z",
    "expirationDate": "",
    "state": "AUTHORIZED",
    "merchant": {
      "shopId": 13235554,
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "paymentId": "42556",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "NORMAL",
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    },
    "payers": [
      {
        "beneficiaryId": 1536923388807,
        "amount": {
          "total": 3500,
          "currency": "978"
        },
        "authorizations": [
          {
            "number": "767909",
            "type": "CVCo",
            "amount": {
              "total": 3000,
              "currency": "978"
            },
            "validationDate": "2018-08-28T11:25:00Z",
            "holder": "15*****8807"
          }
        ]
      }
    ]
  },
  "applicationContext": {
    "returnContext": "1st command from customer",
    "customerId": "customer1236555"
  },
  "responseDate": "2018-08-28T12:22:03Z"
}

```

### How to use

This API can be used when provided [webhook](#) are insufficient or inadequate.

It's recommended to call this API one time per second in order to correctly refresh transactions state.

## 4.5.4 Execute Deferred Payment

If a payment transaction was authorized with a delayed validation time mode, then the **SP** must finalize the transaction in order to have it captured for the Merchant to get paid.

The transaction finalization is done by using a dedicated Web Service method (REST/JSON) provided by the CVCo Acquiring Platform for Delayed payment validation.

Name	Type	Visibility	Version API
/payment-transactions/{id}/execute	POST	Public	V1

The **SP** must provide the CVCO transaction identifier for the CVCo Acquiring Platform (id) to execute the authorized payment.

An authorized payment with a delayed validation time mode is not captured until it has been validated by the **SP**. This means that the Merchant won't receive any refund by the ANCV.

A paiement with a delayed validation time mode that has not been validated by the **SP** after a defined delay, is automatically cancelled on the initiative of the Merchant:

- The Beneficiary is then notified that a cancellation for the CVCO payment was done and that its Beneficiary account was refund.
- The **SP** is informed through the transactions reports that the cancellation was done.

The delay must be set by the SP during the transaction creation process and cannot exceed a maximum delay of **6 calendar days** after the transaction creation date.

When validating the payment, the SP can adjust the real amount to be paid in CVCo so long as it is greater than the minimum amount (€0.01) and less or equal to the authorized amount.

### 4.5.4.1 Request Data

Field	Type	Constraint
id	String	The transaction ID ( <i>transaction</i> → <i>id</i> ) obtained in the response data of the create Transaction method

### 4.5.4.2 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
amount	Object	Mandatory Global and final amount to be paid in CVCo for this transaction
payers	Object List	Optional Can be used to specify the amount to be paid by each payer. If not provided, then the amount for each payer is determined in proportion to each authorized amount

### 4.5.4.3 Ordered list of Fields for sealing

Ordre	Field
1	transaction → id

### 4.5.4.4 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an OK (200) HTTP code. Details on the object are provided in the response body.

Field	Type	Constraint
transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of failure to execute the transaction, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.5.4.5 Call example

```
curl -v -X GET {base_url}/payment-transactions/14fddh1256 \  
-H "Content-Type: application/json" \  
-H "ANCV-Security: {algo.version.seal}"
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 200 OK
- Body:



```

{
  "transaction": {
    "id": "2468135791",
    "creationDate": "2018-05-18T05:18:00Z",
    "updateDate": "2018-05-18T10:41:36Z",
    "state": "VALIDATED",
    "expirationDate": null,
    "merchant": {
      "shopId": 100011118,
      "shopAssistantId": "Magasin",
      "terminalId": "machine-11111",
      "serviceProviderId": 632109876
    },
    "order": {
      "id": "panier-33455",
      "paymentId": "42556",
      "label": "Magasin",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "tspdMode": "001",
      "captureMode": "DEFERRED",
      "captureDate": "2018-05-23T00:00:00+02:00"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    },
    "payers": [
      {
        "beneficiaryId": "james.bond@yopmail.com",
        "amount": {
          "total": 3500,
          "currency": "978"
        },
        "authorizations": [
          {
            "number": "767909",
            "type": "CVCo",
            "amount": {
              "total": 2500,
              "currency": "978"
            },
            "validationDate": "2018-05-18T05:47:25+02:00",
            "holder": "14*****0007"
          }
        ]
      }
    ]
  },
  "applicationContext": {
    "returnContext": "",
    "customerId": "customer1236555"
  },
  "responseDate": "2018-05-18T08:33:14.018+01:00"
}

```

#### 4.5.5 Cancel Transaction

On behalf of the **Merchant**, or its **SP**, it can be possible to send a transaction for the cancellation of a CVCo payment.

The CVCo Acquiring Platform determines if the cancellation transaction can be achieved according to specific circumstances.

The circumstances for a payment transaction to be cancellable are either:

- In NORMAL or DEFERRED mode
  - Transaction was initialized but no authorization was delivered
- In DEFERRED mode (deferred payment)
  - Authorization was delivered (status AUTHORIZED) without validation (/execute)
  - Authorization was delivered (status VALIDATED) with validation (/execute) : the allowed time for transaction cancellation is 4h after the delivery of the /execute call.
- In NORMAL mode (normal payment)
  - Authorization was delivered (status VALIDATED) with validation (/execute) : the allowed time for transaction cancellation is 4h after the delivery of its authorization.

The cancellation transaction is done by using a dedicated Web Service method (REST/JSON) for payment cancellation

Name	Type	Visibility	Version API
/payment-transactions/{id}/cancellation	POST	Public	V1

When a payment is cancelled:

- The Beneficiary will be notified that a cancellation for the CVCo payment was done and that its Beneficiary account was refund.
- The payment's transaction status is in a final state that can't be changed anymore.

#### 4.5.5.1 Request Data

Field	Type	Field
id	String	The transaction ID ( <i>transaction</i> → <i>id</i> ) obtained in the response data of the create Transaction method

#### 4.5.5.2 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
reason	String	Mandatory The accepted values are described in the Appendix (8.2.1)
label	String	Optional Merchant or SP detailed reason for cancellation (free text)
requestDate	String	Optional yyyy-MM-dd'THH:mm:ss.SSS'Z'

#### 4.5.5.3 Ordered list of Fields for sealing

Ordre	Champ
1	Transaction → id
2	Cancellation → reason

#### 4.5.5.4 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an OK (201 Created) HTTP code, meaning cancel operation request was successfully registered. Details on the object are provided in the response body.

Field	Type	Constraint
transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

Calling back the API with the same parameters will generate a 200 OK response, meaning no change was done.

In case of failure to cancel the transaction, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

### Recovery procedures in case of technical error

If a technical error is received (with an HTTP status 408 or 500 for example), the proper procedure is to call the API for [transaction status check](#) to verify whether the cancellation has been processed or not.

Otherwise, it is possible that the cancellation has not been performed and that the SP has not been notified. It is therefore necessary to ensure that the operation has been completed and to relaunch it if not.

#### 4.5.5.5 Call example

```
curl -v -X POST {base_url}/payment-transactions/2468135791/cancellation \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-d '{
  "requestDate": "2018-05-18T10:14:25Z",
  "reason": "COMPLEMENTARY_PAYMENT",
  "label": "Cancellation for lack of complementary payment on holder card"
}'
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 200 OK
- Body:

```

{
  "transaction": {
    "id": "2468135791",
    "creationDate": "2018-05-18T05:18:00Z",
    "updateDate": "2018-05-18T10:41:36Z",
    "state": "CANCELLED",
    "expirationDate": null,
    "merchant": {
      "shopId": 100011118,
      "shopAssistantId": "Magasin",
      "terminalId": "machine-11111",
      "serviceProviderId": 632109876
    },
    "order": {
      "id": "panier-33455",
      "paymentId": "42556",
      "label": "Magasin",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "tspdMode": "001",
      "captureMode": "DEFERRED",
      "captureDate": "2018-05-23T00:00:00+02:00"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    },
    "payers": [
      {
        "beneficiaryId": "james.bond@yopmail.com",
        "amount": {
          "total": 3500,
          "currency": "978"
        },
        "authorizations": [
          {
            "number": "767909",
            "type": "CVCo",
            "amount": {
              "total": 3000,
              "currency": "978"
            },
            "validationDate": "2018-05-18T05:47:25+02:00",
            "holder": "14*****0007"
          }
        ]
      }
    ],
    "cancellation": {
      "effectiveDate": "2018-05-18T10:41:36+02:00",
      "reason": "COMPLEMENTARY_PAYMENT",
      "label": "Cancellation for lack of complementary payment on holder card"
    },
    "applicationContext": {
      "returnContext": "",
      "customerId": "customer1236555"
    },
    "responseDate": "2018-05-18T08:33:14.018+01:00"
  }
}

```

## 4.6 Delayed beneficiary acceptance pre-transactions

### 4.6.1 Create pre-transaction

The first step of any Payment with delayed acceptance using the CVC Co Acquiring Platform is to use a dedicated Web Service method (REST/JSON) for a CVC Co pre-transaction initialization with some data on the Merchant and the Order.

Name	Type	Visibility	Version API
/pre-transactions	POST	Public	V1

#### 4.6.1.1 Precondition

The store must be active.

If it's inactive, request will be refused and a 403 HTTP error with MERCHANT\_NOT\_ALLOWED error code will be returned

#### 4.6.1.2 Request Data

N/A

#### 4.6.1.3 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
merchant	Object	Mandatory The merchant's identity in ANCV CVC Co repository
order	Object	Mandatory The order to be paid Minimum transaction amount in CVC Co is <b>1ct</b> . The data set (shopId, id, prePaymentId) must be unique for the day
paymentMethod	Object	Mandatory When using deferred capture mode, the maximum allowed delay before validation is of <b>6 days</b> after Consumer acceptance
redirectUrls	Object	Optional If filled, URLs are called to notify the service provider of the result of the pretransaction (cf. <a href="#">Webhooks</a> )
expirationDate	String	Mandatory Used to determine the maximum delay for acceptance by the Consumer. Maximum authorized delay is <b>30 days</b> . yyyy-MM-dd'THH:mm:ss.SSS'Z'
applicationContext	Object	Optional Technical data used by the connected device
requestDate	String	Optional Sending date and time of the request by the SP yyyy-MM-dd'THH:mm:ss.SSS'Z'

#### 4.6.1.4 Ordered list of Fields for sealing

Ordre	Champ
1	merchant → shopId
2	merchant → serviceProviderId (only if provided)
3	order → id
4	order → prePaymentId (only if provided)
5	order → amount → total
6	expirationDate

#### 4.6.1.5 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send a created (201) HTTP code. Details on the created object are provided in the response body.

Field	Type	Constraint
pre-transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of request duplication on a shop for the day (same "order → id " and " order → prePaymentId"), the response will be an OK (200) HTTP code still with the same response body as for the previous response.

In case of failure to provide the QR-Code, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.6.1.6 Call example

```
curl -v -X POST {base_url}/pre-transactions \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-d '{
  "merchant": {
    "shopId": 13235554,
    "serviceProviderId": 98232552
  },
  "order": {
    "id": "panier-33455",
    "prePaymentId": "18",
    "amount": {
      "total": 4000,
      "currency": "978"
    }
  },
  "paymentMethod": {
    "captureMode": "DEFERRED",
    "captureTerm": 4,
    "tspdMode": "001"
  },
  "redirectUrls": {
    "returnUrl": "http://ptl.com/response-url",
    "cancelUrl": "http://ptl.com/cancel-url"
  },
  "expirationDate": "2019-04-10T00:00:00Z",
  "applicationContext": {
    "returnContext": "1st command from customer",
    "customerId": "customer1236555"
  },
  "requestDate": "2019-03-26T11:17:58Z"
}'
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 201 Created
- Body :

```

{
  "pre-transaction": {
    "id": "14fjdh1256",
    "creationDate": "2019-03-26T11:18:00Z",
    "updateDate": "2019-03-26T11:18:00Z",
    "expirationDate": "2019-04-10T00:00:00Z",
    "state": "CREATED",
    "merchant": {
      "shopId": 13235554,
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "prePaymentId": "18",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "DEFERRED",
      "captureTerm": 4,
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    },
    "applicationContext": {
      "returnContext": "1st command from customer",
      "customerId": "customer1236555"
    },
    "responseDate": "2019-03-26T11:18:00Z"
  }
}

```

RedirectUrls object is optional, if it's present it can be filled according to these examples (URLs must be replaced with correct values).

```

{
  "pre-transaction": {
    "id": "14fjdh1256",
    "creationDate": "2019-03-26T11:18:00Z",
    "updateDate": "2019-03-26T11:18:00Z",
    "expirationDate": "2019-04-10T00:00:00Z",
    "state": "CREATED",
    "merchant": {
      "shopId": 13235554,
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "prePaymentId": "18",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "DEFERRED",
      "captureTerm": 4,
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    }
  },
  "applicationContext": {
    "returnContext": "1st command from customer",
    "customerId": "customer1236555"
  },
  "responseDate": "2019-03-26T11:18:00Z"
}

```

## 4.6.2 Request Merchant Presented QR-Code

In order to present the Consumer with a Merchant Presented Payment QR-Code, the SP will have to obtain it from the ANCV Acquiring Solution using this method on an initialized pre-transaction.

This QR-Code will then be used by the Consumer in order to accept the payment using her/his ANCV Mobile Application to scan the code

Name	Type	Visibility	Version API
/pre-transactions/{id}/qr-code	GET	Public	V1

The provided QR-Code will remain valid for acceptance until it is either:

- Used for an authorized payment transaction,
- Expired,
- Aborted by the Merchant or SP
- Rejected by the Consumer who chooses to refuse to pay for it from her/his ANCV Mobile Application.

Therefore, the answer will always be the same picture should the SP use this method on the same pre-transaction.

### 4.6.2.1 Request headers

Field	Type	Accepted values
Accept	String	To retrieve PNG formatted QR Code, one of the followings : <ul style="list-style-type: none"> <li>• image/png</li> </ul>



		<ul style="list-style-type: none"> <li>• image/*</li> <li>• */*</li> </ul> <p>To retrieve PNG formatted and base64 encoded QR Code, one of the followings :</p> <ul style="list-style-type: none"> <li>• text/plain</li> <li>• text/*</li> </ul>
--	--	--

#### 4.6.2.2 Request Data

Field	Type	Constraint
id	String	The pre-transaction ID ( <i>pre-transaction</i> → <i>id</i> ) obtained in the response data of the create pre-transaction method

#### 4.6.2.3 Post Data

N/A

#### 4.6.2.4 Ordered list of Fields for sealing

Ordre	Field
1	pre-transaction → id

#### 4.6.2.5 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send in the answer:

- An OK (200) HTTP code.
- The QR-Code picture in the response Body, encoded in PNG format (image size is 300x300 pixels)
- In the response Header, the URL (encoded in the QR-Code) that can be used by the Consumer to accept the payment :

Field	Type	Constraint
pre-transaction-url	String	Mandatory URL that can be used by the Consumer to accept the payment

In case of failure to provide the QR-Code, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.6.2.6 Call example

```
curl -v -X GET {base_url}/pre-transactions/ffuqqy90ii/qr-code \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-H 'Accept: image/png'
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 201 Created
- Header :

```
pre-transaction-url: {base_url}/pre-transactions/ffuqqy90ii/qr-code
```

- Body :
  - Merchant Presented Payment QR-code in PNG format:



### 4.6.3 Retrieve pre-transaction

A CVCO pre-transaction status check is to be done at any time when the **SP** needs to check if an existing CVCO pre-transaction has evolved.

If the **SP** has any doubt on the pre-transaction status; either due to some timeout, or in case of data check; then it is in its responsibility to check the real status with a call to this method.

It is achieved using a dedicated Web Service method (REST/JSON) for pre-transaction status check.

Name	Type	Visibility	Version API
/pre-transactions/{id}	GET	Public	V1

The **SP** must provide the CVCO pre-transaction identifier for the CVCo Acquiring Platform to deliver data about it.

When the pre-transaction is "USED" with an accepted payment transaction linked to it, the **SP** will receive additional data regarding the pre-transaction including the transaction ID.

This transaction ID must be used by the **SP** with the Retrieve Transaction Method (cf. chapter 6.4.3) to receive additional data regarding the authorization request (number, date...) including the real amount paid in CVCO. This data must be used by the **SP** to determine if the Order amount is fully paid or if an additional payment must be done.

#### 4.6.3.1 Request Data

Field	Type	Constraint
id	String	The pre-transaction ID ( <i>pre-transaction</i> → <i>id</i> ) obtained in the response data of the create pre-transaction method

#### 4.6.3.2 Post Data

N/A

#### 4.6.3.3 Ordered list of Fields for sealing

Ordre	Field
1	pre-transaction → id

#### 4.6.3.4 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an OK (200) HTTP code. Details on the object are provided in the response body.

Field	Type	Constraint
preTransaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of failure to provide the QR-Code, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

#### 4.6.3.5 Call Example

```
curl -v -X GET {base_url}/pre-transactions/8rh4q2ujg9 \  
-H "Content-Type: application/json" \  
-H "ANCV-Security: {algo.version.seal}"
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 201 Created
- Body :

```
{
  "pre-transaction": {
    "id": "8rh4q2ujg9",
    "creationDate": "2019-03-26T11:18:00Z",
    "updateDate": "2019-03-26:18:00Z",
    "expirationDate": "2019-04-10T00:00:00Z",
    "state": "USED",
    "validatedPaymentTransactionId": "4gh44ouj47",
    "merchant": {
      "shopId": 95235554,
      "serviceProviderId": 85632552
    },
    "order": {
      "id": "panier-33455",
      "prePaymentId": "18",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "NORMAL",
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    }
  },
  "applicationContext": {
    "returnContext": "1st command from customer",
    "customerId": "customer1236555"
  },
  "responseDate": "2019-03-26T11:20:00Z"
}
```

#### 4.6.4 Send pre-transaction to contact

For reservation purpose, it is possible for the **Merchant** or its **SP** to request the ANCV Acquiring Platform for a sending of email towards a given contact address – that of the Consumer – asking for the payment of an initialized pre-transaction.

This public API sends an email containing the payment QR-code. Either an email address owned by the beneficiary, or its ID is necessary to achieve the operation and is transmitted as a parameter to the API.

The transaction state must be CREATED for this operation to be authorized. After the operation, pre-transaction status is updated to PROCESSING.

Name	Type	Visibility	Version API
/pre-transactions/{id}/contact	POST	Public	V1

##### 4.6.4.1 Request parameters

Field	Type	Constraint
id	String	The pre-transaction ID ( <i>pre-transaction</i> → <i>id</i> ) obtained in the response data of the create pre-transaction method

##### 4.6.4.2 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
contact	String	Mandatory if beneficiaryId is not transmitted. <b>Must not be present otherwise.</b> Email address of the beneficiary to whom the email is addressed to. Does not necessarily correspond to a beneficiary account
beneficiaryId	String	Mandatory if contact is not transmitted. <b>Must not be present otherwise.</b> ID of the beneficiary to whom the email is addressed to

#### 4.6.4.3 Ordered list of Fields for sealing

Ordre	Field
1	pre-transaction → id
2	contact
3	beneficiaryId

#### 4.6.4.4 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an Accepted (202) HTTP code. Details on the object are provided in the response body.

Field	Type	Constraint
pre-transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of failure to provide the QR-Code, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.6.4.5 Call Example

```
curl -v -X POST {base_url}/pre-transactions/14fjdh1256/contact \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-d '{
  {
    "contact": "toto@yopmail.com"
  }
}'
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 201 Created
- Body:

```

{
  "pre-transaction": {
    "id": "14fjdh1256",
    "creationDate": "2019-03-26T11:18:00Z ",
    "updateDate": "2019-03-28T18:00:00Z",
    "expirationDate": "2019-04-10T00:00:00Z",
    "state": "PROCESSING",
    "merchant": {
      "shopId": 13235554,
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "paymentId": "42556",
      "prePaymentId": "18",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "captureMode": "NORMAL",
      "tspdMode": "001"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    }
  },
  "applicationContext": {
    "returnContext": "1ère commande du bénéficiaire",
    "customerId": "customer1236555"
  },
  "responseDate": "2019-03-28T18:00:00Z"
}

```

#### 4.6.5 Abort pre-transaction

On behalf of the **Merchant**, or its **SP**, it can be possible to send an abortion request on a given pre-transaction.

The CVCo Acquiring Platform determines if the abortion can be achieved according to specific circumstances.

The circumstances for a pre-transaction to be aborted are either:

- In NORMAL or DEFERRED mode
  - Transaction was initialized but no authorization was delivered
- In DEFERRED mode (deferred payment)
  - Authorization was delivered (status AUTHORIZED) without validation (/execute)
  - Authorization was delivered (status VALIDATED) with validation (/execute) : the allowed time for pre-transaction cancellation is 4h after the delivery of the /execute call.
- In NORMAL mode (normal payment)
  - Authorization was delivered (status VALIDATED) with validation (/execute) : the allowed time for pre-transaction cancellation is 4h after the delivery of its authorization.

If a payment transaction was authorized on the pre-transaction, then it can't be aborted anymore. In order to refund the Consumer, the **Merchant** or **SP** will have to make use of the Cancel Transaction method

To abort a pre-transaction, the **Merchant** or **SP** must use a dedicated Web Service method (REST/JSON) for pre-transaction abortion.

Nom	Type	Visibility	Version API
/pre-transactions/{id}/abort	POST	Public	V1

When a pre-transaction is aborted:

- The Merchant Presented QR-Code linked with it cannot be used anymore to accept the payment: the Consumer will receive an error message if she/he tries to scan it.
- The pre-transaction status is in a final state that can't be changed anymore.

The abortion of a pre-transaction can also occur when the payment is rejected by the Consumer who chooses to refuse to pay for it from her/his ANCV Mobile Application. In such a situation, the SP will be notified of the abortion using the cancel automatic response service (cf. [Webhooks](#))

#### 4.6.5.1 Request Data

Field	Type	Constraint
id	String	The pre-transaction ID ( <i>pre-transaction</i> → <i>id</i> ) obtained in the response data of the create pre-transaction method

#### 4.6.5.2 Post Data

The data to be provided in the http body are:

Field	Type	Constraint
reason	String	Mandatory The accepted values are described in the Appendix (8.2.1)
label	String	Optional Merchant or SP detailed reason for abortion (free text)
requestDate	String	Optional yyyy-MM-dd'THH:mm:ss.SSS'Z'

#### 4.6.5.3 Ordered list of Fields for sealing

Ordre	Field
1	pre-transaction → id
2	abort → reason → contact
3	abort → reason

#### 4.6.5.4 Response Data

In case of approval for this method request, the CVCo Acquiring Platform will send an OK (200) HTTP code. Details on the object are provided in the response body

Field	Type	Constraint
pre-transaction	Object	Mandatory
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

In case of failure to provide the QR-Code, an appropriate HTTP response code (cf. [Reject codes and root cause](#)) will be provided, as well as a specific error response body providing information on the reject cause.

Field	Type	Constraint
errorCode	String	Mandatory Error Code related with reject cause
errorMessage	String	Mandatory Error message easily read to describe the reject cause

#### 4.6.5.5 Call example

```
curl -v -X POST {base_url}/pre-transactions/14fjdh1256/abort \
-H "Content-Type: application/json" \
-H "ANCV-Security: {algo.version.seal}" \
-d '{
  "requestDate": "2019-03-29T10:14:25Z",
  "reason": "ABORTED_MERCHANT",
  "label": "Abortion due to wrong operator input"
}'
```

A sealing calculation help can be found in appendix [Seal calculation](#).

Response:

- Status: HTTP 200 OK
- Body:

```
{
  "pre-transaction": {
    "id": "14fjdh1256",
    "creationDate": "2019-03-29T12:18:00Z",
    "updateDate": "2019-03-30T16:20:36Z",
    "expirationDate": "2019-04-10T00:00:00Z",
    "state": "ABORTED",
    "merchant": {
      "shopId": 13235554,
      "shopAssistantId": "Magasin",
      "terminalId": "machine-11111",
      "serviceProviderId": 98232552
    },
    "order": {
      "id": "panier-33455",
      "prePaymentId": "18",
      "label": "Magasin",
      "amount": {
        "total": 4000,
        "currency": "978"
      }
    },
    "paymentMethod": {
      "tspdMode": "001",
      "captureMode": "NORMAL",
      "captureDate": "2019-03-25T12:18:00+02:00"
    },
    "redirectUrls": {
      "returnUrl": "http://ptl.com/response-url",
      "cancelUrl": "http://ptl.com/cancel-url"
    },
    "abort": {
      "effectiveDate": "2019-03-29T10:14:30+02:00",
      "reason": "ABORTED_MERCHANT",
      "label": "Abortion due to wrong operator input"
    },
    "applicationContext": {
      "returnContext": "",
      "customerId": "customer1236555"
    },
    "responseDate": "2019-03-29T10:14:30.000+01:00"
  }
}
```



## 5 Specific services

### 5.1 Webhooks

The Automatic Response services are optional services set up by the **Merchant integrator** or by the **Service Provider** and made available to the ANCV platform to provide the result of the Payment transactions or pre-transactions managed by the ANCV either when the transaction is authorized (return), rejected or aborted (cancel).

If the service provider wants to use this functionality, these services must be able to process a request with parameter transmission by POST method in REST/JSON.

The CVCo Acquiring Platform will provide with each call of both services the data necessary to identify the original transaction or pre-transaction and its status.

The situations in which one of the Automatic Response services is contacted are:

- For the return service:
  - When a transaction has received all the required payments authorizations
- For the cancel service:
  - When a delayed acceptance payment expired with no authorized payment,
  - When a Consumer, during the accepting process of a delayed payment, chooses to abort its payment and explicitly refuses to pay it later (before the expiration date)
  - When the CVCo Acquiring Platform rejects the payment transaction for the provided payer
  - When a required payment authorization is refused
  - When a beneficiary cancels the payment, either by an explicit action or by a lack of action before a defined TTL (250 seconds for adjustment step, 250 for authorization step)
    - From the payment page of the Merchant integrator or the Service Provider (lack of action while staying on the page or after closing the page)
    - From the beneficiary application (lack of action after closing of the application or losing connection)

Details on the object provided in the request body are:

Field	Type	Constraint
transaction	Object	Optional Not provided if pre-transaction object is present
preTransaction	Object	Optional Not provided if transaction object is present
applicationContext	Object	Optional
responseDate	String	Mandatory yyyy-MM-dd'THH:mm:ss.SSS'Z'

URLs defined for these automatic response services must follow the rules defined by chapter [Rules](#) dealing with rules applied to the payment process.

If a rule is not respected, no call will be made.

### 5.2 Transactions daily reports

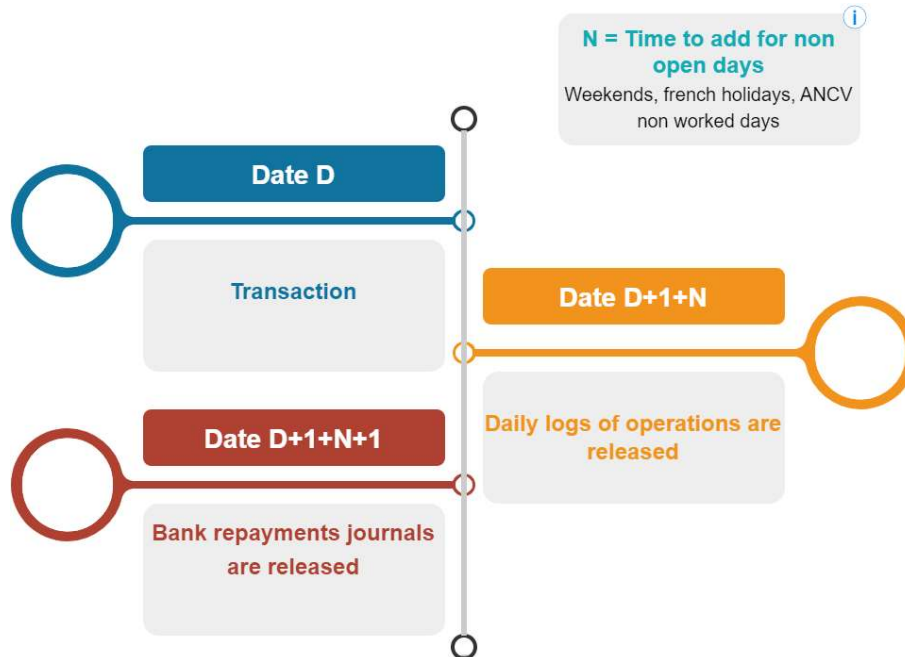
Transactions reports are automatically constructed daily in order to inform the **SP** of the status for transactions done the day before.

These reports are of two kinds and gives for the payment transactions in CVCo:

- The payment accepted or rejected, and all the operations realized on accepted payments (cancellation, capture, refund...)
- The refund status of accepted payments with refund amounts towards the Merchants

Transfer methods for these reports are either:

- Default mode is using an SFTP account on the ANCV Acquiring Platform . Each integrated **SP** will receive an individual account per environment where the log files will be accessible for pull.
- Any other method will have to be studied by the ANCV with the **SP** in order to define the setup conditions.



#### Non worked days :

```
# Jours ferries officiels
01-01-2023
10-04-2023
01-05-2023
08-05-2023
18-05-2023
14-07-2023
15-08-2023
01-11-2023
11-11-2023
25-12-2023
```

```
# Jours ANCV
23-12-2022
30-12-2022
02-01-2023
29-05-2023
22-12-2023
29-12-2023
```



emplacement : /DATA/file\_transfer/in/ancv/acquisition/calendrier\_acquisition.txt

Dernier calendrier communiqué (mise à jour le 06 décembre 2022) / last known calendar (updated on 12/06/2022)

## 5.2.1 Daily log of Operations

This report is provided **daily (on worked days)** if at least one transaction got a status change since the last log was created (otherwise no report is created).

It consists of a log for each **Service Provider** containing all the transactions created on behalf of the Merchants that it operates.

For Merchants directly connected to the CVCo Acquiring Platform, the log contains all the transactions directly created.

The transactions contained in a log are all the ones that have undergone at least one operation since the last log was created.

Only one line is present per transaction. If several changes have been made during the same day, only the last status is transmitted.

The log file format is **CSV** – using ';' separator character.

The encoding of the transmitted fields is identical to that used for the exchanges by REST/JSON.

The first line contains information about the file:

SN	Code	Description
1	type	File type : « DLO »
2	recipient	File recipient ID: · For Merchants: <b>shopId</b> · For SP: <b>serviceProviderId</b>
3	creationDate	File creation date ISO8601 formatted date: yyyy-MM-dd'THH:mm:ss.SSS'Z'
4	transactionNumber	Number of transactions in the log file.

The following lines (from second to last-to-last) relate to all retrieved transactions. Each transaction may, for some fields, not have data to transmit. In this case, the field in the line is not valued (2 separator characters in a row)

The fields transmitted, by serial number, are:

SN	Code	Description
1	transactionId	Transaction ID (transaction → id)
2	updateDate	Latest update date for the transaction (transaction → updateDate)
3	state	Latest status of the transaction (transaction → state)
4+	subState	Latest sub-status of the transaction (transaction → subState)
5	shopId	(transaction → merchant → shopId)
6	shopAssistantId	(transaction → merchant → shopAssistantId)
7	terminalId	(transaction → merchant → terminalId)
8	orderId	(transaction → order → id)
9	paymentId	(transaction → order → paymentId)
10+	orderLabel	(transaction → order → label)
11	amountTotal	(transaction → order → amount → total)
12	amountCurrency	(transaction → order → amount → currency)
13	tspdMode	(transaction → paymentMethod → tspdMode)
14	cancellationDate	(transaction → cancellation → effectiveDate)
15	cancellationReason	(transaction → cancellation → reason)
16*	cancellationLabel	(transaction → cancellation → label)

17*	beneficiaryId	(transaction → payer → beneficiaryId)
18*	authorizationType	(transaction → payer → authorization → type)
19*	authorizationAmount	(transaction → payer → authorization → amount → total)
20*	authorizationNumber	(transaction → payer → authorization → number)
21*	authorizationDate	(transaction → payer → authorization → validationDate)
22*	authorizationHolder	(transaction → payer → authorization → holder)

\* This data block can be repeated several times depending on the number of authorizations requested and obtained (complementary payment...)

The last line contains an end-of-file tag:

SN	Code	Description
1	EOF	Field can only contain « EOF »

Operations journal example

DLO\_{serviceProviderId}\_{datedébut}\_{datefin}.csv

```
DLO;100016;2019-03-02T04:52:01.689Z;2
f9xrsgco;2019-03-
02T03:00:01.783Z;CONSIGNED;;10000065;;PANIERAAAAB;456467;;5500;978;1;;;1
0001001428;CVD;500;f9xrsgco;2019-03-02T03:00:01.783Z;10*****1428
f9xx6cfksq;2019-03-
01T13:40:18.119Z;ABORTED;;10000073;;PANIERAAAAC;456467;;5500;978;1;;;
;;
EOF
```

## 5.2.2 Bank Repayments Journal

This log is made **daily (on worked days)** if refunds were made by the ANCV to the Merchants (otherwise no log is made).

It consists of a log for each **Service Provider** containing all the transactions refunded related to the Merchants that it operates.

For Merchants directly connected to the CVCo Acquiring Platform, the log contains all the transactions directly created that were refunded.

The transactions contained in a log are all the ones that were refunded since the last log was created.

Only one line is present per transaction: it is not possible for a transaction to be refunded twice, or to be split in two refund processes.

The log file format is **CSV** – using ';' separator character.

The encoding of the transmitted fields is identical to that used for the exchanges by REST/JSON.

The first line contains information about the file:

SN	Code	Description
1	Type	File type : « BRJ »
2	recipient	File recipient ID: · For Merchants: <b>shopId</b> · For SP: <b>serviceProviderId</b>
3	creationDate	File creation date ISO8601 formatted date: yyyy-MM-dd'THH:mm:ss.SSS'Z'
4	transactionNumber	Number of transactions in the log file.

The following lines (from second to last-to-last) relate to all retrieved transactions. Each transaction may, for some fields, not have data to transmit. In this case, the field in the line is not valued (2 separator characters in a row)

The fields transmitted, by serial number, are:

SN	Code	Description
1	transactionId	Transaction ID (transaction → id)
2	updateDate	Last update date of the transaction n (transaction → updateDate)
3	shopId	(transaction → merchant → shopId)
4	orderId	(transaction → order → id)
5	orderLabel	(transaction → order → label)
6	paymentId	(transaction → order → paymentId)
7*	amountTotal	(transaction → refund → amount → total)
8*	amountRefund	(transaction → refund → amount → net)
9*	amountFee	(transaction → refund → amount → fee)
10*	amountCurrency	(transaction → refund → amount → currency)
11*	refundDate*	(transaction → refund → effectiveDate)
12*	refundType*	(transaction → refund → type)
13*	slipId*	Slip id used during acquiring operation

\* This data block can be repeated several times depending on the payment methods used (CVCo, CB ...)

The last line contains an end-of-file tag:

SN	Code	Description
1	EOF	Field can only contain « EOF »

Refunds journal example

BRJ\_{serviceProviderId}\_{datedébut}\_{datefin}.csv

```
BRJ;AVIASIMTMACCOUNT;2021-04-14T10:24:49.274Z;1
1000000008;2019-07-15T08:15:00Z;900090009;panier-74215;Achat de
nougat;999888;3000;2800;200;978;2019-07-
15T08:12:01Z;CV_CONNECT;18agt45094718075
EOF
```

### 5.2.3 Specific Bank Repayment Journals and Daily Logs of Operations

These specific journals are meant for an actor, representing a group of merchants for which he needs the Daily Logs of Operations and the Bank Repayment Journals.

The journals are generated only for actors having requested them and for whom ANCV has validated the request.

The list of merchants (shopIds) defined for an actor can be modified by requesting ANCV.

For each shopIds within the list, Daily Logs of Operations and Bank Repayment Journals are generated identically to the usual journals (same frequency, same content, same format).

The specific journals are transferred to an SFTP account dedicated for and validated by the actor.

The "recipient" name appearing on the log headers is the name of the SFTP dedicated account.

*N.B.: these specific journals do not replace the usual journals for merchants, both are generated. The job starting the generation of these specific journals is launched just after the end of the generation of the daily Bank Repayment Journals for all merchants.*

Specific operations journal example

DLO\_{NOMDUCOMPTE}\_{datedébut}\_{datefin}.csv

```
DLO;AVIASIMTMACCOUNT;2021-01-29T05:52:21.483Z;1
1000000008;2019-07-15T08:15:00Z;PAID;;900090009;;machine-44751;panier-
74215;999888;Achat de
nougat;3000;978;001;;;15369233109;CV_CONNECT;3000;800888;2019-07-
15T08:12:30Z;15*****3109
EOF
```

Specific refunds journal example.

BRJ\_{sftp\_account\_name}\_{startDate}\_{endDate}.csv

```
BRJ;AVIASIMTMACCOUNT;2021-04-14T10:24:49.274Z;2

1000000007;2021-04-14T10:24:48.915Z;200065815;panier-10007;Voyage à
Ibiza;888883;150000;148000;2000;978;2018-09-
23T11:02:00Z;CV_CONNECT;46751067

1000000008;2019-07-15T08:15:00Z;900090009;panier-74215;Achat de
nougat;999888;3000;2800;200;978;2019-07-
15T08:12:01Z;CV_CONNECT;18agt45094718075

EOF
```

## 5.2.4 SFTP Space

### 5.2.4.1 Space configuration

An SFTP space is available for merchants / integrators who wish to receive transaction logs.

- The space is composed as such :

Directory	File
<i>./upload/operations_journal</i>	<i>Daily logs of operations</i>
<i>./upload/refunds_journal</i>	<i>Refund journals</i>
<i>./upload/specific_operations_journal</i>	<i>Specific daily logs of operations (when concerned)</i>
<i>./upload/specific_refunds_journal</i>	<i>Specific refund journals (when concerned)</i>

- Only "upload" directory and its children are available to the space users
- **Files are only kept 6 months, and are automatically purged after that**

Limits on moving/creating directories/files

SFT space users are given write permission on directories and files to offer a bit of liberty in managing log organization.

However, a few operations are prohibited in order to maintain the automatic file transfer and purge working :

- within an automatically created directory, every operation other than deleting or renaming of files such as :
  - moving an automatically created directory inside another
  - adding in a compressed file (zip...)
- renaming the automatically created directories
- deleting the automatically created directories

For cleaning / sorting purposes, solutions are available such as :

- retrieving files on another file system and using SFTP space only as an input for transaction logs (recommended approach)
- creating an archive directory outside automatically created directories (and named differently) used as destination for verified log files (the chosen hierarchy can be created within this directory)
- renaming directly the original log files (adding a prefix or a suffix to differentiate verified files)

**Warning** : the 6-month conservation time is applied to every file from its creation date on the space

#### Recommended tools

In order to connect to the SFTP space, an SFTP client should be used. For instance :

- FileZilla
- WinSCP

NB : using a web browser (Chrome/edge) is not a viable solution to access the SFTP space.

#### How to connect

##### Using login / password

The easier way to access SFTP space is using login and password provided by the technical team during staging phase to merchants / integrators.

##### Exchanging SSH keys

A more secure way to access SFTP space is by exchanging SSH keys :

1. Merchant / integrator generates a pair of public / private keys and provides the public one to the technical team
2. The technical team adds it to the authorized keys
3. Merchant / integrator connects using its login and the IP provided by the technical team during the creation of the SFTP account.

Connection using login / password is still possible as a backup solution even if this method is chosen.

## 5.3 Security for payment processes

The layers of security available for the exchanges done in the Payment process using the CVCo Acquiring Platform API are:

- Between the PSP and the ANCV Acceptance Solution : using mutual authentication certificates for access to the Web Services
- Between the Merchant and the ANCV Acceptance Solution: using a sealing key dedicated to the Merchant, and the encryption of sensitive data.

### 5.3.1 Rules

Security between the Merchant's IT or its Service Provider and the CVCo Acquiring Platform is achieved by using 2 protections:

- The Platform uses an HTTPS connection with an externally trusted server authentication certificate for the website on which the Web Service is available. The protocol used will be at least TLS 1.2.
- A seal must be provided with each request on the Open API using a dedicated key for each **SP** or, if no **SP** is used, **Merchant**.
  - Encryption is achieved by sealing the mandatory and sensitive parameters of the request.
  - The algorithm uses a key (or salt) that will be provided by the ANCV to the merchant or Service Intermediary. A renewal procedure can be used for salt change

A key is referenced by:

**{versionKey} = {key value}**

## 5.3.2 Sealing Algorithm

### 5.3.2.1 Fields concatenation

An ordered list of fields is specified for each API method.

Before performing a call to the CVCo Acquiring Platform, the integrator retrieves the values of the data of this list and concatenates them in a string of characters using the separator "&"

*Example:*

*If required fields are transactionID = "trx123" & shopId = 12544, then the concatenated string = "trx123&12544"*

If one of the fields is empty, then it shall not be used in the concatenation: the « & » separator can't be positioned either in the beginning or end of the concatenation string, nor be used twice in a row.

*Example:*

*If required fields are transactionID = "trx123" & shopId = 12544 & serviceProviderId = "", then the concatenated string = "trx123&12544"*

### 5.3.2.2 String sealing

Next step for the integrator is to encode the string using **base64Url** type.

After encoding the string, the new string is to be signed with a valid seal key using the **HMAC-SHA256** algorithm. The result is the **seal** required.

Important Note:

- If the parameter **Merchant** → **serviceProviderId** is specified in the creation of a transaction, then for this request as for all subsequent requests on the same transaction, the signature key to be used is that of the Service Provider (SP).
- If not, the signature key to be used is that of the Merchant (PTL).

N.B.: When using String objects to create the seal, the encoding must be set to "UTF-8" as described in chapter [Input & Output encoding types](#).

### 5.3.2.3 Requesting the CVCo Acquiring Platform

For each call to one of the API methods, the Integrator will provide an HTTP header: **ANCV-Security** containing the following data: **algorithm.versionKey.seal**

Call to the CVCo Acquiring Platform can be done using the calculated header.

*Header example:*

**HMAC256.version-3620.f536975d06c0309214f805bb90ccff089ef**

### 5.3.2.4 Request Authorization

The CVCo Acquiring Platform will ensure the consistency and validity of each request by calculating the seal according to the provided parameters and matching it with the one provided in the header by the Integrator.

Only when the match is valid will the request be authorized.

## 5.4 Reading Consumer ANCV ID from presented Bar Codes

In order to read the Consumer's ANCV ID, the SP can use one of two methods in its integrated systems. Either one will allow the SP to operate payment transactions with immediate acceptance.



The methods are:

- 1D Bar Code (Code 128)
- QR-Code

### 5.4.1 Bar Code

The format used to encode the data in the Bar Code is: Code 128.

The encoded data are: CVCoId=**{ANCV ID}**

The ANCV ID is a numeric of 11 digits, the last of which is a Luhn key.

*Example* : CVCoId=**10001001576**



### 5.4.2 QR Code

The format used to encode the data in the QR-Code is standard.

The encoded data are: CVCoId=**{ANCV ID}**

The ANCV ID is a numeric of 11 digits, the last of which is a Luhn key.

*Example* : CVCoId=**10001001576**



## 6 Documentation appendix

Some appendices are provided to all the actors interested with the ANCV online holiday voucher in order to provide a common language reference and data definition.

### 6.1 Glossary

The glossary regroups the definition for terms, acronyms, and context-specific expressions in relation with the CVCo Acquiring Platform.

This glossary is available as an independent documentation and by displaying an overlay block in the other documentation.

Term	Definition
<b>ANCV</b>	Agence Nationale pour les Chèques-Vacances
<b>CV</b>	Chèque-Vacances
<b>CVCo</b>	« Chèque-Vacances Connect », the online holiday voucher
<b>Merchant</b>	Leisure and Travel Professionals providing goods or services to Customers in exchange for payment in CVCo voucher or other payment method
<b>MOTO</b>	Mail Order / Telephone Order.
<b>PTL</b>	Professionnels du Tourisme et des Loisirs : Leisure & Travel industry
<b>PSP</b>	Payment Service Provider.
<b>SP</b>	Service Provider: can be PSP, Marketplace, Software or Hardware editors

### 6.2 Data definition

#### 6.2.1 transaction

##### 6.2.1.1 id

This is the unique identifier for a given transaction with payment by CVCo.

It is a **mandatory** parameter.

The format is **String** with a constraint of alphanumeric characters only.

##### 6.2.1.2 merchant

This object contains the data of the Merchant which is the initiator of the payment transaction.

It is a **mandatory** parameter.

It is an **Object** (cf. 8.2.1.1).

##### 6.2.1.3 creationDate

This is the Date and Time of the transaction's creation by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'.

##### 6.2.1.4 updateDate

This is the Date and Time of the transaction last update by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'.

#### 6.2.1.5 state

This field contains the actual status of a transaction.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
<b>INITIALIZED</b>	Initial State. The transaction is created
<b>PROCESSING</b>	Transitional state. The transaction is being processed by the CVC Co Acquiring Platform
<b>AUTHORIZED</b>	Transitional state. The transaction's authorization is delivered
<b>VALIDATED</b>	Transitional state. The transaction is authorized and valid for capture
<b>DELAYED or NO_SLIP_FOUND</b>	Transitional state. The transaction is validated and cannot be cancelled anymore. Slips delivering ANCV service was not able to deliver on time for last acquiring operation. Slip retrieval will be retried during the next acquiring operation.
<b>CONSIGNED</b>	Transitional state. The transaction was captured for Merchant refund by the ANCV accounting department on the CVC Co validated amount
<b>PAID</b>	Final state. The transaction's CVC Co validated amount was refund by the ANCV accounting department towards the Merchant
<b>REJECTED</b>	Final state. The transaction failed due to a reject by the CVC Co Acquiring Platform
<b>ABORTED</b>	Final state. The transaction failed due to the customer abortion of the CVC Co payment process
<b>CANCELLED</b>	Final state. The transaction was cancelled by the Merchant for a refund of the authorized amounts (before the effective capture of the transaction)
<b>EXPIRED</b>	Final state. Initialized transaction expired
<b>CONFLICTED</b>	Abnormal transitional state. Calculated available amount was insufficient to consign transaction (transition to CONSIGNED state). Controls are in place on previous treatments but the state can be reached in a case of on an error occurring on these treatments. In such cases, manual operations must be managed by the technical team in order to resume the correct workflow of the transaction (capture for Merchant refund by the ANCV accounting department).

#### 6.2.1.6 subState

This field contains the actual sub-status of a transaction available for some specific transaction statuses.

It is an **optional** parameter.

The format is **String** with a constraint of restricted values:

Status code	subStatus Code	Description
-------------	----------------	-------------

PROCESSING	IN_ADJUSTMENT	The Customer must validate the amount to be paid using CVCo
	AUTHORIZATION_REQUEST	The transaction is being authorized with a Customer validation.
REJECTED	REJECTED_DEVICE	The transaction failed due to a lack of registered and non-blocked payment device for the Customer
	REJECTED_SECURITY	The transaction failed due to a reject during the Customer security checks
	REJECTED_TIMEOUT	The transaction failed due to a reject by timeout during the Customer required actions
	REJECTED_INTERNAL	The transaction failed due to a reject by the CVCo Acquiring Platform
	REJECTED_COMPLEMENT	The transaction failed due to a reject by the complementary acquiring service
ABORTED	ABORTED_TSPD	The transaction was aborted by the Customer during the CVCo payment process
	ABORTED_COMPLEMENT	The transaction was aborted by the Customer during the complementary payment process

#### 6.2.1.7 [expirationDate](#)

The expiration Date and Time of the transaction depending on its actual status and expected actions :

Status - subStatus	Actor	Expected Action	Max Delay	Status – subStatus when expired
INITIALIZED	SP	Request payment by payer	300s	EXPIRED
PROCESSING	Acquiring Platform	Perform Acquiring	100s	REJECTED – REJECTED_INTERNAL
PROCESSING – IN_ADJUSTMENT	Customer	Validate the amount	250s	REJECTED – REJECTED_TIMEOUT
PROCESSING – AUTHORIZATION_REQUEST	Customer	Authorize the payment	250s	REJECTED – REJECTED_TIMEOUT

For other statuses, the expirationDate is empty

It is an **optional** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

#### 6.2.1.8 [order](#)

This object contains the data of the order to be paid.

It is a **mandatory** parameter.

It is an **Object** (cf. [order](#))

#### 6.2.1.9 [paymentMethod](#)

This object contains the data on the payment method for this transaction.

It is a **mandatory** parameter.

It is an **Object** (cf. [paymentMethod](#))

#### 6.2.1.10 `redirectUrls`

This object contains the data of the return URL provided by the Merchant or SP.

It is an **optional** parameter.

It is an **Object** (cf. [redirectUrls](#))

#### 6.2.1.11 `payers`

This object list contains the data of the payers for part or whole order payment. It is only provided if a request for payment was accepted by the CVCo Acquiring Platform.

It is an **optional** parameter.

It is an **Object** list (cf. [payer](#))

#### 6.2.1.12 `refunds`

This object list contains the data of the scheduled refunds towards the merchant according to its contract for the payment means used.

It is an **optional** parameter that will only be provided and **mandatory** if the transaction is in status "PAID". At least one of the refunds will be of CVCo type.

It is an **Object** list (cf. [refund](#))

#### 6.2.1.13 `cancellation`

This object contains the data of the cancellation request on the transaction. It is only provided if a cancel transaction was accepted by the CVCo Acquiring Platform.

It is an **optional** parameter.

It is an **Object** (cf. [cancellation](#))

#### 6.2.1.14 `merchant`

##### `shopId`

This is the unique identifier of the merchant's shop in the Acquiring Platform for acceptance of payment in CVCo. It is provided by ANCV.

It is a **mandatory** parameter.

The format is **Long**.

*Example: 10000008*

##### `shopAssistantId`

This is an identifier of the sales machine.

It is an **optional** parameter.

The format is **String** with a constraint of 50 characters max

*Example: machine-71111*

##### `terminalId`

This is an identifier of the sales machine.

It is an **optional** parameter.

The format is **String** with a constraint of 50 characters max

*Example: machine-71111*

##### `serviceProviderId`

This is the unique identifier of the Service Provider in the Acquiring Platform for processing payment in CVCo on behalf of the merchant. It is provided by ANCV.

It is an **optional** parameter that will only be provided and **mandatory** if the processing is not done directly by a Merchant for its shop but by a service Provider.

The format is **Long**.

*Example: 20200113*

#### 6.2.1.15 order

id

This is the unique identifier of the order (cart reference) in the Merchant or SP referential.

It is a **mandatory** parameter.

The format is **String** with a constraint of 64 characters max

*Example: cart-54441*

paymentId

This is the unique identifier of the payment transaction for this order in the Merchant or SP referential.

It is a **mandatory** parameter.

The format is **String** with a constraint of 40 characters max

*Example: 90001*

label

This is the description of the order in the Merchant or SP referential.

It is an **optional** parameter.

The format is **String** with a constraint of 255 characters max

*Example: "2 nights booking"*

amount

This is the complete amount to pay for the order (a part of which will be paid in CVCo).

It is a **mandatory** parameter.

It is an **Object** (cf.8.2.1.3)

#### 6.2.1.16 amount

Generic object used to represent amounts in a chosen currency

total

It represents the total amount of money using the smallest currency (cents).

It is a **mandatory** parameter.

The format is **Integer**

*Example: 8000 (means 80 €)*

net

It represents the net amount of money using the smallest currency (cents) that is refund to the Merchant.

It is equal to the total amount minus the fee amount.

It is an **optional** parameter that will only be provided and **mandatory** if used from a refund object

The format is **Integer**

*Example: 200 (means 2 €)*

fee

It represents the amount of money using the smallest currency (cents) that is paid by the acceptor (Merchant) towards the scheme fee. For transaction using CVCo, this amount relates to the commission fee described in the contract between the Merchant and ANCV.

It is an **optional** parameter that will only be provided and **mandatory** if used from a refund object (cf, 8.2.1.8)

The format is **Integer**

Example: 7800 (means 78 €)

currency

It contains the currency to use for this amount, using ISO4217 codification.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Currency Code	Currency label
978	EUR	Euros

#### 6.2.1.17 paymentMethod

tspdMode

It contains the voucher payment method to be used by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Acquiring Method
001	Payment only with CVCo. The amount to be authorized can be adjusted.
002	Payment only with CVCo. The amount to be authorized can't be adjusted.

*The complementary payment processed by the ANCV is only available for Online payments using the ANCV PayPage and for Merchants under specific contract with ANCV.*

captureMode

Transaction payment capture mode (normal or deferred).

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
NORMAL	Normal payment : automatic validation with authorization
DEFERRED	Deferred payment : after authorization, the validation must be done by the merchant before a given capture date

captureDate

This is the Date and Time of the expected time for the execution of the deferred payment on transaction, and only provided if the captureMode is "DEFERRED".

The maximum authorized value for this parameter is of **6 calendar days** after the transaction's creationDate.

It is an **optional** parameter that must only be provided and **mandatory** if captureMode is "DEFERRED".

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

#### 6.2.1.18 redirectUrls

##### returnUrl

If provided, this is the URL that will be used by the CVCo Acquiring Platform to push the transaction status upon valid payment authorization.

It is an **optional** parameter.

The format is **String** with a constraint of 512 characters max

Example: <https://01net.fr/notif-payment>

##### cancelUrl

If provided, this is the URL that will be used by the CVCo Acquiring Platform to push the transaction status upon abortion or rejection of a payment authorization.

It is an optional parameter.

The format is **String** with a constraint of 512 characters max

Example: <https://01net.fr/notif-payment-cancel>

#### 6.2.1.19 payer

##### beneficiaryId

This is the identifier of the CVCo holder that was transmitted when requesting for payment.

It is a **mandatory** parameter.

The format is **String** which can be either:

- The beneficiary e-mail address linked with its CVCo account (max 254 characters)
- The CVCo account number (composed of 11 digits – Luhn coded)

##### amount

This is the amount that were requested to be paid by the beneficiary for the order (a part of which will be paid in CVCo). If no specific amount was specified when requesting the payment, then the full order amount is applied.

It is a **mandatory** parameter.

It is an **Object** (cf.8.2.1.3)

##### authorizations

This object list contains the data of the payment authorizations that were obtained by the CVCo Acquiring Platform for each payment type (CVCo, Bank Card ...). It is only provided if all the payment authorizations requested were delivered.

It is an **optional** parameter.

It is an **Object** list (cf. [authorization](#))

#### 6.2.1.20 authorization

Payment authorization is delivered for a given amount, using a given payment type and for an approved holder.

##### type

This is the payment type for which the authorization was delivered.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
CVCo	Payment using CVCo on ANCV scheme
VISA	Payment using Bank Card on VISA scheme



MASTERCARD	Payment using Bank Card on MasterCard scheme
CB	Payment using Bank Card on CB scheme

amount

This is the authorized amount to be paid by the holder.

It is a **mandatory** parameter.

It is an **Object** (cf.8.2.1.3)

number

This is the authorization number.

It is a **mandatory** parameter.

The format is **String**.

validationDate

This is the Date and Time of the authorization delivery.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations:  
yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

holder

This parameter contains masked data of the holder: 2 first digits and 4 last digits are visible; others are replaced by the '\*' character.

It is a **mandatory** parameter.

The format is **String**.

#### 6.2.1.21 refund

Data on the refund towards the Merchant that is scheduled by an acquiring scheme involved in the transaction. At least one of the refunds must be of CVCo type.

amount

This is the amount used for refund towards the Merchant.

It is a **mandatory** parameter.

It is an **Object** (cf.8.2.1.3)

effectiveDate

This is the Date and Time of the scheduled refund by the scheme towards the Merchant

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations:  
yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

type

This is the payment type indicating the scheme performing the refund.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
CVCo	Payment using CVCo on ANCV scheme
VISA	Payment using Bank Card on VISA scheme
MASTERCARD	Payment using Bank Card on MasterCard scheme
CB	Payment using Bank Card on CB scheme

#### 6.2.1.22 cancellation

effectiveDate

This is the Date and Time of the cancellation by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations:  
yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

reason

It is the coded reason for cancellation provided by the Merchant or Service Provider.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
COMPLEMENTARY_PAYMENT	Complementary payment was not obtained
CUSTOMER_ABORT	Customer chose to abort the payment process
OTHER	Any other cancellation reason

label

It is a description of the cancellation reason that can be provided by the Merchant or Service Provider.

It is an **optional** parameter.

The format is **String**.

*Example: "Customer asked for order refund"*

### 6.2.2 payer

#### 6.2.2.1 beneficiaryId

This is the CVCo identifier of the customer provided when choosing to pay using its CVCo account.

It is a **mandatory** parameter.

The format is **String** which can be either:

- The beneficiary e-mail address linked with its CVCo account
- The CVCo account number (composed of 11 digits – Luhn coded)

#### 6.2.2.2 amount

This is the amount requested to be paid by the beneficiary for the order (a part of which will be paid in CVCo). If no specific amount is specified, then the full order amount is applied.

The minimum amount that can be paid in CVCo is **1ct**, and the maximum amount is the order's amount.

It is an **optional** parameter.

It is an **Object**

### 6.2.3 pre-transaction

#### 6.2.3.1 id

This is the unique identifier for a given pre-transaction for payment request by CVCo.

It is a **mandatory** parameter.

The format is **String** with a constraint of alphanumeric characters only.

#### 6.2.3.2 validatedPaymentTransactionId

This is the unique identifier for the payment transaction in CVCo that was validated for the pre-transaction

It is an **optional** parameter.

The format is **String** with a constraint of alphanumeric characters only.

#### 6.2.3.3 merchant

This object contains the data of the Merchant which is the initiator of the payment pre-transaction.

It is a **mandatory** parameter.

It is an **Object** (cf. [merchant](#))

#### 6.2.3.4 creationDate

This is the Date and Time of the pre-transaction's creation by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'.

#### 6.2.3.5 updateDate

This is the Date and Time of the pre-transaction last update by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'.

#### 6.2.3.6 state

This field contains the actual status of a pre-transaction.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
<b>CREATED</b>	Initial State. The pre-transaction is created It can't be used yet to perform a payment transaction.
<b>PROCESSING</b>	Transitional state. The pre-transaction can be used to perform a payment transaction
<b>USED</b>	Final state. The pre-transaction was used for a payment transaction that was authorized. It can't be used anymore to perform a payment transaction.
<b>AUTHORIZING</b>	Transitional state. The pre-transaction is being authorized
<b>ABORTED</b>	Final state. The pre-transaction is cancelled due to either the merchant payment request abortion or the customer abortion of the CVCo payment process with no will to pay it later. It can't be used anymore to perform a payment transaction.
<b>EXPIRED</b>	Final state. The pre-transaction expired It can't be used anymore to perform a payment transaction.

#### 6.2.3.7 expirationDate

The Date and Time of the pre-transaction expected action from the Customer (TTL) before it expires.

The value can't exceed ANCV defined limit of 30 days after the creation date. Moreover, when associated with a situation where the QR Code is presented directly to a customer, the delay should not exceed a day (or even a few minutes if the use case permits it).

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations: yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

#### 6.2.3.8 order

This object contains the data of the order to be paid.

It is a **mandatory** parameter.

It is an **Object** (cf. [order](#))

#### 6.2.3.9 paymentMethod

This object contains the data on the payment method for this pre-transaction.

It is a **mandatory** parameter.

It is an **Object** (cf. [paymentMethod](#))

#### 6.2.3.10 redirectUrls

This object contains the data of the return URL provided by the Merchant or SP.

It is an **optional** parameter.

It is an **Object** (cf. [redirectUrl](#))

#### 6.2.3.11 abort

This object contains the data of the abortion of the pre-transaction.

It is an **optional** parameter that will only be provided and **mandatory** if the pre-transaction was aborted.

It is an **Object** (cf. [abort](#))

#### 6.2.3.12 merchant

shopId

This is the unique identifier of the merchant's shop in the Acquiring Platform for acceptance of payment in CVCo. It is provided by ANCV.

It is a **mandatory** parameter.

The format is **Long**.

Example: 10000008

shopAssistantId

This is an identifier of the salesperson or eCommerce site.

It is an **optional** parameter.

The format is **String** with a constraint of 50 characters max

Example: [01.net](#)

terminalId

This is an identifier of the sales machine.

It is an **optional** parameter.

The format is **String** with a constraint of 50 characters max

*Example: machine-71111*

serviceProviderId

This is the unique identifier of the Service Provider in the Acquiring Platform for processing payment in CVCo on behalf of the merchant. It is provided by ANCV.

It is an **optional** parameter that will only be provided and **mandatory** if the processing is not done directly by a Merchant for its shop but by a service Provider.

The format is **Long**.

*Example: 20200113*

#### 6.2.3.13 order

id

This is the unique identifier of the order (cart reference) in the Merchant or SP referential.

It is a **mandatory** parameter.

The format is **String** with a constraint of 64 characters max

*Example: cart-54441*

prePaymentId

This is the unique identifier of the pre-transaction for this order in the Merchant or SP referential.

It is an **optional** parameter, if no value is given, the prePaymentId will be set to "0".

The format is **String** with a constraint of 40 characters max

*Example: 90001*

label

This is the description of the order in the Merchant or SP referential.

It is an **optional** parameter.

The format is **String** with a constraint of 255 characters max

*Example: "2 nights booking"*

amount

This is the complete amount to pay for the order (a part of which will be paid in CVCo).

It is a **mandatory** parameter.

It is an **Object**

#### 6.2.3.14 amount

total

It represents the amount of money using the smallest currency (cents).

It is a **mandatory** parameter.

The format is **Integer**

*Example: 8000 (means 80 €)*

currency

It contains the currency to use for this amount, using ISO4217 codification.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Currency Code	Currency label
978	EUR	Euros

#### 6.2.3.15 paymentMethod

tspdMode

It contains the voucher payment method to be used by the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Acquiring Method
001	Payment only with CVCo. The amount to be authorized can be adjusted.
002	Payment only with CVCo. The amount to be authorized can't be adjusted.

captureMode

Transaction payment capture mode (normal or deferred).

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
NORMAL	Normal payment : automatic validation with authorization
DEFERRED	Deferred payment : after authorization, the validation must be done by the merchant before a given capture date

captureTerm

This is the expected time for the execution of the deferred payment on transaction after it is authorized and only provided if the captureMode is "DEFERRED".

This parameter represents Calendar Days and must be between **1** and **6**

It is an **optional** parameter that must only be provided and **mandatory** if captureMode is "DEFERRED".

The format is **Integer**

#### 6.2.3.16 redirectUrls

returnUrl

If provided, this is the URL that will be used by the CVCo Acquiring Platform to push the transaction status linked with this pre-transaction upon valid payment authorization.

It is an **optional** parameter.

The format is **String** with a constraint of 512 characters max

Example: <https://01net.fr/notif-payment>

cancelUrl

If provided, this is the URL that will be used by the CVCo Acquiring Platform to push the pre-transaction status upon abortion by the merchant or customer, or expiration.

It is an **optional** parameter.

The format is **String** with a constraint of 512 characters max

Example: <https://01net.fr/notif-payment-cancel>

#### 6.2.3.17 abort

effectiveDate

This is the Date and Time of the abortion in the CVCo Acquiring Platform.

It is a **mandatory** parameter.

The format is **Date and Time with Time zone** according to ISO8601 recommendations:  
yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

reason

It is the coded reason for abortion depending on its origin.

It is a **mandatory** parameter.

The format is **String** with a constraint of restricted values:

Value	Description
ABORTED_BENEFICIARY	The pre-transaction was aborted by the Customer during the payment process with no will to pay for it later
ABORTED_MERCHANT	The pre-transaction was aborted by the Merchant before a payment transaction was authorized and consigned for it

label

It is a description of the abortion reason that can be provided by the Merchant or Service Provider.

It is an **optional** parameter.

The format is **String**.

Example: "Customer asked for order refund"

## 6.2.4 applicationContext

### 6.2.4.1 returnContext

It can receive an inner Context for the Merchant or Service Provider to be returned with the response.

It is an **optional** parameter.

The format is **String** with a constraint of 255 characters max

### 6.2.4.2 customerId

It can be used to store the internal customer's ID in the Merchant or Service Provider referential.

It is an **optional** parameter.

The format is **String** with a constraint of 34 characters max using only:

- alphabetic characters : aA-zZ
- digits : 0-9

Example: clientJohnDoe01

## 6.3 Reject codes and root cause

Error messages are only provided in English.

HTTP Status	errorCode	errorMessage
400 Bad Request	BAD_REQUEST	Bad request
403 Forbidden	INVALID_SEAL	The seal is invalid
403 Forbidden	MERCHANT_NOT_ALLOWED	The merchant is not allowed

403 Forbidden	OPERATION_TRANSACTION_NOT_ALLOWED	The operation on transaction is not allowed
403 Forbidden	OPERATION_TRANSACTION_NOT_ALLOWED	The operation on transaction is not allowed
403 Forbidden	OPERATION_PRE_TRANSACTION_NOT_ALLOWED	The operation on pre-transaction amount is not allowed
403 Forbidden	INSUFFICIENT_BALANCE	The balance is insufficient
403 Forbidden	NO_TA_TRANSACTION_PENDING	No TA transaction is pending for this authorization
403 Forbidden	TRANSACTION_NOT_VALIDATED_ON_TA_SIDE	Transaction was not validated on TA side
403 Forbidden	TRANSACTION_VALIDATED_ON_TA_SIDE	Transaction was already validated on TA side
404 Not Found	TRANSACTION_NOT_FOUND	The transaction was not found
404 Not Found	BENEFICIARY_NOT_FOUND	The beneficiary was not found
404 Not Found	PRE_TRANSACTION_NOT_FOUND	The pre-transaction was not found
404 Not Found	POINT_OF_SALE_NOT_FOUND	The shopID is not recognized as a known point of sale
409 Conflict	Not Acceptable detail	Could not find acceptable representation
412 Precondition Failed	OTHER_TRANSACTION_PENDING	Another transaction is pending
412 Precondition Failed	INVALID_TRANSACTION_AMOUNT	The transaction amount is invalid
412 Precondition Failed	INVALID_TRANSACTION_CURRENCY	The transaction currency is invalid
412 Precondition Failed	INVALID_PAYER_AMOUNT	The payer amount is invalid
412 Precondition Failed	INVALID_TSPD_MODE	The TSPD mode amount is invalid
412 Precondition Failed	MISSING_CAPTURE_DATE	The capture date is mandatory for deferred capture mode
412 Precondition Failed	INVALID_CAPTURE_DATE	The capture date is invalid
412 Precondition Failed	MISSING_CAPTURE_TERM	The capture term is mandatory for deferred capture mode



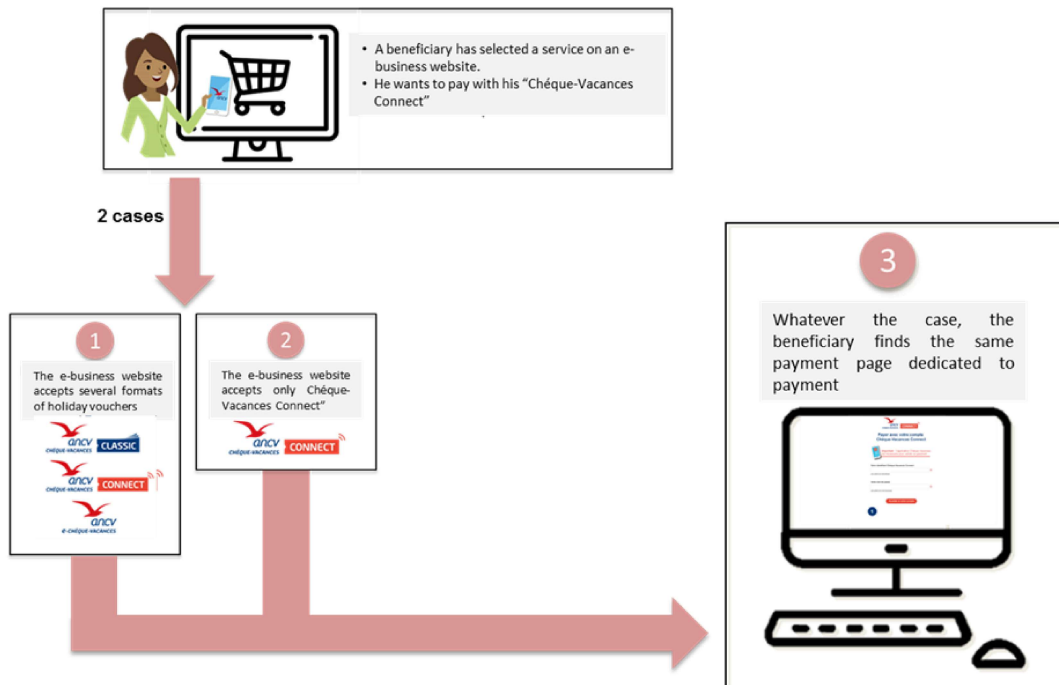
412 Precondition Failed	INVALID_CAPTURE_TERM	The capture term is invalid
412 Precondition Failed	INVALID_EXPIRATION_DATE	The expiration date is invalid
412 Precondition Failed	TRANSACTION_EXPIRED	The transaction has expired
412 Precondition Failed	PRE_TRANSACTION_EXPIRED	The pre-transaction has expired
412 Precondition Failed	INVALID_PRE_TRANSACTION_AMOUNT	The pre-transaction amount is invalid
412 Precondition Failed	INVALID_PRE_TRANSACTION_CURRENCY	The pre-transaction currency is invalid
412 Precondition Failed	VALIDATION_DEADLINE_EXCEEDED	The validation deadline has been exceeded
412 Precondition Failed	INVALID_PAYERS_AUTHORIZATIONS	Payers' authorizations are invalid
500 Server error	NO_ACTIVE_DEVICE	The beneficiary has no active devices
502 Bad Gateway	INTERNAL_SERVER_ERROR	internal server error
502 Bad Gateway	TA_COMMUNICATION_ERROR	Failed to communicate with TA

Other error codes may be received by the integrator in the case of a malfunction between the integrator and the CVCo Platform. In these cases, refer to the online documentation (i.e. [https://en.wikipedia.org/wiki/List\\_of\\_HTTP\\_status\\_codes](https://en.wikipedia.org/wiki/List_of_HTTP_status_codes))

## 6.4 Graphic charter payment pages

Ensure the best user experience by:

- Offering a similar path regardless of the payment page
- Harmonizing the design of payment pages developed by our payment services partners and software publishers



#### 6.4.1 Case #1 - Several formats are accepted

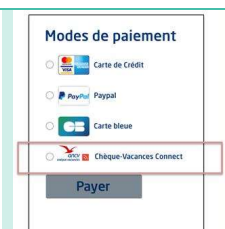
<p><b>Step 1:</b> selection of the payment method in case the professional of tourism and leisure accepts <b>several formats of the Chèque-Vacances</b></p> <ul style="list-style-type: none"> <li>The logo to appear on the payment method choice page is Chèque-Vacances.</li> <li>In case where the means of payment logos are accompanied by a descriptive text, "<b>Chèque-Vacances</b>" must be written in full.</li> </ul>	
<p><b>Step 2:</b> Select the type of « Chèque-Vacances » - by clicking on the « Chèque-Vacances » button, you will access a new page to select the format of Chèques-Vacances available to the beneficiary.</p> <ul style="list-style-type: none"> <li>Description text of the action to be performed: « Sélectionnez le format de vos Chèques-Vacances » ("Select the format of your Chèques-Vacances")</li> <li>Logos used must respect the rules of use of the Graphic Charter of the ANCV</li> <li>In case of a format less than or equal to 15mm or 100 pixels, reduced versions of the logos Chèque-Vacances Connect, and Chèque-Vacances Classic must be used</li> </ul>	
<p><b>Step 3:</b> choice other than Chèque-Vacances Connect</p> <ul style="list-style-type: none"> <li>Clicking on the "Chèque-Vacances Classic" button will take you to a page describing the procedure for accepting Chèques-Vacances Classic (printed form)</li> <li>Click on the "eChèque-Vacances" button to go to a description page of the product acceptance procedure eChèques-Vacances</li> </ul>	

## 6.4.2 Case #2 - Only the Chèque-Vacances Connect is accepted

**Only one Step:** Select the method of payment "Chèque-Vacances Connect"

In the case where the professional of tourist and leisure only accepts the Chèque-Vacances Connect. The logo to be displayed on the page of choice of the method of payment is that of Chèque-Vacances Connect.

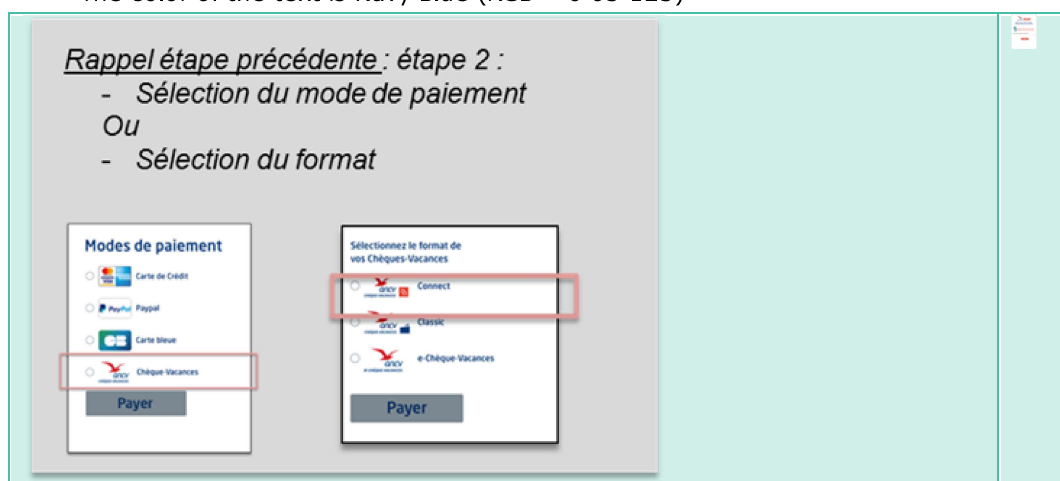
- In case where the means of payment logos are accompanied by a descriptive text, "Chèque-Vacances Connect" must be written in full.
- Logos used must respect the rules of use of the Graphic Charter of the ANCV
- In case of a format less than or equal to 15mm or 100 pixels, reduced versions of the logos Chèque-Vacances Connect



## 6.4.3 Next step is common to both cases - access to the payment page

Format selection "**Chèque-Vacances Connect**" goes to the login page of the following ANCV payment space

- The building blocks of this page are available online (graphic references)
- The color of the button and the bottom of the message is Coral (RGB = 230 76 64)
- The color of the text is Navy Blue (RGB = 0 63 125)



A holder of "Chèque-Vacances Classic" (printed form) could use that payment page. We recommend highlighting the CVCo payment option by offering more detailed information and easy access to our Exchange zone.

(Exchange zone launched in January 2021)

- Text box:

You are an existing « Chèques-Vacances » beneficiary holder of "Chèque-Vacances Classic" (printed form) or "e-Chèque-Vacances".

Find out all the "Chèque-Vacances Connect" (CVCo) advantages, a dematerialized payment solution.

Via "Chèque-Vacances" mobile application, you can now pay your leisure and travels services to the nearest cent, as well as on an e-commerce website or from one of our partner stores in the network.

For more information on the "Chèque-Vacances Connect", visit our website dedicated to you by [clicking this link](#).

[The above underscored string is the hyperlink support opening a new browser tab]

- Current URL set: <http://cheque-vacances-connect.com/collaborateur/>

In order to identify its CVCo Account, the Consumer can use either:


- The email address used with her/his CVCo account.
- Her/his CVCo ID.

The field "identifiant" must use the following regex:

If the input is not a numeric, then it must consist of an email with at least a format check: "[xxxx]@[xxxxx].[xxx]".

If the input is numeric, then 11 digits are required with a Luhn check.

Following **transaction success**, a success page is displayed.




Pictogram in the folder available at the link below  
Summarizing requested and received amounts, merchant name  
transaction reference.

**Rules:**

- If the requested amount equals received amount, then remaining amount is not displayed
- If the requested amount does not equal the amount of the requested amount, then remaining amount amounts to requested amount minus received amount

Following **transaction failure**, a failure page is displayed.



Pictogram in the folder available at the link below

**Rules:**

- If the requested amount equals received amount, then remaining amount is not displayed
- If the requested amount does not equal the amount of the requested amount, then remaining amount amounts to requested amount minus received amount

**Note:** All page templates (GIF and PSD), text to use in tooltips and logos used in the pages are available online at following link.

<https://static.ancv.com/ddmc/connect/paypage/paypage.zip>

#### 6.4.4 Specific behavior for some reject causes

When using the Acquiring Platform messages, some of the reject situations (cf. [Reject codes and root cause](#)) can be caused by a human error to be indicated with explicit messages for the Consumer to have it understood and facilitate her/his reaction in order to finalize the payment.

In case of request payment on transaction by a payer (cf. [Request Payment on Transaction by a payer](#)) some information must be displayed according to the received HTTP status:

http status	errorCode	Message to display
403 Forbidden	INSUFFICIENT_BALANCE	This Chèque-Vacances Connect Account has an insufficient balance.

		<u>Balance calculation for a Beneficiary takes into accounts the following elements :</u> <ul style="list-style-type: none"> <li>• Transactions waiting merchant validation are deducted from the available balance</li> <li>• Available balance is the sum of the remaining amounts from the endowments with a validity date that has not been reached when : <ul style="list-style-type: none"> <li>• Payment request by the merchant (when the beneficiary notification is sent) has been made, or when a QR Code has been scanned</li> <li>• Payment has been authorized by the beneficiary</li> </ul> </li> </ul>
404 Not Found	BENEFICIARY_NOT_FOUND	There is no existing Chèque-Vacances Connect Account for this ID You must have a Chèque-Vacances Connect Account to pay using Chèque-Vacances Connect. To learn more, <a href="#">click here</a> . ( <i>&lt;- insert the link to the help page: <a href="http://static.ancv.com/ddmc/connect/aide.html">http://static.ancv.com/ddmc/connect/aide.html</a> )</i> )
409 Conflict	OTHER_TRANSACTION_PENDING	There is a pending transaction for this Chèque-Vacances Connect Account. Please finalize or cancel the pending transaction before you can perform this one.
412 Precondition Failed	NO_ACTIVE_DEVICE	You don't have any registered and active device with the Chèque-Vacances Application

When controlling the transaction state after this request and in case of reject, some information must be displayed according to the transaction state:

Status code	Sub-status code	Message to display
REJECTED	REJECTED_DEVICE	You don't have any registered and active device with the Chèque-Vacances Application
REJECTED	REJECTED_SECURITY	The payment was not completed because the personal code you entered was incorrect.
REJECTED	REJECTED_TIMEOUT	The payment was not completed within the time limit. The operation was cancelled.

## 6.5 Route and graphic charter requested for the implementation of the Connect Holiday Vouchers on terminals and payment terminals

This chapter details the route and the graphic charter requested by the ANCV for the Chèques-Vacances Connect in order to offer a similar route regardless of the Payment Service Provider, the Software Publishers or the merchant operating the payment.

3 cases can be encountered:

- Case #1: Display of the QR code on the receipt
- Case #2: Payment on an automatic kiosk
- Case #3: Payment on a payment terminal

### 6.5.1 Integration of the payment QR Code in the receipt

Display of a QR code with ANCV logo in the center.

Our API returns this QR Code corresponding to the requested payment in the form of an image.



## 6.5.2 Integration of the payment QR Code on an interactive kiosk

### 6.5.2.1 Automatic kiosk: Choice of payment method

The logo to appear on the payment method selection page is that of the Chèque-Vacances Connect.

- If the logos of payment methods are accompanied by a text description, « Chèque-Vacances Connect » must be written in full
- The logos used must respect the rules of use of the ANCV Graphic Charter
- In the case of a size smaller than or equal to 15 mm or 100 pixels, the smaller versions of Chèque-Vacances Connect and Chèque-Vacances Classic logos to be used



#### 6.5.2.2 Automatic kiosk: Payment request

Display of a QR code with ANCV logo in the center.

Our API returns this QR Code corresponding to the requested payment in the form of an image.

The user is accompanied in his buying journey with the display of the sentence: "Scannez ce QR Code de paiement avec votre application Chèque-Vacances".



#### 6.5.2.3 Automatic kiosk: Confirmation of payment

Management rules :

- If Amount requested = Amount received => no display of "Balance due"
- If Amount requested < Amount received => display of "Balance due" = Amount requested - Amount received



#### 6.5.2.4 Automatic kiosk: Payment error

The messages to display to the user according to the cause of the error, are listed in [Specific behavior for some reject causes](#).



### 6.5.3 Integration of the payment QR Code on a payment terminal

#### 6.5.3.1 Payment terminal: Payment request

Display of a QR code with ANCV logo in the center.

Our API returns this QR Code corresponding to the requested payment in the form of an image.

The user is accompanied in his buying journey with the display of the sentence: "Scannez ce QR Code de paiement avec votre application Chèque-Vacances".





### 6.5.3.2 Payment terminal: Payment confirmation

Management rules :

- If Amount requested = Amount received => no display of "Balance due"
- If Amount requested < Amount received => display of "Balance due" = Amount requested - Amount received



### 6.5.3.3 Payment terminal: Payment error



**Note:** All the pages (in GIF and PSD format), the text to be included in the help bubble and the logos used on these pages are available online in the zipped file accessible below.

<https://static.ancv.com/ddmc/connect/paypage/paypage.zip>

## 6.6 SlipId number

SlipId is usually composed of 8 numerical digits (constraint: maximum 15 alphanumeric characters)

## 6.7 Seal calculation

In order to succeed a request to a payment API, you must apply **HmacSHA256** algorithm over you sealing key.

Then, apply **Base64** encoding over the previous result and the concatenation of fields specified in section **"Ordered list of Fields for sealing"**.

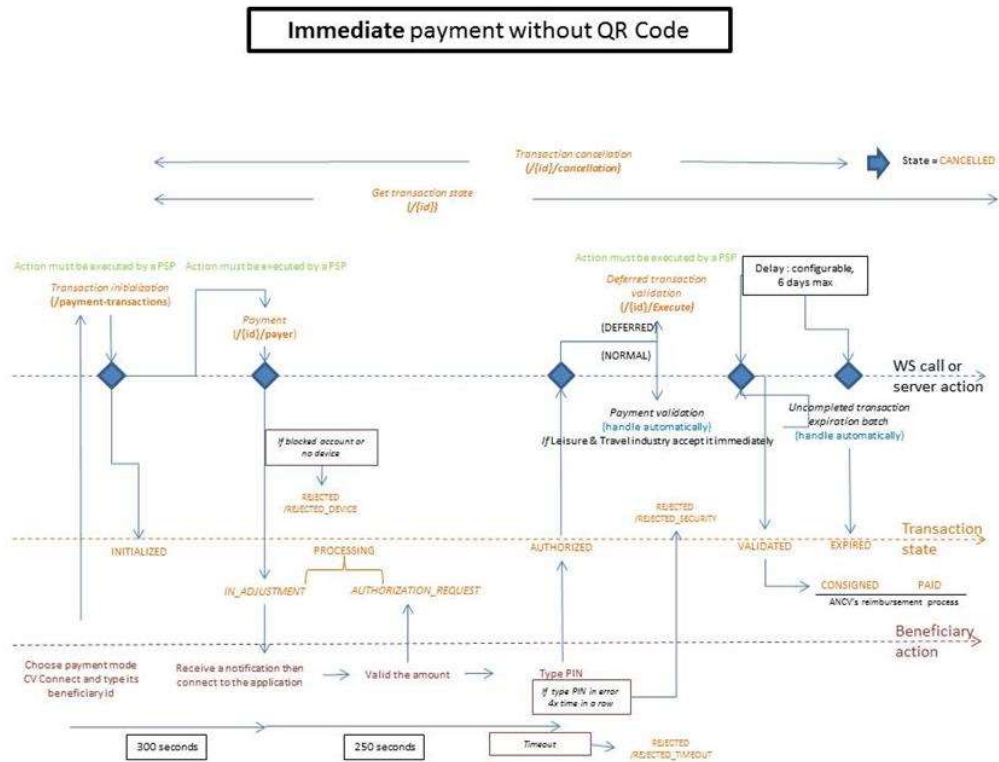
Sealing calculation example in Java in order to **initialize a payment transaction**

```
// The sealing key
String secretKey = "663768ff68ad8ea6768bbf65163e9b0a";
// Fields values concatenate with an ampersand
// In that example, shopId, serviceProviderId, orderId, paymentId and
total
String messageToHash = "10000065&100016&panier-33455&42556&500";
Mac sha256HMAC = Mac.getInstance("HmacSHA256");
sha256HMAC.init(new SecretKeySpec(secretKey.getBytes("UTF-8"),
"HmacSHA256"));
Base64.encodeBase64URLSafeString(sha256HMAC.doFinal(messageToHash.getBytes
("UTF-8")));
```

That should return : **mfy6VhbdyiErpfvQ3AvnKwU39W\_ae9MfuaVurEg-KjE**

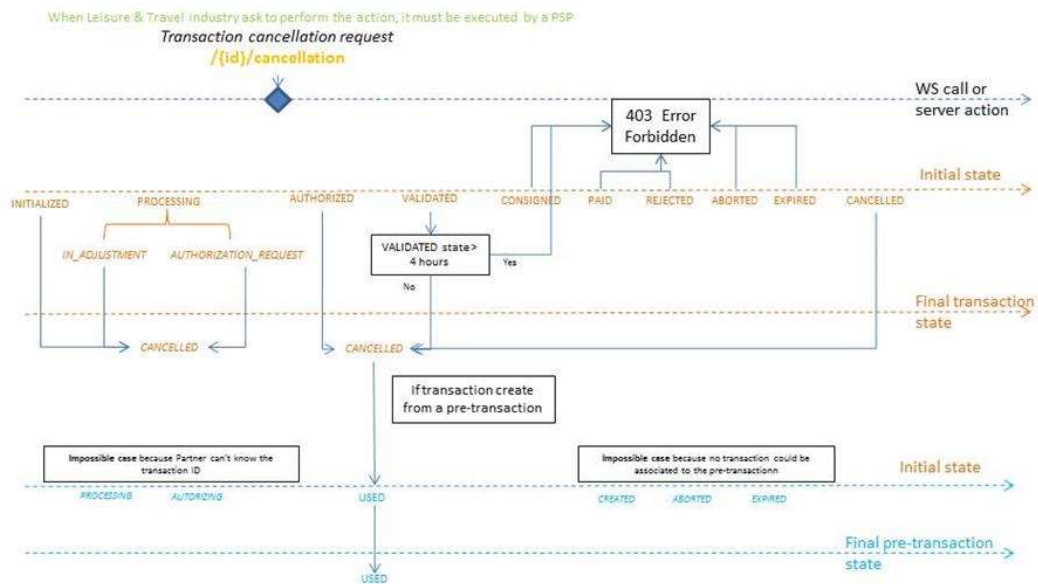
## 6.8 Timeline

### 6.8.1 Immediate payment without QR Code

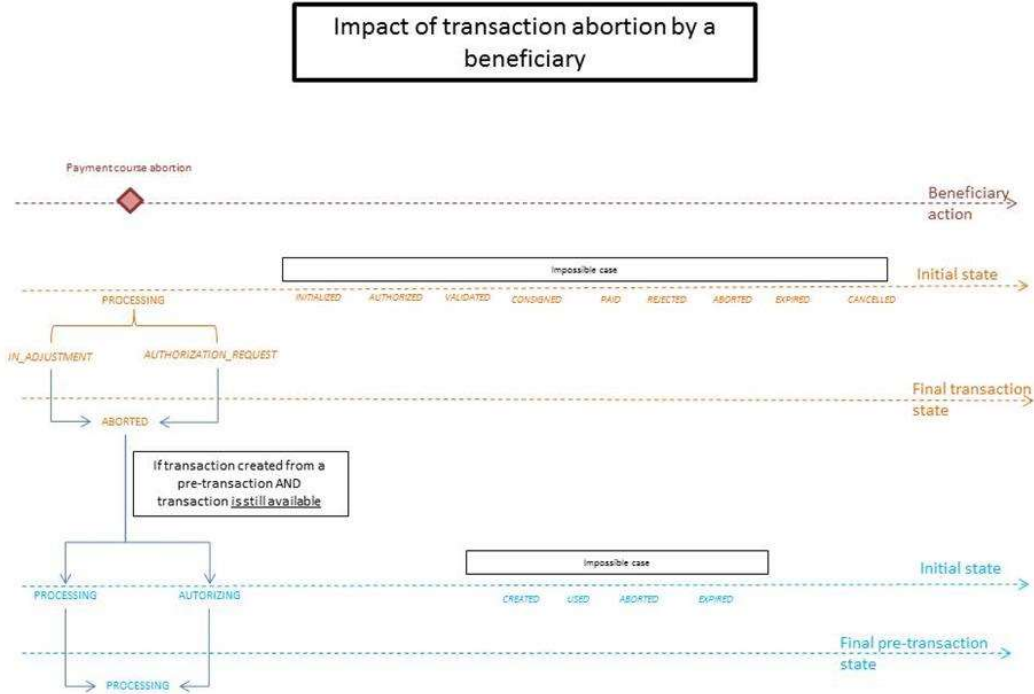


## 6.8.2 Impact of transaction cancellation by a Partner

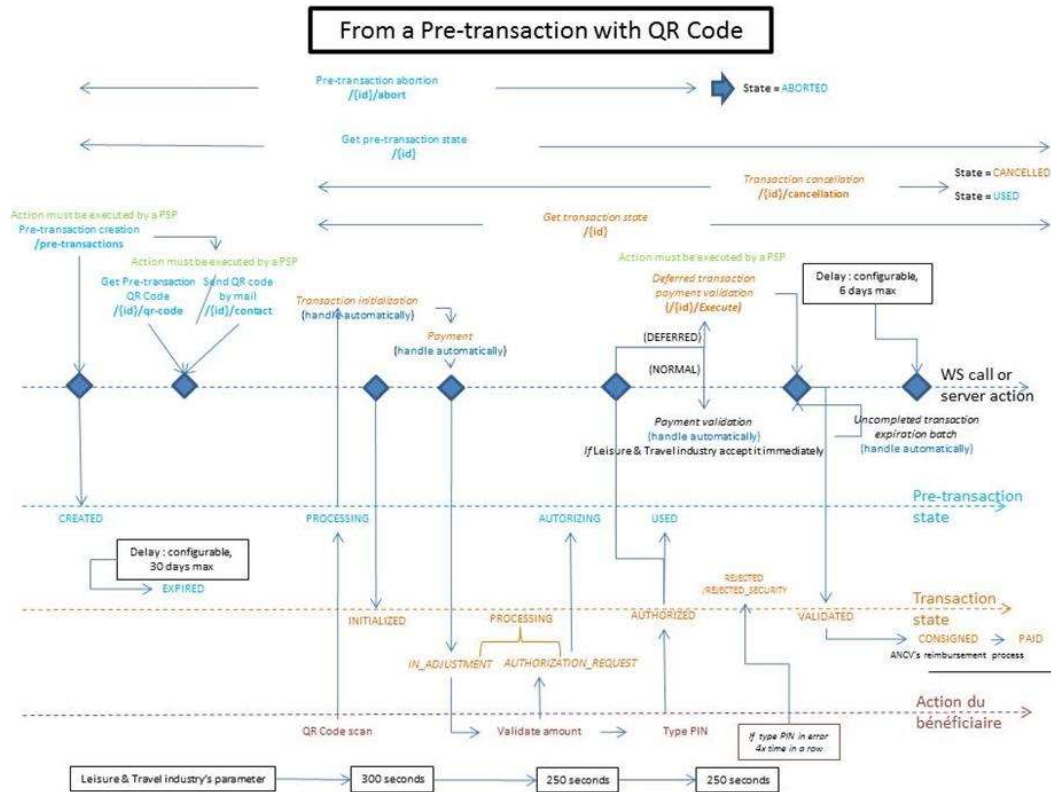
### Impact of transaction cancellation by a Partner



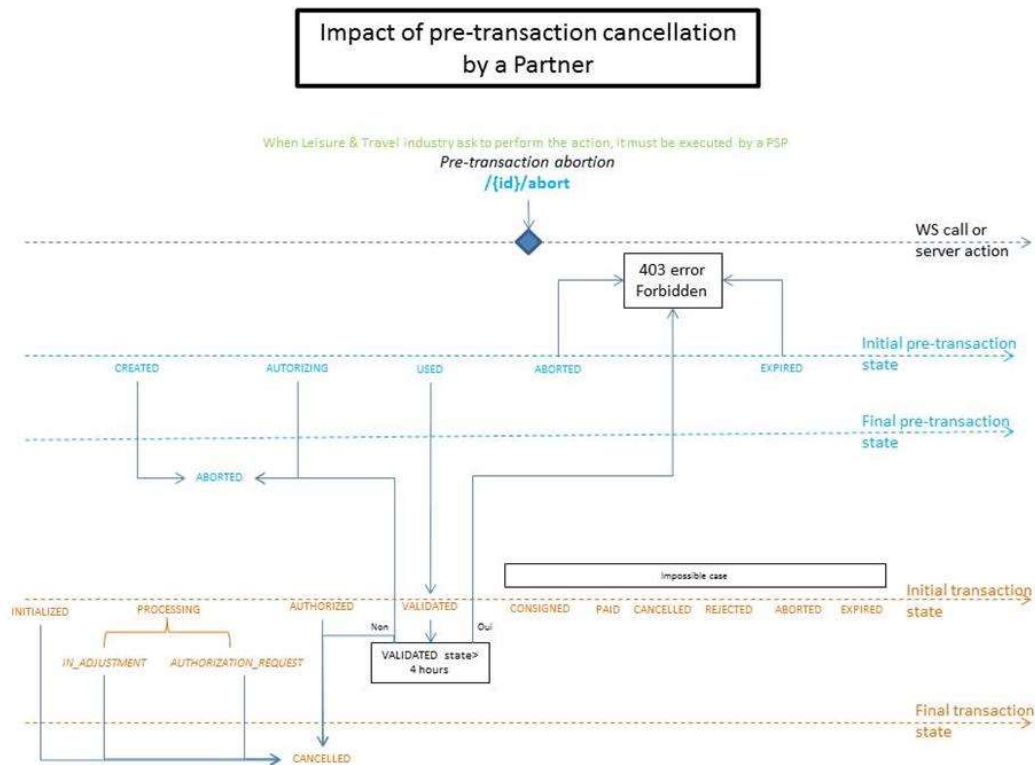
6.8.3 Impact of transaction abortion by a beneficiary



## 6.8.4 From a Pre-Transaction with QR Code



### 6.8.5 Impact of a pre-transaction cancellation by a Partner



## 6.9 Acceptance test plan

Not yet released

## 6.10 Application installation

See DR3