Network Partner API v1

Table of Contents

Chapter 1. Overview

The application programming interface for network partners (Network Partner API) allows communication with the eBill infrastructure on the "biller side". It is therefore the central entry point for billers.

The eBill service comprises of electronic bills, reminders, credit notes and advices which are summarized under the term "business case". Business cases are delivered from network partners to the SIX eBill infrastructure and can be received online by bill recipients. The eBill infrastructure is the runtime system at SIX. Core functionality is the management of system participants and the processing of business cases. It consists of software and hardware needed to provide the entire service.

Documentation of the Network Partner API comprises of three parts:

Handbook for network partners
Business-level description targeting product and IT management.

Network Partner API documentation
Technical description targeting IT architects and developers.

OpenAPI Specification
Technical specification of the interface for developers and code generators.

1.1. General Note

SIX reserves the right to amend this documentation as required within the scope of the applicable contractual conditions.

All rights are reserved with respect to this documentation, including with regard to photocopying and storage on electronic media as well as translation into foreign languages.

This documentation has been compiled with the greatest care, but may nevertheless contain errors or inaccuracies. SIX cannot assume any legal responsibility or any liability for erroneous information or its consequences.

If you notice any errors in this documentation or have any suggestions for improvements, we would be grateful to receive your feedback.

Contact
SIX
eBill & Direct Debit Support
Pfingstweidstrasse 110
8005 Zürich
Switzerland

Phone: 0041 58 399 4800

1.2. Version information

Version : 1.7.0

1.3. Contact information

Contact : SIX eBill & Direct Debit Support
Contact Email : support.billing-payments@six-group.com

1.4. URI scheme

Host : api.six-group.com
BasePath : /api/pns/networkpartner/v1
Schemes : HTTPS

1.5. Tags

- billers : **Biller Management**
  All operations that are associated with a biller are located within the biller resource, including the creation of business cases.
  The operations are designed to be self contained, because of this, the data objects can be rather large. However this allows for complete validation and avoids chains of calls that depend on each other.
- bill-recipient-subscription : **Bill Recipient Subscription**
  View 'bill recipient subscription requests' sent to the biller, using the subscription token on the subscription url from the biller. More information about the subscription process can be found in Section 3.3.3, “Subscriptions and Subscription Cancellations”.
- events : **Notification Events**
  The event resource allows the network partner to retrieve all changes addressed to it. There is one specific operation for every type of event (for example: Subscription status changes or business case status changes), where the network partner can pull new changes from.

Guidelines for Polling the Events
Depending on the network partner, different types of events may be of interest. We recommend the network partner to consume those events of interest on a regular basis.
Generally it is recommended to poll each events endpoint once per hour, whereas it’s important to consume all events until the endpoint does not return any more events.
Please avoid querying events unnecessary often.

- sectors : **Industry Sector**
  Industry sectors are valid system wide. Each biller will reference one or several industry sectors.
- healthcheck : **System Healthcheck**
  This allows to check the basic state of the system (can it be reached, does authentication and auhorization work, does it respond).
  As some infrastructures block certain HTTP methods by default, the healthcheck allows to test if

all required methods (GET, POST, PUT, DELETE) work across all layers.

1.6. Documentation Structure

This documentation consists of the following sections:

Overview
Overview of the document.

Introduction
High-level information about the eBill service with roles, interfaces, business introduction.

General Documentation
Basic design principles, general concepts applied during API and model design like multilingual support, event and error handling.

Resources
Describes the endpoints of the API. Generated from the Swagger definition and enriched with additional information.

Definitions
Request and Response definitions. Generated from the Swagger definition.

Chapter 2. Introduction

The main goal of the Network Partner API is to offer network partners an easy to use interface to deliver electronic invoices in the name of their customers (billers) to the eBill infrastructure. Electronic invoices delivered to this channel target online consumers on the financial institution / online banking side.

System participants of the eBill service are:

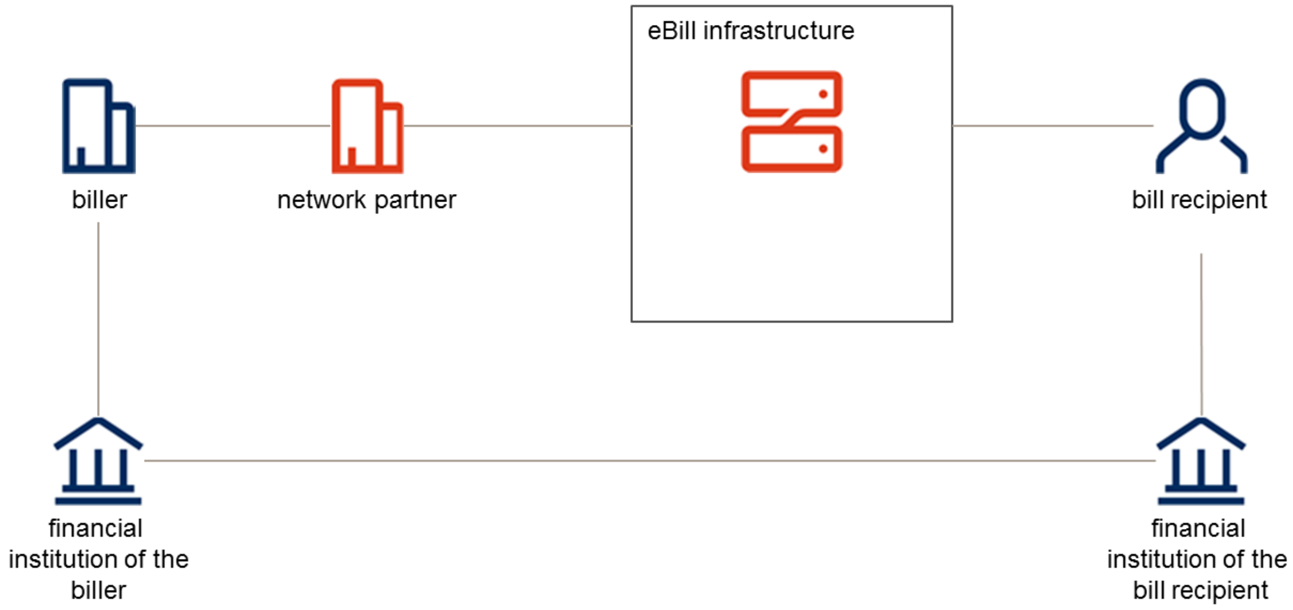

eBill infrastructure
biller
network partner
bill recipient
financial institution of the biller
financial institution of the bill recipient

Figure 1. Participants of the eBill service

- **Biller** (creditor) is a system participant that uses the eBill service to send electronic invoices to bill recipients.
- **Network partners** offer the eBill service to billers. A system participant that delivers electronic invoices to the eBill infrastructure.
- **eBill infrastructure** offers the eBill service of SIX to network partners and financial institutions. The eBill infrastructure manages master data, receives and processes business cases, creates payment instructions and sends them to financial institutions. Furthermore it offers an Events endpoint which consumers can poll to get information on changes that occurred asynchronously.
- **Financial institution of the bill recipient** offers the eBill service to its customers (bill recipients). It provides banking services including the processing of payment instructions generated by the eBill infrastructure.
- **Bill recipients** get access to the electronic invoices while using the online banking functionality of their financial institution.
- **Financial institution of a biller** receive and book the payment on behalf of the biller. There is no direct relation to the eBill infrastructure.

2.1. Scope

This documentation focuses on the eBill service, which offers electronic invoices that can be viewed and processed by bill recipients using the eBill portal.
The documentation is specifically targeting network partners and does not include details for billers and financial institutions.

Other services of SIX, namely direct debit service ("LSV"), "E-Rechnung EDI" and "E-Rechnung Workflow" are not in scope.

2.2. System Interfaces

System participants use different interfaces to communicate with the eBill infrastructure. The most important interfaces are described below:

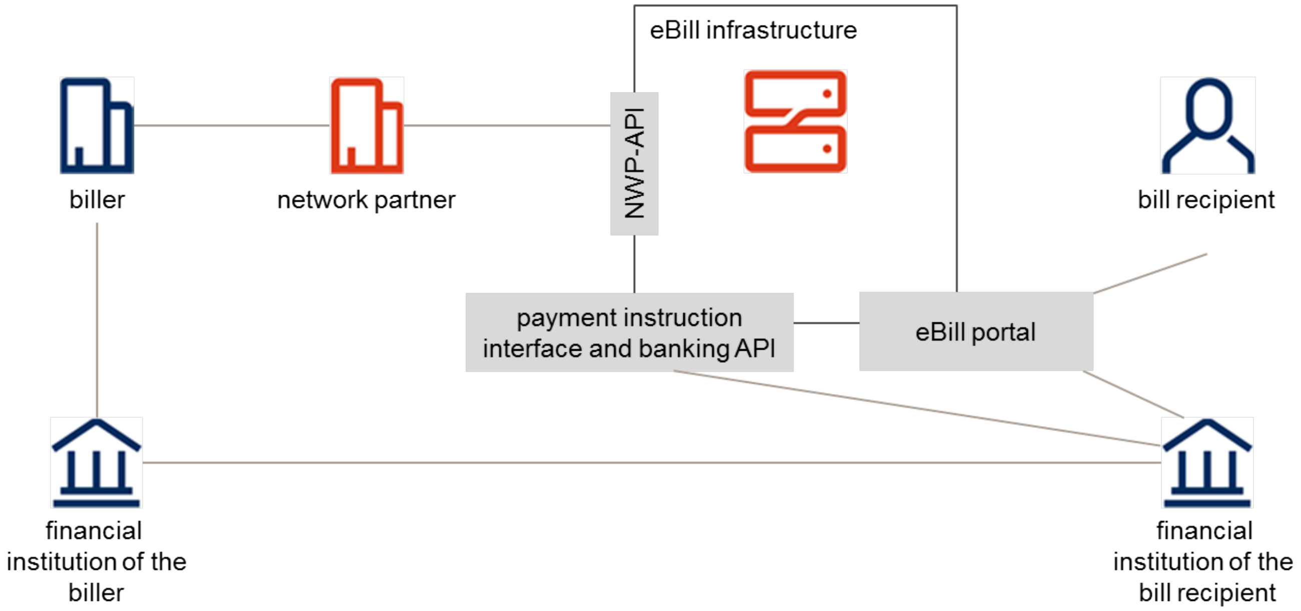

eBill infrastructure
NWP-API
biller
network partner
bill recipient
payment instruction interface and banking API
eBill portal
financial institution of the biller
financial institution of the bill recipient

Figure 2. Interfaces of the eBill infrastructure

- **Network Partner API** allows network partners and - indirectly - the billers to interact with the system.
- **Banking API** is a webservice interface for financial institutions on the bill recipient side. It e.g. allows for single sign on with the eBill infrastructure.
- **Payment instruction interface** is an asynchronous communication channel to send payment instruction messages (pain.001) to financial institutions and receive status report messages (pain.002) from financial institutions.
- **eBill portal** is a central web application that can be used by all participating financial institutions. It allows bill recipients to use the eBill functionality in the web. Access to the eBill portal is always initiated from an online banking session.

NOTE	Network partners can offer additional services to their customers, e.g. a web portal for billers. This kind of additional functionality is not part of the service offering of SIX and therefore not depicted above.

2.3. Primary Network Partners

The eBill infrastructure does not restrict billers to work with a single network partner. Specifically, it is possible for a biller to deliver business cases through several network partners.

However, to allow secure data management there are some restrictions and it is necessary that each biller assigns one network partner as primary network partner.

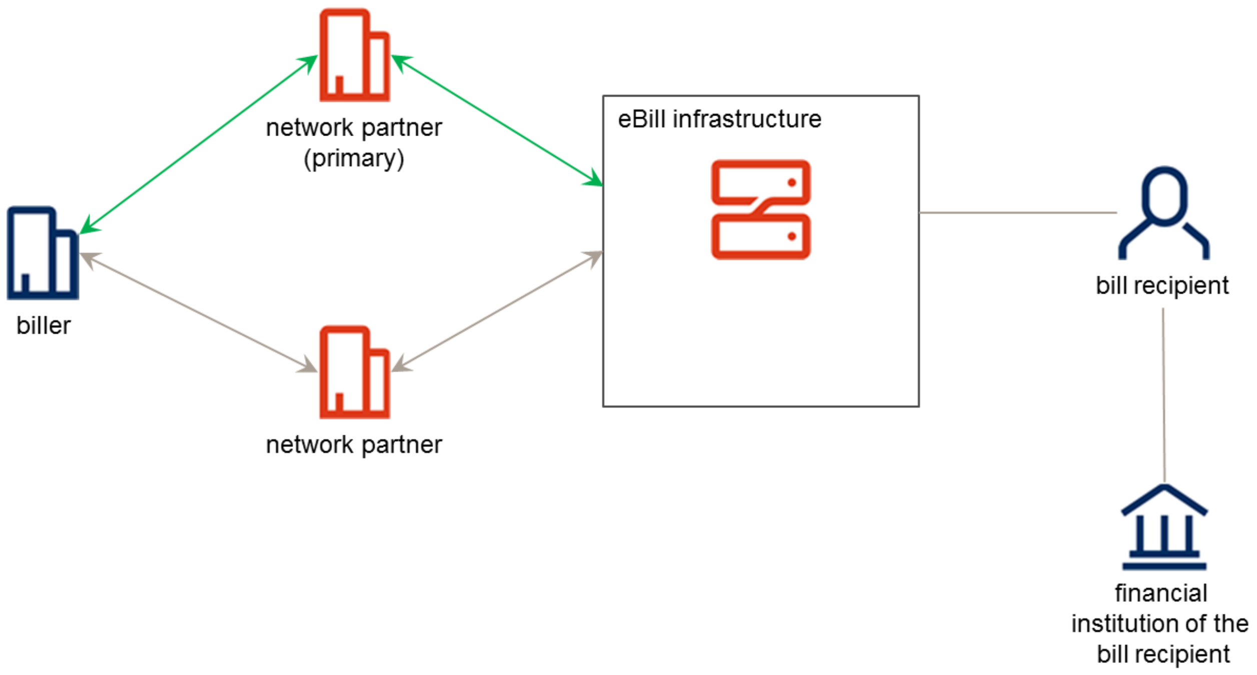

network partner (primary)
eBill infrastructure
biller
bill recipient
network partner
financial institution of the bill recipient

Figure 3. Network Partner and Primary Network Partner

The following functionality of the Network Partner API must be executed by the primary network partner (green above):

- Creation of new billers in the eBill infrastructure. The creation of a biller will assign the registering network partner as primary network partner.
- Management of biller master data and bill attachments of a biller.
- Subscriptions and cancellation of subscriptions of bill recipients with billers.

After a biller was created by his primary network partner, the eBill infrastructure allows to deliver business cases using other network partners, too (grey above). The necessary contractual agreements and technical setup have to be completed between biller and network partner.

2.4. Billing and Payment Process Overview

An entire billing and payment process using the eBill infrastructure is shown in the following overview:

NOTE Roles within a colored area can be represented by the same party. Example: Often, the bill recipient will be the same party as the one finally paying the bill with his account.

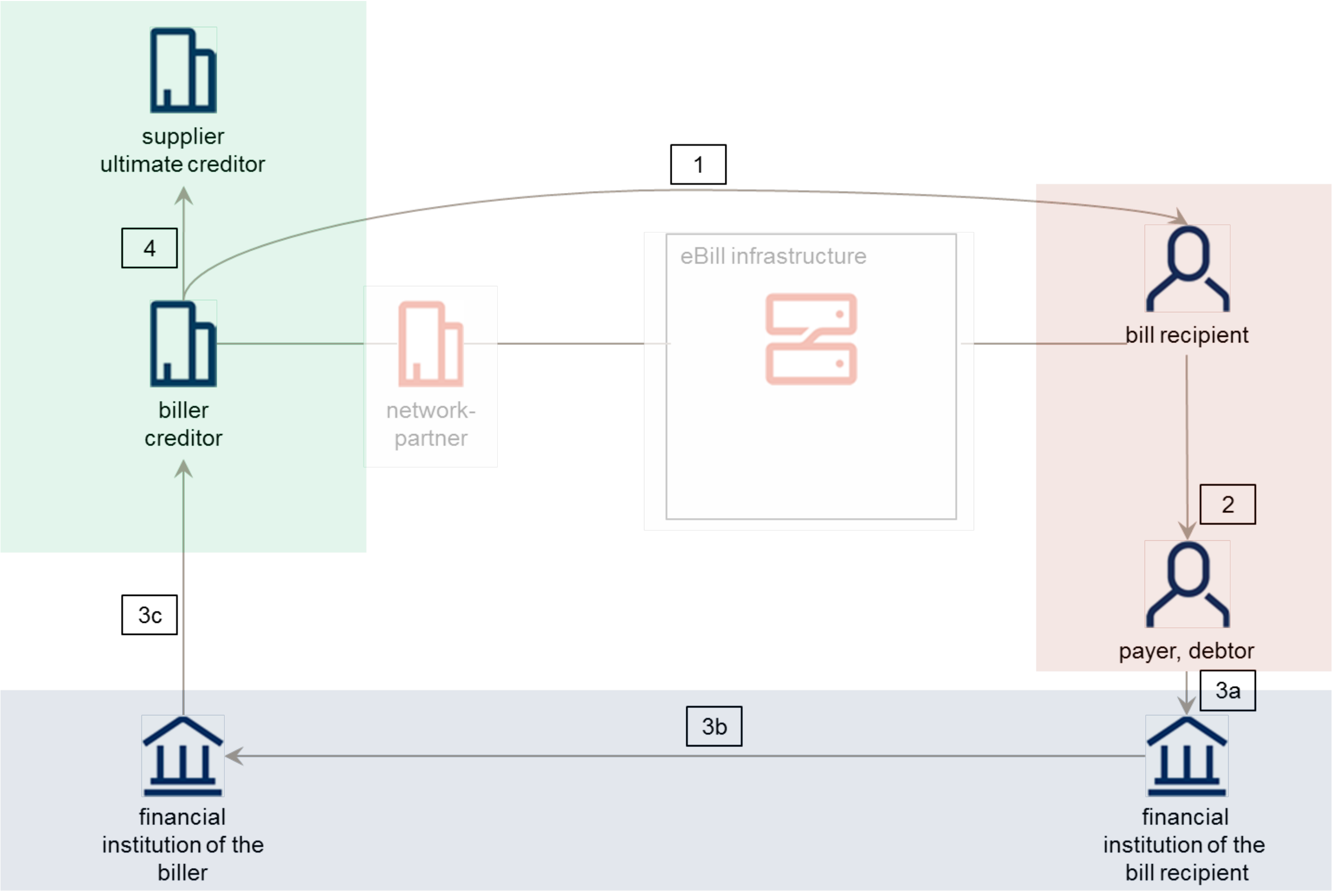


Figure 4. Billing and Payment Process

1. The biller sends an electronic invoice to a bill recipient using the services of his network partner and the eBill infrastructure. If the supplier or ultimate creditor is a different party than the biller, this has to be handled outside of the eBill infrastructure.
2. The bill recipient approves the electronic invoice and a payment instruction is sent by the eBill infrastructure to his or her financial institution. If the bill recipient is not the owner of the account being debited, a debtor party may be involved. However, these structures are implemented by the financial institution and are not known to the eBill infrastructure.
3. The debtor pays the bill and at the financial institutions the following steps occur:
    a. The financial institution of the debtor debits the account of the debtor.
    b. The financial institution of the debtor initiates the funds transfer to the financial institution of the biller.
    c. The financial institution of the biller credits the account of the biller and sends a credit note to the biller.
4. A potential ultimate creditor may be informed by the biller (out of scope of the eBill infrastructure).

Chapter 3. General Documentation

General information about the Network Partner API.

3.1. Design Principles

The Network Partner API is designed and implemented as a RESTful API.

The REST resources are usually designed to be self-contained. However, complex business-objects (for example: a biller with multiple properties like logos and shared attachments) may be split into different resources and sub-resources.

3.1.1. Definition Language

The Network Partner API is available as an OpenAPI Specification Version 2.0 (Swagger).
This detailed specification of the API is in itself also the documentation of the API.
Furthermore consumers of the API have the possibility to generate the client-side code from the specification.

The specification is provided in a separate file:

FileName	Description
networkpartner-api-v1-swagger.yaml	OpenAPI Specification of the Network Partner API. The OpenAPI Specification is best viewed in an editor such as https://editor.swagger.io/

3.1.2. Payload

The payload of the Network Partner API is defined in a format independent way in the Network Partner API specification.
The implementation of the Network Partner API expects and produces JSON-Payloads.

3.1.3. API Versioning

```
API Versioning (Version number in Swagger and XML-Schema):
1.2.1
| | |
| | +--- Defines the Patch Version, is incremented in case of a Bugfix in the API or a change of the documentation
| +----- Defines the Minor Version, is incremented in case of a Non-Breaking-Change in the API
+------- Defines the Major-Version, is incremented in case of a Breaking-Change in the API

Versioning in Namespace and in URLs:
1
|
+------ Defines the Major-Version and is incremented in case of a Breaking-Change in the API
```

The major version of the API is defined in the basepath URL.
Example: "https://api.six-group.com/api/pns/networkpartner/v1"

The following API changes are defined as backward compatible, meaning not a Breaking-Change, and will not lead to a new major version of the API specification:

- adding a new resource
- new, optional headers
- new, optional query parameters
- new, optional properties in a request
- new properties in a response
- mandatory property, query parameter or header becomes optional
- changing the order of response properties
- adding new problem types

Minor and patch level versions can be redeployed in the backend at any time.
API consumers have to deal with these backward compatible changes

The following API changes are examples of breaking-changes:

- existing functionality is removed
- mandatory data structure changes
- renaming of a property, query parameter or header
- optional property, query parameter or header becomes mandatory

Major version changes are always done with a transitional period, in this period both versions of the API are accessible.
Requests are always answered with the same major version as the requests were given (via basepath URL).

3.2. Conventions

3.2.1. Models

In Chapter 6, *Definitions*, all models that define the resources of the Network Partner API are listed.

3.2.2. Id-Handling

Identifications ("id") within the Network Partner API are always generated by the Network Partner API. This means that when the consumer of a resource creates a new object, such as billers, attachments or business cases, the "id" field in these models are optional.
When the creation of these new objects is successful, the endpoint will respond with an unique "id" that is generated by the Network Partner API.
This "id" can then be used as a unique shared identifier for this particular object and will be mandatory in any update or deletion actions.

3.2.3. Multilingual Support

To support a biller that operates in multiple language regions, the API offers the concept of

localizedData. Within localizedData, the biller has to specify its default language as well as provide at least the localizedData belonging to the selected default language.
There is direct support for multiple languages and the setting of a default language in the following resources:
- Section 4.1.2, “Create a biller”
- Section 4.1.4, “Update biller”
- Section 4.1.9, “Create biller attachment”
- Section 4.1.10, “Update biller attachment”

The localizedData models for biller and biller attachments can be found here: Section 6.3, “LocalizedBillerData”, Section 6.8, “LocalizedBillerAttachment”.

3.2.4. Assets

The Network Partner API supports the creation and management of binary assets such as images or PDF files. Management of the assets is split into two separate steps: First an entity (e.g. a biller or an attachment) is created using a POST call. This returns asset identifications (`assetId`), usually one `assetId` per language. In a second step the binary data for the `assetId` can be uploaded.
These calls are separated to avoid embedding binary data in Base64 encoded fields or MIME/multipart uploads.

Biller example: When creating a biller via Section 4.1.2, “Create a biller”, an empty asset is created for each logo in the Section 6.3, “LocalizedBillerData” entry. When the biller is created successfully, the operation will respond with a unique `assetId` for each one of these assets.
This returned `assetId` can then be used in combination with the Section 4.1.6, “Add/update asset” operation to upload the logo for the biller.
When changing a biller via Section 4.1.4, “Update biller”, without changes to the logo, an existing `assetId` can be reused.

Attachment example: The operation Section 4.1.9, “Create biller attachment” works similarly. It also returns an `assetId` if the attachment object is created successfully. At this time, there is no binary data stored for the attachment. The operation Section 4.1.6, “Add/update asset” can then be used to upload the binary data for the attachment.

3.2.5. HTTP Headers

The network partner must provide the following HTTP header fields with each request:

HTTP-Header-Field	Description	Example

authorization	This field contains the OAuth 2.0 client credentials Access-Token with the prefix "Bearer". A more detailed description of the OAuth 2.0 implementation and the handshake needed to acquire the access token is described in Chapter 5, *Security*	Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9Cg==.eyJpc3MiOiJhcGkuc2l4LWdyb3VwLmNvbSIsImF1ZCI6Ii9hcGkvbmV0d29ya3BhcnRuZXIiLCJleHAiOjE1Mjc3ODU1OTUsIm5iZiI6MTUyNzY4NTU5NSwiaWF0IjoxNTI3Njg1NTk1LCJuYW1lIjoiTmV0d29ya3BhcnRuZXIgTldJRDAwMDAwMCIsInN1YiI6Ik5XSUQwMDAwMDAifQo=.ZHVtbXktc2lnbmF0dXJlCg==
x-correlation-id	This field contains an ID, which will unambiguously identify this request to the API, this field is mandatory. When the field is not included in the request the API will respond with a Code 400 (Bad Request). The uniqueness of this value is validated for each network partner, the api will keep a buffer of the last 1000 correlation ids per network partner and validate the given value against this set. The ID is used in to analyze problem cases and enables the eBill infrastructure to uniquely identify requests between a network partner and the API. For responses, the ID is taken from the request. This allows queries and answers to be linked. We recommend to use a RFC 4122 Version 4 UUID as correlation id.	d36d37e7-bfad-45cf-b59c-b6c2ee16c446

3.2.6. HTTP Status Codes

Each HTTP request of the client is answered to with a HTTP status code by the Network Partner API. The status code is an indication for the client whether or not the request was successful.
In the event of an error, the response body contains additional information about how the error can be resolved, see Section 3.2.7, “Error Handling”.

3.2.7. Error Handling

If a request was not successful, the according HTTP status code provides basic information (see Section 3.2.6, “HTTP Status Codes”).

Further details are in the response body, see https://tools.ietf.org/html/rfc7807.

Example of an error response:

```
   HTTP/1.1 400 Bad Request ①
   Content-Type: application/problem+json
   Content-Language: en
   {
    "type": "/problems/REQUEST_BODY_VALIDATION_FAILED", ②
    "title": "Payload has missing or invalid values", ③
    "status": 400, ①
    "detail": "The submitted request contains invalid or missing data which can not be
processed.", ④
    "instance": "/api/pns/networkpartner/v1/billers/errors/NWID0090000001/ef46fa53-
6377-40d3-9f35-39a4a507792e" ⑤
    "technicalReason": "some field validations failed" ⑥
    "fieldErrors": [ ⑦
        {
            "fieldName": "billRecipientSubscriptionUrl",
            "message": "size must be between 1 and 1000",
            "rejectedValue": ""
        }
     ]
    }
```

① The HTTP status provides basic information.

② A problem URI specifies the type of problem that occurred. See Chapter 7, *Problem Descriptions Overview* for details.

③ Human readable title of the problem.

④ Details of the problem that occurred.

⑤ A specific reference of this occurrence. A combination of request url, network partner id and correlation id.

⑥ A more detailed technical reason of the problem, if available.

⑦ An object containing validation errors on field level, if available. The object contains the field name, a message and optionally a rejected value.

3.2.8. Event Handling

All events are accessible on the specific operations of the resource Section 4.3, “Events”.
The eBill infrastructure will generate events specifically for each consumer (e.g. network partner).

Each operation offers the following parameters to control which and how many events are returned in a single response:

Parameter	Description

lastEventId	when provided, the operation will only respond with events that occurred after the provided lastEventId
limit	the maximum number of events the endpoint will respond with in a single call

3.3. Use cases

3.3.1. Biller Management

Billers are managed through the Section 4.1, “Billers” endpoint which offers a number of different operations:

- The operation Section 4.1.1, “Find billers” is the public listing of all billers. This means it exposes all public biller data to all network partners, not only the biller’s primary network partner.
- The operation Section 4.1.3, “Get biller by id” is used to retrieve a single specific biller by its Id. It returns the complete biller information, but can only be called by the primary network partner of this biller.
- The operation Section 4.1.2, “Create a biller” is used to create a new biller. The network partner that created the biller is automatically assigned as primary network partner. If its status is `ACTIVE`, a duplicate check will be performed on existing `ACTIVE` billers.
- The operation Section 4.1.4, “Update biller” is used by the primary network partner of the biller to change the data of the biller. It is not possible to delete a biller, but one can set its status to `INACTIVE`. If its status is `ACTIVE`, a duplicate check will be performed on existing `ACTIVE` billers.

3.3.2. Business Case Management

Business cases can be submitted through the operation Section 4.1.13, “Create business case in PDF/A-3b-format”. After a successful creation, the operation answers with a Section 6.17, “BusinessCase”.
The id assigned to the business case by the Network Partner API can later be used to retrieve the business case information from the Section 4.1.14, “Get business case” resource.
There are five business case types Section 6.12, “Bill”, Section 6.15, “Reminder”, Section 6.14, “CreditNote”, Section 6.16, “Advice” and Section 6.13, “InstalmentBill”.

Credit note and advice only have the status `OPEN` and `COMPLETED`, which change, as soon as the user has viewed the business case.

The following state diagram shows the lifecycle of bills, reminders and each instalment of an instalment bill:

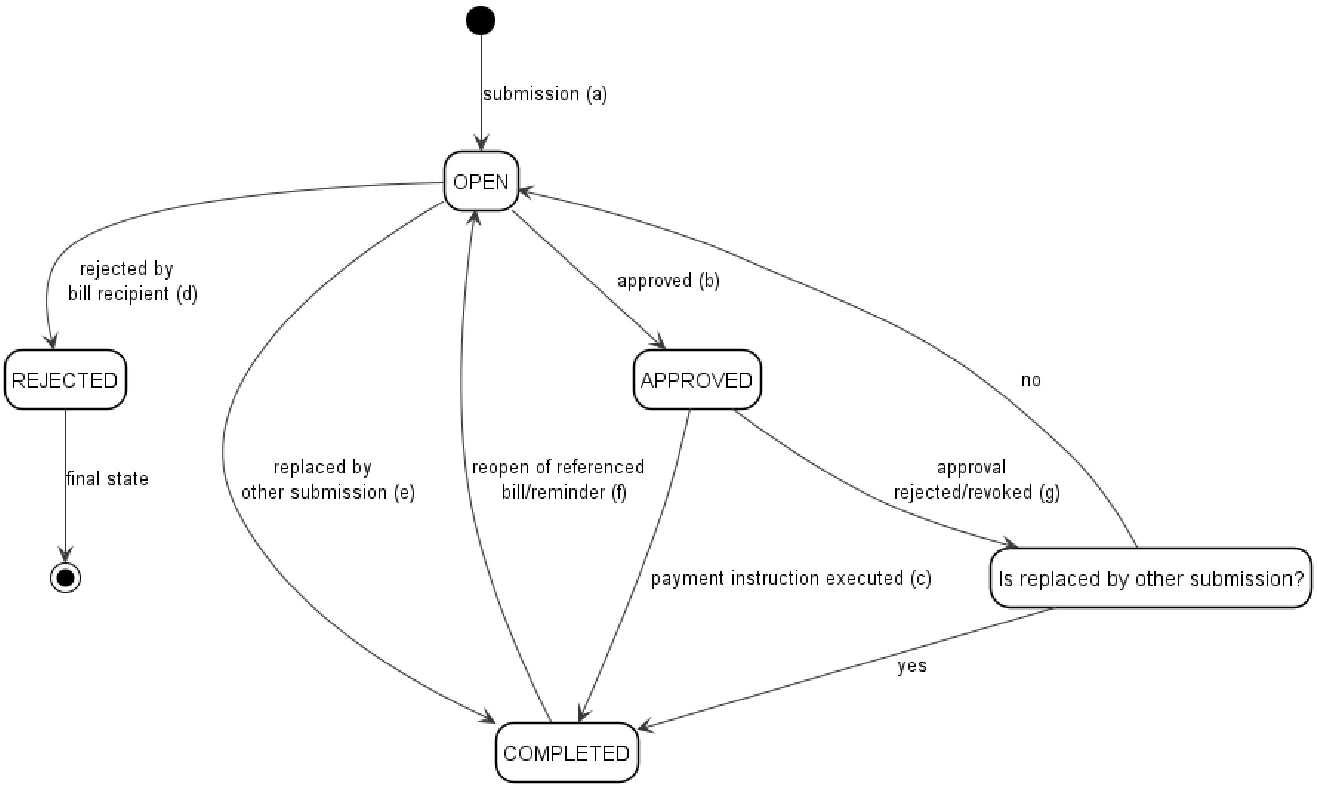


Figure 5. Lifecycle of bills, reminders and instalments

Happy Path

a. A new bill, reminder or instalment was created.
b. The bill, reminder or instalment was approved, either by the user or a standing approval.
c. The payment instruction was executed. The bill, reminder or instalment gets completed.

Rejection by User

d. The user rejected the bill, reminder or instalment. If more information is needed, the biller must contact the bill recipient.

Replacement and Referencing Business Cases

e. A bill or an instalment bill can be replaced by a new submission, which will complete the referenced bill or all instalments of the instalment bill.
But if the new submission is a reminder, it will not replace the bill or the instalment, but only reference it and both cases stay open. If either the reminder, the bill or the instalment is approved, the other one is set to completed.
Referencing ist possible in eBill-SIX_v1.xml with the element `referencedBill`, where the reference number of the referenced bill has to be provided.

f. If as in (g), a bill or an instalment bill was referenced by a reminder and the approved case of the pair gets reopened, the other one gets reopened as well.

Revocation of an Approvement

g. If the bill, reminder or instalment has been revoked/rejected in the online banking it changes back to open (re-opened). But, if in the meantime (between approval and approval-rejection/-

revocation), it has been replaced by another submission (bill reminder or instalment), it gets completed.
There are multiple reasons for re-opening:

1. The approval was revoked in the online banking by the user.
2. The payment instruction was not valid.
3. The payment instruction could not be executed.

Each status change will be reported by a specific event as described in Section 4.3, “Events”.
It’s distinguished between business case status change events (for bill, reminder, credit note and advice) and instalment status changed events (for instalment bills only).

Further information on the format of the PDF needed for the Section 4.1.13, “Create business case in PDF/A-3b-format” can be found in Section 3.3.2.5, “eBill-Format”.

eBill-Format

The eBill-format is the specification of the PDF and the attachment included in the PDF. The PDF with the included attachment is the required payload of the Section 4.1.13, “Create business case in PDF/A-3b-format” operation.

Business Considerations

The business requirements that are considered by this standard are summarized as follows:

- It should be possible to easily convert existing electronic billing standards to the eBill-format.
- VAT details and instalments information are supported on the business case level in the format.
- Addresses are required and structured in compliance with new regulations
- Relevant information concerning bill recipient ERP systems are supported under the term "workflow".
  1. This workflow information is modeled as optional values in the eBill-SIX_v1.xml format and supports information pertaining to VAT, delivery of goods/services, and reference identifiers see Section 3.3.2.5.5, “Specification of the Structured Data”.
  2. Only the structure of this data is validated by the Network Partner API.
- The information contained in the QR-bill must be sufficient for the creation of a simple electronic invoice. A minimal set of base attributes will be enriched by the network partner (for example, bill recipient identification, biller identification and business case type).

Basic Structure

To facilitate the transfer of business cases from a network partner to the eBill-infrastructure a standard-format has been defined.
The eBill-infrastructure only handles this standard-format and does not convert or support any other formats.
Transformations from other message formats to eBill-format can be offered by the network partner as a service.

The eBill-format consists of a PDF-container with an XML attachment and embedded signatures, see the figure below.

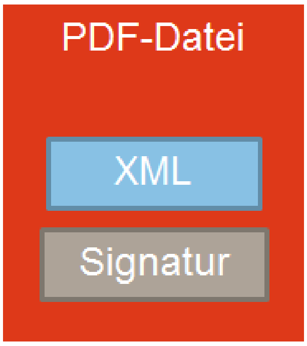


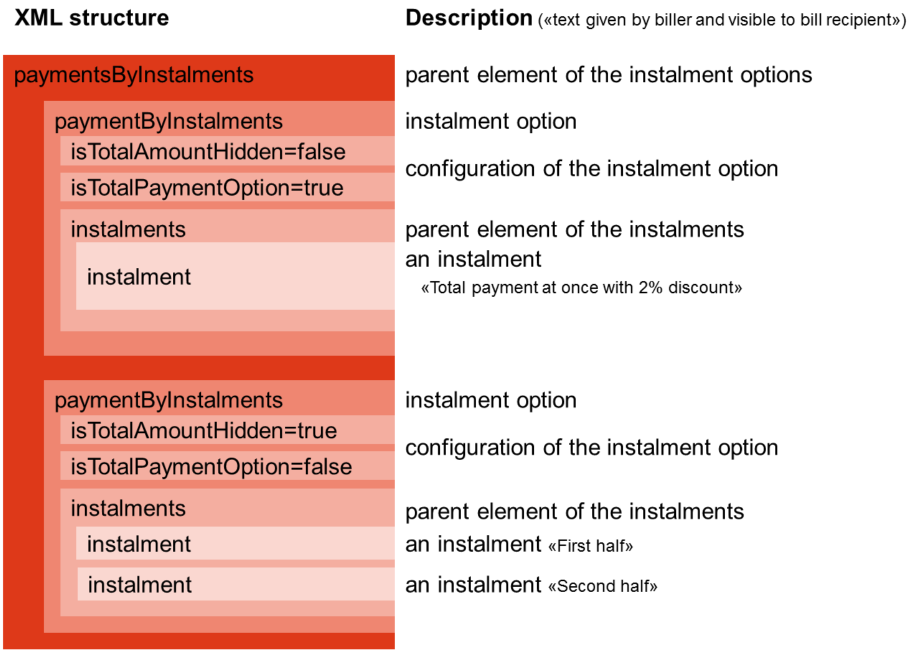
Figure 6. A representation of the PDF-container

Table 1. Glossary:

Name	**Description**
eBill-Format	The complete PDF (PDF incl. XML and signatures)
eBill-SIX_v1.xml	The Structured business case information in XML included as attachment in the PDF
eBill-SIX_v1.xsd	The XSD-Schema that describes the structure of the XML attachment (eBill-SIX_v1.xml)

Instalments

The structured data offers support for instalment bills. A biller can offer to pay his bill by various instalment options. Each of the instalment options can contain multiple instalments:



XML structure	**Description** («text given by biller and visible to bill recipient»)
paymentsByInstalments	parent element of the instalment options
paymentByInstalments	instalment option
isTotalAmountHidden=false	configuration of the instalment option
isTotalPaymentOption=true	
instalments	parent element of the instalments
instalment	an instalment «Total payment at once with 2% discount»
paymentByInstalments	instalment option
isTotalAmountHidden=true	configuration of the instalment option
isTotalPaymentOption=false	
instalments	parent element of the instalments
instalment	an instalment «First half»
instalment	an instalment «Second half»

Figure 7. Example of two instalments options

The bill recipient will pick one of the instalment options in the UI and will subsequently approve all of the instalments.

File Specification and Signatures

The supplied PDF needs to be a PDF/A-3b conforming document. This type of document fulfills a number of requirements "out of the box" such as the support for long term archiving, embedding data, document signing and wide tools support.
A PDF is allowed to have other embedded documents. The eBill-format requires that an XML attachment is included with the name "eBill-SIX_v1.xml" that is compliant with the eBill XSD. The PDF must not contain an XML attachment with name "eBill-SIX_v2.xml" or any other version.
The document needs to be signed with a PAdES-B-B-level conforming electronic signature. The documentation of the PAdES standard can be downloaded from https://www.etsi.org/.
The signature must be included in the PDF document and encompasses the whole PDF document including the embedded documents and at least the eBill-SIX_v1.xml.

Note: Accessibility is not required in the PDF/A-3b. It is up to the biller to ensure accessibility, if desired.

Specification of the Structured Data

The specification of the eBill-SIX_v1.xml is created and maintained by SIX. The XML adheres to the following ground principles:

- Simplicity of the format (No detailed invoice positions)
- Limited user group (network partner to infrastructure)
- Possibility to convert from existing, simple formats, through the use of QR-Bills
- The format is based on the recommendations of "swissDIGIN content standard light"

A specification of the structured data (XML) including a detailed description is available.

This specification is delivered separately:

FileName	Description
eBill-SIX_v1.xsd	XSD-Schema of the structured data (eBill-SIX_v1.xml)
D0474_eBill-SIX_v1_advice.pdf D0475_eBill-SIX_v1_bill.pdf D0476_eBill-SIX_v1_creditNote.pdf D0480_eBill-SIX_v1_instalmentBill.pdf D0477_eBill-SIX_v1_reminder.pdf	documentation of the XML schema

Character set

The eBill-SIX_v1.xml content will be parsed using the UTF-8 character set. Please note that ISO 20022 messages according to the Swiss Payment Standard only allow the Latin Character Set to be used. Therefore a conversion as documented in the pain.001 specifications will be applied for non latin characters in the relevant fields.

eBill-Format Versioning

The major version of the eBill-Format is defined in the namespace of the XML schema (eBill-

SIX_v1.xsd).
Example: xmlns:ebill="http://six-group.com/pns/networkpartner/**v1**/ebill/xml"

The following XML schema changes are defined as backward compatible and will not lead to a new major version of the XML schema:

- new, optional fields
- adding new error codes

Consumers of the eBill-Format have to deal with such changes without prior notice.

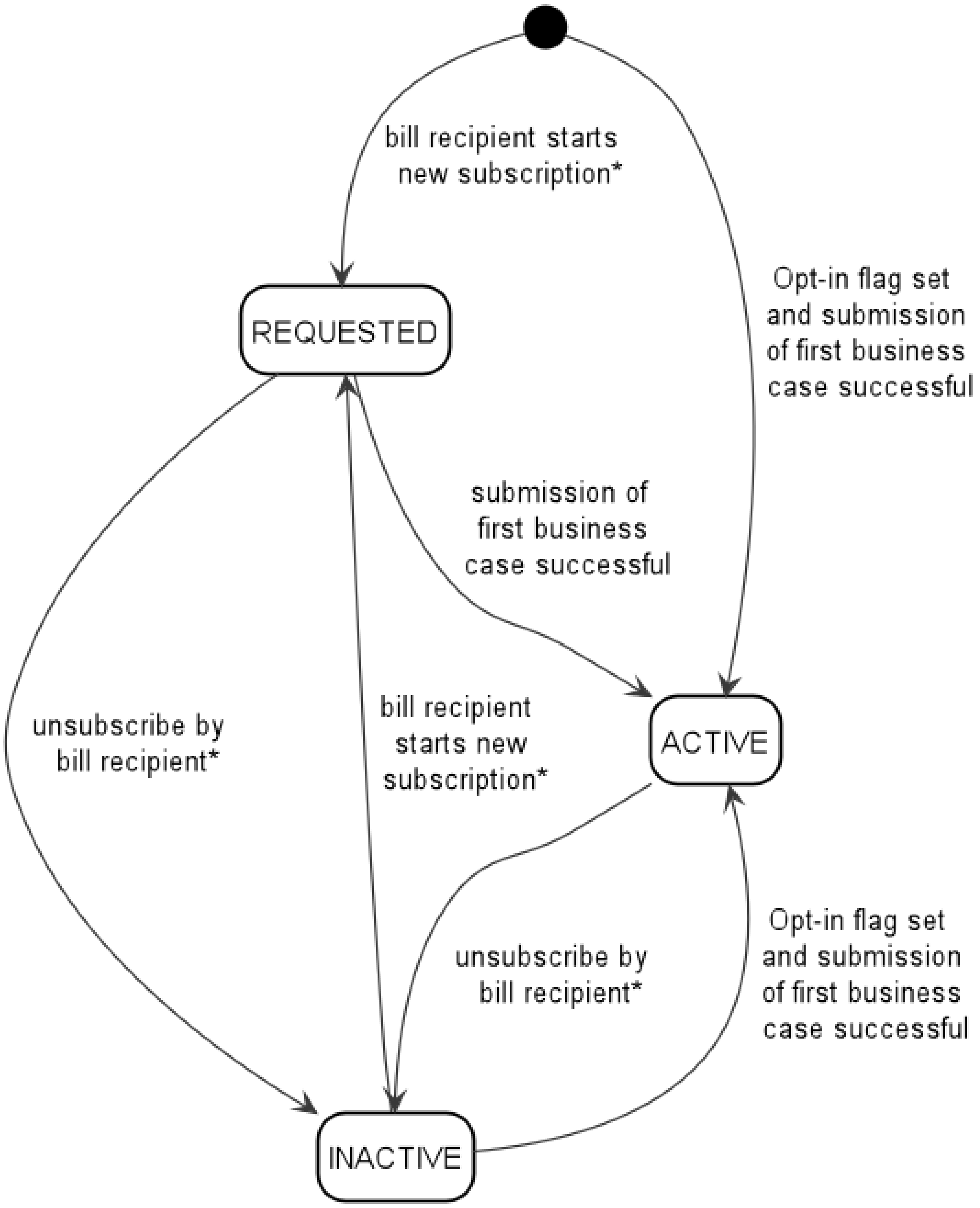
3.3.3. Subscriptions and Subscription Cancellations

The transfer of business cases from a biller to a bill recipient requires both parties to agree on this process. The resulting connection is called "biller to bill recipient relation" or for simplicity just "biller relation". Initiating a biller to bill recipient relation is called "subscription" and there are various ways to achieve this:

1. Bill recipient driven subscription. Depending on the biller configuration there are two possibilities:
    a. subscription *without* custom forms
    b. subscription using custom forms
2. Biller driven subscription (also known as "Look-Up").

The following sections provide details on these processes.

Events for subscriptions and subscription cancellations are triggered after a bill recipient has subscribed or unsubscribed to a biller. It is also possible for the bill recipient to subscribe multiple times to the same biller, which results in multiple events that can have the same status but maybe a different email address or a different address.
The following state diagram of the biller to bill recipient relation illustrates the status transitions and reasons for it.


bill recipient starts new subscription*
Opt-in flag set and submission of first business case successful
REQUESTED
submission of first business case successful
unsubscribe by bill recipient*
bill recipient starts new subscription*
ACTIVE
unsubscribe by bill recipient*
Opt-in flag set and submission of first business case successful
INACTIVE

status transitions marked with a * trigger an event for the network partner

Figure 8. State diagram for biller to bill recipient relations

Bill recipient driven subscription without custom forms

Avoiding custom subscription forms simplifies the subscription process for the bill recipient and the network partner considerably. If a network partner creates a biller *without* an `billRecipientSubscriptionUrl` value, the event-based flow as described in this section is used.

If the bill recipient starts the subscription process in the eBill portal, the flow is as follows:

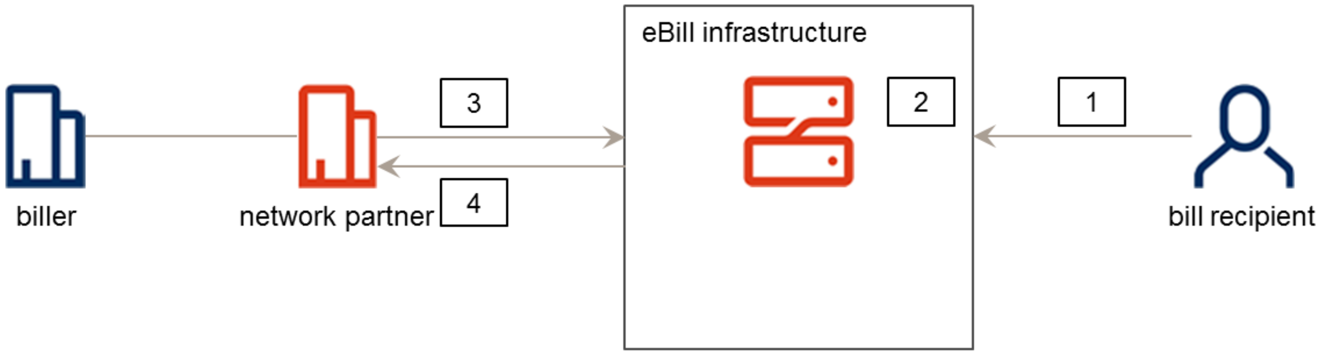


Figure 9. Subscription initiated by bill recipient

1. A bill recipient finds the biller and subscribes to the biller.
2. The eBill infrastructure creates a bill recipient subscription in status `REQUESTED` and creates a subscription event.
3. The network partner polls for new subscription events….
4. …and receives the subscription data as a response.

A similar process is started if the bill recipient is using in the online banking system of his financial institution:

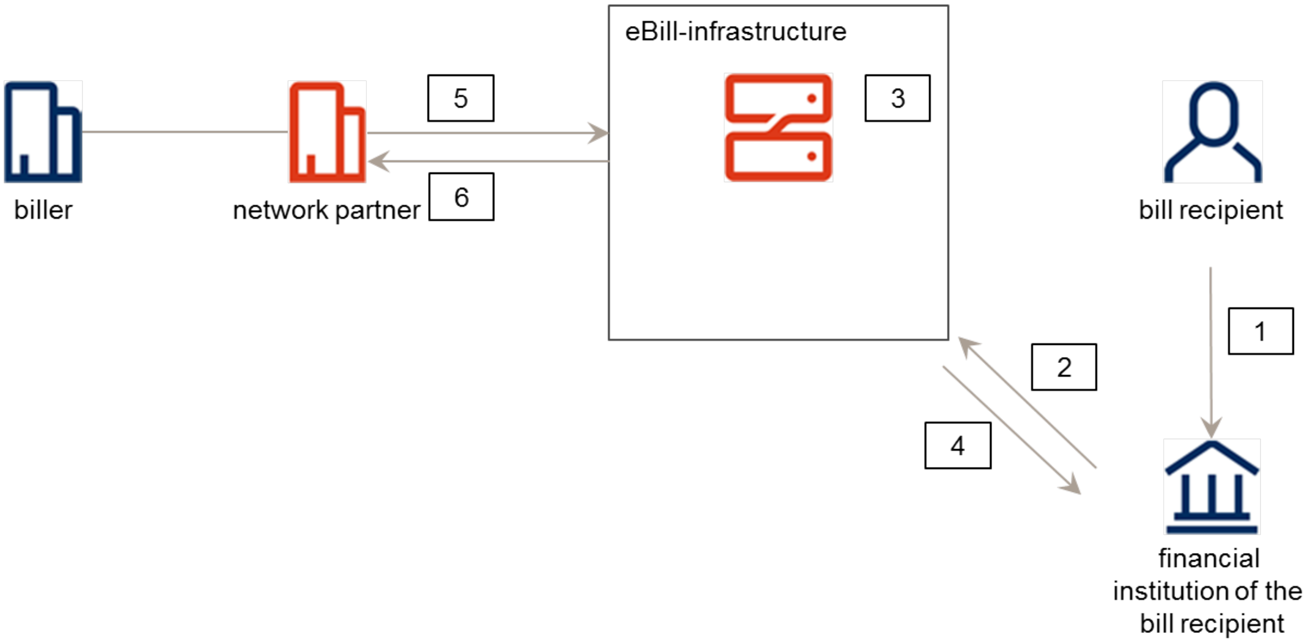


Figure 10. Subscription initiated by the online banking of the bill recipient

1. A bill recipient enters payment information in the online banking system of his financial institution. The financial institution checks if the payment information (account / part of the reference number) refers to an entry in the biller directory. If a match is found, the e-banking system provides the user the option to subscribe to the biller.
2. The user decides to subscribe to the biller in the online banking. The online banking system calls the eBill infrastructure to subscribe.
3. The eBill infrastructure creates a bill recipient subscription in status REQUESTED and creates a subscription event for the network partner.

4. The response of the eBill infrastructure returns an OK code (or - just for backward compatibility - an URL to a subscription confirmation page).
5. The network partner polls for new subscription events....
6. ...and receives the subscription data as a response.

Bill recipient driven subscription using custom form

Currently, eBill relies on custom subscription forms. They are used to collect additional information that only the bill recipient knows: reference number of a past bill, social security number, MAC address of a modem, ... The Network Partner API subscription process has to be designed in a way that custom subscription forms for billers *can* be supported by the network partners.

If the bill recipient starts the subscription process in the eBill portal, the flow is as follows:

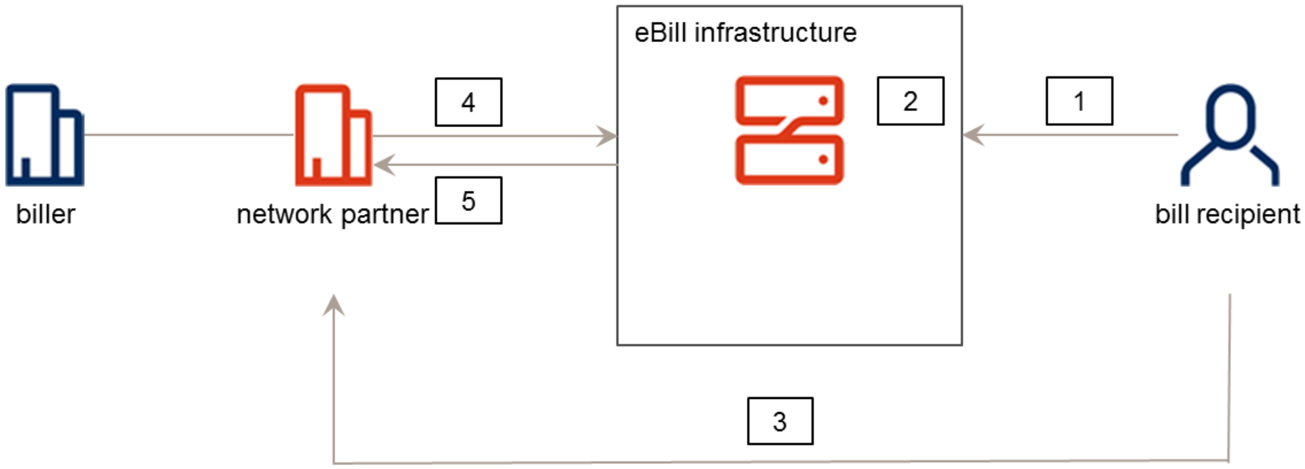


Figure 11. Subscription initiated by bill recipient with a biller that requires a custom form

1. A bill recipient finds the biller in the eBill portal and clicks "subscribe"
2. eBill infrastructure returns the URL as defined in the biller `billRecipientSubscriptionUrl` property. The `subscriptionToken` is added to the url as a query parameter.

   Example-URL:

   https://subscription-url.nwp.ch/?token=9fbff442-6720-4083-9bca-0f9b00b1f362

3. The bill recipient clicks the subscription form link. This causes the custom subscription form at the network partner to be called.
   For security reasons, the network partner has to validate that the subscription form can only be displayed once with the same token.
4. The network partner can use the token contained in the URL to call the `/bill-recipient-subscriptions/{subscriptionToken}` endpoint.

   Example-Call:

   `/bill-recipient-subscriptions/9fbff442-6720-4083-9bca-0f9b00b1f362`

   As soon as this call is received by the eBill infrastructure a biller relation in status `REQUESTED` is created.

5. Data returned by the Network Partner API can be used to pre-fill the custom subscription form. As the bill recipient completes and submits the form, all the data can be forwarded to the biller.

A similar process is started if the bill recipient is working in the online banking system of his financial institution:

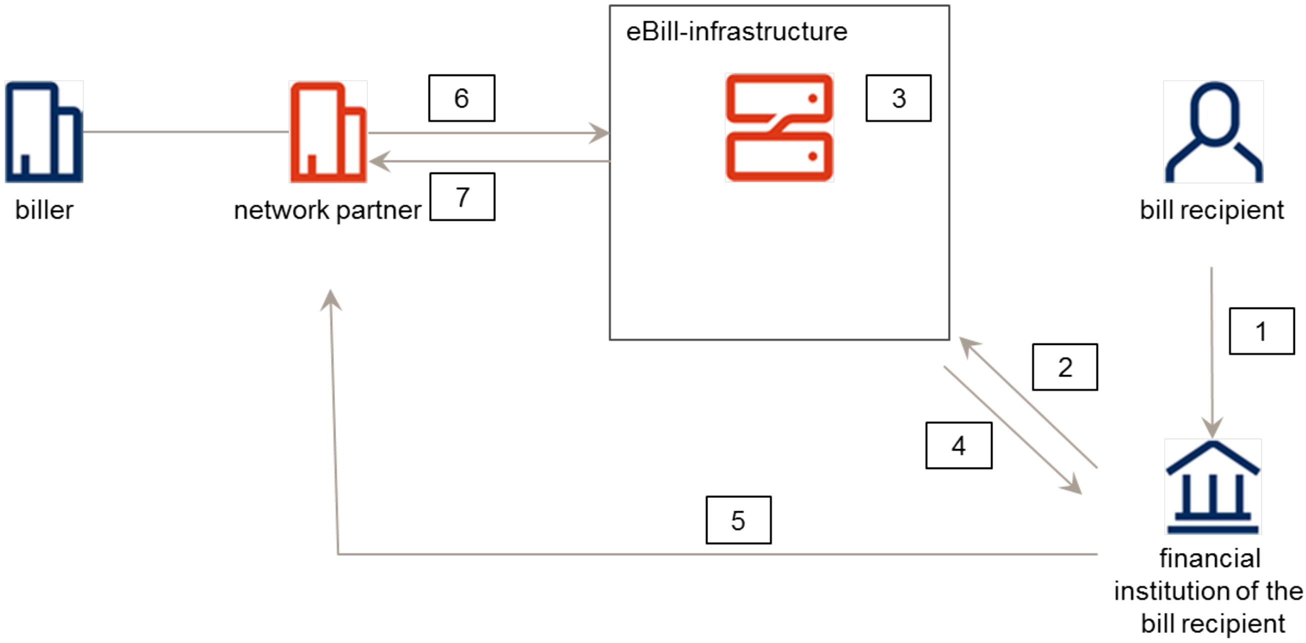


Figure 12. Subscription initiated in the online banking of the bill recipient with a biller that requires a custom form

1. A bill recipient enters payment information in the online banking system of his financial institution. The financial institution checks if the payment information (account / part of the reference number) refers to an entry in the biller directory.
2. If a match is found, the online banking system requests the subscription URL of the biller.
3. eBill infrastructure creates a URL as defined in the biller `billRecipientSubscriptionUrl` property. The `subscriptionToken` is added to the url as query parameter.

   Example-URL:

   `https://subscription-url.nwp.ch/?token=9fbff442-6720-4083-9bca-0f9b00b1f362`
4. The URL is returned to the online banking system and displayed to the user.
5. The bill recipient clicks the subscription form link. This causes the custom subscription form at the network partner to be called.
   For security reasons, the network partner has to validate that the subscription form can only be displayed once with the same token.
6. The network partner can use the token contained in the URL to call the `/bill-recipient-subscriptions/{subscriptionToken}` endpoint.

   Example-Call:

   `/bill-recipient-subscriptions/9fbff442-6720-4083-9bca-0f9b00b1f362`

   As soon as this call is received by the eBill infrastructure a biller relation in status `REQUESTED` is created.

7. Data returned by the Network Partner API can be used to pre-fill the custom subscription form. As the bill recipient completes and submits the form, all the data can be forwarded to the biller.

Biller driven subscription ("Look-Up")

Billers get the possibility to search for bill recipients using the Network Partner API. With a positive match the biller can immediately submit business cases to the bill recipient.

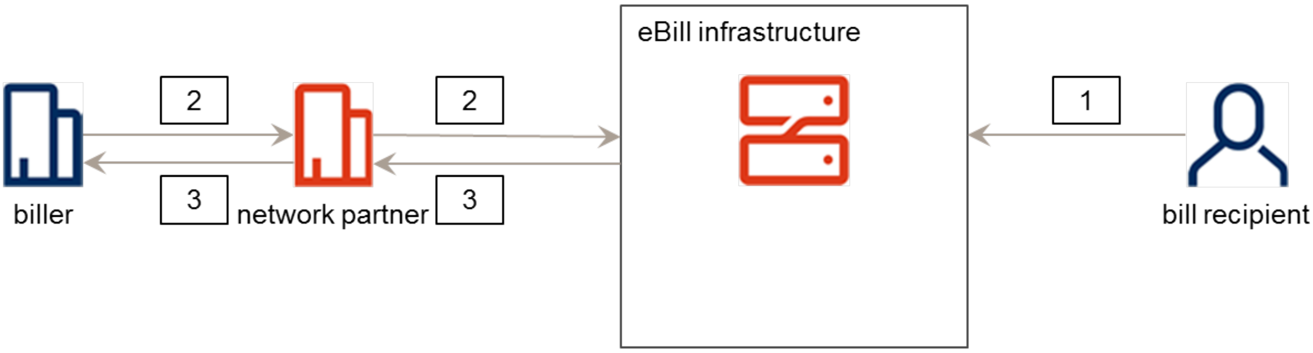


Figure 13. Subscription initiated by biller using Look-Up

1. A bill recipient has to enable, to be found in the eBill infrastructure.
2. The biller may query the eBill infrastructure to find bill recipients (e.g. using the bill recipient’s e-mail address as search criteria)
3. If all conditions are met a positive response is returned and it is possible to submit a business case immediately.

Subscription Cancellation

The cancellation of a subscription is always initiated by a bill recipient (a biller will simply decide not to send eBills any more - this is not represented in the eBill infrastructure).
Note that subscription cancellation forms are not supported any more by the eBill infrastructure. Subscription cancellation will simply lead to an event on the Network Partner API.

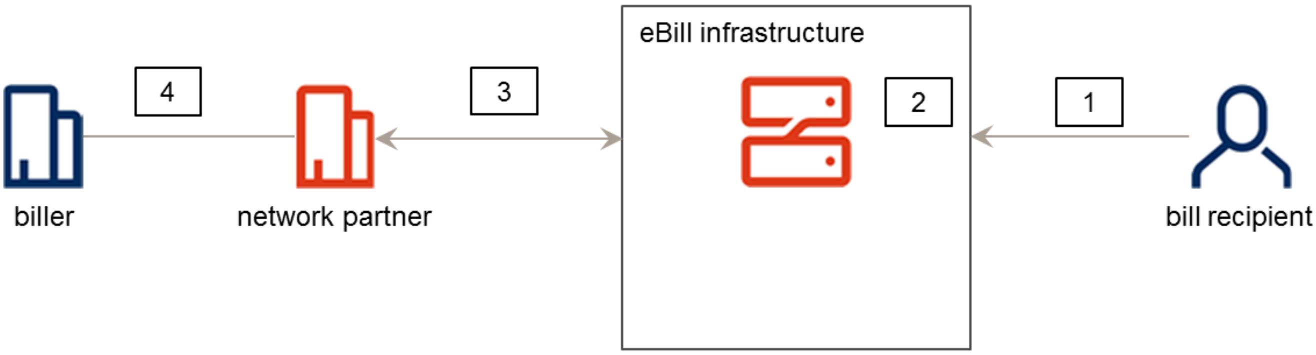


Figure 14. Subscription cancellation initiated by bill recipient

1. A bill recipient clicks "unsubscribe" in the eBill portal.
2. The eBill infrastructure sets the biller relation to `INACTIVE` and creates a subscription changed event with the necessary information.

3. The network partner polls for new subscription cancellation events…
4. …and receives the subscription cancellation data as a response.

Chapter 4. Resources

4.1. Billers

Biller Management
All operations that are associated with a biller are located within the biller resource, including the creation of business cases.
The operations are designed to be self contained, because of this, the data objects can be rather large. However this allows for complete validation and avoids chains of calls that depend on each other.

4.1.1. Find billers

```
GET /billers
```

Description

a public listing of all billers in the eBill infrastructure. Can be used by all network partners to retrieve public data about each biller.

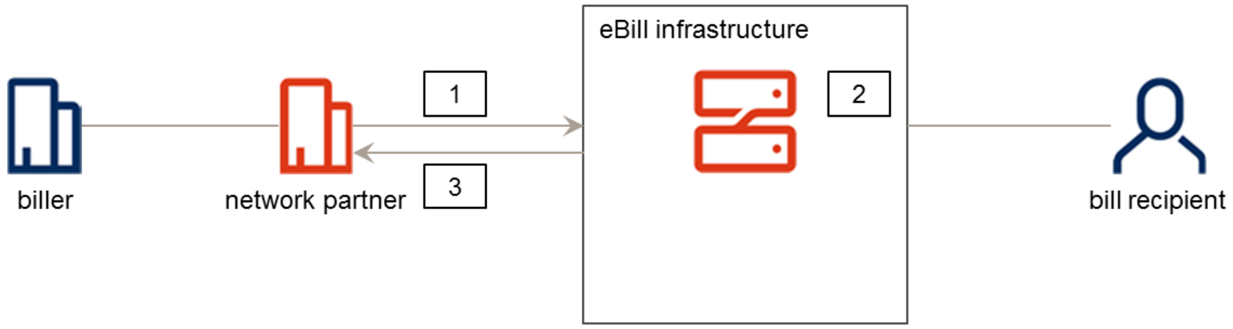


Figure 15. Find billers

1. A network partner sends the request to the eBill infrastructure
2. The eBill infrastructure finds billers (active + inactive)
3. and returns the answer to the network partner.

Validations

All validations according to Section 7.1, “Basic Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string

Type	Name	Description	Schema
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Billers found	< Biller > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.1.2. Create a biller

```
POST /billers
```

Description

This operation attempts to create a new biller in the system.

Before creating a new biller, a series of validations are executed (these rules apply even if the biller exists in INACTIVE status, except for duplicate checks, which only apply for billlers in ACTIVE status).
If all validations pass, the biller is created and the response will contain the newly created biller ID.
If there is a conflict with an existing biller, its biller ID is provided in the technicalReason of the error response.

The network partner that successfully creates a biller becomes the primary network partner for this biller.
All other network partners will immediately be able to see a limited set of data of the the newly created biller.

The following diagram gives an overview of the organizational and technical steps of the biller creation:

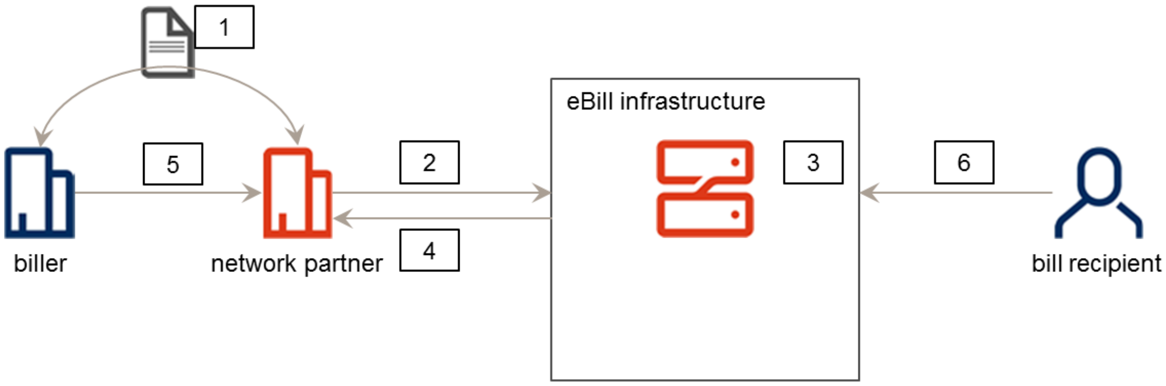


Figure 16. Create a biller

1. Biller signs a contract with the network partner to use his services.
2. The network partner creates the biller on the eBill infrastructure using the operation Section 4.1.2, “Create a biller”.
3. The eBill infrastructure validates the request.
4. After a successful validation, the API returns a new biller ID.
5. The biller can start to use the eBill services via its network partner as soon as the registration is completed (e.g. searching bill recipients, creating business cases).
6. Bill recipients can search for the newly created biller and subscribe with it.

Validations

All validations according to Section 7.1, “Basic Validations” and Section 7.3, “Biller Validations” except INVALID_ASSET_ID and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Data for the biller to be added	BillerDetail

Responses

HTTP Code	Description	Schema
201	Biller created	BillerDetail

HTTP Code	Description	Schema
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/json`

Produces

- `application/json`

4.1.3. Get biller by id

```
GET /billers/{billerId}
```

Description

Get all information for a specific biller. Calls are validated and only network partners are allowed to retrieve information for billers where they are assigned as primary network partner.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Biller found	BillerDetail

HTTP Code	Description	Schema
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.1.4. Update biller

```
PUT /billers/{billerId}
```

Description

Updates a biller with the given biller details. Only the primary network partner of a biller is allowed to update biller details.

Note that if the status of the biller is set to `INACTIVE` the biller cannot submit business cases any more and it will be removed from the biller listing in the eBill infrastructure.

Validations

All validations according to Section 7.1, “Basic Validations”, Section 7.3, “Biller Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Biller, which should be updated	BillerDetail

Responses

HTTP Code	Description	Schema
200	Biller updated	BillerDetail

HTTP Code	Description	Schema
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.1.5. Get asset data by id

```
GET /billers/{billerId}/assets/{assetId}
```

Description

Get the binary data for a specific asset. Depending on the content type of the asset, it produces the response accordingly. Only the primary network partner of a biller is allowed to manage its assets.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Path	**assetId** *required*	asset id **Maximal length**: `14` **Example**: `"ASID0000000001"` **Pattern**: `ASID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Biller asset found	string (binary)

HTTP Code	Description	Schema
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `image/jpeg`
- `image/png`
- `image/gif`
- `application/pdf`

4.1.6. Add/update asset

```
PUT /billers/{billerId}/assets/{assetId}
```

Description

Update the asset's binary data or adds new binary data to the asset. Depending on the accept header the corresponding content type is stored along with the asset's binary data.
Note the different content types that are accepted: Biller logo assets have to be one of image/jpeg, image/png or image/gif. Attachment assets have to be application/pdf.
Only the primary network partner of a biller is allowed to manage its assets.

Validations

All validations according to Section 7.1, "Basic Validations" and Section 7.4, "Asset Validations" and some of Section 7.2, "Shared Validations".

For image assets:

ASSET_IMAGE_INVALID
ASSET_MIME_TYPE_DOES_NOT_CORRESPOND_TO_CONTENT_TYPE
ASSET_IMAGE_EXCEEDS_MAXIMUM_SIZES
ASSET_CONTENT_TYPE_MUST_BE_IMAGE

For PDF assets:

Section 7.9, "PDF Validations"
ASSET_PDF_EXCEEDS_MAXIMUM_SIZE
ASSET_CONTENT_TYPE_MUST_BE_PDF

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Path	**assetId** *required*	asset id **Maximal length**: `14` **Example**: `"ASID0000000001"` **Pattern**: `ASID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Binary of biller asset	string (binary)

Responses

HTTP Code	Description	Schema
204	Biller asset updated	No Content
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `image/jpeg`
- `image/png`
- `image/gif`
- `application/pdf`

4.1.7. Delete asset

```
DELETE /billers/{billerId}/assets/{assetId}
```

Description

Delete the asset, this includes its binary and content type data. Only the primary network partner of a biller is allowed to manage its assets.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Path	**assetId** *required*	asset id **Maximal length**: `14` **Example**: `"ASID0000000001"` **Pattern**: `ASID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
204	Biller asset deleted	No Content
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

4.1.8. Find attachments of the biller

```
GET /billers/{billerId}/attachments
```

Description

Attachments are documents of a biller that have to be shown to the bill recipient at the same time as the bill itself. Examples of such documents are tariff lists, terms and conditions and promotions. Attachments are only shown within a certain time window. Several attachments with overlapping time windows can be available and will be shown. Only the primary network partner of a biller is allowed to manage its attachments.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Attachments for the biller found	< BillerAttachment > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.1.9. Create biller attachment

```
POST /billers/{billerId}/attachments
```

Description

When creating a new attachment, one will get back one or many asset ids. With these asset ids one can upload the corresponding binaries using /billers/{billerId}/assets/{assetId}. Only the primary network partner of a biller is allowed to manage its attachments.

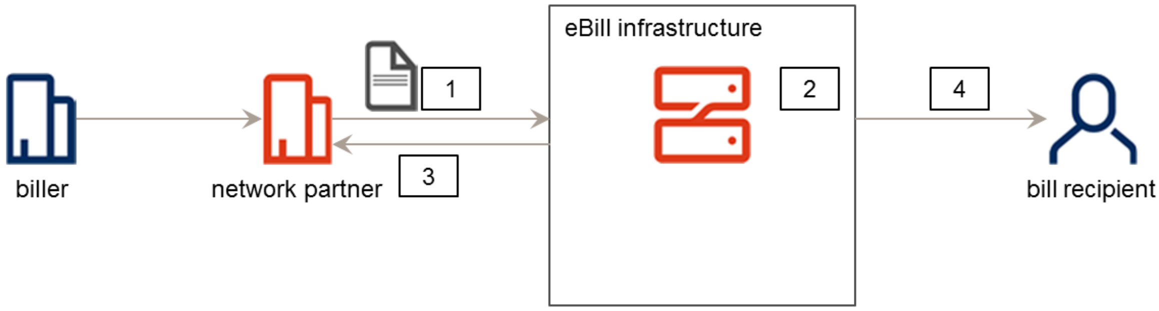


Figure 17. Create an attachment

1. Meta data for a new attachment is created in the eBill infrastructure. It is mandatory to provide meta data for the defined default language.
2. The meta data is validated.
3. If successful the response contains references to the assets that have been created for the meta data. Binary data can be uploaded with this information.
4. As bill recipients look at business cases the eBill infrastructure will check for attachments that are valid at this point in time. All matching attachments are shown to the bill recipient together with the business case. The system tries to show attachments in the matching language or falls back to the language marked as "default".

Validations

All validations according to Section 7.1, “Basic Validations”, Section 7.5, “Attachment Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Biller attachment, which should be added	BillerAttachment

Responses

HTTP Code	Description	Schema
201	Biller attachment created	BillerAttachment
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/json`

Produces

- `application/json`

4.1.10. Update biller attachment

```
PUT /billers/{billerId}/attachments/{attachmentId}
```

Description

Update a biller attachment by its id. Only the primary network partner of a biller is allowed to manage its attachments.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Path	**attachmentId** *required*	attachment id **Maximal length**: `14` **Example**: `"ATID0000000001"` **Pattern**: `"ATID[0-9]{10}"`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Biller attachment, which should be updated	BillerAttachment

Responses

HTTP Code	Description	Schema
200	Updated biller attachment	BillerAttachment
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/json`

Produces

- `application/json`

4.1.11. Delete biller attachment

```
DELETE /billers/{billerId}/attachments/{attachmentId}
```

Description

Delete a biller attachment by its id. Only the primary network partner of a biller is allowed to manage its attachments.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Path	**attachmentId** *required*	attachment id **Maximal length**: `14` **Example**: `"ATID0000000001"` **Pattern**: `"ATID[0-9]{10}"`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
204	Deleted biller attachment	No Content
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

4.1.12. Search for bill recipients for this biller

```
POST /billers/{billerId}/bill-recipients/search
```

Description

Returns a list of one or more bill recipients.
The response will only list a bill recipient if he allows bills to be posted by the biller or allows bills to be posted in general.
The call will return an empty list if no matching bill recipient is found or the biller is not allowed to send bills to the bill recipient.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Parameters for the search, at least one of the parameters is required	BillRecipientsForBillerSearch

Responses

HTTP Code	Description	Schema
200	Bill recipients found	< BillRecipientsForBillerSearch > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/json`

Produces

- `application/json`

4.1.13. Create business case in PDF/A-3b-format

```
POST /billers/{billerId}/business-cases
```

Description

The creation of a business case in PDF/A-3b format works with a simple POST request where the PDF is the binary payload of the request. Note that only one PDF can be submitted at a time.
Validation of the submitted data will take place immediately and the response either confirms the successful creation of the business case or contains the error details of the validation.

Technical Hint:
Note that this POST operation is not officially supported by the specification language 'OAS 2.0' ('Swagger 2.0') as described in detail here -> 'https://swagger.io/docs/specification/2-0/file-upload/'

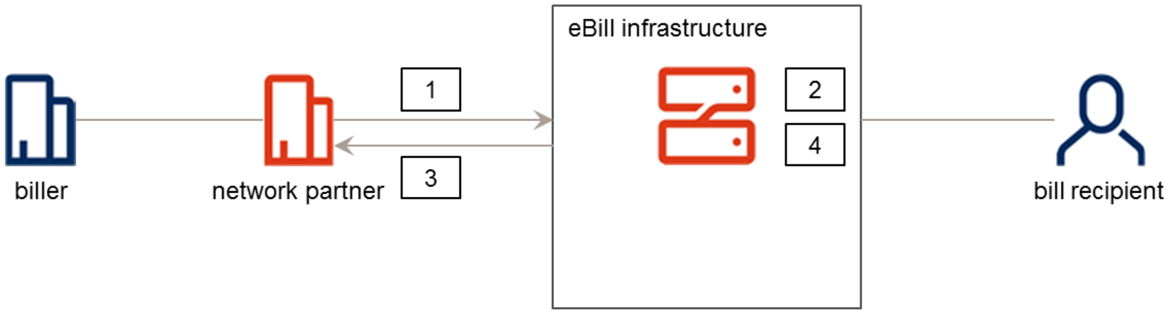


Figure 18. Create a business case

1. A network partner submits a business case on behalf of a biller.
2. Processing takes place in the eBill infrastructure.
3. The response contains the generated business case id.
4. As of this moment the business case is visible to bill recipients.

Addressing of a Business Case to a Bill Recipient

A submitted business case can be addressed to a bill recipient by the bill recipient’s email address or by its billRecipientId or by its swiss enterprise identification number (UID) from the commercial register (Handelsregister). Business cases addressing a bill recipient by its email address can be successfully submitted even if the bill recipient’s email address has been changed in the meantime (at least for a limited time). For further details see: Section 4.3.4, “Find events for bill recipients email address changes”

Validations

All validations according to Section 7.1, “Basic Validations”, Section 7.6, “Business Case Validations”, Section 7.7, “Business Case Payment Validations”, Section 7.8, “Business Case Instalment Validations”, Section 7.9, “PDF Validations” and some of Section 7.2, “Shared Validations”.

Biller Reference Number Validation

For all business case types (bill, reminder, advice and credit), the reference number must be unique per biller with regard to the submission date and the business case type. This means that no business case of the same type with the same reference number can be submitted on the same day.

In the case of business case type "bill", an additional requirement regarding the uniqueness of reference number (RN) must also be met:

The following image illustrates the corresponding validation rule for unique reference number on submission of business case (BC) of type "bill".
First of all: Each BC, in particular bills, has a submission date (SD), due date (DD) and a check date (CD). The check date of a business case is defined as the maximum of its submission date (SD) and its due date (DD), i.e. the most recent of the two dates, SD and DD.

When a bill BILLx is submitted, there must be no other bill with the same reference number whose CD is less than 90 days old. This means that no other bill can exist in the system with the same reference number and check date (CD) between x-90 days and EOT. If this is the case, the submission of bill BILLx will be rejected.

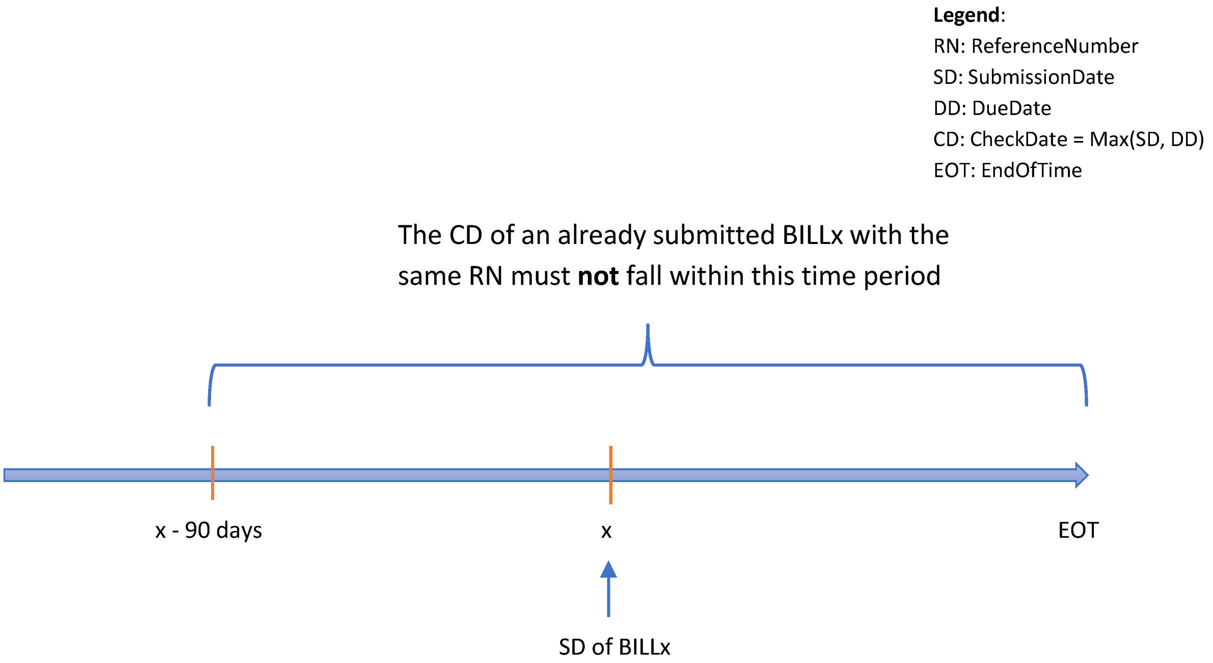

Legend:
RN: ReferenceNumber
SD: SubmissionDate
DD: DueDate
CD: CheckDate = Max(SD, DD)
EOT: EndOfTime
The CD of an already submitted BILLx with the same RN must not fall within this time period
x - 90 days
x
EOT
SD of BILLx

Figure 19. Validation rule for unique reference number

The following examples represent different scenarios for reference number validation on submission of business case (BC) of type "bill"

Scenario 1:

. Let a bill, BILLy with RN=1234 and check date, CD=y=today-30 days be present in the system.

. Todays submission of bill, BILL0 with the same RN=1234 leads to rejection of BILL0.

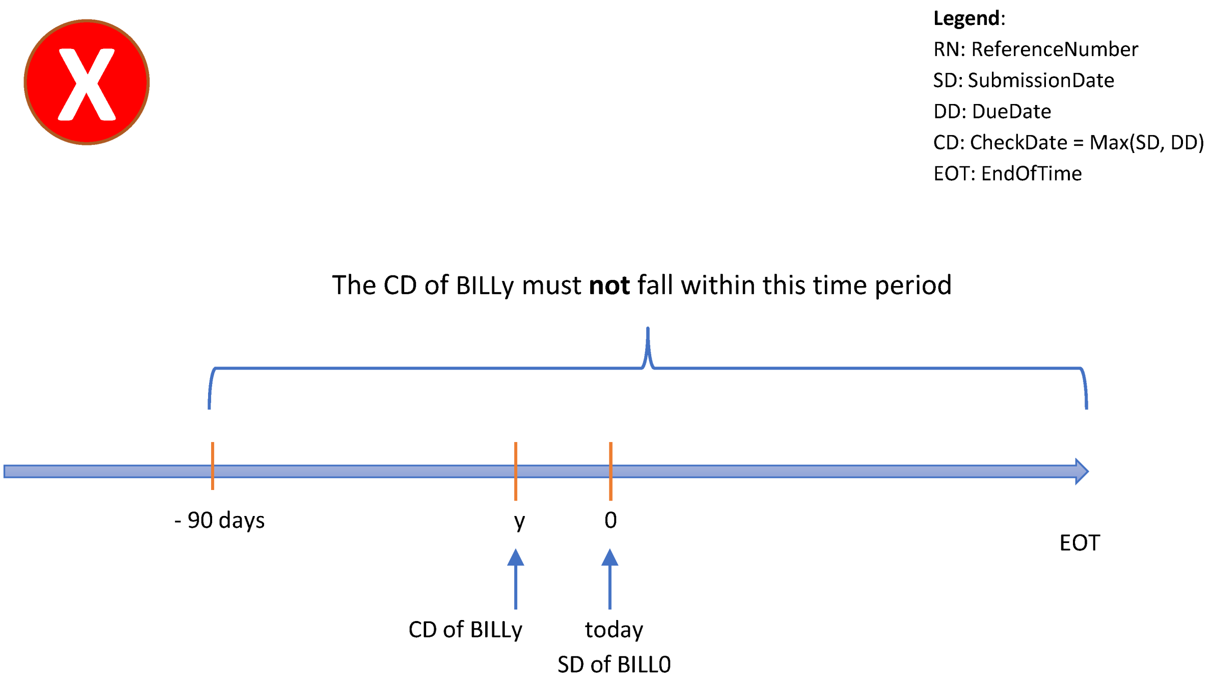


Figure 20. Scenario 1

Scenario 2:

. Let a BILL, BILLz with RN=1234 and check date, CD=z=today+30 days be present in the system

. Todays submission of bill, BILL0 with the same RN=1234 leads to rejection of BILL0.

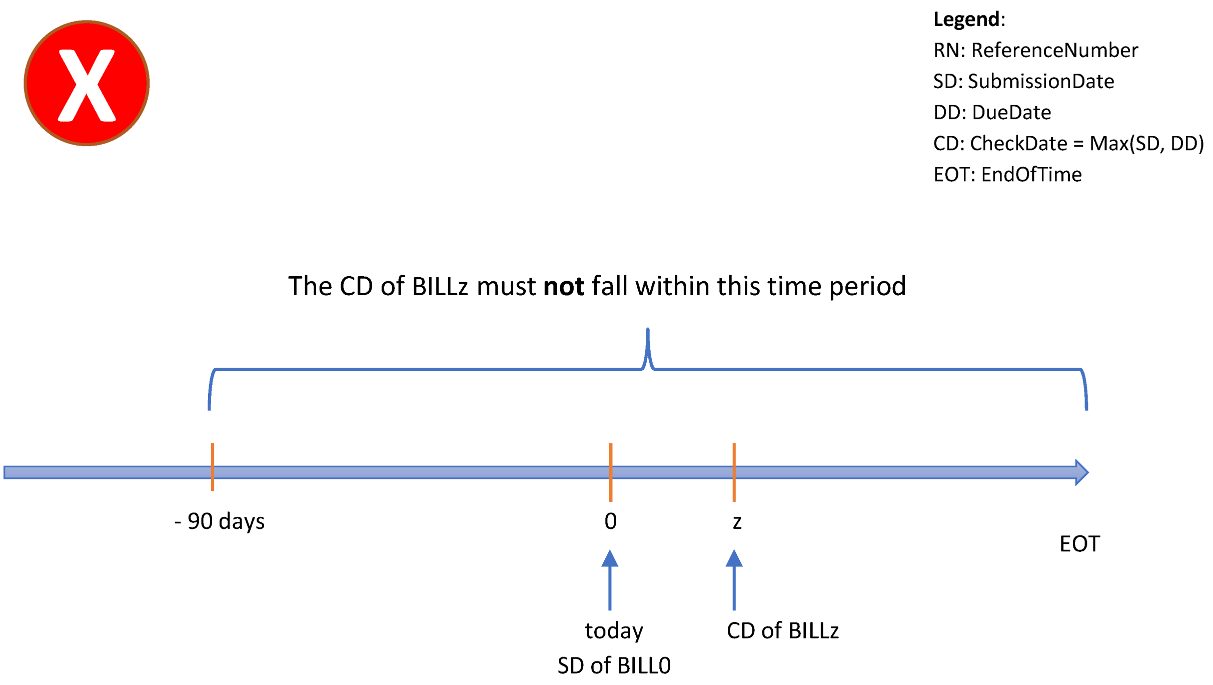


Figure 21. Scenario 2

Scenario 3:
. Let a bill, BILLw with RN=1234 and check date, CD=w=today-120 days be present in the system
. Todays submission of bill, BILL0 with the same RN=1234 is allowed.

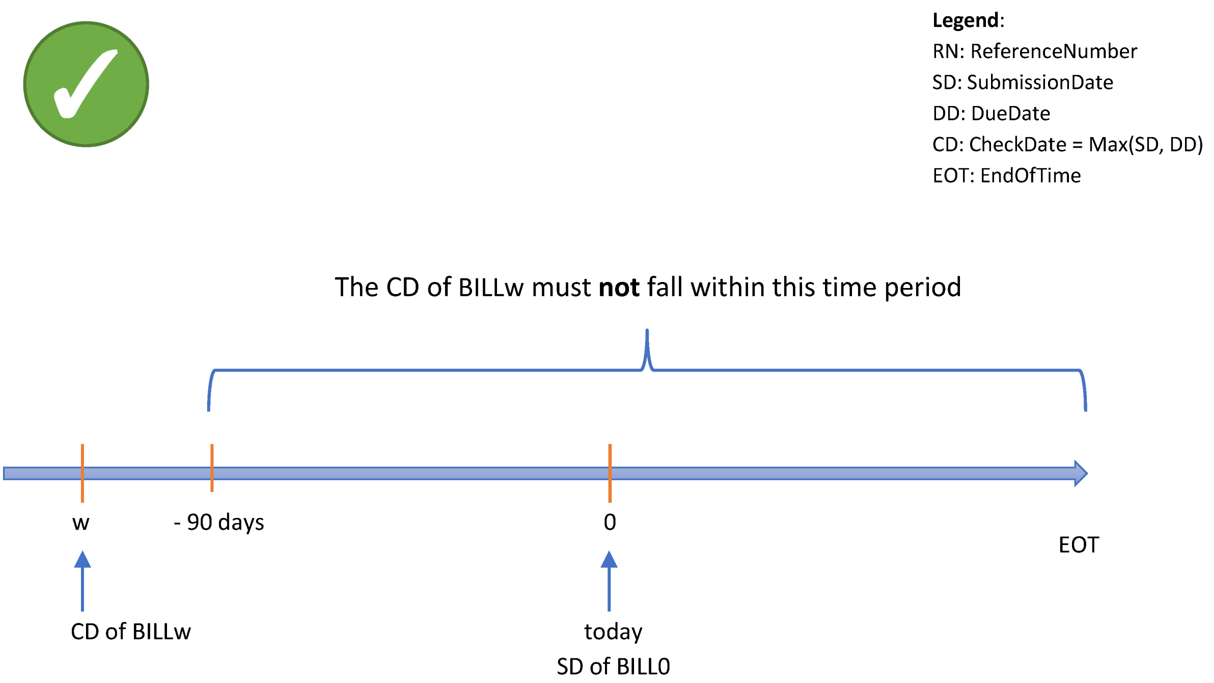


Figure 22. Scenario 3

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Header	**x-filename** *optional*	Filename for the business case PDF. This is only used for analytical purposes and support. The filename is not visible for the bill recipient. **Minimal length**: `1` **Maximal length**: `99`	string
Body	**body** *required*	PDF as binary data	string (binary)

Responses

HTTP Code	Description	Schema
201	Business case created	BusinessCase
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/pdf`

Produces

- `application/json`

4.1.14. Get business case

```
GET /billers/{billerId}/business-cases/{businessCaseId}
```

Description

Depending on the accept header, this operation either returns a JSON business case object or the original PDF.
The returned JSON object contains one of the business case subtypes: Bill, InstalmentBill, Reminder, CreditNote or Advice.

Validations

All validations according to Section 7.1, "Basic Validations" and some of Section 7.2, "Shared Validations".

Parameters

Type	Name	Description	Schema
Path	**billerId** *required*	biller id **Maximal length**: `14` **Example**: `"BIID0000123456"` **Pattern**: `BIID[0-9]{10}`	string
Path	**businessCaseId** *required*	business case id **Maximal length**: `36` **Example**: `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"` **Pattern**: `BCID[0-9A-Z]{32}`	string
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string

Type	Name	Description	Schema
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Business case found	BusinessCase
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`
- `application/pdf`

4.2. Bill-recipient-subscription

Bill Recipient Subscription
View 'bill recipient subscription requests' sent to the biller, using the subscription token on the subscription url from the biller. More information about the subscription process can be found in Section 3.3.3, “Subscriptions and Subscription Cancellations”.

4.2.1. Get bill recipient subscription information

```
GET /bill-recipient-subscriptions/{subscriptionToken}
```

Description

The `{subscriptionToken}` has a defined life time and can only be consumed once. If a token is too old or used more than once the call will fail with an validation error.

More details regarding bill recipient subscriptions can be found in Section 3.3.3, “Subscriptions and Subscription Cancellations”.

Validations

All validations according to Section 7.1, “Basic Validations” and Section 7.10, “Subscription Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Path	**subscriptionToken** *required*	the token from the biller subscription url	string

Responses

HTTP Code	Description	Schema
200	Bill recipient found	BillRecipientSubscription
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.3. Events

Notification Events
The event resource allows the network partner to retrieve all changes addressed to it. There is one specific operation for every type of event (for example: Subscription status changes or business case status changes), where the network partner can pull new changes from.

Guidelines for Polling the Events
Depending on the network partner, different types of events may be of interest. We recommend the network partner to consume those events of interest on a regular basis.
Generally it is recommended to poll each events endpoint once per hour, whereas it’s important to consume all events until the endpoint does not return any more events.
Please avoid querying events unnecessary often.

4.3.1. Find events for business cases which changed status

```
GET /events/business-case-status-changed
```

Description

Events for status changes of bills, advices, credit notes and reminders.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema	Default
Query	**lastEventId** *optional*	Id of the last received event. If omitted, the result starts with the oldest available event. **Example**: `"EVID82A65938766547EBBBA39BA6F7B07F24"` **Maximal length**: `36` **Pattern**: `"EVID[0-9A-Z]{32}"`	string	
Query	**limit** *optional*	maximum number of events one wants to receive (technical maximum is 10000, no more will be returned at once and you have to fetch again to check if there are more)	integer	`1000`
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string	
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string	

Responses

HTTP Code	Description	Schema
200	Business case status changed events found	< BusinessCaseStatusChangedEvent > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.3.2. Find events for instalments which changed status

```
GET /events/instalment-status-changed
```

Description

Events for status changes of instalment bills only.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema	Default
Query	**lastEventId** *optional*	Id of the last received event. If omitted, the result starts with the oldest available event. **Example**: `"EVID82A65938766547EBBBA39BA6F7B07F24"` **Maximal length**: `36` **Pattern**: `"EVID[0-9A-Z]{32}"`	string	
Query	**limit** *optional*	maximum number of events one wants to receive (technical maximum is 10000, no more will be returned at once and you have to fetch again to check if there are more)	integer	`1000`
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string	
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string	

Responses

HTTP Code	Description	Schema
200	Instalment status changed events found	< InstalmentStatusChangedEvent > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.3.3. Find events for bill recipient subscriptions which changed status

```
GET /events/bill-recipient-subscription-status-changed
```

Description

More information about the subscription process can be found in «Subscriptions and Subscription

Cancellations».

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema	Default
Query	**lastEventId** *optional*	Id of the last received event. If omitted, the result starts with the oldest available event. **Example**: `"EVID82A65938766547EBBBA39BA6F7B07F24"` **Maximal length**: `36` **Pattern**: `"EVID[0-9A-Z]{32}"`	string	
Query	**limit** *optional*	maximum number of events one wants to receive (technical maximum is 10000, no more will be returned at once and you have to fetch again to check if there are more)	integer	`1000`
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string	
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string	

Responses

HTTP Code	Description	Schema
200	Bill recipient subscription changed events found	< BillRecipientSubscriptionStatusChangedEvent > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.3.4. Find events for bill recipients email address changes

```
GET /events/bill-recipient-email-address-changed
```

Description

This event is triggered after a biller has submitted a business case with an outdated, so called historically available email address. It notifies about the changed email address of a bill recipient, which has been adjusted in eBill.

An email address is considered to be historically available if it was present up to 15 months prior to the submission time.

The billers are able to submit business cases with historically available email addresses of a bill recipient. However, latest 15 months after the email address changed, the billers are required to submit the business cases with the currently valid email address of the bill recipient.

Validations

All validations according to Section 7.1, “Basic Validations” and some of Section 7.2, “Shared Validations”.

Parameters

Type	Name	Description	Schema	Default
Query	**lastEventId** *optional*	Id of the last received event. If omitted, the result starts with the oldest available event. **Example**: `"EVID82A65938766547EBBBA39BA6F7B07F24"` **Maximal length**: `36` **Pattern**: `"EVID[0-9A-Z]{32}"`	string	
Query	**limit** *optional*	maximum number of events one wants to receive (technical maximum is 10000, no more will be returned at once and you have to fetch again to check if there are more)	integer	`1000`
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string	
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string	

Responses

HTTP Code	Description	Schema
200	Bill recipient email address changed events found	< BillRecipientEmailAddressChangedEvent > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.4. Sectors

Industry Sector
Industry sectors are valid system wide. Each biller will reference one or several industry sectors.

4.4.1. Find Sectors

```
GET /sectors
```

Description

Get the industry sector list of the eBill infrastructure. The sectors are more or less static and can therefore be cached.

Validations

All validations according to Section 7.1, “Basic Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Sectors found	< Sector > array
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.5. Healthcheck

System Healthcheck

This allows to check the basic state of the system (can it be reached, does authentication and auhorization work, does it respond).
As some infrastructures block certain HTTP methods by default, the healthcheck allows to test if all required methods (GET, POST, PUT, DELETE) work across all layers.

4.5.1. Health check using GET method

```
GET /healthcheck
```

Description

Returns a status message of the system.

Validations

All validations according to Section 7.1, “Basic Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
200	Healthcheck successful	HealthCheckResponse
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Produces

- `application/json`

4.5.2. Health check using PUT method

```
PUT /healthcheck
```

Description

Returns the request body. This operation will not modify the system.

Validations

All validations according to Section 7.1, “Basic Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Any message which is expected in the response	HealthCheckRequest

Responses

HTTP Code	Description	Schema
200	Healthcheck successful	HealthCheckResponse
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/json`

Produces

- `application/json`

4.5.3. Health check using POST method

```
POST /healthcheck
```

Description

Returns the request body. This operation will not modify the system.

Validations

All validations according to Section 7.1, “Basic Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string
Body	**body** *required*	Any message which is expected in the response	HealthCheckRequest

Responses

HTTP Code	Description	Schema
200	Healthcheck successful	HealthCheckResponse
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Consumes

- `application/json`

Produces

- `application/json`

4.5.4. Health check using DELETE method

```
DELETE /healthcheck
```

Description

Returns without further content. This operation will not modify the system.

Validations

All validations according to Section 7.1, “Basic Validations”.

Parameters

Type	Name	Description	Schema
Header	**authorization** *required*	OAuth 2.0 client credentials Access-Token with the prefix "Bearer".	string

Type	Name	Description	Schema
Header	**x-correlation-id** *required*	ID which will unambiguously identify this request to the API. **Minimal length**: `1` **Maximal length**: `36`	string

Responses

HTTP Code	Description	Schema
204	Healthcheck successful	No Content
default	See section Error Handling of the Network Partner API documentation for further details about errors and error handling.	Problem

Chapter 5. Security

5.1. HTTPS

The REST-API is exposed on an HTTPS endpoint supporting TLS 1.2 **only**.

5.2. oauth2_cc

OAuth2 based authentication with client credentials.

Type : oauth2
Flow : application
Token URL : https://api.six-group.com/auth/oauth/token

Name	Description
default-nwp	default scope for network partner

The OAuth2 Client Credentials implementation for the Network Partner API consists of two endpoints: an Authorization Server and the API Platform. The latter acts as a gateway to the Network Partner API.

As a first request, the network partner communicates to the Authorization Server to obtain a JWT token, all following requests (Network Partner API REST-calls) are directed to the API Platform:

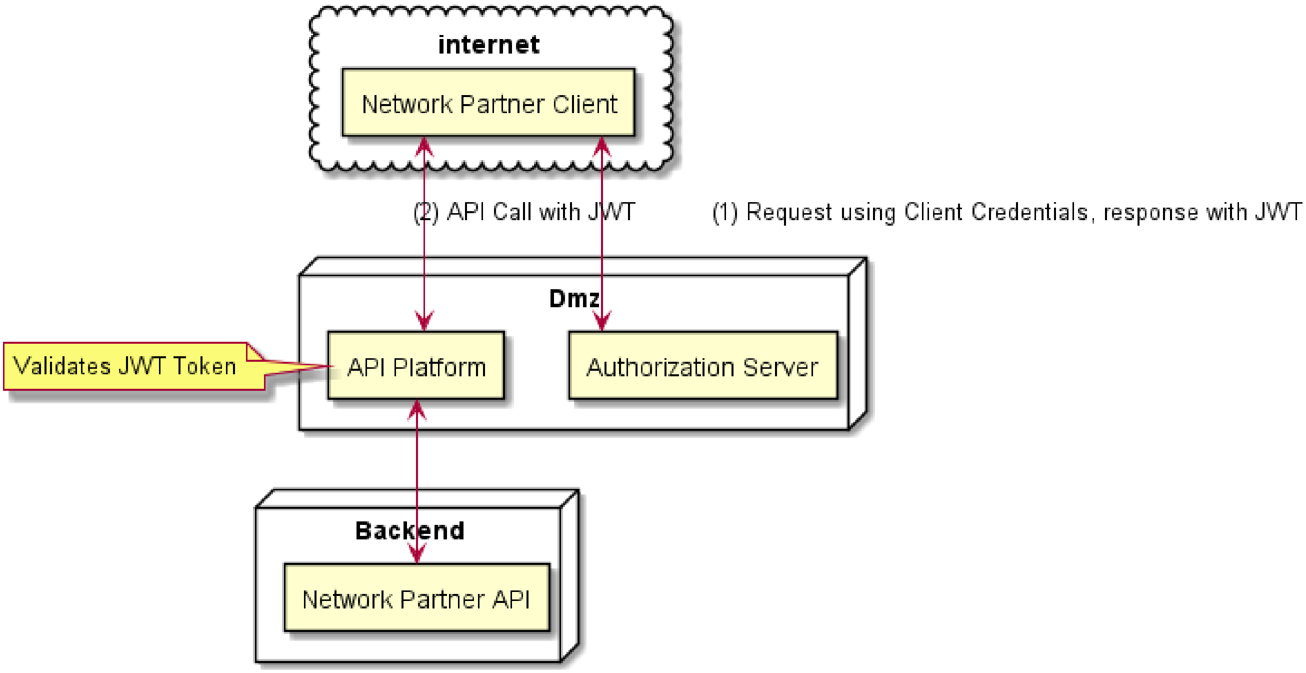


Figure 23. OAuth2 Flow Overview

The following sequence diagram shows the flow for OAuth2 Client Credentials:

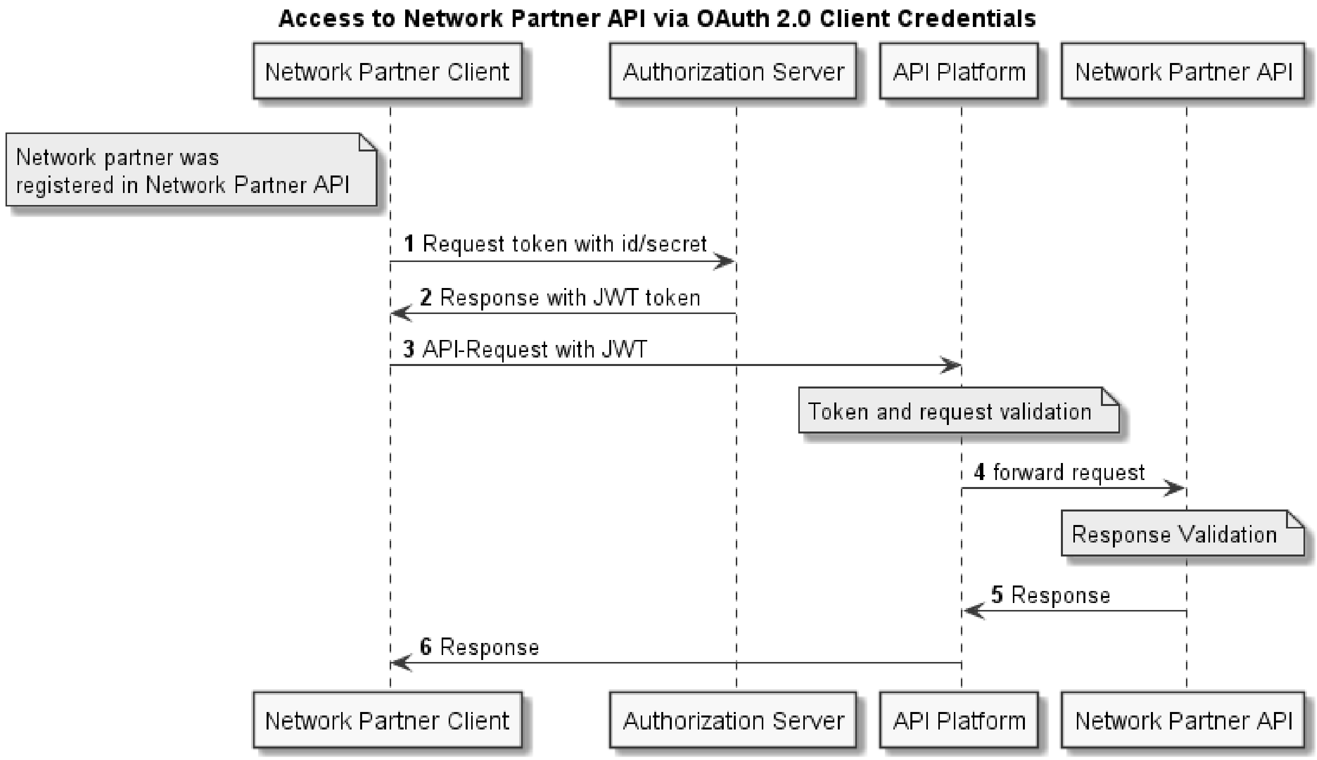


Figure 24. OAuth2 Flow Client Credentials

Requesting a JWT from the Authorization Server:

1. The Network Partner Client requests a valid token at the authorization server by sending a request with:
   a. The Base64 encoded "client_id:client_secret ", prefixed by "Basic" in the authorization header.
   b. The "grant_type" request parameter must be set to "client_credentials".
   c. Example:

   ```
   curl --request POST \
     --url https://api.six-group.com/auth/oauth/token \
     --cacert /network-partner-client/trust-store/ca-certs.crt \
     --header 'content-type: application/json' \
     --header 'authorization: Basic dGVzdDoxMjM0NTY=' \
     --data grant_type=client_credentials \
   ```

2. The Authorization Server Endpoint responds with a valid JWT token.
3. The Network Partner Client sends a request to the API Platform
   a. Sending the JWT token in the authorization header: "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhcGkuc2l4LWdyb3VwLmNvbSIsImV4cCI6MTMwMDgxOTM4MCwibmFtZSI6Ik5ldHdvcmtwYXJ0bmVyIDEiLCJzdWIiOiJOV0lEMDA5MDAwMDAwMSJ9.f73dL96IN9IUXbp-iqNhR6ARmOZ76e728pjOJzK7sik"
4. The API Platform validates the JWT Token and forwards the request
5. The Network Partner API sends a response

6. The API Platform forwards the response

5.2.1. Details about the Usage of JWT

A valid JWT-Token consists of three parts:

- Header:
  {"alg":"HS256","typ":"JWT"}
- Payload:
  {"iss":"api.six-group.com","exp":1300819380,"name":"Networkpartner 9001","sub":"NWID0090000001"}
- Signature:
  HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload),<your-256-bit-secret>)

Each of the three parts is itself a json-content that will be Base64-encoded:

- eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
- eyJpc3MiOiJhcGkuc2l4LWdyb3VwLmNvbSIsImV4cCI6MTMwMDgxOTM4MCwibmFtZSI6Ik5ldHdvcmtwYXJ0bmVyIDEiLCJzdWIiOiJOV0lEMDA5MDAwMDAwMSJ9
- f73dL96IN9IUXbp-iqNhR6ARmOZ76e728pjOJzK7sik

At the end the three parts are composed together using a dot in between:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhcGkuc2l4LWdyb3VwLmNvbSIsImV4cCI6MTMwMDgxOTM4MCwibmFtZSI6Ik5ldHdvcmtwYXJ0bmVyIDEiLCJzdWIiOiJOV0lEMDA5MDAwMDAwMSJ9.f73dL96IN9IUXbp-iqNhR6ARmOZ76e728pjOJzK7sik

Then, the composed Base64 encoded string is prefixed with Bearer and used as authorization header as explained above.

References:

For details on headers that the eBill infrastructure expects see Section 3.2.5, “HTTP Headers”.
For details on the OAuth Client Credentials Flow see https://tools.ietf.org/html/rfc6749 and https://www.oauth.com/oauth2-servers/access-tokens/client-credentials/
For details on JSON Web Tokens (JWT) see https://tools.ietf.org/html/rfc7519

Chapter 6. Definitions

6.1. BillerDetail

Polymorphism : Composition

Name	Description	Schema
id *optional*	A unique id for this biller. Property must not be given when creating a new biller. **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
status *required*	the status of the biller	enum (ACTIVE, INACTIVE)
legalName *required*	legal name of the company **Length** : `1 - 70` **Example** : `"Verlag Neuer"`	string
enterpriseIdentificationNumber *optional*	Swiss enterprise identification number (UID) without dashes, dots or extension. Note that this has to contain the swiss enterprise identification number (UID) from the commercial register (Handelsregister) which may be different from the VAT UID (Mehrwertsteuer UID). **Maximal length** : `12` **Pattern** : `"CHE[0-9]{9}"` **Example** : `"CHE123456789"`	string
billRecipientSubscriptionStatus *required*		BillRecipientSubscriptionStatus
localizedData *required*		localizedData
sectorIds *required*	list of assigned sector ids to the biller	< string > array
accounts *required*	list of accounts from the biller	< BillerAccount > array
billRecipientSubscriptionUrl *optional*	The subscription url for this biller. It is called with a subscriptionToken added to the billRecipientSubscriptionUrl. The token can be used to get bill recipient information. More information about the subscription process are in the general section. **Length** : `1 - 1000`	string

localizedData

Name	Schema
defaultLanguage *required*	DefaultLanguage
ger *optional*	LocalizedBillerData
fre *optional*	LocalizedBillerData
ita *optional*	LocalizedBillerData
eng *optional*	LocalizedBillerData

6.2. Biller

Name	Description	Schema
id *optional*	A unique id for this biller. Property must not be given when creating a new biller. **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
status *required*	the status of the biller	enum (ACTIVE, INACTIVE)
legalName *required*	legal name of the company **Length** : `1 - 70` **Example** : `"Verlag Neuer"`	string
enterpriseIdentificationNumber *optional*	Swiss enterprise identification number (UID) without dashes, dots or extension. Note that this has to contain the swiss enterprise identification number (UID) from the commercial register (Handelsregister) which may be different from the VAT UID (Mehrwertsteuer UID). **Maximal length** : `12` **Pattern** : `"CHE[0-9]{9}"` **Example** : `"CHE123456789"`	string
billRecipientSubscriptionStatus *required*		BillRecipientSubscriptionStatus
localizedData *required*		localizedData
sectorIds *required*	list of assigned sector ids to the biller	< string > array
accounts *required*	list of accounts from the biller	< BillerAccount > array

localizedData

Name	Schema
defaultLanguage *required*	DefaultLanguage
ger *optional*	LocalizedBillerData
fre *optional*	LocalizedBillerData
ita *optional*	LocalizedBillerData
eng *optional*	LocalizedBillerData

6.3. LocalizedBillerData

Name	Description	Schema
displayName *required*	display name which will be used for presentation **Length** : `1 - 100` **Example** : `"Neuers Neuste Nachrichten"`	string
emailAddress *optional*	Email address to contact the biller. It is recommended to provide a specific address for questions concerning eBill, if available. **Length** : `1 - 256` **Example** : `"nnn@nnn-verlag.info"`	string (email)
phone *optional*	Phone number to contact the biller. It is recommended to provide a specific phone number for questions concerning eBill, if available. Both the countryCode and the nationalNumber are as defined in International Telecommunication Union (ITU) Recommendation E.164, without any leading zeros.	phone
website *optional*	Website to contact the biller. It is recommended to provide a specific website for questions concerning eBill, if available. **Length** : `1 - 1000` **Example** : `"http://www.nnn-verlag.info"`	string
address *required*		BillerAddress
logo *optional*	Biller logos may be provided and will be scaled down to the necessary size. Contains the reference to a logo as binary from the /biller/{billerId}/assets endpoint. For create/update: if assetId is empty, a new assetId will be returned, were the logo can be uploaded.	logo

phone

Name	Description	Schema
countryCode *required*	**Maximal length**: `6` **Maximum value** : `999999` **Example** : `41`	integer
nationalNumber *required*	**Maximal length**: `18` **Example** : `446681800`	integer (int64)

logo

Name	Description	Schema
name *optional*	the name of the asset for create/update: will be taken as asset name if provided **Length** : `1 - 256` **Example** : `"company-logo-de"`	string
assetId *optional*	asset id **Maximal length** : `14` **Pattern** : `"ASID[0-9]{10}"` **Example** : `"ASID0000000001"`	string

6.4. BillerAddress

Name	Description	Schema
streetName *required*	street name **Length** : `1 - 70` **Example** : `"Neustadtstrasse"`	string
buildingNumber *optional*	building number **Length** : `1 - 16` **Example** : `"20a"`	string
postalCode *required*	postal code **Length** : `1 - 9` **Example** : `"6025"`	string
city *required*	city name **Length** : `1 - 35` **Example** : `"Neudorf"`	string
countryCode *required*	in format ISO 3166-1 alpha 2 **Maximal length** : `2` **Pattern** : `"[A-Z]{2}"` **Example** : `"CH"`	string

6.5. RecipientAddress

Name	Description	Schema
streetName *required*	street name **Length** : `1 - 70` **Example** : `"Neustadtstrasse"`	string
postalCode *required*	postal code **Length** : `1 - 9` **Example** : `"6025"`	string
city *required*	city name **Length** : `1 - 35` **Example** : `"Neudorf"`	string
countryCode *required*	in format ISO 3166-1 alpha 2 **Maximal length** : `2` **Pattern** : `"[A-Z]{2}"` **Example** : `"CH"`	string

6.6. BillerAccount

Name	Description	Schema
accountNumber *required*		AccountNumber
iid *required*	The required institution identification of the associated financial institute. Please use leading zeroes to reach the 5 digit value. **Length** : `5` **Pattern** : `"[0-9]{5}"` **Example** : `"00100"`	string
besr *optional*	BESR-ID of the account. The BESR-ID is used during business case validation (if a BESR-ID is defined the ESR reference *has to* start with this exact digits) and during direct subscription from online banking (if a BESR-ID is defined the ESR reference *has to* start with these exact digits to produce a match for direct subscription). If the accountNumber object contains an ESR participant number and the IID is a financial institution other than PostFinance (IID != 09000) then the BESR field has to contain a value. **Length** : `6 - 11` **Pattern** : `"[0-9]{6,11}"` **Example** : `"123456"`	string
currencyCode *required*	currency code in ISO-4217 **Maximal length** : `3` **Pattern** : `"[A-Z]{3}"` **Example** : `"CHF"`	string

6.7. BillerAttachment

Biller attachments have to be provided in at least one language. The attachment in the default language has to be provided. The default language will be used to select a fallback if the requested language is not available.

It contains the reference to a binary from the /biller/{billerId}/assets endpoint.

For create/update: if assetId is empty, a new assetId will be returned, that can be used to upload the corresponding binaries of the attachment.

Name	Description	Schema
id *optional*	A unique id for this attachment. Property must not be given when creating a new attachment. **Maximal length** : `14` **Pattern** : `"ATID[0-9]{10}"` **Example** : `"ATID0000123456"`	string
billerId *optional*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
presentFrom *required*	the valid from date (ISO-8601 format) **Example** : `"2015-01-01"`	string (date)
presentTo *required*	the valid to date (ISO-8601 format) **Example** : `"2015-12-31"`	string (date)
localizedData *required*		localizedData

localizedData

Name	Schema
defaultLanguage *required*	DefaultLanguage
ger *optional*	LocalizedBillerAttachment
fre *optional*	LocalizedBillerAttachment
ita *optional*	LocalizedBillerAttachment
eng *optional*	LocalizedBillerAttachment

6.8. LocalizedBillerAttachment

Name	Description	Schema
label *required*	label of the attachment **Length** : `1 - 256` **Example** : `"Informationen zu Steuern"`	string

Name	Description	Schema
name *required*	the name of the asset for create/update: will be taken as asset name if provided **Length** : `1 - 256` **Example** : `"steuer-pdf-2018"`	string
assetId *optional*	asset id **Maximal length** : `14` **Pattern** : `"ASID[0-9]{10}"` **Example** : `"ASID0000000001"`	string

6.9. BillRecipientsForBillerSearch

Name	Description	Schema
emailAddress *optional*	email address of the bill recipient **Length** : `1 - 256` **Example** : `"peter@muster.ch"`	string (email)
billRecipientId *optional*	id of the bill recipient **Length** : `1 - 17` **Pattern** : `"([0-9])*"` **Example** : `"41010560425610173"`	string
enterpriseIdentificationNumber *optional*	Swiss enterprise identification number (UID) without dashes, dots or extension. Note that this has to contain the swiss enterprise identification number (UID) from the commercial register (Handelsregister) which may be different from the VAT UID (Mehrwertsteuer UID). **Maximal length** : `12` **Pattern** : `"CHE[0-9]{9}"` **Example** : `"CHE123456789"`	string

6.10. BillRecipientSubscription

Name	Description	Schema
billerId *required*	**Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"`**Example** : `"BIID0000123456"`	string
billRecipient *required*		BillRecipient
accountNumber *optional*	Account number of the biller (e.g. iban), if provided from the financial institution **Maximal length** : `21`	string

Name	Description	Schema
esrReference *optional*	ESR reference number or QR reference number, if provided from the financial institution. *Note for future API versions: esrReference will be renamed to referenceStructured in version 2.* **Length** : `16 - 27` **Pattern** : `"([0-9])*"` **Example** : `"123456123456789012345678901"`	string

6.11. BillRecipient

Name	Description	Schema
emailAddress *optional*	email address of the bill recipient **Length** : `1 - 256` **Example** : `"peter@muster.ch"`	string (email)
billRecipientId *required*	id of the bill recipient **Length** : `1 - 17` **Pattern** : `"([0-9])*"` **Example** : `"41010560425610173"`	string
enterpriseIdentificationNumber *optional*	Swiss enterprise identification number (UID) without dashes, dots or extension. Note that this has to contain the swiss enterprise identification number (UID) from the commercial register (Handelsregister) which may be different from the VAT UID (Mehrwertsteuer UID). **Maximal length** : `12` **Pattern** : `"CHE[0-9]{9}"` **Example** : `"CHE123456789"`	string
type *required*	the type of the bill recipient	enum (PRIVATE, COMPANY)
name *required*	last name, if private bill recipient company name, if company bill recipient **Length** : `1 - 70` **Example** : `"for private bill recipient: Muster, for company name: Muster AG"`	string
firstName *optional*	first name, if private bill recipient empty, if company bill recipient **Maximal length** : `35` **Example** : `"Peter"`	string
correspondenceLanguage *required*	language for correspondence with this bill recipient ISO-639-2/B **Length** : `1 - 3` **Example** : `"ger"`	string
address *required*		RecipientAddress

6.12. Bill

A business case of type Bill

Polymorphism : Inheritance
Discriminator : type

Name	Description	Schema
id *required*	A unique id for this business case. Property must not be given when creating a new business case. **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
type *required*	the type of the business case	enum (Bill, InstalmentBill, Advice, CreditNote, Reminder)
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
referenceNumber *required*	A business case reference given by the biller. Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string
referencedBill *optional*		ReferencedBill
businessCaseDate *required*	The business case date (ISO-8601 format), can not be more than 90 days in the past on the date it was created. Cannot be in the future. **Example** : `"2017-12-22"`	string (date)
status *optional*	the status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
totalAmount *required*		AmountWithCurrency

6.13. InstalmentBill

A business case of type InstalmentBill

Polymorphism : Inheritance
Discriminator : type

Name	Description	Schema
id *required*	A unique id for this business case. Property must not be given when creating a new business case. **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
type *required*	the type of the business case	enum (Bill, InstalmentBill, Advice, CreditNote, Reminder)
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
referenceNumber *required*	A business case reference given by the biller. Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string
referencedBill *optional*		ReferencedBill
businessCaseDate *required*	The business case date (ISO-8601 format), can not be more than 90 days in the past on the date it was created. Cannot be in the future. **Example** : `"2017-12-22"`	string (date)
status *optional*	the status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
totalAmount *required*		AmountWithCurrency

6.14. CreditNote

A business case of type CreditNote

Polymorphism : Inheritance
Discriminator : type

Name	Description	Schema
id *required*	A unique id for this business case. Property must not be given when creating a new business case. **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string

Name	Description	Schema
type *required*	the type of the business case	enum (Bill, InstalmentBill, Advice, CreditNote, Reminder)
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
referenceNumber *required*	A business case reference given by the biller. Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string
referencedBill *optional*		ReferencedBill
businessCaseDate *required*	The business case date (ISO-8601 format), can not be more than 90 days in the past on the date it was created. Cannot be in the future. **Example** : `"2017-12-22"`	string (date)
status *optional*	the status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
totalAmount *required*		AmountWithCurrency

6.15. Reminder

A business case of type Reminder

Polymorphism : Inheritance
Discriminator : type

Name	Description	Schema
id *required*	A unique id for this business case. Property must not be given when creating a new business case. **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
type *required*	the type of the business case	enum (Bill, InstalmentBill, Advice, CreditNote, Reminder)

Name	Description	Schema
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
referenceNumber *required*	A business case reference given by the biller. Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string
referencedBill *optional*		ReferencedBill
businessCaseDate *required*	The business case date (ISO-8601 format), can not be more than 90 days in the past on the date it was created. Cannot be in the future. **Example** : `"2017-12-22"`	string (date)
status *optional*	the status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
totalAmount *required*		AmountWithCurrency

6.16. Advice

A business case of type Advice

Polymorphism : Inheritance
Discriminator : type

Name	Description	Schema
id *required*	A unique id for this business case. Property must not be given when creating a new business case. **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
type *required*	the type of the business case	enum (Bill, InstalmentBill, Advice, CreditNote, Reminder)
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string

Name	Description	Schema
referenceNumber *required*	A business case reference given by the biller. Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string
referencedBill *optional*		ReferencedBill
businessCaseDate *required*	The business case date (ISO-8601 format), can not be more than 90 days in the past on the date it was created. Cannot be in the future. **Example** : `"2017-12-22"`	string (date)
status *optional*	the status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
totalAmount *required*		AmountWithCurrency

6.17. BusinessCase

the abstract business case object containing the shared properties

Name	Description	Schema
id *required*	A unique id for this business case. Property must not be given when creating a new business case. **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
type *required*	the type of the business case	enum (Bill, InstalmentBill, Advice, CreditNote, Reminder)
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
referenceNumber *required*	A business case reference given by the biller. Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string
referencedBill *optional*		ReferencedBill

Name	Description	Schema
businessCaseDate *required*	The business case date (ISO-8601 format), can not be more than 90 days in the past on the date it was created. Cannot be in the future. **Example** : `"2017-12-22"`	string (date)
status *optional*	the status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
totalAmount *required*		AmountWithCurrency

6.18. ReferencedBill

The business case can only bills or instalment bills.

Name	Description	Schema
businessCaseId *optional*	business case id **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
referenceNumber *required*	Must be unique for 90 days, calculated from the due date, together with the billerId. Old invoice numbers can then be recycled. **Length** : `1 - 120` **Example** : `"2018-123456-22"`	string

6.19. Sector

Name	Description	Schema
id *optional*	A unique id for this sector. **Maximal length** : `14` **Pattern** : `"SCID[0-9]{10}"` **Example** : `"SCID0000000000"`	string
localizedData *optional*		localizedData

localizedData

Name	Schema
ger *optional*	LocalizedSectorData
fre *optional*	LocalizedSectorData
ita *optional*	LocalizedSectorData

Name	Schema
eng *optional*	LocalizedSectorData

6.20. LocalizedSectorData

Name	Description	Schema
name *required*	name of the sector **Length** : `1 - 36` **Example** : `"Transport"`	string

6.21. Event

contains common fields for all events

Name	Description	Schema
eventId *required*	event id **Maximal length** : `36` **Pattern** : `"EVID[0-9A-Z]{32}"` **Example** : `"EVID82A65938766547EBBBA39BA6F7B07F24"`	string
timestamp *required*	timestamp of the event **Example** : `"2015-01-01T10:00:00.000Z"`	string (date-time)

6.22. BusinessCaseStatusChangedEvent

An Event describing the status change of a business case
These events are sent for bills, reminders, advices and credit notes.
The approved amount is only provided for the status APPROVED.

Polymorphism : Composition

Name	Description	Schema
eventId *required*	event id **Maximal length** : `36` **Pattern** : `"EVID[0-9A-Z]{32}"` **Example** : `"EVID82A65938766547EBBBA39BA6F7B07F24"`	string
timestamp *required*	timestamp of the event **Example** : `"2015-01-01T10:00:00.000Z"`	string (date-time)
billerId *optional*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
businessCaseId *optional*	business case id **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string

Name	Description	Schema
newStatus *optional*	the new status of the business case	enum (OPEN, APPROVED, REJECTED, COMPLETED)
approvedAmount *optional*		AmountWithCurrency

6.23. BillRecipientEmailAddressChangedEvent

An Event describing the change of the email address of a bill recipient.

Polymorphism : Composition

Name	Description	Schema
eventId *required*	event id **Maximal length** : `36` **Pattern** : `"EVID[0-9A-Z]{32}"` **Example** : `"EVID82A65938766547EBBBA39BA6F7B07F24"`	string
timestamp *required*	timestamp of the event **Example** : `"2015-01-01T10:00:00.000Z"`	string (date-time)
oldEmailAddress *optional*	the old email address of the bill recipient which has been used in the submission of a business case **Length** : `1 - 256` **Example** : `"peter@muster.ch"`	string (email)
newEmailAddress *optional*	the new email address of the bill recipient **Length** : `1 - 256` **Example** : `"peter_new@muster.ch"`	string (email)
triggeredBy *optional*		triggeredBy

triggeredBy

Name	Description	Schema
businessCaseId *required*	business case id **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
billerId *required*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string

6.24. InstalmentStatusChangedEvent

An Event describing the status change of an instalment.
These events are only sent for instalment bills.
The approved amount is only provided for the status APPROVED.

Polymorphism : Composition

Name	Description	Schema
eventId *required*	event id **Maximal length** : `36` **Pattern** : `"EVID[0-9A-Z]{32}"` **Example** : `"EVID82A65938766547EBBBA39BA6F7B07F24"`	string
timestamp *required*	timestamp of the event **Example** : `"2015-01-01T10:00:00.000Z"`	string (date-time)
billerId *optional*	biller id **Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"` **Example** : `"BIID0000123456"`	string
businessCaseId *optional*	business case id **Maximal length** : `36` **Pattern** : `"BCID[0-9A-Z]{32}"` **Example** : `"BCID0FB909852BBC4D06AD8336AAE87D7FC9"`	string
externalPaymentByInstalmentsId *optional*	external id of the respective paymentByInstalment **Length** : `1 - 36` **Example** : `"298031-2999"`	string
externalInstalmentId *optional*	external id of the instalment **Length** : `1 - 36` **Example** : `"298031-2999-ACX01"`	string
newStatus *optional*	the new status of the instalment	enum (OPEN, APPROVED, REJECTED, COMPLETED)
approvedAmount *optional*		AmountWithCurrency

6.25. AmountWithCurrency

Name	Description	Schema
value *required*	The amount value. Take care when using JavaScript libraries to parse this value - it should be treated as a financial amount and therefore not as a floating point number but rather using a precise decimal representation (like BigDecimal in Java). **Maximum value**: `99'999'999.99` **Maximal length**: `10` **Example** : `99.99`	number
currencyCode *required*	The amount currency code according to ISO-4217. **Maximal length** : `3` **Pattern** : `"[A-Z]{3}"` **Example** : `"CHF"`	string

6.26. BillRecipientSubscriptionStatusChangedEvent

An Event describing the status change of a bill recipient subscription.

Polymorphism : Composition

Name	Description	Schema
eventId *required*	event id **Maximal length** : `36` **Pattern** : `"EVID[0-9A-Z]{32}"` **Example** : `"EVID82A65938766547EBBBA39BA6F7B07F24"`	string
timestamp *required*	timestamp of the event **Example** : `"2015-01-01T10:00:00.000Z"`	string (date-time)
billerId *required*	**Maximal length** : `14` **Pattern** : `"BIID[0-9]{10}"`**Example** : `"BIID0000123456"`	string
billRecipient *required*		BillRecipient
accountNumber *optional*	Account number of the biller (e.g. iban), if provided from the financial institution **Maximal length** : `21`	string
esrReference *optional*	ESR reference number or QR reference number, if provided from the financial institution. *Note for future API versions: esrReference will be renamed to referenceStructured in version 2.* **Length** : `16 - 27` **Pattern** : `"([0-9])*"` **Example** : `"123456123456789012345678901"`	string
newStatus *optional*	the new status of the bill recipient subscription, currently only REQUESTED and INACTIVE are used, see «Subscriptions and Subscription Cancellations» for further information	enum (INACTIVE, REQUESTED, ACTIVE)

6.27. AccountNumber

For the account number, either IBAN or ESR participant number has to be specified (not both).

Name	Description	Schema
esrParticipantNumber *optional*	ESR participant number, format according to the specifications: * numerical value with mandatory 9 digits long (unhyphenated) * structure: VV999999P * VV = preceding digits, either 01 or 03 * 999999: reference number where a reference number is only 6 digits long, additional zeros must be inserted on the left hand side reference number should be greater than 000000 * P: check digit in accordance with Modulo 10, recursive **Maximal length** : `9` **Pattern** : `"[0-9]{9}"` **Example** : `"010001628"`	string
iban *optional*	credit account * The requirements for IBAN usage are limited to the country codes CH and LI, otherwise the business case will be rejected. * The IBAN should match an already existing credit account of the biller, otherwise the business case will be rejected. * See also ISO-13616-1 **Maximal length** : `21` **Pattern** : `"(CH|LI)\\d{7}[0-9A-Z]{12}"` **Example** : `"CH100023000A109822346"`	string

6.28. Problem

Name	Description	Schema
type *optional*	An absolute URI that identifies the problem type. We may provide human-readable documentation for the problem type in the future, when the URI is dereferenced. For now consult the network partner API documentation for further information. The URI consists of the /problems endpoint and the documented problem type, see example. **Default** : `"about:blank"` **Example** : `"/problems/REQUEST_BODY_VALIDATION_FAILED"`	string (uri)
title *optional*	A short, human readable summary of the problem type. **Example** : `"Payload has missing or invalid values"`	string
status *optional*	The HTTP status code generated by the origin server for this occurrence of the problem. **Minimum value** : `100` **Maximum value (exclusive)** : `600` **Example** : `400`	integer (int32)
detail *optional*	A human readable explanation specific to this occurrence of the problem. **Example** : `"The submitted request contains invalid or missing data which can not be processed."`	string
instance *optional*	An absolute URI that identifies the specific occurrence of the problem. It may or may not yield further information if dereferenced. **Example** : `"/api/pns/networkpartner/v1/billers/errors/NWID0090000001/provided-x-correlation-id"`	string (uri)
technicalReason *optional*	The Technical Description/Reason for engineers might contain addition system information on the problem. **Example** : `"Some field validations failed"`	string
fieldErrors *optional*		< fieldErrors > array

fieldErrors

Name	Description	Schema
fieldName *optional*	the name of the field with the error **Example** : `"localizedData.ger.address.city"`	string
message *optional*	the message describing the error **Example** : `"size must be between 1 and 35"`	string
rejectedValue *optional*	the provided value which was rejected if available **Example** : `"Very Long Invalid City Name Which Is Rejected"`	string

6.29. HealthCheckRequest

Name	Description	Schema
message *required*	expected response message from health check **Length** : `1 - 100` **Example** : `"any string"`	string

6.30. HealthCheckResponse

Name	Description	Schema
message *required*	response message from health check **Maximal length** : `100` **Example** : `"The healthcheck GET request was successfully received and processed"`	string
requestDateTime *required*	according to RFC3339, section 5.6 in ISO 8601 with timezone and milliseconds **Example** : `"2018-10-03T16:03:09.101+02:00"`	string (date-time)
receivedHeaders *required*		< receivedHeaders > array
environmentStage *required*	the instance which the request was sent to **Example** : `"XE"`	string
applicationVersion *required*	the version of the eBill infrastructure **Example** : `"1.4.3.0-desire-20180927091004161-71-5e3ca91"`	string
apiVersion *required*	the version of the network partner api **Example** : `"1.0.23"`	string

receivedHeaders

Name	Description	Schema
headerName *optional*	the name of the provided header **Example** : `"x-correlation-id"`	string
headerValue *optional*	As received **Example** : `"9bcd4351-4b7b-4017-9b63-9613414c6ff1"`	string

6.31. DefaultLanguage

From the provided localizedData, one has to be marked as the default language.
The eBill infrastructure will use this localizedData in case a user requests a language where no localizedData have been provided.

Type : enum (ger, fre, ita, eng)

6.32. BillRecipientSubscriptionStatus

If allowed, the biller can be found and bill recipients can subscribe with this biller.

Type : enum (ALLOWED, NOT_ALLOWED)

Chapter 7. Problem Descriptions Overview

7.1. Basic Validations

UNAUTHORIZED
Status code: 401
Request failed authorization validation
All requests require a valid JWT token.
For more information on the JWT token, see network partner API documentation in Chapter 5, *Security*.

REQUEST_URL_MALFORMED
Status code: 400
Request failed url parsing
The request URL must be a valid normalized URL.
Request URLs must not contain path traversal sequences or multiple forward slashes.

NETWORK_PARTNER_UNKNOWN
Status code: 401
Network partner is unknown
The NWID in the JWT of the request has to reference a known and active network partner in the system.
This also requires a valid JWT token, see network partner API documentation in Chapter 5, *Security*.

NETWORK_PARTNER_NOT_ACTIVE
Status code: 401
Network partner is not active
The NWID in the JWT of the request has to reference an active network partner in the system.
This also requires a valid JWT token, see network partner API documentation in Chapter 5, *Security*.

CORRELATION_ID_NOT_UNIQUE
Status code: 400
HTTP header x-correlation-id is not unique
The correlation id must be unique for every network partner in the the system.

REQUEST_CONTENT_TYPE_UNSUPPORTED
Status code: 415
The content type of the request is unsupported
Content type of a POST or PUT request should be set to the allowed content type for this resource, see network partner API documentation for details.
For JSON content this has to be application/json, an image asset must be one of image/png, image/jpeg or image/gif and for PDF content it has to be application/pdf. Other content types are not supported.

REQUEST_METHOD_NOT_ALLOWED
Status code: 405
HTTP method of the request is not allowed
HTTP method of a request is validated and has to be one of the supported methods for this

resource, see network partner API documentation for details.

INVALID_RESOURCE
Status code: 404
Resource not found
The requested resource or endpoint does not exist

REQUEST_PAYLOAD_EXCEEDS_MAXIMUM_SIZE
Status code: 413
Payload of the request is too large
Submitted POST or PUT request payload data should be smaller than 10'000'000 bytes, otherwise it is discarded without further analysis.

REQUEST_PAYLOAD_IS_EMPTY
Status code: 400
Payload of the request is empty
Empty payload content is not allowed for POST or PUT.

REQUEST_BODY_VALIDATION_FAILED
Status code: 400
Payload has missing or invalid values
The submitted request contains invalid or missing data in its payload and can not be processed.

REQUEST_HEADER_VALIDATION_FAILED
Status code: 400
Header is missing or has invalid value
The submitted request contains an invalid or missing header and can not be processed.

7.2. Shared Validations

NETWORK_PARTNER_OPERATION_NOT_ALLOWED
Status code: 403
Requested operation not allowed for network partner
There are restrictions regarding access to some resources. See network partner API documentation in Section 2.3, “Primary Network Partners” for details.

REQUEST_QUERY_PARAMETER_VALIDATION_FAILED
Status code: 400
Request query parameter has missing or invalid values
The submitted request contains invalid or missing query parameters which can not be processed. This problem type is used in case of missing or invalid input values.

INVALID_EMAIL
Status code: 400
Payload contains not well formed email address
The submitted request contains a not well formatted email address which can not be processed.

INVALID_CURRENCY_CODE
Status code: 400

Payload contains invalid currency code
The submitted request contains a currency code which can not be processed.
The currency code must be according to ISO-4217.

INVALID_ASSET_ID
Status code: 400
The assetId is invalid
The provided assetId is invalid as it doesn't match the existing assetId, which was generated by the system.

ASSET_ID_MUST_NOT_BE_PROVIDED
Status code: 400
The assetId must not be provided by the client
The assetId must not be provided by the client, it will be generated by the system.

LOCALIZED_DATA_FOR_DESIRED_DEFAULT_LANGUAGE_MISSING
Status code: 400
Payload does not contain the localized data for the given default language
For the specified default language no localized data could be found in the request. Make sure for the referenced localizedData.language you also provide the data set.

7.3. Biller Validations

BILLER_ACCOUNT_MISSING_BESRID
Status code: 400
Biller account definition is missing a BESR-ID
If the accountNumber contains an ESR participant number and the IID is a financial institution other than PostFinance (IID != 09000) then the BESR field has to contain a value.

BILLER_ACCOUNT_BESRID_NOT_ALLOWED
Status code: 400
Biller account BESR-ID is not supported as the account is of type IBAN or ESR having IID is the financial institution PostFinance (IID = 09000)
The BESR-ID is only allowed and required for ESR account types and if the IID of the account is a financial institution other than PostFinance (IID != 09000).

BILLER_ACCOUNT_BESRID_COMBINATION_ALREADY_EXISTS
Status code: 400
Biller account number / BESR-ID combination already exists in the system
None of the biller account and optional BESR-ID combinations are allowed to match with any existing account information.

BILLER_ACCOUNT_CURRENCY_CODE_COMBINATION_REJECTED
Status code: 400
Biller account / currency code combination is not allowed
The currency code of the biller account has to match the allowed currencies for the account type. Account type ESR only supports payment type 1 and therefore currency code is limited to CHF or EUR. Account type QR-IBAN only supports payment type 3 and therefore currency code is limited to CHF or EUR.

See the definition of payment types in https://www.six-interbank-clearing.com/dam/downloads/en/standardization/iso/swiss-recommendations/implementation-guidelines-ct.pdf.
More information about QR in
https://www.paymentstandards.ch/dam/downloads/ig-qr-bill-en.pdf.

BILLER_ACCOUNT_ESR_PARTICIPANT_NUMBER_CURRENCY_INCONSISTENCY
Status code: 400
Biller account ESR participant number is not consistent with currency code
Biller account inconsistency between ESR participant number and currency code found. Participant numbers starting with 01 must refer to CHF, those with 03 to EUR.

BILLER_ACCOUNT_INVALID_ACCOUNT_NUMBER_COMBINATION
Status code: 400
Validation of biller account number failed
The submitted biller account number must be either IBAN or ESR participant number.
The submission of ESR participant number and IBAN at the same time is not allowed.

BILLER_NAME_HRUID_COMBINATION_ALREADY_EXISTS
Status code: 400
Biller name / UID combination already exists in the system
The combination of display name and swiss enterprise identification number (UID) from the commercial register (Handelsregister) must be unique for every biller in the system.

BILLER_NAME_STREET_TOWN_COMBINATION_ALREADY_EXISTS
Status code: 400
Biller name / street / town combination already exists in the system
Display name, street and town combination must be unique for every biller in the system.

BILLER_REFERENCED_SECTOR_DOES_NOT_EXIST
Status code: 400
One or more referenced sectorIds do not exist
One or more referenced sectorIds do not exist. Make sure you use only existing sectorIds.

BILLER_INVALID_IBAN_NUMBER
Status code: 400
Payload contains invalid IBAN number
The submitted request contains an IBAN number which can not be processed.

BILLER_INVALID_ESR_PARTICIPANT_NUMBER
Status code: 400
Validation of ESR participant number failed
The submitted ESR participant number must be valid.
See https://www.postfinance.ch/content/dam/pfch/doc/zv/impl_iso_man_nov_de.PDF for more details.

7.4. Asset Validations

ASSET_IMAGE_INVALID

Status code: 400
Asset payload does not seem to be a valid image
The submitted binary data has to be a valid image.

ASSET_MIME_TYPE_DOES_NOT_CORRESPOND_TO_CONTENT_TYPE
Status code: 415
Asset mime type does not correspond to content type
The mime type of the uploaded asset does not correspond to the provided content type.

ASSET_CONTENT_TYPE_MUST_BE_IMAGE
Status code: 400
Only images are allowed for the given asset
The asset is connected to an image. Only the content types image/jpeg, image/gif or image/png are allowed

ASSET_CONTENT_TYPE_MUST_BE_PDF
Status code: 400
Only PDFs are allowed for the given asset
The asset is connected to a PDF. Only the content type application/pdf is allowed for the given asset.

ASSET_IMAGE_EXCEEDS_MAXIMUM_SIZES
Status code: 413
Image asset maximum sizes exceeded
Image assets are required to be smaller than 100'000 bytes and smaller than 1024x1024 pixels.

ASSET_PDF_EXCEEDS_MAXIMUM_SIZE
Status code: 413
PDF asset too large
PDF assets are required to be smaller than 3'000'000 bytes.

7.5. Attachment Validations

ATTACHMENT_MAXIMUM_NUMBER_REACHED
Status code: 400
Maximum number of attachments reached
A maximum of 5 attachments can be assigned to a biller.

ATTACHMENT_INVALID_DATE_RANGE
Status code: 400
Invalid date range defined
The 'presentTo' date mustn’t be before the 'presentFrom' date.

7.6. Business Case Validations

BC_PDF_ATTACHMENT_VALIDATION_FAILED
Status code: 400
Business case PDF does not contain a valid attachment
PDF file contains exactly one embedded attachment with the exact name `eBill-SIX_v1.xml` and

none of any other version.
More attachments are allowed - and ignored - as long as they have different names.

BC_FILENAME_HEADER_EXCEEDS_MAXIMUM_LENGTH
Status code: 400
HTTP header x-filename exceeds maximum length
The request can contain a custom HTTP header `x-filename` of max length 99.

BC_EBILL_SIX_XML_SCHEMA_VALIDATION_FAILED
Status code: 400
XML schema validation of the eBill-SIX_v1.xml attachment failed
The eBill-SIX_v1.xml file has to be schema valid.

BC_BILLER_NOT_ACTIVE
Status code: 400
Biller is not active
The request has to refer to an active biller in the system.

BC_BILLER_SUBMISSION_FORBIDDEN
Status code: 403
Biller is currently suspended from submitting business cases
Submissions are not permitted for suspended billers.
Due to detected anomaly the biller has been suspended from submitting business cases.
For further details please contact the infrastructure provider.

BC_BILLER_ID_INCONSISTENT
Status code: 400
Business case has inconsistent biller id
The biller id in url must match the value in the submitted xml file.

BC_BILLER_NAME_INCONSISTENT
Status code: 400
Business case has inconsistent biller legal name.
The biller legal name must match the value from the system.
The submitted biller legal name must match the value from the biller data managed in system. Otherwise the business case will be rejected.

BC_BILL_RECIPIENT_EMAIL_NOT_FOUND
Status code: 400
Provided bill recipient email not found
The bill recipient email must be currently or historically available in the system.
An e-mail address is considered to be historically available if it was present in the period of 15 months up to the submission time.
For further details see network partner API documentation in Section 4.3.4, “Find events for bill recipients email address changes”.

BC_BILL_RECIPIENT_ID_NOT_FOUND
Status code: 400
Provided bill recipient id not found
The bill recipient id must be available in the system.

BC_BILL_RECIPIENT_HRUID_NOT_FOUND
Status code: 400
Provided bill recipient enterprise identification number (UID) not found
The provided swiss enterprise identification number (UID) of the bill recipient must be available in the system.

BC_BILLER_BILLRECIPIENT_RELATION_INSUFFICIENT
Status code: 400
Biller and bill recipient do not have the required relation
Check for a sufficient biller - bill recipient relation, see network partner API documentation for details.
If none of the following conditions are met, the business case is rejected:
. There is an active biller - bill recipient relation (say, the subscription process was completed beforehand).
. There is a biller - bill recipient relation in 'initiated' status (say, a registration process was at least started by the bill recipient).
. There is no biller - bill recipient relation in 'inactive' status and the bill recipient has enabled to be found.

BC_INVALID_DATE
Status code: 400
Business case has invalid date
On the submission date, the business case date can not be more than 90 days in the past and it can not be in the future.

BC_INVALID_REFERENCE_NUMBER
Status code: 400
Business case has invalid reference number
The reference number must be unique per biller with regard to the submission date and the business case type.
In the case of business case type 'bill' the reference number must be unique for 90 days, calculated from the business case due date. After this period of 90 days old reference numbers may then be reused.

BC_REFERENCED_BILL_IS_NOT_A_BILL
Status code: 400
Provided Business case referenced bill is not a bill.
The business case must not reference other business cases types but bill or instalment bill.

BC_REFERENCE_NUMBER_FOR_REFERENCED_BILL_REQUIRED
Status code: 400
Referenced number for referenced bill is missing
The reference number of the referenced bill is mandatory in case of reminder.

BC_INVALID_REFERENCE_NUMBER_FOR_REFERENCED_BILL
Status code: 400
Referenced business case has invalid reference number
The reference number of the referenced business case has to be valid.

BC_INVALID_TOTAL_AMOUNT
Status code: 400
Business case has invalid total amount
The total amount of the business case must fulfill the requirements for minimum and maximum value.
See eBill-SIX_v1.xsd for details about total amount validation.

7.7. Business Case Payment Validations

BC_PAYMENT_INFORMATION_INVALID_PAYMENT_DUE_DATE
Status code: 400
Business case has invalid payment due date
On the submission date, the payment due date can not be more than 3 years (1095 days) in the future and can not be more than 90 days in the past.

BC_INCONSISTENT_CURRENCY_CODES
Status code: 400
Business case has inconsistent currency codes
In the entire business case only one currency is allowed.
It is not allowed to submit instalments, total amount or workflow specifications with different currencies.

BC_PAYMENT_INFORMATION_INVALID_CURRENCY_CODE_FOR_ESR
Status code: 400
The business case contains a payment information with an invalid currency code for the ESR payment type
Account type ESR only supports payment type 1 and therefore currency code is limited to CHF or EUR.
See the definition of payment types in
https://www.six-interbank-clearing.com/dam/downloads/en/standardization/iso/swiss-recommendations/implementation-guidelines-ct.pdf.

BC_PAYMENT_INFORMATION_INVALID_CURRENCY_CODE_FOR_QR_IBAN
Status code: 400
The business case contains a payment information with an invalid currency code for the QR-IBAN payment type
Payment type QR-IBAN only supports payment type 3 and therefore currency code is limited to CHF or EUR.
See the definition of payment types in
https://www.six-interbank-clearing.com/dam/downloads/en/standardization/iso/swiss-recommendations/implementation-guidelines-ct.pdf.
More information about QR in
https://www.paymentstandards.ch/dam/downloads/ig-qr-bill-en.pdf.

BC_PAYMENT_INFORMATION_ACCOUNT_INFORMATION_INCONSISTENT
Status code: 400
The business case contains a biller account that has inconsistent information
The biller account in the business case 'payment information' must match the account information

in the biller master data.

BC_PAYMENT_INFORMATION_INVALID_REFERENCE_TYPE
Status code: 400
The business case has an invalid reference type
In case of structured reference the biller account has to contain a valid reference type. See definitions and documentation in eBill-SIX_v1.xsd for more details.
The allowed values for reference type are: QRR: QR-Reference, SCOR: Creditor Reference (according to ISO 11649), NON: no reference, IPI: IPI-Reference (according to ISO 7064) (Note: The IPI-Reference must be withdrawn from 31.03.2020 at the latest.)

BC_PAYMENT_INFORMATION_INVALID_STRUCTURED_REFERENCE
Status code: 400
The business case contains invalid structured reference
Validation of the structured reference failed for the given referenceType.
QRR: QR-Reference, the submitted QR-Reference must be numeric and 27 characters long. The 27th digit (check digit) must be in accordance with Modulo 10, recursive.
SCOR: Creditor Reference (according to ISO 11649), it must be alphanumeric and its maximal length is 25.
NON: no reference, only reference type NON allows empty structured reference.
IPI: IPI-Reference, the submitted IPI-Reference must be alphanumeric and its maximal length is 20. The first two digit are check digits, according to Modulo 97-10 (ISO 7064). Note: The IPI-Reference must be withdrawn from 31.03.2020 at the latest.

BC_PAYMENT_INFORMATION_ESR_INVALID_FOR_BESR
Status code: 400
Payment information contains ESR reference which is not valid for the exsiting BESR-ID
The submitted ESR reference number must begin with the BESR-ID of the matching biller account.

BC_PAYMENT_INFORMATION_INVALID_ESR_REFERENCE_NUMBER
Status code: 400
Payload contains invalid ESR reference number
The submitted request contains an ESR reference number which can not be processed.
See https://www.postfinance.ch/content/dam/pfch/doc/cust/download/inpayslip_isr_man_en.pdf for more details.

7.8. Business Case Instalment Validations

BC_INVALID_TOTAL_PAYMENT_OPTION
Status code: 400
Payload contains invalid total payment option for instalments.
Validation of the total payment option failed.

BC_EXTERNAL_PAYMENT_BY_INSTALMENTS_ID_NOT_UNIQUE
Status code: 400
The identifications for the instalment options are not unique.
The instalment options must be uniquely identified within a business case.

BC_EXTERNAL_INSTALMENT_ID_NOT_UNIQUE

Status code: 400
The identifications for the instalments are not unique.
The instalments must be uniquely identified within a business case.

BC_INVALID_INSTALMENT_AMOUNT
Status code: 400
Business case has instalment with invalid amount.
The instalment amount must be positive.

7.9. PDF Validations

PDF_INVALID
Status code: 400
Payload does not seem to be a valid PDF
The submitted binary data has to be a valid PDF document.

PDF_IS_NOT_PDFA3B
Status code: 400
PDF is not PDF/A-3b
PDFs must be provided in a valid PDF/A-3b format.

PDF_INVALID_SIGNATURE
Status code: 400
PDF does not contain a valid signature
The PDF must contain a valid PAdES-B-B-level signature.
For more information, see network partner API documentation in Section 3.3.2.5.4, “File Specification and Signatures”.

7.10. Subscription Validations

SUBSCRIPTION_TOKEN_EXPIRED
Status code: 400
The subscription token has been expired
The subscription token can only be consumed once and is only valid for a limited time.

7.11. Technical Errors

TECHNICAL_ERROR
Status code: 500
Technical error on server side
Processing yielded a technical error. We are sorry and will debug the issue.

Appendix A: Changelog

A.1. Changes between 1.6.0 and 1.7.0

A.1.1. Changes of the Swagger definitions

- added new problem type 'BC_BILLER_SUBMISSION_FORBIDDEN' for the 'createBusinessCase' operation.

A.1.2. Changes of the Documentation

- adjusted all header examples to lower-case

A.2. Changes between 1.5.0 and 1.6.0

A.2.1. Changes of the Swagger definitions

- added line about duplicate checks for active billers

A.2.2. Changes of the Documentation

- added line about duplicate checks for active billers
- documented version- and cardinality constraints of ebill pdf-attachments .

A.3. Changes between 1.4.0 and 1.5.0

A.3.1. Changes of the Swagger definitions

- renamed Address to BillerAddress
- added RecipientAddress without building number

A.3.2. Changes of the Documentation

- added guidelines for consuming the notification events
- adjusted documentation about the QR-reference check digit and ISR participant number check digit

A.3.3. Changes for the QR-bill

The following changes are relevant for the introduction of the QR-bill as of May 2020

- eBill XSD: the value of the amountType is now optional, this enables QR bills without any value
- Swagger definitions: introduced the new problem type BC_PAYMENT_INFORMATION_INVALID_CURRENCY_CODE_FOR_QR_IBAN (see documentation for details)

- Swagger definitions: extended the description of the problem type BILLER_ACCOUNT_CURRENCY_CODE_COMBINATION_REJECTED with QR-IBAN which only supports CHF and EUR
- Documentation: adjusted the description of esrReference as it can contain the QR reference number as well as the ESR reference number

A.4. Changes between 1.3.0 and 1.4.0

eBill for Business
The feature release 'eBill for Business' gives swiss companies the opportunity to authorize multiple company users to view, approve and reject invoices on behalf of the company.
Based on this feature, the Network Partner API has been adjusted correspondingly.
The swiss enterprise identification number (UID) from the commercial register (Handelsregister) can be used for:

- biller subscriptions and subscription cancellations
- biller driven subscription ("Look-up") / bill recipient search
- business case submission

These adjustments result in a **breaking change** to the Network Partner API: the field 'emailAddress' in the definition of **BillRecipient** had to be declared `optional`.

A.4.1. Changes of the eBill XSD

- extended billRecipient with the additional option 'enterpriseIdentificationNumber' (UID), analogously to the 'emailAddress' and the 'billRecipientId', the 'enterpriseIdentificationNumber' can be used for business case submission
- fixed typo vatEntrpriseIdentificationNumberType -→ vatEnterpriseIdentificationNumberType

A.4.2. Changes of the Documentation

- adjusted the description of the section Subscriptions and Subscription Cancellations
- adjusted the description of the section Addressing of a Business Case to a Bill Recipient
- added new 'ProblemType' BC_BILL_RECIPIENT_HRUID_NOT_FOUND to **Business Case Validations**
- adjusted the title/detail of the 'ProblemType' BILLER_NAME_HRUID_COMBINATION_ALREADY_EXISTS

A.4.3. Changes of the Swagger definitions

- **BillRecipient**
  - added optional attribute 'enterpriseIdentificationNumber'(UID)
  - **breaking change**: changed attribute 'emailAddress' to be optional
- **BillRecipientsForBillerSearch**

    - added optional attribute 'enterpriseIdentificationNumber'(UID)
- adjusted the problem types as follows:
    - renamed CORRELATION_ID_MISSING → REQUEST_HEADER_VALIDATION_FAILED

The new problem type 'REQUEST_HEADER_VALIDATION_FAILED' is more general and is applied to both missing and invalid header values.
For example in case of missing header for correlationId this problem type is used.

A.5. Changes between 1.2.0 and 1.3.0

A.5.1. Changes of the eBill XSD

- changed element street in complexType postalAddress from mandatory to optional
- added explicit min and max inclusive restriction to complexType amountType
- adjusted description of biller reference number validation considering the new requirements
- added optional billRecipientAddress to workflow information
- removed pain.001 and qr-mapping from schema (available in the implementation guidelines)

A.5.2. Changes of the Swagger definitions

- BillerAttachment: billerId is no longer required since it is already present in the path for operations like createBillerAttachment and therefore ignored in the body

A.6. Changes between 1.0.36 and 1.2.0

A.6.1. Changes of the eBill XSD

- various text improvements and clarifications after user feedback
- added mappings to pain.001 and QR-bill in description
- introduced additional reference type IPI in case of accountIBANAndReference
- changed specifications in case of QR-IBAN, usage only with the reference type QRR possible

A.6.2. Changes of the Documentation

- various improvements and clarifications after user feedback
- restructured and reorganised the problem types for business case validations

A.6.3. Changes of the Swagger definitions

- declared fields in responses as `required` if they cannot be empty - this allows clients to rely on values in these fields.
- added first name on `BillRecipient` for improved identification (name for private bill recipient is now split into name and first name)

- introduced the following new problem types (see documentation for details):
    - BC_INVALID_TOTAL_PAYMENT_OPTION
    - BC_EXTERNAL_PAYMENT_BY_INSTALMENTS_ID_NOT_UNIQUE
    - BC_EXTERNAL_INSTALMENT_ID_NOT_UNIQUE
    - BC_INVALID_INSTALMENT_AMOUNT
    - BC_REFERENCE_NUMBER_FOR_REFERENCED_BILL_REQUIRED
- renamed/clarified the following problem types (see documentation for details):
    - PDF_INVALID_PADES_LTV_SIGNATURE → PDF_INVALID_SIGNATURE
    - BC_INVALID_ESR_REFERENCE_NUMBER → BC_PAYMENT_INFORMATION_INVALID_ESR_REFERENCE_NUMBER
    - BC_REFERENCED_BILL_NOT_FOUND → BC_REFERENCED_BILL_IS_NOT_A_BILL

A.7. Changes between Beta 2 and 1.0.36

A.7.1. Changes of the eBill XSD

- participantNumber renamed to esrParticipantNumber
- biller name renamed to legalName
- MWST-UID maxlength 16 and added pattern
- Changed amountType to maximum ten digits
- Changes for instalments
    - introduction of business case type **InstalmentBill**
    - business case type **Bill** ist now always SinglePayment
    - added id’s for instalment and paymentByInstalments
    - limited number of instalments and paymentByInstalments to 99

A.7.2. Changes of the Swagger operations

- renamed 'updateAttachment' to 'updateBillerAttachment'
- renamed 'deleteBiller' to 'deleteBillerAttachment'
- added 'findBillRecipientEmailAddressChangedEvent'
- added 'findInstalmentStatusChangedEvents'
- replaced 'findBillRecipientSubscriptionEvents' and 'findBillRecipientUnsubscribeEvents' with 'findBillRecipientSubscriptionStatusChangedEvents'
- introduction of 'X-FILENAME' optional header parameter for 'createBusinessCase' operation

A.7.3. Changes of the Swagger definitions

- **HealthCheckRequest**

  - introduced 'HealthCheckRequest'
- **HealthCheckResponse**
  - added fields requestDateTime, receivedHeaders, environmentStage, applicationVersion and apiVersion
- **Biller**
  - added attribute 'legalName'
  - changed size of 'enterpriseIdentificationNumber' from 15 to 12
  - added specific pattern to 'enterpriseIdentificationNumber'
- **BillerDetail**
  - Made 'countryCode' a required property of phone numbers, as well as validating that it equals the address 'countryCode'
  - Added upper and lower bounds for array of 'sectorIds' and array of biller accounts to avoid unbounded array sizes
- **Sector**
  - fix wrong pattern prefix from ASID to SCID
- **SectorId**
  - simplified the sectorIds array to contain only strings instead of SectorId objects
- **LocalizedBillerData**
  - removed attribute 'legalName', as it is now available on biller level.
  - renamed 'email' to 'emailAddress'
- **ProblemType**
  - definition was removed because of backward compatibility. A list of problem types can be found in the documentation.
- **Problem**
  - added 'fieldErrors' to describe validation failures per field
- **BusinessCase**
  - changed total amount of business case to new 'AmountWithCurrency' definition
- **InstalmentBill**
  - introduced new business case type 'InstalmentBill'
- **AccountNumber**
  - renamed 'participantNumber' to 'esrParticipantNumber'
- **AmountWithCurrency**
  - introduced new 'AmountWithCurrency' definition type
  - maximum length 10
- **BillRecipientSubscription**
  - simplified by having only one 'accountNumber' for either IBAN or participantNumber

- **BusinessCaseStateChangedEvent**
- renamed **BusinessCaseStateChangedEvent** to **BusinessCaseStatusChangedEvent**
    - added 'approvedAmount' property of type 'AmountWithCurrency'
- **InstalmentStatusChangedEvent**
    - introduced new 'InstalmentStatusChangedEvent' for new operation 'findInstalmentStatusChangedEvents'
- **BillRecipientEmailAddressChangedEvent**
    - introduced new 'BillRecipientEmailAddressChangedEvent' for new operation 'findBillRecipientEmailAddressChangedEvent'
- **BillRecipientUnsubscribedEvent** and **BillRecipientSubscriptionEvent**
    - replaced by 'BillRecipientSubscriptionStatusChangedEvent'