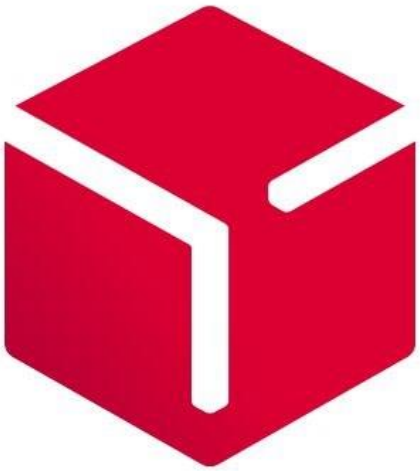


dpd

Contents

Client documentation

Descriptive label

Project	DPD Services Documentation	Document version:	0.1
Author:	Grzegorz Mendala, Albert Ambroziewicz, DaNol Pustuła, Tomasz Szeglowski	Abbreviated project name:	
Document date:	2017-10-23[2]	Document status[1]	Approved

[1] The following document status is acceptable: draft, approved
[2] Online version of the document is maintaNod online on: ...

Revision history

Author of the change	Date changed	Version	Change description
Grzegorz Mendala, Albert Ambroziewicz, DaNol Pustuła, Tomasz Szeglowski	2017-12-22	0.1	DostarczeNo dokumentu

Introduction

Purpose of the document

The purpose of this document is to describe the mode of operation and use of WebService type interfaces for DPD clients. The developed solution provides all-purpose tool (not dependant on the system type) for transfer of information between client services and DPD.

The document contains the description of the following WebService methods:

- Validation of parcel details and assigning waybill numbers
- Generating waybill labels
- Generating receipt protocol and data notification
- Ordering a courier
- Post codes search
- Checking courier availability
- Import of business events

Every call of the Web Service method is authorised by a login and password and is labelled with one of three policies of operation if an error occurs:

- Interruption of processing when the first error is encountered
- Ignoring of parcels with wrong data
- Interruption of processing when the first error is encountered and cancellation of parcels processed before the error occurred (option available only in some Web Service methods).

The client who is integrated with this DPD Polska solution is assured of consistency of the prepared parcels with the standing standard, which implies improvement to capacity and reliability of services offered by our courier company.

Glossary

Name	Definition
notification	Delivery of information regarding the parcel being handed over
label	Parcel details (e.g. recipient, sender, weight, courier route code, date of events, weight, parcel number, processing depot, service codes) in a descriptive form and a barcode label attached to the parcel.
waybill number	Number by which a specific parcel is identified
numkat	Number by which the DPD client is identified in other words: fid
OpenUMLF	Format of data used for DPD system integration
Delivery protocol	Document confirming the packages/parcels have been received
package	A package is an element of the parcel. A package has specified dimension (length, width, height, weight) . A package is identified by the waybill number.
parcel	A parcel – appropriately packed goods delivered from the sender to the recipient by the courier company. The Parcel consists of one or more packages. One package of the parcel is described as a master package. The parcel is identified by a master waybill (waybill number for the master package in the parcel). There are following types of parcels: • domestic, • international. A parcel has the following client roles assigned: • sender, • recipient, • payer.
routing system	DPD system which sets the parcel route (from dispatch to delivery, including transitional depots)
service	modification of parcel processing and handling mode (e.g. guaranteed delivery within a specific time slot, cash on delivery, carry in).
event	an event which occurred in connection with the parcel or package life cycle (e.g. dispatch, delivery).

Service addresses

Interfaces are available at the following URL addresses:

Object version:

https://dpdservices.dpd.com.pl/DPDPackageObjServicesService/DPDPackageObjServices?WSDL

xml/zip version:

https://dpdservices.dpd.com.pl/DPDPackageXmlServicesService/DPDPackageXmlServices?WSDL

For demo environment the URL addresses are as follows:

Object version:

https://dpdservicesdemo.dpd.com.pl/DPDPackageObjServicesService/DPDPackageObjServices?WSDL

xml/zip version:

https://dpdservicesdemo.dpd.com.pl/DPDPackageXmlServicesService/DPDPackageXmlServices?WSDL

Login details for the DEMO version

- Login: test
- Masterfid: 1495
- Password: thetu4Ee

Safety standard and permissions

The transferred data is secured by SSL standard with 128-bit asymmetric key. Each call is secured with login and password stored by DPD Polska in a LDAP based database. Also the user must be authorised to operate on the specific client number which is checked upon WebService call.

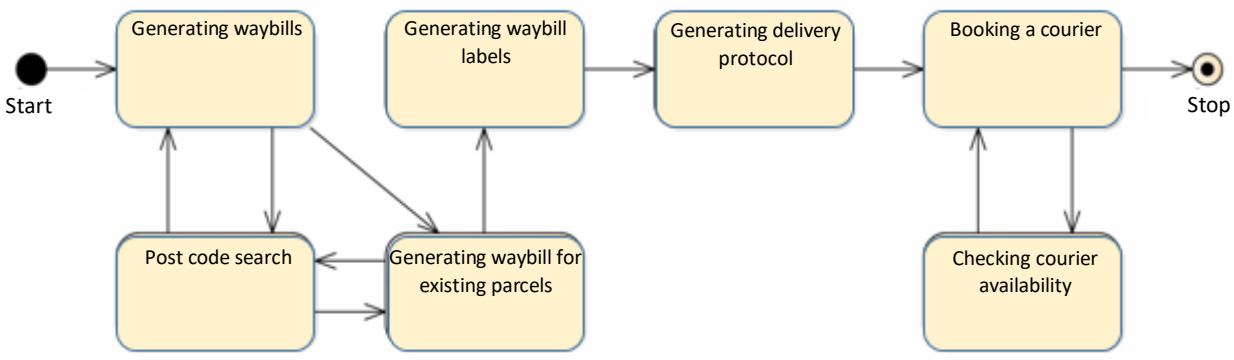
Full configuration of user account with permission to use web interfaces is performer in DPD Polska systems, after prior arrangement with the client. Authorisation is performed by placing login data and client number inside the sent query. More information is provided in the description of methods called from the service interfaces.

1.1 Business description

DPDServices is an all-purpose tool (not dependant on the system type) for transfer of information between client services and DPD. The tool is based on WebService type interfaces accessible to DPD clients.

Order of performed operations

The diagram below depicts the order of operation call within the business process.



The following business operations are available for the clients:

- Generating waybills (as well as existing parcels)
- Generating waybill labels
- Generating delivery protocol
- Booking a courier
- Post code search (auxiliary operation for data validation)
- Checking courier availability (auxiliary operation for booking the courier)
- Import of business events (not shown on the diagram)

Every call of the Web Service method is authorised by a login and password and is labelled with one of three policies of operation if an error occurs:

- Interruption of processing when the first error is encountered
- Ignoring of parcels with wrong data
- Interruption of processing when the first error is encountered and cancellation of parcels processed before the error occurred (option available only in some Web Service methods).

Linking of operations at the business process with interface methods is described below.

Short description	Method name	Business description	Technical description
Generating waybills Version V1, V2, V3, V4	generatePackagesNumbersCV1	Version V1-V4	
	generatePackagesNumbersXV1		
	generatePackagesNumbersV1		
	generatePackagesNumbersCV2		
	generatePackagesNumbersXV2		
	generatePackagesNumbersV2		
	generatePackagesNumbersCV3		
	generatePackagesNumbersXV3		
	generatePackagesNumbersV3		
	generatePackagesNumbersCV4		
	generatePackagesNumbersXV4		
	generatePackagesNumbersV4		

Generating labels	generateSpedLabelsCV1		
	generateSpedLabelsXV1		
	generateSpedLabelsV1		
	generateSpedLabelsCV2		
	generateSpedLabelsXV2		
	generateSpedLabelsV2		
	generateSpedLabelsCV3		
	generateSpedLabelsXV3		
	generateSpedLabelsV3		
	generateSpedLabelsCV4		
	generateSpedLabelsXV4		
	generateSpedLabelsV4		
Generating delivery protocols Version V1, V2	generateProtocolCV1		
	generateProtocolXV1		
	generateProtocolV1		
	generateProtocolCV2		
	generateProtocolXV2		
	generateProtocolV2		
	generaleProtocolsWithDestionationsV1		
	generaleProtocolsWithDestionationsCV1		
	generaleProtocolsWithDestionationsXV1		
	generaleProtocolsWithDestionationsV2		
	generaleProtocolsWithDestionationsCV2		
	generaleProtocolsWithDestionationsXV2		
Booking a courier Version V1, V2, V3 i V4	packagesPickupCallXV1		
	packagesPickupCallXV2		
	packagesPickupCallXV3		
	packagesPickupCallXV4		
	packagesPickupCallV1		
	packagesPickupCallV2		
	packagesPickupCallV3		
	packagesPickupCallV4		
Checking the post code accuracy	findPostalCodeV1		
	findPostalCodeXV1		
Checking courier availability	getCourierOrderAvailabilityV1		

	getCourierOrderAvailabilityXV1		
Import of delivery event	importDeliveryBusinessEventsV1		
	importDeliveryBusinessEventV1		
Generating waybills for existing parcels	appendParcelsToPackageV1		
	appendParcelsToPackageCV1		
	appendParcelsToPackageXV1		

General description of interfaces and difference between them

DPDServices has two available interfaces, which differ from each other mainly by the mode of sending input parameters and format of returned files:

1. Object interface DPDPackageObjServicesPortBinding, which receives and returns parameters provided directly (non-encoded and non-compressed)
2. Interface DPDPackageXmlServicesPortBinding, where the parameters are encoded with base64 (an option of compressed zip format, which affects the size of sent requests and received responses)

When preparing input data, close attention must be paid to upper/lower case letters used in tag names:

- Lower case in object version, e.g..: <weight>1.2</weight>
- Upper case at the beginning of the Yes in base base64/zip version, e.g.: „<Weight>1.2</Weight>”

Also, minor differences in the data structure must be taken into account, e.g..:

- Object version - „<packages><parcels><content>…”
- base64/zip version - „<Packages><Package><Parcels><Parcel><Content>…”

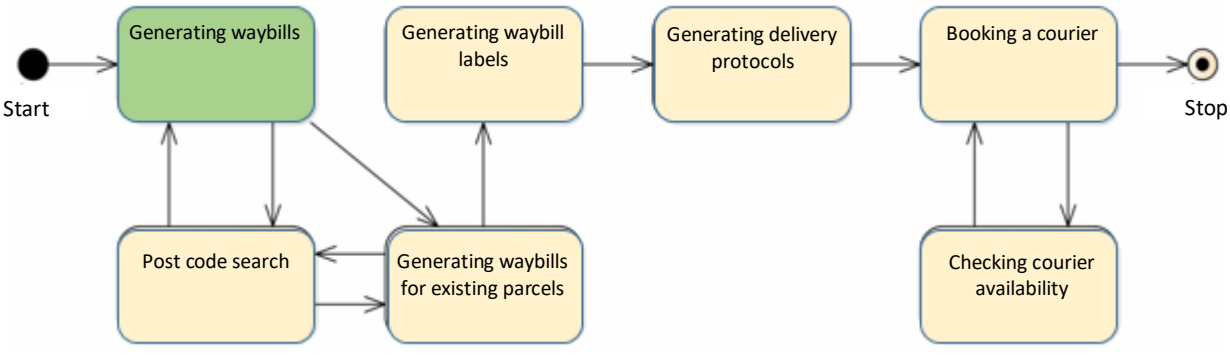
Two variations exist for most of available DPDPackageXmlServicesPortBinding interface methods:

- containing prefix „x” after the number, which receive and return data coded in the base64 form
- containing prefix “C” in the number, which receive and return data compressed by zip and are at the same time in the base64 form

Description of interface models

This description concerns DPDPackagesObjServices and DPDPackagesXMLServices interfaces.

Business description of the methods is combined for both of them because the differences between them are only structural which does not affect their significance from the business description point of view.



Generating waybills

Generating a waybill is the first step of the process of executing the order. Parcel details are sent as input parameters and they differ by the level of details depending on the version.

There are four versions of generating a waybill. The newer the version the higher the number and the latest version is recommended for use. Older versions are maintained in order to maintain consistency with the systems which has not updated its services.

Method name	generatePackagesNumbers		
Available versions	V1 - V4	CV1 - CV4	XV1 - XV4

The method assumes the following input parameters:

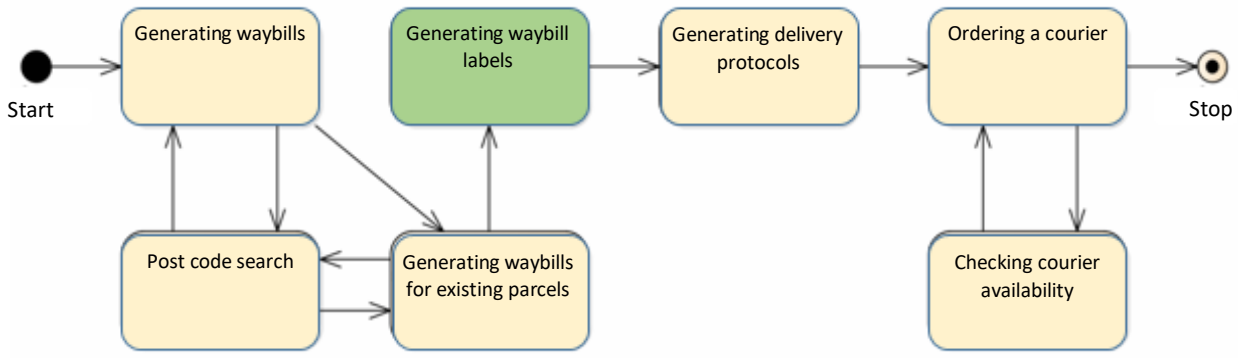
Input parameter	Method version
List of parcels and packages (optionally with client's reference number)	1 - 4
Error handling policy	1 - 4
Permission data	1 - 4
Country code	2 - 4

The following operations are performed from the DPD system:

Operation	Method version
Validation of transferred data	1 - 4
Generating waybill numbers for parcels	1 - 4

The method returns the following values:

Returned value	Method version
Session with a list of parcels and packages with assigned waybill numbers, client's reference numbers and validation status	1 – 4
Unique DPD system identifier	1 – 4



Generating labels

Method name	generateSpedLabels		
Available versions	V1 - V4	CV1 - CV4	XV1 - XV4

The method assumes the following input parameters:

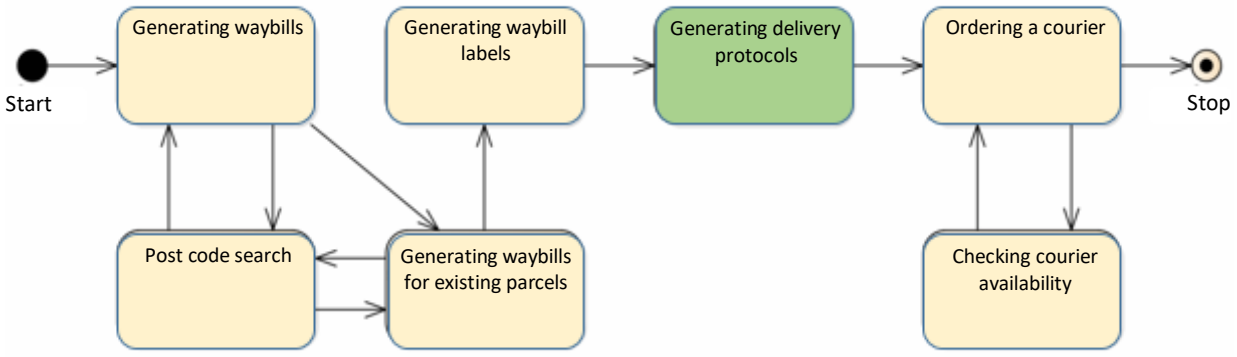
Input parameter	Method version
List of parcels to be processed by WS	1 - 4
Error handling policy	1 - 4
Format in which the labels are to be generated	1 - 4
Format of the page in which the labels are to be generated	1 - 4
Permission data	1 - 4
Label type: normal or extended	2 - 4

The following operations are performed from the DPD system:

Operation	Method version
Generating labels in the required format	1 - 4

The method returns the following values:

Returned value	Method version
Session with the list of parcels and packages with processing status	1 - 4
Labels generated in the required format	1 - 4



Generating the protocol

There are two methods available - basic (generateProtocol) and extended (generateProtocolWithDestinations)

With respect to generateProtocols methods, the generateProtocolsWithDestinations method enables generating the protocol together with description of destinations to which the protocol will be divided.

Method name	generateProtocol			generateProtocolWithDestinations		
Available versions	V1 - V2	CV1 - CV2	XV1 - XV2	V1 - V2	CV1 - CV2	XV1 - XV2

The method assumes the following input parameters:

Input parameter	Method version
Session with the list of parcels and packages to be processed by WS	1 - 2
Error handling policy	1 - 2
Format in which the protocol is to be generated	1 - 2
Format of the page into which the protocol is to be generated (in current version only A4 is available)	1 - 2
Permission data	1 - 2

generateProtocolWithDestinations method assumes the following input parameters:

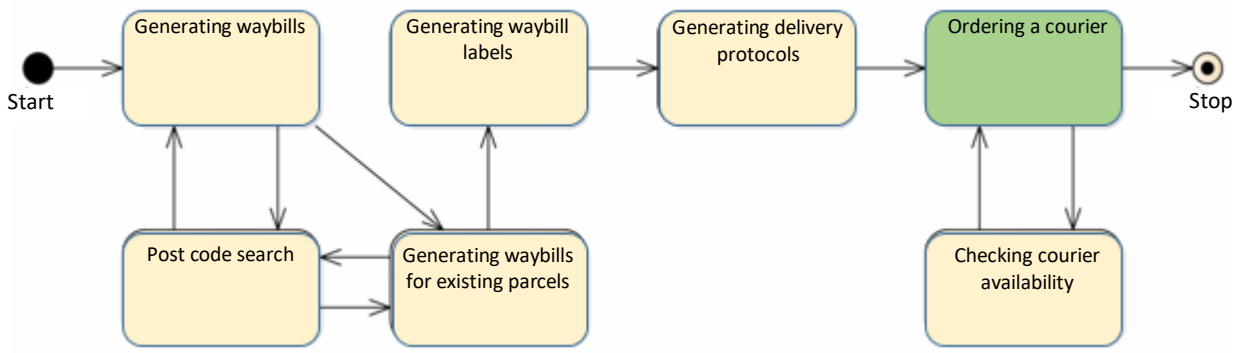
Input parameter	Method version
Session with the list of parcels and packages to be processed by WS	1 - 2
Description of destinations into which the protocol will be divided	1 - 2
Error handling policy	1 - 2
Permission data	1 - 2

The following operations are performed from the DPD system for both methods:

Operation	Method version
Generating the protocol with relevant information	1 - 2

Both methods return the following values:

Returned value	Method version
Protocol generated in the required format	1 - 2



Booking a courier

Method name	packagesPickupCall	
Available versions	V1 - V4	XV1 - XV4

Version: V1 – V2 (XV1 – XV2)

The method assumes the following input parameters:

Input parameter	Method version
Session with the list of parcels and packages to be processed by WS	1
Error handling policy	1
Place of dispatch	1
Preferred date and time of pickup	1 - 2
Permission data	1 - 2
Simplified parcel and package data, including: • Payer's details (payer number, cost centre, name) • Ordering party's details (name, first name and surname, phone number) • Sender's details (name, first name and surname, address, city, post code, phone number) Parcel/package parameter details (more information in the description of courier booking interfaces)	2

The following operations are performed from the DPD system:

Operation	Method version
Validation of accuracy of provided data	1 - 2
Receipt of information regarding incoming order in the DPD systems	1 - 2

The method returns the following values:

Returned value	Method version
Session with the list of parcels and packages with processing status	1
System confirmation of correct/incorrect order	1 - 2

Version: V3 – V4 (XV3 – XV4)

The method performs one of three operations selected by the user:

- INSERT – adding a new courier booking
- UPDATE – modification of an existing booking
- CANCEL – removing an existing booking

The method assumes the following input parameters (for the specific operation type):

Input parameter	INSERT	UPDATE	CANCEL
Order number, checksum and action plan if the order has already been closed		YES	
Simplified parcel and package data, including: • Payer's details (payer number, cost centre, name) • Ordering party's details (name, first name and surname, phone number) • Sender's details (name, first name and surname, address, city, post code, phone number • Parcel/package parameter details (more information in the description of courier booking interfaces)	YES	YES	
Preferred date and time of pickup	YES	YES	
Permission data	YES	YES	YES
CANCEL operation details: order number and checksum			YES

The following operations are performed from the DPD system:

Operation	Method version
Validation of accuracy of provided data	3 - 4
Approval of information regarding recording/modification/cancellation of the order in the DPD systems	3 - 4

The method returns the following values:

Returned value	Method version
System confirmation or correct/incorrect arrival/modification/cancellation of the order	3 - 4
Order number for INSERT/UPDATE operation	3 - 4
Checksum for INSERT/UPDATE operation	3 - 4

Other methods

Checking courier availability

Method name	getCourierOrderAvailability	
Available versions	V1	XV1

The method assumes the following input parameters:

Input parameter	Method version
Place of dispatch	1
Permission data	1

The following operations are performed from the DPD system:

Operation	Method version
Checking the courier availability in the specific area	1

The method returns the following values:

Returned value	Method version
Information regarding courier availability or information regarding errors in parameters	1

Import of the delivery event

Method name	importDeliveryBusinessEvents	
Available versions	V1	V1

The method assumes the following input parameters:

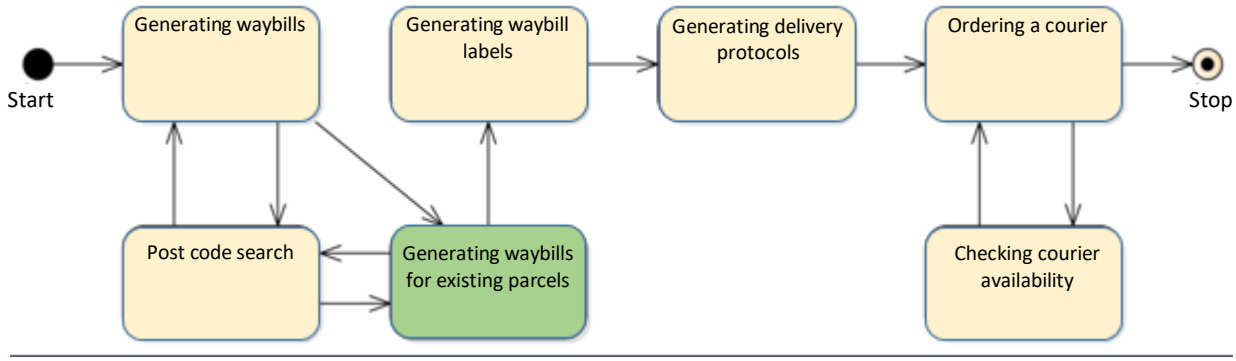
Input parameter	Method version
Delivery event details: • Delivery place details (country code, post code) • Delivered parcel details (waybill number) • Event code • Time of occurrence of the event • Additional event data (e.g. recipient name)	1
Permission data	1

The following operations are performed from the DPD system:

Operation	Method version
Checking the client's permissions to add the specific event type	1
Checking the client's permissions to add an event for the specific parcel	1
Receiving the information on delivery event for the specific parcel	1

The method returns the following values:

Returned value	Method version
Status of the performed operation	1
Exception at the wrong input data	1
Exception if the client hasn't got the permission to use the service	1



Generating waybills for existing parcels

Method name	appendParcelsToPackage		
Available versions	V1	CV1	XV1

The method assumes the following input parameters:

Input parameter	Method version
The search condition for searching the existing parcel for whom the new waybill is being generated (parcel number, reference number or waybill number)	1
List of parcels and packages (optionally with client reference numbers)	1
Permission data	1

The following operations are performed from the DPD system:

Operation	Method version
Validation or transferred data	1
Generating waybill numbers for parcels and adding them to the existing parcel	1

The method returns the following values:

Returned value	Method version
List of parcels with waybill numbers, client's reference numbers and validation status	1

Checking the accuracy of post code

Method name	findPostalCodeV1	
Available versions	V1	XV1

The method assumes the following input parameters:

Input parameter	Method version
Post code	1
Permission data	1

The following operations are performed from the DPD system:

Operation	Method version
Validation of transferred data	1

The method returns the following values:

Returned value	Method version
Confirmation of data accuracy	1

1.2 Description of service structure

The service consists of two main interfaces containing methods which execute the same business goals. The difference is between the mode of formatting and sending information. Detailed description of these interfaces and their methods is provided below.

The methods are described in the following order:

- Signature – form of the method on the webservice part
- Parameters – all parameters existing in the structure of the request
- Comments and suggestions – ways of implementation and suggestions for the specific method (e.g. sending group parcels rather than single requests of each of them).
- Limitations and comments – limitations in connected with method parameters (boundary values)
- Validation- concerns the feedback in the event the form which makes the query has been filled in incorrectly.
- Request results – different feedback variants
- Method call – examples of ready requests in different parameter configurations

1.2.1 Permissions

Since the permissions are shared for all methods in different interfaces, the description is separate and provided below::

In their structure, all methods described in this chapter have the authDataV1 (type AuthDataV1) parameter which stores the permission data for the service. Each recipient should have own logging parameters. AuthDataV1 consists of three elements:

- login (String) – a unique identifier assigned to the recipient by the service provider.
- password (String) – a password assigned to the login.
- masterFid (Integer) – a unique client catalogue number provided by the service provider. Also known as numkat.

His parameter is required and lack of it results in the „AuthData is null” message. The following conditions of its internal structure are checked during validation:

Field name	Validation condition	Field requested	Additional information
login	No field or incorrect login.	Yes	Error results in „Login failed” message
password	No field or incorrect login.	Yes	Error results in „Login failed” message
masterFid	No validation	No	

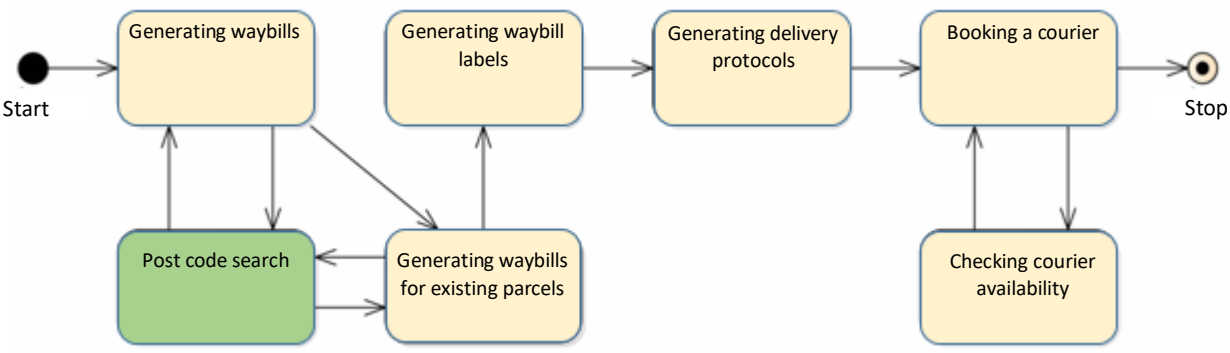
Due to repeatability the parameter will not be included in description of specific methods below.

1.2.2 Error codes

Error codes are described in the attachment to this documentation

1.2.3 Description of DPDPackageObjService interface methods

1.2.3.1 Auxiliary methods



Method name	findPostalCodeV1
Signature	FindPostalCodeResponseV1 findPostalCodeV1(PostalCodeV1 postalCodeV1, AuthDataV1 authDataV1)
Output	FindPostalCodeResponseV1

Parameters:

postalCodeV1(PostalCodeV1)

countryCode(String) – country code

zipCode(String) – postal code (only characters, excluding e.g. '-')

Validation

No validation

Query results

XML response structure for checking the post code accuracy in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:findPostalCodeV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <status>OK</status>
      </return>
    </ns2:findPostalCodeV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as FindPostalCodeResponseV1 and be FindPostalCodeResponseV1 type

The response returns the following elements:

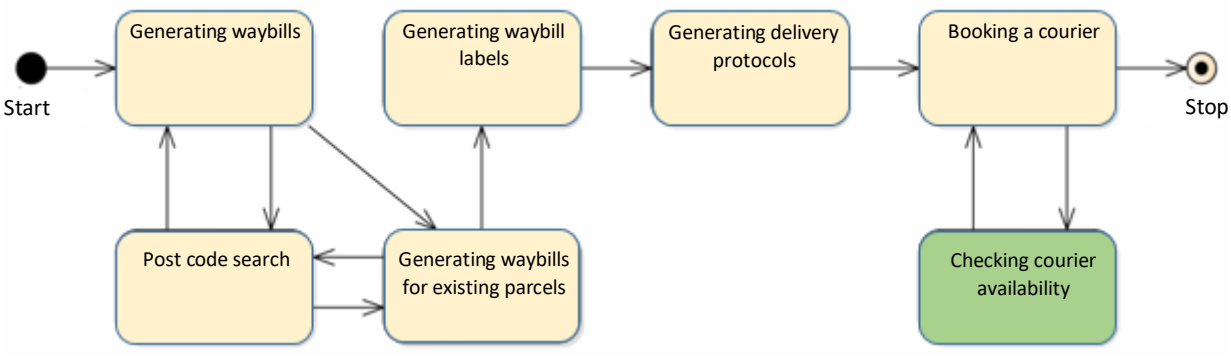
status(String) :

- OK – post code is correct
- NONEXISTING_POSTAL_CODE – post code does not exist
- NONEXISTING_COUNTRY_CODE – country code does not exist
- WRONG_POSTAL_PATTERN – in the event the given parameters are wrong

Method call

The example below checks whether post code 00-275 exists in Poland:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:findPostalCodeV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <postalCodeV1>
        <countryCode>PL</countryCode>
        <zipCode>02275</zipCode>
      </postalCodeV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:findPostalCodeV1>
  </S:Body>
</S:Envelope>
```



Method name	getCourierOrderAvailabilityV1
Signature	GetCourierOrderAvailabilityResponseV1 getCourierOrderAvailabilityV1(SenderPlaceV1 senderPlaceV1, AuthDataV1 authDataV1)
Output	GetCourierOrderAvailabilityResponseV1

The method checks the courier availability.

Parameters:

senderPlaceV1(SenderPlaceV1)

- countryCode(String) – Country code
- zipCode(String) – Post code (only significant characters, without e.g. '-')

Validation

For getCourierOrderAvailabilityV1:

Field name	Validation condition	Field required	Additional information
SenderPlaceV1	Internal structure conditions met	Yes	

For SenderPlaceV1 field:

Field name	Validation condition	Field required	Additional information
CountryCode		Yes	
ZipCode	Integer value	Yes	

Query results

XML response structure for correctly generated courier availability check in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getCourierOrderAvailabilityV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <ranges/>
        <ranges>
```

```
          <offset>120</offset>
          <range>14:00-16:00</range>
        </ranges>
        <ranges>
          <offset>120</offset>
          <range>15:00-17:00</range>
        </ranges>
        <ranges>
          <offset>120</offset>
          <range>17:00-19:00</range>
        </ranges>
       <status>OK</status>
    </return>
    </ns2:getCourierOrderAvailabilityV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as getCourierOrderAvailabilityResponseV1 and should be GetCourierOrderAvailabilityResponseV1 type.

The response returns the following elements:

- status(String) – response status
- ranges(List<CourierOrderAvailabilityRangeV1>) – list of availability ranges
    - range(String) – availability range format (hh:mm-hh:mm)
    - offset(Integer) – minimum advance time (in minutes), when the courier needs to be booked to execute the order.

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:getCourierOrderAvailabilityV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <senderPlaceV1>
        <countryCode>PL</countryCode>
        <zipCode>02275</zipCode>
      </senderPlaceV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:getCourierOrderAvailabilityV1>
  </S:Body>
</S:Envelope>
```

Method name	importDeliveryBusinessEventV1
Signature	ImportDeliveryBusinessEventResponseV1 importDeliveryBusinessEventV1(DPDParcelBusinessEventV1 dpdParcelBusinessEventV1, AuthDataV1 authDataV1)
Output	ImportDeliveryBusinessEventResponseV1

Parameters:

dpdParcelBusinessEventV1(DPDParcelBusinessEventV1)

- countryCode(String) – country code (ISO 3166-1 alpha-2)
- postalCode(String) – post code (only characters, without e.g. „-" for post codes)
- eventCode(String) – event code
- waybill(String) – parcel waybill number
- eventTime(Date) – time of the event (yyyy-MM-ddTHH-mm-ss)
- eventDataList(DPDParcelBusinessEventDataV1) – list of additional data for the event
    - code(String) – code of the additional data for the event
- value(String) – value of the additional data for the event

Query results

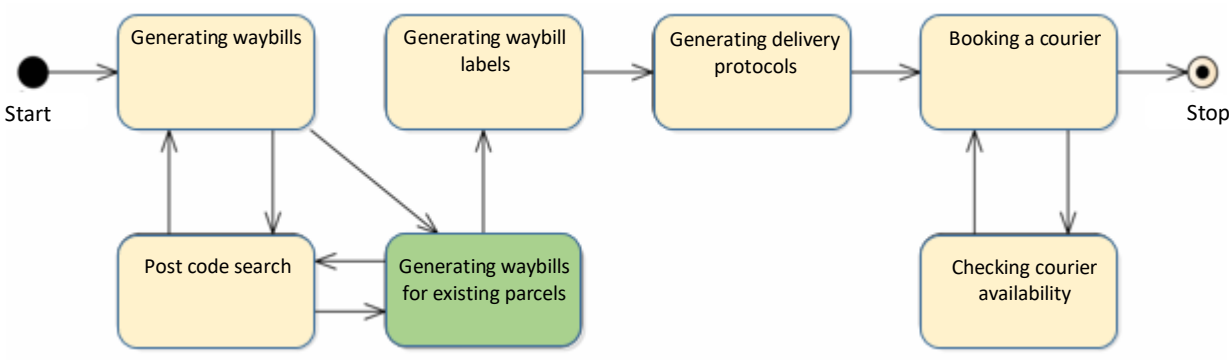
Correctly generated response should be labelled as importDeliveryBusinessEventResponseV1 and be ImportDeliveryBusinessEventResponseV1 type.

Response returns the following elements:

- status(ImportDeliveryBusinessEventStatusEnumV1) – operation status
    - OK - operation successful
    - UNKNOWN_ERROR – unknown internal error in the system
    - PARCEL_NOT_FOUND – no parcel for which the delivery event was to be added
    - PARCEL_PERMISSION_DENOD – no permissions to add events for any parcels
    - EVENT_PERMISSION_DENOD – no permissions to add events with the specific code
    - INCORRECT_DATA – wrong semantics of the data
- description(String) – additional description

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:importDeliveryBusinessEventV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdParcelBusinessEventV1>
        <countryCode>PL</countryCode>
        <eventCode>190101</eventCode>
        <eventDataList>
          <code>RECIPIENT_SURNAME</code>
          <value>Surname</value>
        </eventDataList>
        <eventTime>2018-01-01T14:00:00</eventTime>
        <postalCode>02274</postalCode>
        <waybill>0000001052524S</waybill>
      </dpdParcelBusinessEventV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:importDeliveryBusinessEventV1>
  </S:Body>
</S:Envelope>
```



Method name	appendParcelsToPackageV1(
Signature	ParcelsAppendResponseV1 appendParcelsToPackageV1(ParcelsAppendV1 parcelsAppend, AuthDataV1 authDataV1)
Output	ParcelsAppendResponseV1

Method is used for adding packages to an existing parcel. During an attempt to add a package, the possibility to add is assessed, validated and if such option is confirmed, the parcels are added and waybill numbers are generated for new packages.

Parameters:

parcelsAppend(ParcelsAppendV1)

- packagesearchCriteria(ParcelsAppendSearchCriteriaPAV1) – parcel search criteria.
- packageId(long) – parcel IDi.
- reference(String) – parcel reference code – unique key.
- waybill(String) – unique parcel ID – waybill number.
- parcels(List<ParcelAppendPAV1>) – list of packages in the parcel.
    - reference(String) – package reference code.
    - weight(Double) – package weight.
    - sizeX(Integer) – package dimension X cm.
    - sizeY(Integer) - package dimension Y cm.
    - sizeZ(Integer) - package dimension Z cm.
    - content(String) – description of contents.
    - customerData1(String) – Notes for delivery
    - customerData2(String) – client data.
    - customerData3(String) – client data.

Validation

For appendParcelsToPackageV1:

Field name	Validation condition	Field required	Additional information
parcelsAppend	No field	Yes	Lack thereof results in message: "Parameter parcelsAppend cannot be null"

For ParcelsAppendV1 field:

Field name	Validation condition	Field required?	Additional information
PackageSearchCriteria	Internal structure conditions met	Yes	
Parcels	Internal structure conditions met	Yes	

For PackageSearchCriteria field:

Field name	Validation condition	Field required?	Additional information
PackageId	Integer value	No	Lack of at least one fields results in: „PACKAGE_NOT_FOUND message
Reference	Maximum number of characters100.	No	
Waybill	Maximum number of characters14.	No	

For Parcels field:

Field name	Validation condition	Field required?	Additional information
Parcel	One or more occurrences	Yes	

For Parcel field:

Field name	Validation condition	Field required?	Additional information
Reference		No	
Weight	Floating point value	Yes	
Waybill	Maximum number of characters 14.	No	
SizeX	Integer value	No	
SizeY			
SizeZ			
Content	Maximum number of characters 300.	Yes	
CustomerData1	Maximum number of characters 100.	No	
CustomerData2			
CustomerData3			

During processing, the following status field value can be encountered:

OK – object has been processed correctly

UNKNOWN_ERROR – unknown error has occurred

PACKAGE_NOT_FOUND – no package found

NOT_PROCESSED – unprocessed data

BLOCKING_EVENT_EXISTS – there is a blocking status for at least one package in the parcel

PROTOCOL_GENERATED – the protocol has been generated for the specific parcel

INCORRECT_DATA – incorrect data. Details in InvalidFields section.

Query results

parcelsAppendResponseV1(ParcelsAppendResponseV1)

- status(String) – general status of the operation
- parcels(List<ParcelsAppendParcelPAV1>) – number of packages in the parcel
    - status(String) – parcel status.
    - reference(String) – unique key
    - parcelId(Long) – new parcel ID.
    - waybill(String) – waybill number
    - invalidFields(List<InvalidFieldPAV1>) – detailed error description
        - fieldName(String) - field name to whom the error message applies
        - info(String) – information regarding the cause of the error
        - status(String) – status code

XML response structure for generating waybills for existing parcels in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:appendParcelsToPackageV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <parcels>
          <parcelId>1093642</parcelId>
          <status>OK</status>
          <waybill>0000001052525S</waybill>
        </parcels>
        <status>OK</status>
      </return>
    </ns2:appendParcelsToPackageV1Response>
  </S:Body>
</S:Envelope>
```

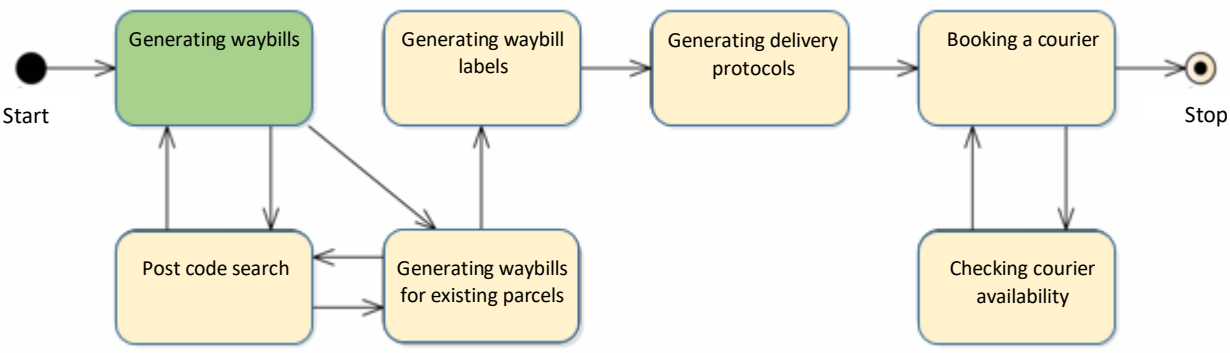
Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:appendParcelsToPackageV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <parcelsAppend>
        <packagesearchCriteria>
          <packageId>640505</packageId>
        </packagesearchCriteria>
        <parcels>
          <content>philosopher's stone</content>
          <customerData1>client data 11</customerData1>
          <customerData2>client data 22</customerData2>
          <customerData3>client data 33</customerData3>
          <sizeX>1</sizeX>
```

```
          <sizeY>3</sizeY>
          <sizeZ>3</sizeZ>
          <weight>1.2</weight>
        </parcels>
      </parcelsAppend>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:appendParcelsToPackageV1>
  </S:Body>
</S:Envelope>
```

1.2.3.2 Methods relating to generating waybills

Each of 4 versions presented below have an example of the call. Please note that not all elements used in the higher method versions can be used in lower versions (e.g. no service for version 1 of dpdPickup), however the higher versions are compatible with the lower ones.



Method name	generatePackagesNumbersV1
Input signature	PackagesGenerationResponseV1 generatePackagesNumbersV1(OpenUMLFeV1 openUMLV1, PkgNumsGenerationPolicyV1 policyV1, AuthDataV1 authDataV1)
Output	PackagesGenerationResponseV1

Parameters

openUMLV1(OpenUMLFeV1) – it is the first version of the parameter responsible for data concerning the parcel, selected services and other key information, which point to the mode of processing parcel information. More information can be found in an attachment describing all versions of OpenUMLF parameter. The attachment is called „OpenUMLF description”.

pkgNumsGenerationPolicyV1(PkgNumsGenerationPolicyV1) – glossary parameter defining the service policy regarding reaction to errors occurring during method call. Depending on the selected option, the service will take a different course when an error occurs. The possible values are as follows:

- STOP_ON_FIRST_ERROR – means that the service will stop the execution of the operation when the first error occurs for any of the prepared parcels. Import of the first parcels will be successful.
- IGNORE_ERRORS – means ignoring the general error for the specific record/parcel. It means that the incorrect parcel will be ignored, and the remaining parcels, with correct data, will be correctly prepared by the service.
- ALL_OR_NOTHING – stops the parcel import for any error. Import will be effected if all parcels are successfully verified.

Notes and suggestions

Access levels:

When the user tries to perform operations which are not available for his permission level, the following message will be displayed: „User has no privileges to execute this operation".

Exceeded number of generated waybills:

The user will receive the message: „Waybill counter exceeded for fid:" and the client ID – masterFid (numkat), if he/she exceeds the limit set up for them by the service provider.

Error in OpenUMLF format:

Parameter OpenUMLF is very different in the subsequent versions of methods for generating waybills. There is a number of cases for which the OpenUMLF format error can appear. It usually means incorrect interpretation of the parameter component. Message „Incorrect OpenUMLFe format: " precedes the detailed content of format error.

Empty OpenUMLF input parameter:

In the event of problems with transmitting the OpenUMLF parameter, the following message will be generated: „Parameter openUMLV1 cannot be null".

Error in saving input data:

During generating waybills the request is registered. If there is problem with registering, the following message will appear: „Exception while saving request data".

Error in saving output data:

During generating waybills the response is registered. If there is problem with registering, the following message will appear: „Exception while saving response data".

Unknown error:

In the event of internal problems in the service, the 'Unknown error' message may appear. It relates to all unclassified errors in the service. If such error occurs, the service provider must be contacted.

Validation

For generatePackagesNumbersV1:

Field name	Validation condition	Field required?	Additional information

OpenUMLV1	Internal structure conditions met	Yes	Lack thereof result in the message : "Parameter openUMLV1 cannot be null"
pkgNumsGenerationPolicyV1	No validation	No	

For OpenUMLV1:

Field name	Validation condition	Field required?	Additional information
packages	No field	Yes	Lack thereof result in the message: „One of '(Package)' is expected."
	Internal structure conditions met		

For packages field:

Field name	Validation condition	Field required?	Additional information
Reference	Maximum number of characters100.	No	
Receiver	Internal structure conditions met	Yes	
Sender	Internal structure conditions met	Yes	
PayerType	Must be one of acceptable values	Yes	
ThirdPartyFID	Integer value	No	
Ref1		No	
Ref2		No	
Ref3		No	
Services		No	
Parcels	Internal structure conditions met	Yes	

For Receiver field:

Field name	Validation condition	Field required?	Additional information
FID	Integer value	No	
Company	Maximum 50 characters	No	
Name	Maximum 30 characters	No	
Address	Maximum 30 characters	Yes	
City	Maximum 35 characters	Yes	
CountryCode		No	Default value - „PL"
PostalCode		Yes	
Phone	Maximum 30 characters	No	
Email		No	

For Sender field:

Field name	Validation condition	Field required?	Additional information
FID	Integer value	No	
Company	Maximum 30 characters for domestic service Maximum 24 characters for international service	No	
Name	Maximum 30 characters for domestic service Maximum 24 characters for international service	No	
Address	Maximum 30 characters for domestic service Maximum 24 characters for international service	No	
City	Maximum 20 characters for domestic service Maximum 15 characters for international service	No	
CountryCode		No	Default value - „PL”
PostalCode		No	
Phone	Maximum 20 characters	No	
Email		No	

For PayerType field:

Field name	Validation condition	Field required?	Additional information
RECEIVER	One of the values allowed	No	
SENDER			
THIRD_PARTY			

For Services field:

Field name	Validation condition	Field required?	Additional information
DeclaredValue	Internal structure conditions met	No	
Guarantee	Internal structure conditions met	No	
COD	Internal structure conditions met	No	
CUD		No	
ROD		No	
InPers		No	
SelfCol	Internal structure conditions met	No	
PrivPers		No	
CarryIn		No	
Duty		No	
Pallet		No	
DOX		No	
RequestedDelivDate		No	

DedicatedDelivery		No	
Tires		No	
TiresExport		No	

For SelfCol field:

Field name	Validation condition	Field required?	Additional information
PRIV	One of the values allowed	No	
COMP			

For Guarantee field:

Field name	Validation condition	Field required?	Additional information
Attr1	One of the values allowed	No	
type			

For type field (for Guarantee):

Field name	Validation condition	Field required?	Additional information
TIME0930			
TIME1200			
TIMEFIXED			
B2C			
SATURDAY			
INTER			

For DeclaredValue field and COD:

Field name	Validation condition	Field required?	Additional information
Amount	Floating point value	No	
Currency		Yes	

For Currency field:

Field name	Validation condition	Field required?	Additional information
PLN	One of the values allowed	No	
EUR			

USD			

For Parcels field:

Field name	Validation condition	Field required?	Additional information
Parcel	One or more occurrences	Yes	

For Parcels field:

Field name	Validation condition	Field required?	Additional information
Reference	Maximum number of characters 100.	No	
Weight	Floating point value.	Yes	
SizeX	Integer value	No	
SizeY	Integer value	No	
SizeZ	Integer value	No	
Content	Maximum number of characters 50. Itis possible to enter 300 characters, but only 50 can be placed on the label.	Yes	
CustomerData1	Maximum number of characters 100	No	
CustomerData2		No	
CustomerData3		No	

Query results

Query results are divided into two parts. The first concerns correct waybill generation and includes a detailed description of components. The second part describes incorrect generation of waybills and complements the first case with additional components.

XML response structure for correctly generated waybill in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
 <S:Body>
  <ns2:generatePackagesNumbersV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
   <return>
    <packages>
     <packageId>640522</packageId>
     <parcels>
     <parcelId>1093661</parcelId>
     <status>OK</status>
     <waybill>0000001052539S</waybill>
   </parcels>
   <status>OK</status>
  </packages>
  <sessionId>610355</sessionId>
    <status>OK</status>
```

```
    </return>
   </ns2:generatePackagesNumbersV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as generatePackagesNumbersV1Response and be PackagesGenerationResponseV1 type. The response returns the following elements:

- Packages(List<PackagePGRV1>) – it is a collection of parcels declared for generating waybills. Parcels can contain a number of packages further referred to as parcels.

The parcels consist of the following elements:

- packageId(Long) – Unique parcel ID, assigned in the process of generating waybills.
- reference(String) – if a reference parameter which is a component of Packages parameter within Open UMLF was defined in the process of generating waybills, it will also appear as an element of response. Please note, that this parameter should only be set up intentionally, since it overwrites the standard recorded unique ID by which the parcel is searched.
- status(ValidationStatusPGREnumV1) – status of waybill generation for a specific parcel. Corresponding to the status of the entire waybill (described below).
- Parcels(List<ParcelPGRV1>) – a list of packages belonging to the parcel. Packages consist of the following elements::
    - parcelId(Long) – unique package ID, assigned in the process of generating waybills.
    - reference(String) – unique package ID given by the user, similar to the process for the parcel. It is used as a search key - SearchKey.
    - waybill(String) – unique ID used as public ID for the parcel and assigned during the process of generating waybills.
    - status(ValidationStatusPGREnumV1) – similar as in the parcel status (see below).
- sessionId(Long) – a unique session number concerning the performed operation. It is necessary for the business process during performing further steps of the process.
- status (ValidationStatusPGREnumV1) – parameter which specifies the status of a specific parcel during generating waybills. The following values are possible:

    OK – the parcel was correctly described and there were no problems during generating waybills.
    UNKNOWN_ERROR – an unknown error occurred during the process of generating waybills.
    DB_ERROR – problem with connection between the service to database. It is necessary to contact the service provider.
    INCORRECT_DATA – incorrect data entered for the parcel. Please verify and correct the data.
    NOT_PROCESSED – this status appears in incorrect circumstances, when parcel processing is not possible, for example no parcel record in the database.

DUPLICATED_PACKAGE_SEARCH_KEY – A parcel with this unique search key already exists. Search key is set up with the input parameter called „Reference" if it was entered by the user. If it was not provided it is set up as a conjunction of string of characters consisting of „##" tab and parcel ID. If such message appears, then it is probably necessary to assign a new value to 'reference' parameter or the waybill has already been generated for the parcel.

DUPLICATED_PARCEL_SEARCH_KEY – similar to the parcel, but it relates to a single parcel within the package.

DISALLOWED_FID – in the shape provided by the user it is not possible to create a waybill for the user with the provided masterFid number. The following error message can appear: 'Payer number not allowed'

DUPLICATED_WAYBILL – a parcel with this unique ID already exists. The unique Waybill ID is created in the process of ordering waybills and such error should not appear at that time.

UNSUPPORTED_LANG_CODE – this value for generating waybills applies to versions higher than V1 and concerns langCode parameter, which is responsible for language used for creating waybills. This error appears if the above parameter has not been set up or has a value other than 'PL'.

XML response structure for the unsuccessful attempt to generate a waybill.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <packages>
          <invalidFields>
            <fieldName>Sender/Address;</fieldName>
            <info>Spender name exceeds acceptable limit(100)</info>
            <status>VALUE_INCORRECT</status>
          </invalidFields>z
          <parcels>
            <reference/>
            <status>OK</status>
          </parcels>
        <status>INCORRECT_DATA</status>
      </packages>
      <status>INCORRECT_DATA</status>
    </return>
  </ns2:generatePackagesNumbersV1Response>
 </S:Body>
</S:Envelope>
```

Data returned for this case does not differ significantly from the data returned for the case of correct generation of waybills. The main difference is the new InvalidFields parameter within the parcel described as packages. It is used for transferring information regarding any input parameter fields which did not pass the validation and must be modified. InvalidFields is a list of InvalidFieldPGRV1 type and consists of:

- fieldName(String) – field name, to which the submitted validation problem applies.
- Info(String) – a short explanation of the problem.
- status(FieldValidationStatusPGREnumV1) – status for the specific field. It can assume the following values:
    - OK – validation was successful, there are no errors for this field.
    - UNKNOWN_ERROR – unknown error.
    - DB_ERROR – an error caused by problems with connection to the database.
    - DONT_MATCH_DICTIONARY – the error occurs if a value, which does not belong to the glossary assigned to this field, is used.
    - DONT_MATCH_PATTERN – wrong pattern e.g. post code. Entered value does not match the pattern required for the specific parameter.
    - VALUE_EMPTY – error of entering an empty value for a field which requires a value.
    - VALUE_ZERO – error of entering a zero value which requires a non-zero value.
    - VALUE_OUT_OF_RANGE – error regarding the value exceeding the range allowed for the specific field.
    - VALUE_INCORRECT – incorrect field value.

- UNKNOWN_RDB_ERROR – unknown error of the routing system. The system is not able to determine the route for parcel delivery.
- DUPLICATED_KEY – error of duplicating the value key for the specific field.

Method call

An example of a method call for generating waybills in version V1 is shown below:

- directed to a domestic recipient at the address: ul. Polna 200 in Radom.
- the sender at the address: ul. Leśna 100 in Warszawa and numkat (catalogue number assigned by DPD, also known as FID): 1495, is also the payer
- containing two domestic services:
  - COD
  - guarantee (type TIME1200 – delivery attempt before 12:00)

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header/>
<S:Body>
  <ns2:generatePackagesNumbersV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
    <openUMLV1>
      <packages>
        <parcels>
          <content>philosopher's stone</content>
          <customerData1>client data 11</customerData1>
          <customerData2>client data 22</customerData2>
          <customerData3>client data 33</customerData3>
          <sizeX>1</sizeX>
          <sizeY>2</sizeY>
          <sizeZ>3</sizeZ>
          <weight>1.2</weight>
        </parcels>
        <payerType>SENDER</payerType>
        <receiver>
          <address>Ul. Polna 200</address>
          <city>Radom</city>
          <company>XYZ</company>
          <countryCode>PL</countryCode>
          <email>ktos@pocztowy.pl</email>
          <name>Antoni Nowak</name>
          <phone>999999999</phone>
          <postalCode>26605</postalCode>
        </receiver>
        <ref1>Nr ref. 111/01</ref1>
        <ref2>Nr ref. 111/02</ref2>
        <ref3>FVAT 17/03</ref3>
        <sender>
          <address>Ul. Leśna 100</address>
          <city>Warszawa</city>
          <company>Sklep ABC</company>
          <countryCode>PL</countryCode>
          <email>pan@chcepaczke.pl</email>
          <fid>1495</fid>
          <name>Jan Kowalski</name>
          <phone>777777777</phone>
          <postalCode>02274</postalCode>
        </sender>
        <services>
        <cod>
```

```
            <amount>123.00</amount>
            <currency>PLN</currency>
          </cod>
          <guarantee>
             <type>TIME1200</type>
          </guarantee>
        </services>
      </packages>
      </openUMLV1>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersV1>
  </S:Body>
</S:Envelope>
```

Method name	generatePackagesNumbersV2
Signature	PackagesGenerationResponseV2 generatePackagesNumbersV2(OpenUMLFeV1 openUMLV1, PkgNumsGenerationPolicyV1 policyV1, String langCode, AuthDataV1 authDataV1)
Output	PackagesGenerationResponseV2

Parameters

OpenUMLFeV1 openUMLV1 – this parameter does not change in comparison to version V1.

PkgNumsGenerationPolicyV1 policyV1 - this parameter does not change in comparison to version V1.

langCode(String) – abbreviated code of language used for entering data. ‘PL’ code is the allowed value.

Notes and suggestions

Any errors and comments in connection with this method are the same as for the method version generatePackagesNumbersV1. The following additional errors can also be encountered:

Unspecified error:

It is a general class of a relative broad spectrum of internal errors encountered while processing the entry. The error is: ‘UNSPECIFIED_ERROR’. Such error is usually connected with unspecified internal problems of the application.

Authorisation service error:

It is a rare error connected with possible disruption of the user authorisation service. If it occurs the following message will appear: ‘Unexpected error during executing webserwice. Try again later’.

Validation

It is the same as for generatePackagesNumbersV1 method.

Query results

The response structure is changed and PackagesGenerationResponseV2 is assigned.

XML response structure for correctly generated waybill in version V2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <Status>OK</Status>
        <SessionId>610349</SessionId>
        <Packages>
          <Package>
            <Status>OK</Status>
            <PackageId>640516</PackageId>
            <Parcels>
              <Parcel>
                <Status>OK</Status>
                <ParcelId>1093655</ParcelId>
                <Waybill>13039600210032</Waybill>
              </Parcel>
            </Parcels>
          </Package>
        </Packages>
      </return>
    </ns2:generatePackagesNumbersV2Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as generatePackagesNumbersV2Response and be PackagesGenerationResponseV2 type. The response returns the following elements:

- Packages (List<PackagePGRV2>) – a collection of parcels declared for generation of waybills. The parcels can contain a number pf packages further referred to as parcels.

The packages consists of the following elements:

- packageId(Long) – unique parcel ID, assigned in the process of generating waybills.
- reference(String) – if a reference parameter which is a component of Packages parameter within Open UMLF was defined in the process of generating waybills, it will also appear as an element of response. Please note, that this parameter should only be set up intentionally, since it overwrites the standard recorded unique ID by which the parcel is searched.
- status(String) – status of waybill generation for a specific parcel. Corresponding to the status of the entire waybill (described below).
- Parcels(List<ParcelPGRV2>) – a list of packages belonging to the parcel. Packages consist of the following elements:
  - parcelId(Long) unique package ID, assigned in the process of generating waybills.
  - reference(String) – unique package ID given by the user, similar to the process for the parcel. It is used as a search key - SearchKey.
  - waybill(String) – unique ID used as public ID for the parcel and assigned during the process of generating waybills.
  - status(String) – similar as in the parcel status (see below).
  - validationInfo(ValidationInfoPGRV2) – similar to the parcel (see below).

- validationInfo(ValidationInfoPGRV2) – responsible for transmitting information regarding status of validation of waybill submission. This element includes data if an error occurred during processing of the entry. It consists of:
    - ErrorId(Integer) – error ID.
    - ErrorCode(String) – error code/name.
    - fieldNames(String) – stores the form field name which refers to Validation.
    - Info(String) – short description of the error.
- sessionId(Long) – a unique session number concerning the performed operation. It is necessary for the business process during performing further steps of the process.
- status (String) – parameter which specifies the status of a specific parcel during generating waybills. In this version of the method - generatePackagesNumbersV2, the use of ValidationStatusPGREnumV1 type has been discontinued and replaced with a string of characters which can assume the following values:
    - OK – the parcel was correctly described and there were no problems during generating waybills.
    - NOT_VALIDATED – general validation error, which means problems with the query structure or content.
    - UNKNOWN_ERROR – an unknown error occurred during the process of generating waybills.
    - DB_ERROR – problem with connection between the service to database. It is necessary to contact the service provider.
    - INCORRECT_DATA – incorrect data entered for the parcel. Please verify and correct the data.
    - NONEXISTENT_POSTAL_CODE – the query used a post code which is not registered in the application database.
    - SERVICE_NOT_AVAILABLE – the service is not available due to unavailability of the service subsystem.
    - NONEXISTENT_PACKAGE – nonexistent package ID has been used.
    - NONEXISTENT_PARCEL – nonexistent parcel ID has been used.
    - NOT_PROCESSED – this status appears in incorrect circumstances, when parcel processing is not possible, for example no parcel record in the database.
    - DUPLICATED_PACKAGE_REFERENCE – A parcel with this unique search key already exists. Search key is set up with the input parameter called „Reference" if it was entered by the user. If it was not provided it is set up as a conjunction of string of characters consisting of „##" tab and parcel ID. If such message appears, then it is probably necessary to assign a new value to 'reference' parameter or the waybill has already been generated for the parcel.
    - DUPLICATED_PARCEL_REFERENCE – similar to the parcel, but it relates to a single parcel within the package.
    - UNKNOWN_RDB_ERROR – unknown error of the internal parcel routing system.
    - DISALLOWED_FID – in the form provided by the user it is not possible to create a waybill for the user with the provided masterFid number. A the following error message can appear: 'Payer number not allowed'
    - DUPLICATED_WAYBILL – a parcel with this unique ID already exists. The unique Waybill ID is created in the process of ordering waybills and such error should not appear at that time.
    - NONEXISTENT_COUNTRY_CODE – the country code provided is not registered in the application glossary.
    - WRONG_POSTAL_CODE_PATTERN – incorrect post code pattern has been used.
    - UNSUPPORTED_LANG_CODE – this value for generating waybills applies to versions higher than V1 and concerns langCode parameter, which is responsible for language used for creating waybills. This error appears if the above parameter has not been set up or has a value other than 'PL'.

XML response structure for the unsuccessful attempt to generate waybills:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <Status>INCORRECT_DATA</Status>
        <Packages>
          <Package>
            <Status>INCORRECT_DATA</Status>
            <Reference>FVAT 1234</Reference>
            <ValidationDetails>
              <ValidationInfo>
                <ErrorId>1502</ErrorId>
                <ErrorCode>INCORRECT_RECEIVER_POSTAL_CODE</ErrorCode>
                <FieldNames>ReceiverPostalCode</FieldNames>
                <Info>Incorrect recipient post code format(055s00)</Info>
              </ValidationInfo>
            </ValidationDetails>
            <Parcels>
              <Parcel>
                <Status>OK</Status>
                <Reference/>
              </Parcel>
            </Parcels>
          </Package>
        </Packages>
      </return>
    </ns2:generatePackagesNumbersV2Response>
  </S:Body>
</S:Envelope>
```

Data returned for this case does not differ significantly from the data returned for the case of correct generation of waybills. The main difference is the new InvalidFields parameter within the parcel described as packages. It is used for transferring information regarding any input parameter fields which did not pass the validation and must be modified. InvalidFields is a list of InvalidFieldPGRV1 type and consists of::

- fieldName(String) – field name, to which the submitted validation problem applies.
- Info(String) – a short explanation of the problem.
- status(FieldValidationStatusPGREnumV1) – status for the specific field. It can assume the following values:
    - OK – validation was successful, there are no errors for this field.
    - UNKNOWN_ERROR – unknown error.
    - DB_ERROR – an error caused by problems with connection to the database.
    - DONT_MATCH_DICTIONARY – the error occurs if a value, which does not belong to the glossary assigned to this field, is used.
    - DONT_MATCH_PATTERN – wrong pattern e.g. post code. Entered value does not match the pattern required for the specific parameter.
    - VALUE_EMPTY – error of entering an empty value for a field which requires a value.
    - VALUE_ZERO – error of entering a zero value which requires a non-zero value.
    - VALUE_OUT_OF_RANGE – error regarding the value exceeding the range allowed for the specific field.
    - VALUE_INCORRECT – incorrect field value.

- UNKNOWN_RDB_ERROR – unknown error of the routing system. The system is not able to determine the route for parcel delivery.
- DUPLICATED_KEY – error of duplicating the value key for the specific field.

Method call

An example of method call for generating waybills in version V2 is shown below:

- directed to a foreign recipient
- where the payer is a third party
- including an international service: tiresExport

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLV1>
        <packages>
          <parcels>
            <content>philosopher's stone</content>
            <customerData1>client data 11</customerData1>
            <customerData2>client data 22</customerData2>
            <customerData3>client data 33</customerData3>
            <sizeX>1</sizeX>
            <sizeY>2</sizeY>
            <sizeZ>3</sizeZ>
            <weight>1.2</weight>
          </parcels>
          <payerType>SENDER</payerType>
          <receiver>
            <address>Ul. Polna 200</address>
            <city>Radom</city>
            <company>XYZ</company>
            <countryCode>PL</countryCode>
            <email>ktos@pocztowy.pl</email>
            <fid>1495</fid>
            <name>Antoni Nowak</name>
            <phone>999999999</phone>
            <postalCode>26605</postalCode>
          </receiver>
          <ref1>Nr ref. 111/01</ref1>
          <ref2>Nr ref. 111/02</ref2>
          <ref3>FVAT 17/03</ref3>
          <sender>
            <address>Ul. Leśna 100</address>
            <city>Warszawa</city>
            <company>Sklep ABC</company>
            <countryCode>PL</countryCode>
            <email>pan@chcepaczke.pl</email>
            <fid>1495</fid>
            <name>Jan Kowalski</name>
            <phone>777777777</phone>
            <postalCode>02274</postalCode>
          </sender>
          <services>
            <cud/>
          </services>
        </packages>
      </openUMLV1>
```

```
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
       </authDataV1>
      </ns2:generatePackagesNumbersV2>
  </S:Body>
</S:Envelope>
```

Method name	generatePackagesNumbersV3
Signature	PackagesGenerationResponseV2 generatePackagesNumbersV2(OpenUMLFeV2 openUMLV2, PkgNumsGenerationPolicyV1 policyV1, String langCode, AuthDataV1 authDataV1)
Output	PackagesGenerationResponseV2

Parameters

OpenUMLFeV2 – is the second version of the parameter responsible for details of the parcel, selected services and other key information which show processing mode for parcel details. This parameter changes in comparison to method versions V1 and V2. (More information in the attachment 'Description of OpenUMLF').

PkgNumsGenerationPolicyV1 policyV1 – this parameter does not change in comparison to method version

V1.

langCode(String) – this parameter changes in comparison to method version V2.

Notes and suggestions

All errors and comments concerning this method are similar to methods in generatePackagesNumbersV1 and generatePackagesNumbersV2 versions.

Validation

For generatePackagesNumbersV3:

Field name	Validation condition	Field required?	Additional information
OpenUMLFeV2	Internal structure conditions met	Yes	Lack thereof results in the message: "Parameter openUMLV1 cannot be null"
pkgNumsGenerationPolicyV1	No validation	No	

For OpenUMLFeV2:

Field name	Validation condition	Field required?	Additional information
packages	No field	Yes	Lack thereof results in the message: „One of '(Package)' is expected."
	Internal structure conditions met		

For Receiver field:

Field name	Validation condition	Field required?	Additional information
FID	Integer value	No	
Company		No	
Name		No	
Address		No	
City		No	
CountryCode		No	Default value - „PL"
PostalCode		No	
Phone		No	
Email		No	

For Services field:

Field name	Validation condition	Field required?	Additional information
DeclaredValue	Internal structure conditions met	No	
Guarantee	Internal structure conditions met	No	
COD	Internal structure conditions met	No	
CUD		No	
ROD		No	
InPers		No	
SelfCol	Internal structure conditions met	No	
PrivPers		No	
CarryIn		No	
Duty		No	
Pallet		No	
DOX		No	
RequestedDelivDate		No	
DedicatedDelivery		No	
Tires		No	

TiresExport		No	
DpdPickup		No	

For Pudo field:

Field name	Validation condition	Field required?	Additional information
pudo	Floating point value	No	

Query results

XML structure for correctly generated waybill in version V3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      ...(no changes to V2)...
    </ns2:generatePackagesNumbersV3Response>
  </S:Body>
</S:Envelope>
```

The response's structure has not been changed in comparison to the method version generatePackagesNumbersV2 and the PackagesGenerationResponseV2 is assigned to it. Detailed information on the service response is available in the description of query results for the method.

Method call

An example of method call for generating waybills in version V3 is shown below:

- directed at the foreign recipient
- where the recipient is the payer
- including domestic service: dpdPickup (collection from the point with code PL11805)

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLFeV2>
      <packages>
        <parcels>
          <content>philosopher's stone</content>
          <customerData1>client data 11</customerData1>
          <customerData2>client data 22</customerData2>
          <customerData3>client data 33</customerData3>
          <sizeX>1</sizeX>
          <sizeY>2</sizeY>
          <sizeZ>3</sizeZ>
          <weight>1.2</weight>
        </parcels>
        <payerType>RECEIVER</payerType>
        <receiver>
          <address>Ul. Polna 200</address>
          <city>Radom</city>
          <company>XYZ</company>
          <countryCode>PL</countryCode>
          <email>ktos@pocztowy.pl</email>
          <fid>1495</fid>
```

```
          <name>Antoni Nowak</name>
          <phone>999999999</phone>
          <postalCode>26605</postalCode>
        </receiver>
        <ref1>Nr ref. 111/01</ref1>
        <ref2>Nr ref. 111/02</ref2>
        <ref3>FVAT 17/03</ref3>
        <sender>
          <address>Ul. Leśna 100</address>
          <city>Warszawa</city>
          <company>Sklep ABC</company>
          <countryCode>PL</countryCode>
          <email>pan@chcepaczke.pl</email>
          <name>Jan Kowalski</name>
          <phone>777777777</phone>
          <postalCode>02274</postalCode>
        </sender>
        <services>
        <dpdPickup>
          <pudo>PL11805</pudo>
            </dpdPickup>
          </services>
        </packages>
      </openUMLFeV2>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersV3>
  </S:Body>
</S:Envelope>
```

Method name	generatePackagesNumbersV4
Signature	PackagesGenerationResponseV2 generatePackagesNumbersV2(OpenUMLFeV3 openUMLV3, PkgNumsGenerationPolicyV1 policyV1, String langCode, AuthDataV1 authDataV1)
Output	PackagesGenerationResponseV2

Parameters

OpenUMLFeV3 openUMLFeV3 – it is the third version of the parameter responsible for details of the parcel, selected services and other key information which show processing mode for parcel details. This parameter changes in comparison to method version V3. (More information in the attachment 'Description of OpenUMLF').

PkgNumsGenerationPolicyV1 policyV1 - this parameter does not change in comparison to method version V1.

langCode(String) – this parameter changes in comparison to method versions V2 and V3.

Notes and suggestions

All errors and comments concerning this method are similar to method versions V1 and V2.

Validation

For generatePackagesNumbersV4:

Field name	Validation condition	Field required?	Additional information
OpenUMLFeV3	Internal structure conditions met	Yes	Lack thereof results in the message: "Parameter openUMLV cannot be null"
pkgNumsGenerationPolicyV1	No validation	No	

For OpenUMLFeV3:

Field name	Validation condition	Field required?	Additional information
packages	No field.	Yes	Lack thereof results in the message: „One of '(Package)' is expected."
	Internal structure conditions met		

For Currency field:

Field name	Validation condition	Field required?	Additional information
PLN	One of the values allowed	No	
EUR			
USD			
HUF			
HRK			
BGN			
DKK			
GBP			
RSD			
RUB			
TRY			
RON			
CHF			
NOK			
SEK			
CZK			

For Services field:

Field name	Validation condition	Field required?	Additional information
DeclaredValue	Internal structure conditions met	No	
Guarantee	Internal structure conditions met	No	
COD	Internal structure conditions met	No	
CUD		No	
ROD		No	
InPers		No	
SelfCol	Internal structure conditions met	No	
PrivPers		No	
CarryIn		No	
Duty	Internal structure conditions met	No	
Pallet		No	
DOX		No	
documentsInternational		No	
dpdExpress		No	
RequestedDelivDate		No	
DedicatedDelivery		No	
Tires		No	
TiresExport		No	
DpdPickup		No	

For field type (for Guarantee):

Field name	Validation condition	Field required?	Additional information
TIME0930		No	
TIME1200		No	
TIMEFIXED		No	
B2C		No	
SATURDAY		No	
INTER		No	
DPDNEXTDAY		No	

Query results

XML response structure for correctly generated waybill in version V4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
```

```
  <S:Body>
    <ns2:generatePackagesNumbersV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
     ...(no changes to V2)...
    </ns2:generatePackagesNumbersV4Response>
  </S:Body>
</S:Envelope>
```

No changes to the response structure in comparison to version V1. (detailed information on the service response is available in the description of query results for the earlier version).

Method call

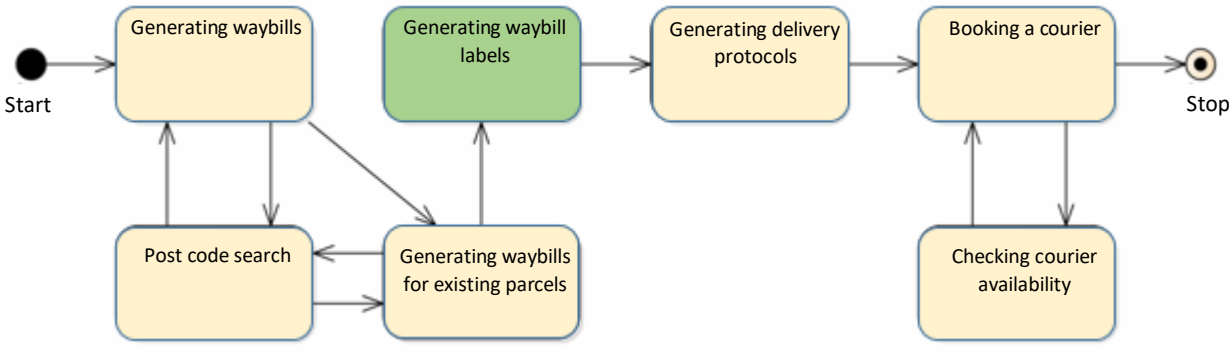
An example of method call for generating waybills in version V4 is shown below:

- directed at the foreign recipient
- where the sender is the payer
- including international services:
    - dpdExpress – air freight
    - duty – customs clearance

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLFeV3>
        <packages>
          <parcels>
            <content>philosopher's stone</content>
            <customerData1>client data 11</customerData1>
            <customerData2>client data 22</customerData2>
            <customerData3>client data 33</customerData3>
            <sizeX>1</sizeX>
            <sizeY>2</sizeY>
            <sizeZ>3</sizeZ>
            <weight>1.2</weight>
          </parcels>
          <payerType>SENDER</payerType>
          <receiver>
            <address>Ul. Polna 200</address>
            <city>Radom</city>
            <company>XYZ</company>
            <countryCode>CH</countryCode>
            <email>ktos@pocztowy.pl</email>
            <name>Antoni Nowak</name>
            <phone>999999999</phone>
            <postalCode>8200</postalCode>
          </receiver>
          <ref1>Nr ref. 111/01</ref1>
          <ref2>Nr ref. 111/02</ref2>
          <ref3>FVAT 17/03</ref3>
          <sender>
            <address>Ul. Leśna 100</address>
            <city>Warszawa</city>
            <company>Sklep ABC</company>
            <countryCode>PL</countryCode>
            <email>pan@chcepaczke.pl</email>
            <fid>1495</fid>
            <name>Jan Kowalski</name>
            <phone>777777777</phone>
            <postalCode>02274</postalCode>
```

```
          </sender>
          <services>
            <dpdExpress/>
            <duty>
              <amount>100</amount>
              <currency>PLN</currency>
            </duty>
          </services>
        </packages>
      </openUMLFeV3>
    <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
       <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersV4>
  </S:Body>
</S:Envelope>
```

1.2.3.3 Methods relating to generating labels



Method name	generateSpedLabelsV1
Signature	DocumentGenerationResponseV1 generateSpedLabelsV1(DPDServicesParamsV1 dpdServicesParamsV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV1 -

Parameters:

dpdServicesParamsV1(DPDServicesParamsV1) – list of references to parcels and processing policy. The references may be the following: session ID, parcel ID/keys, waybill ID/key/number.

policy(PolicyDSPEnumV1) – processing policy (required)

- STOP_ON_FIRST_ERROR – processing has been stopped at the occurrence of the first error
- IGNORE_ERRORS – processing has been stopped in the event of an error and the wrong parcel will be ignored

session(SessionDSPV1) – Parcel and package session (required)

- sessionId(Long) – session IDi (optional)
- packages(List<PackageDSPV1>) – List of parcels to be processed (optional). (At least one of the following fields is required: Parcel, Reference or Waybill)
    - packageId(Long) – parcel ID (optional)
    - reference(String)- parcel key (optional, ignored if parcel ID is provided)
    - parcels(List<ParcelDSPV1>) – the list of parcels to be processed (if the Parcels section does not exist, Parcel or Reference field is required)
        - parcelId(Long) – parcel ID (optional)
        - reference(String) – parcel key (optional, ignored if parcel ID is provided)
        - waybill(String) – waybill number (optional, ignored if parcel ID or key is provided)
    - sessionType(SessionTypeDSPEnumV1) – session type (required domestic or international)
        - DOMESTIC – session with domestic parcels
        - INTERNATIONAL – session with international parcels

pickupAddress(PickupAddressDSPV1) – parcel pick-up address (required for generating the protocol)

- name (String) – first name and surname for the dispatch address
- company (String) – company name for the dispatch address
- address (String) - address for the dispatch address
- city (String) – city name for the dispatch address
- countryCode (String) – country code for the dispatch address
- postalCode (String) – post code for the dispatch address
- phone (String) – phone number for the dispatch address
- email (String) – e-mail address for the dispatch address
- fid(Integer) - Numkat, whose details will be displayed in the protocol as dispatch address

documentId(String) – document ID (required for printing protocol duplicate)

outputDocFormatV1(OutputDocFormatDSPEnumV1) – format of the return document

- PDF – PDF file
- TIFF – TIFF file
- PS – PS file
- EPL – EPL file
- ZPL – ZPL file

OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1 – format of the printout page

- A4 – file in A4 format
- LBL_PRINTER – file in the label format

Comments and suggestion

While generating waybills it is possible to call for the expanded reference ref1 on the label (reference value will be displayed in large font)

It is obtained by entering ###$$$$$$$### phrase in ref2 in the form of generating waybills (see Methods in connection with generating waybills)

As a result, the label shows only ref 1 and the second reference is unavailable.

Extended label with client barcode in the middle. After the below call, barcode with ref1 field content will be printed on the extended label.

<outputDocFormatV1>PDF</outputDocFormatV1>

<outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>

<outputLabelType>EXTENDED</outputLabelType>

<labelVariant>RUCH</labelVariant>

Validation

For generateSpedLabelsV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV1	No field	Yes	Lack thereof results in the message "Parameter dpdServicesParamsV1 cannot be null"

For DPDServicesParamsV1 field:

Field name	Validation condition	Field required?	Additional information
Policy	Internal structure conditions met	Yes	
PickupAddress	Internal structure conditions met	No	
DocumentId	Maximum number of characters 10.	No	
Session	Internal structure conditions met	Yes	

For Policy field:

Field name	Validation condition	Field required?	Additional information
STOP_ON_FIRST_ERROR	One of the values allowed	No	
IGNORE_ERRORS			

For PickupAddress field

Field name	Validation condition	Field required?	Additional information
FID	Integer value	No	
Name	Maximum number of characters 100.	No	
Company	Maximum number of characters 100.	No	
Address	Maximum number of characters 100.	No	
City	Maximum number of characters 50.	No	
CountryCode	Maximum number of characters 2.	No	
PostalCode	Maximum number of characters 10.	No	
Email	Maximum number of characters 100.	No	

Phone	Maximum number of characters 100.	No	

For Session field:

Field name	Validation condition	Field required?	Additional information
SessionId	Integer value (long)	No	
SessionType	Internal structure conditions met	Yes	
Packages	Internal structure conditions met	No	

For SessionType field:

Field name	Validation condition	Field required?	Additional information
DOMESTIC	One of the values allowed	No	
INTERNATIONAL			

For Packages field:

Field name	Validation condition	Field required?	Additional information
Package	Zero or more occurrences, Internal structure conditions met	No	

For Package field:

Field name	Validation condition	Field required?	Additional information
PackageId	Integer value (long)	No	
Reference	Maximum number of characters 50.	No	
Parcels	Internal structure conditions met	No	

For Parcels field:

Field name	Validation condition	Field required?	Additional information
Parcel	Zero or more occurrences, Internal structure conditions met	No	

For Parcel field:

Field name	Validation condition	Field required?	Additional information
ParcelId	Integer value (long)	No	
Reference	Maximum number of characters50.	No	
Waybill	Maximum number of characters20.	No	

Query results

XML response structure for correctly generated label in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
    <return>

<documentData>JVBERi0xLjQKJaqrrK0KNCAwIG9iago8PAovUHJvZHVjZXIgKEFwYWNoZSBGT1AgVmVyc2lvbiBT
Vk4gdGFncy9mb3AtMV8wKQovQ3JlYXRpb25EYXRlIChEOjIwMTcxMjEyMDMzNzMyKzAxJzAwJykKPj4KZW5kb2Jq
CjUgMCBvYmoKPDwKICAvTiAzCiAgLOxlbmd0aCAxMSAwIFIKICAvRmlsdGVyIC9GbGF0ZURlY29kZQo+PgpzdHJl
YWOKeJydIndUU9kWh8…
        (data cut due to length)
      </documentData>
      <session>
        <packages>
          <packageId>640485</packageId>
          <parcels>
            <parcelId>1093618</parcelId>
            <reference>##1093618</reference>
            <statusInfo>
               <status>OK</status>
            </statusInfo>
            <waybill>0000001052503S</waybill>
          </parcels>
          <reference>##640485</reference>
          <statusInfo>
            <status>OK</status>
          </statusInfo>
        </packages>
        <sessionId>610311</sessionId>
        <statusInfo>
          <status>OK</status>
        </statusInfo>
        </session>
      </return>
    </ns2:generateSpedLabelsV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as generateSpedLabelsV1Response and be type DocumentGenerationResponseV1. The response returns the following elements:

documentData(byte[]) – Byte table containing the document

session(SessionDGRV1) – Session with parcels to be processed

- sessionId(Long) - Session ID
- statusInfo(StatusInfoDGRV1) – information regarding session processing status
    - status(StatusDGREnumV1) – processing status
        - OK – object processed correctly

- NOT_FOUND – object not found
- NOT_PROCESSED – object not processed
- INCORRECT_PKGS_FOR_SESSION_TYPE – selected parcels are not consistent with the session type
- INCORRECT_PICKUP_ADDRESS_FID – FID number (PickupAddress/FID) is not allowed or incorrect
- INCORRECT_PICKUP_ADDRESS_NAME - name (PickupAddress/Name) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_COMPANY – company name (PickupAddress/Company)) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_ADDRESS - address (PickupAddress/Address)) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_CITY - city (PickupAddress/City)) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_COUNTRY – country code (PickupAddress/Country)) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_POSTAL_CODE – post code (PickupAddress/PostalCode)) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_EMAIL – e-mail address (PickupAddress/Email)) is incorrect (not consistent with the protocol)
- INCORRECT_PICKUP_ADDRESS_PHONE – phone number (PickupAddress/Phone)) is incorrect (not consistent with the protocol)
- PARCEL_LIMIT_EXCEEDED – maximum number of parcels exceeded
- ACCESS_DENOD_FOR_FID – selected parcels have the FID number not allowed for the user
- UNKNOWN_ERROR – unknown error has occurred
- ALREADY_ADVICED – the parcel has already been advised
- ADVICE_ERROR – advice error
- DB_ERROR – database error

- Description(String – additional description
- Packages(List<PackageDGRV1>) – list of processed parcels

documentId(String) – Generated document ID (only for protocols)

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsV1>
        <policy>IGNORE_ERRORS</policy>
        <session>
          <sessionId>610311</sessionId>
          <sessionType>DOMESTIC</sessionType>
        </session>
      </dpdServicesParamsV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsV1>
```

```
  </S:Body>
</S:Envelope>
```

Method name	generateSpedLabelsV2
Signature	DocumentGenerationResponseV1 generateSpedLabelsV2(DPDServicesParamsV1 dpdServicesParamsV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, String outputLabelTypeV2, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV1

Parameters:

No changes regarding the method version V1 for parameters:

- dpdServicesParamsV1(DPDServicesParamsV1)
- outputDocFormatV1(OutputDocFormatDSPEnumV1)
- outputDocPageFormatV1(OutputDocPageFormatDSPEnumV1)

outputLabelTypeV2(String) – label type: BIC3, BIC3_EXTENDED1

Validation

For generateSpedLabelsV2:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV1	No field.	Yes	Lack thereof results in the "Parameter dpdServicesParamsV1 cannot be null" message

Validation for DPDServicesParamsV1 field has been described in the chapter concerning generateSpedLabelsV1 method.

Query results

XML response structure for correctly generated label in version V2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
    ...(no changes to V1)...
    </ns2:generateSpedLabelsV2Response>
  </S:Body>
</S:Envelope>
```

No changes to the response structure as compared to the method in version V1 (detailed description can be found in the section concerning query result for its earlier version).

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsV1>
        <policy>IGNORE_ERRORS</policy>
        <session>
          <sessionId>610311</sessionId>
          <sessionType>DOMESTIC</sessionType>
        </session>
      </dpdServicesParamsV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <authDataV1>
        <login>userlogin</login>
          <masterFid>1495</masterFid>
          <password>xxxxxxxxx</password>
        </authDataV1>
      </ns2:generateSpedLabelsV2>
    </S:Body>
</S:Envelope>
```

Method name	generateSpedLabelsV2
Signature	DocumentGenerationResponseV1 generateSpedLabelsV2(DPDServicesParamsV1 dpdServicesParamsV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, String outputLabelTypeV2, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV1

Parameters:

No changes as compared to the methods in versions V1 and V2 for parameters:

- dpdServicesParamsV1(DPDServicesParamsV1)
- outputDocFormatV1(OutputDocFormatDSPEnumV1)
- outputDocPageFormatV1(OutputDocPageFormatDSPEnumV1)

outputLabelType(OutputLabelTypeEnumV1) – label type

- BIC3
- EXTENDED

labelVariant(String) – responsible for various version of the specific label type.

Notes and suggestions

Note 1: to obtain ZPL or EPL format outputDocPageFormatV1 = LBL_PRINTER must also be set up

Note 2: type BIC3_EXTENDED1 can only be set up for outputDocPageFormatV1 = LBL_PRINTER

It is possible to print the additional information contained in the additional field for the client.

The content field can have up to 300 characters, which will be printed on the extended label.

This is performed with the use of input parameters in the form of:

<outputDocFormatV1>PDF</outputDocFormatV1>
<outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
<outputLabelType>EXTENDED</outputLabelType>
<labelVariant>APOLLO</labelVariant>

Validation

For generateSpedLabelsV3:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV1	No field	Yes	Lack thereof results in "Parameter dpdServicesParamsV1 cannot be null" message

Validation for the DPDServicesParamsV1 field has been described in the section concerning generateSpedLabelsV1 method

Query results

XML response structure for correctly generated label in version V3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
    ...(no changes to V1)...
  </ns2:generateSpedLabelsV3Response>
  </S:Body>
</S:Envelope>
```

No changes to the response structure as compared to the methods in versions V1 and V2 (detailed description can be found in query results section for method V1).

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsV1>
        <policy>IGNORE_ERRORS</policy>
        <session>
          <sessionId>610311</sessionId>
          <sessionType>DOMESTIC</sessionType>
        </session>
        </dpdServicesParamsV1>
        <outputDocFormatV1>PDF</outputDocFormatV1>
        <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
        <outputLabelType>BIC3</outputLabelType>
        <labelVariant>2</labelVariant>
        <authDataV1>
          <login>userlogin</login>
          <masterFid>1495</masterFid>
          <password>xxxxxxxxx</password>
        </authDataV1>
```

```
    </ns2:generateSpedLabelsV3>
  </S:Body>
</S:Envelope>
```

Method name	generateSpedLabelsV4
Signature	DocumentGenerationResponseV1 generateSpedLabelsV4(DPDServicesParamsV1 dpdServicesParamsV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, OutputLabelTypeEnumV1 outputLabelType, String labelVariant, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV1

Parameters:

No changes to the input parameter structure as compared to method version V3.

Validation

For generateSpedLabelsV4:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV1	No field.	Yes	Lack thereof results in the: "Parameter dpdServicesParamsV1 cannot be null" message

Validation for DPDServicesParamsV1 field has been described in the section concerning generateSpedLabelsV1 method.

Notes and suggestions:

As per V2 and V3.

Query results

XML response structure for correctly generated label in version V4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
     ...(no changes to V1)...
    </ns2:generateSpedLabelsV4Response>
  </S:Body>
</S:Envelope>
```

No changes to the response structure as compared to method version V1, V2 and V3 (detailed description can be found in the section concerning query results for method V1).

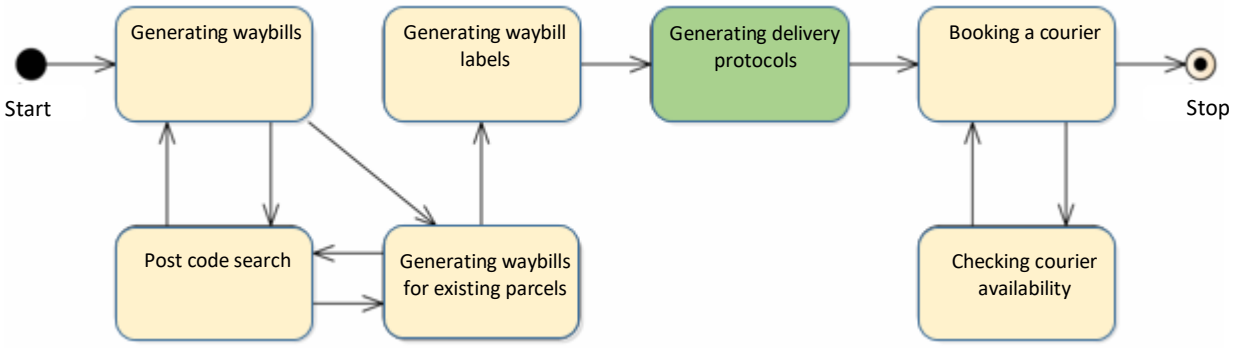
Method call

No changes to Method call as compared to version V3.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
    ...(no changes to V3)...
    </ns2:generateSpedLabelsV4>
  </S:Body>
</S:Envelope>
```

Extended label with client barcode in the middle. After the call as below, the barcode with ref1 field content will be printed on the extended label
<outputDocFormatV1>PDF</outputDocFormatV1>
<outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
<outputLabelType>EXTENDED</outputLabelType>
<labelVariant>RUCH</labelVariant>

1.2.3.4 Methods relating to generating protocols



Method name	generateProtocolV1
Signature	DocumentGenerationResponseV1 generateProtocolV1(DPDServicesParamsV1 dpdServicesParamsV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV1

Parameters:

dpdServicesParamsV1(DPDServicesParamsV1) – list of references for parcels and processing policy.

OutputDocFormatDSPEnumV1(outputDocFormatV1) - format of returned document.

OutputDocPageFormatDSPEnumV1(outputDocPageFormatV1) - format of the document page

Detailed description of these parameters is available in the chapter on generateSpedLabelsV1 method

For generateProtocolV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV1 cannot be null"

Validation:

Validation for DPDServicesParamsV1 field has been described in the chapter on generateSpedLabelsV1 method.

The errors INCORRECT_PICKUP_ADDRESS_FID or INCORRECT_PICKUP_ADDRESS_COMPANY, are errors concerning data entered as pickupAddress. During generating waybills by generatePackagesNumbers method the company address and its fid (optionally) is entered. It is required to provide consistency of that data with the data entered during the protocol generation. The company name and the company address must be identical. INCORRECT_PICKUP_ADDRESS_FID error appears if there are differences between client fid entered in the protocol and the previously entered fid number during waybill generation by one of the generatePackagesNumbers method types. The sender's data can be submitted in two ways:

- by fid – then the sender data is downloaded from the database through numkat and they can't be changed
- manually - fid is then commented and the appropriate sender fields are filled it. Please note that it is necessary to enter data identical to the data used for generating labels. Each character is important, including white spaces.

Query results

XML response structure for correctly generated protocol by generateProtocol method version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <documentData>
JVBERi0xLjQKJaqrrK0KNCAwIG9iago8PAovUHJvZHVjZXIgKEFwYWNoZSBGT1AgVmVyc2lvbiBTVk4gdGFncy9mb
3AtMV8wKQovQ3JlYXRpb25EYXRlIChEOjIwMTcxMjEyMDM1MzE1KzAxJzAwJykKPj4KZW5kb2JqCjUgMCBvYmoK
PDwKICAvTiAzCiAgL0xlbmd0aCAxMSAwIFIKICAvRmlsdGVyIC9GbGF0ZURlY29kZQo+PgpzdHJlYW0KeJydlndUU
9kWh8...
(data cut due to length)
      </documentData>
      <documentId>274966</documentId>
      <session>
        <packages>
          <packageId>640485</packageId>
          <parcels>
            <parcelId>1093618</parcelId>
            <reference>##1093618</reference>
            <statusInfo>
              <status>OK</status>
            </statusInfo>
              <waybill>0000001052503S</waybill>
            </parcels>
            <reference>##640485</reference>
            <statusInfo>
              <status>OK</status>
            </statusInfo>
          </packages>
          <sessionId>610311</sessionId>
```

```
          <statusInfo>
            <status>OK</status>
          </statusInfo>
        </session>
      </return>
    </ns2:generateProtocolV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as generateProtocolV1Response and be DocumentGenerationResponseV1 type. The query returns the result of the same structure as in generateSpedLabels method. Detailed description is available in the chapter on generateSpedLabelsV1 method.

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsV1>
        <pickupAddress>
          <fid>1495</fid>
        </pickupAddress>
        <policy>IGNORE_ERRORS</policy>
        <session>
          <sessionId>610311</sessionId>
          <sessionType>DOMESTIC</sessionType>
        </session>
      </dpdServicesParamsV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
      </ns2:generateProtocolV1>
  </S:Body>
</S:Envelope>
```

Method name	generateProtocolV2
Signature	DocumentGenerationResponseV1 generateProtocolV2(DPDServicesParamsV1 dpdServicesParamsV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV1

Parameters:

dpdServicesParamsV1(DPDServicesParamsV1) – list of references for parcels and processing policy.

OutputDocFormatDSPEnumV1(outputDocFormatV1) - format of returned document

OutputDocPageFormatDSPEnumV1(outputDocPageFormatV1) - format of document page

Validation:

For generateProtocolV2:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV1 cannot be null"

Validation for DPDServicesParamsV1 field is described in the chapter on generateSpedLabelsV1 method.

The list of accepted parameters has not changed as compared to version V1. Detailed description of the above parameters is available in the chapter on generateSpedLabelsV1 method.

are errors concerning data entered as pickupAddress. During generating waybills by generatePackagesNumbers method the company address and its fid (optionally) is entered. It is required to provide consistency of that data with the data entered during the protocol generation. The company name and the company address must be identical. INCORRECT_PICKUP_ADDRESS_FID error appears if there are differences between client fid entered in the protocol and the previously entered fid number during waybill generation by one of the generatePackagesNumbers method types.

The sender's data can be submitted in two ways:

- by fid – then the sender data is downloaded from the database through numkat and they can't be changed
- manually - fid is then commented and the appropriate sender fields are filled it. Please note that it is necessary to enter data identical to the data used for generating labels. Each character is important, including white spaces.

Query results:

XML response structure to correctly generated protocol by generateProtocol method version V2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
    ...(no changes to V1)...
    </ns2:generateProtocolV2Response>
  </S:Body>
</S:Envelope>
```

No changes to the structure of returned result as compared to method version V1. Detailed description is available in the chapter on generateSpedLabelsV1 method.

Method call:

No changes to method call mode in comparison to version V1.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
```

```
    <S:Body>
      <ns2:generateProtocolV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
    ...(no changes to V1)...
    </ns2:generateProtocolV2>
  </S:Body>
</S:Envelope>
```

Method name	generateProtocolsWithDestinationsV1
Signature	DocumentGenerationResponseV2 generateProtocolsWithDestinationsV1(DPDServicesParamsV2 dpdServicesParamsV2, AuthDataV1 authDataV1)
Output	DocumentGenerationResponseV2

Notes:

With regard to generateProtocols methods, the generateProtocolsWithDestinations method enables generating the protocol together with description of destinations into which the protocol will be divided. It is executed by an additional parameter which is a list of objects of DeliveryDestination type.

Parameters:

dpdServicesParamsV2(DPDServicesParamsV2) – list of references to parcels, processing policy and destination definitions. The reference can be as follows: Session ID, parcel ID/keys, parcel waybills ID/keys/numbers

- policy(PolicyDSPEnumV2) – processing policy (required)
  - STOP_ON_FIRST_ERROR – processing stopped at the time of the first error
  - IGNORE_ERRORS – processing stopped in the event of an error and the incorrect parcel is ignored
  - session(SessionDSPV2) – parcel and package session (required)
    - sessionId(Long) - session ID (optional)
    - packages(List<PackageDSPV2>) – list of parcels to be processed (optional). If the Parcels section does not exist, packageId or reference field is required.)
      - packageId(Long) – parcel ID (optional)
      - reference(String) – parcel key (optional, ignored if parcel ID is provided)
      - parcels(List<ParcelDSPV2>) – list of parcels to be processed (at least one of the fields is required: parcel, reference or waybill.)
    - parcelId(Long) – parcel ID (optional)
    - reference(String) – parcel key (optional, ignored if the parcel ID is provided)
    - waybill(String) – waybill number (optional, ignored if parcel ID or key is provided)
    - pickupAddress(PickupAddressDSPV2) – parcel pick-up address (required for generating the protocol)
      - fid - numkat, whose address details will be displayed on the protocol and dispatch place
      - name – first name and surname to dispatch place
      - company – company name to dispatch place
      - address – address to dispatch place

- city – city name to dispatch place
- countryCode – country code to dispatch place
- postalCode – post code to dispatch place
- phone – phone number to dispatch place
- email – e-mail address to dispatch place
- deliveryDestinations(List<DeliveryDestination>) – Definition of destinations
    - destinationName(String) – destination name (required)
    - depotList(List<ProtocolDepot>) – list of depots for the destination
        - Number(String) – depot number

- genProtForNonMatching(Boolean) – are the parcel protocols to be generated for parcels not included in defined destinations

Validation

For generateProtocolsWithDestinationsV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV2	No field	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV2 cannot be null"

DPDServicesParams parameter methods in version V2, which was extended by new fields in relation to the previous version. Changes with respect to the previous version are shown below.

In DPDServicesParamsV2 field two parameters were added (DeliveryDestinations and GenProtForNonMatching):

Field name	Validation condition	Field required?	Additional information
Policy		Yes	
PickupAddress	Internal structure conditions met	No	
DocumentId		No	
Session	Internal structure conditions met	Yes	
DeliveryDestinations	Internal structure conditions met	Yes	
GenProtForNonMatching	Boolean value type	No	

SessionType was removed from the Session field:

Field name	Validation condition	Field required?	Additional information
SessionId	Integer value (long)	No	
Packages	Internal structure conditions met	No	

Limitation to Reference field was removed in Packages:

Field name	Validation condition	Field required?	Additional information
PackageId	Integer value (long)	No	
Reference		No	

Parcels	Internal structure conditions met	No	

Limitation to Reference and Waybill fields was removed in Parcel:

Field name	Validation condition	Field required?	Additional information
ParcelId	Integer value (long)	No	
Reference	Maximum number of characters50.	No	
Waybill	Maximum number of characters20.	No	

For DeliveryDestinations field:

Field name	Validation condition	Field required?	Additional information
DeliveryDestination	Zero or more occurrences, Internal structure conditions met	No	

For DeliveryDestination field:

Field name	Validation condition	Field required?	Additional information
DeliveryName			
DepotList	Internal structure conditions met	Yes	

For DepotList field:

Field name	Validation condition	Field required?	Additional information
ProtocolDepot	Zero or more occurrences, Internal structure conditions met	Yes	

For ProtocolDepot field:

Field name	Validation condition	Field required?	Additional information
Number		No	

In the event of errors INCORRECT_PICKUP_ADDRESS_FID or INCORRECT_PICKUP_ADDRESS_COMPANY the situation is the same as in the event of validation of generateProtocolV1 i V2 methods.

Query results:

XML response structure for the correctly generated protocol by generateProtocolWithDestinations method in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolsWithDestinationsV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
    <return>
      <DestinationDataList>
        <DestinationsData>
          <DestinationName/>
```

```
          <DocumentId>1979</DocumentId>
          <Domestic>true</Domestic>
        </DestinationsData>
      </DestinationDataList>
    <DocumentData>
JVBERi0xLjQKJaqrrKOKNCAwIG9iago8PAovUHJvZHVjZXIgKEFwYWNoZSBGT1AgVmVyc2lvbiBTVk4gdGFncy9mb
3AtMV8wKQovQ3JlYXRpb25EYXRlIChEOjIwMTcxMjE5MDA1NzAzKzAxJzAwJykKPj4KZW5kb2JqCjUgMCBvYmoK
PDwKICAvTiAzCiAgLOxlbmdOaCAxMSAwIFIKICAvRmlsdGVyIC9GbGF0ZURlY29kZQo+PgpzdHJlYWOKeJydlndUU
9kWh8...
(data cut due to length)
        </DocumentData>
        <Session>
          <Packages/>
          <StatusInfo>
            <Status>OK</Status>
          </StatusInfo>
        </Session>
      </return>
    </ns2:generateProtocolsWithDestinationsV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as generateProtocolsWithDestinationsV1Response and be DocumentGenerationResponseV2 type. The response returns the following elements:

DocumentGenerationResponseV2

- documentData(byte[]) – byte table containing the document
- session(SessionDGRV2) – session with parcels to be processed
    - sessionId(Long) - Session ID
    - statusInfo(StatusInfoDGRV2) – information regarding session processing status (details below)
    - packages(List<PackageDGRV2>) – list of processed parcels
        - packageId(Long) – parcel ID
        - reference(String) – parcel key
        - statusInfo(StatusInfoDGRV2) – information regarding parcels processing status
        - parcels(List<ParcelDGRV2>) – list of processed packages
            - parcelId(Long) – package ID
            - reference(String) – package key
            - waybill(String) – waybill number
            - statusInfo(StatusInfoDGRV2) - information regarding package processing status
            - destinationDataList(List<DestinationData>) – list of destinations used on the protocols
                - documentId(String)
                - destinationName(String)
                - domestic(Boolean)
                - nonMatchingDataList(List<NonMatchingData>) – list of package numbers which are not included in the protocols
                    - waybill(String)

StatusInfoDGRV2

- status(String) – processing status
- description(String) – additional description

Method call:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolsWithDestinationsV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsV2>
        <Policy>STOP_ON_FIRST_ERROR</Policy>
        <Session>
          <SessionId>610311</SessionId>
            <Packages>
              <PackageId>640485</PackageId>
              <Parcels>
                <Parcel>
                  <ParcelId>1093618</ParcelId>
                </Parcel>
              </Parcels>
            </Packages>
          </Session>
          <PickupAddress>
            <Address>Słoneczna 11/22</Address>
            <City>Kraków</City>
            <Company>Sklep u Ewy</Company>
            <CountryCode>PL</CountryCode>
            <Email>email@skrzynkapickup.pl</Email>
            <Fid>1495</Fid>
            <Name>Sklep u Ewy</Name>
            <Phone>PPhone</Phone>
            <PostalCode>30001</PostalCode>
          </PickupAddress>
        <DeliveryDestinations>
          <DeliveryDestination>
            <DestinationName>destination name</DestinationName>
              <DepotList>
                <ProtocolDepot>
                  <Number>1305</Number>
                </ProtocolDepot>
              </DepotList>
            </DeliveryDestination>
          </DeliveryDestinations>
        <GenProtForNonMatching>true</GenProtForNonMatching>
      </dpdServicesParamsV2>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolsWithDestinationsV1>
  </S:Body>
</S:Envelope>
```

Method name	generateProtocolsWithDestinationsV2
Signature	DocumentGenerationResponseV2 generateProtocolsWithDestinationsV2(DPDServicesParamsV2 dpdServicesParamsV2, AuthDataV1 authDataV1)

Output	DocumentGenerationResponseV2

Parameters:

dpdServicesParamsV2(DPDServicesParamsV2) – list of package references, processing policy and destination definition.

Validation

For generateProtocolsWithDestinationsV2:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsV2	No field.	Yes	Lack thereof results in the message: "parameter dpdServicesParamsV2 cannot be null"

Validation for DPDServicesParamsV2 field is described in the chapter regarding method in version V1.

List of assumed parameters with respect to method version V1 has not changed.

In the event of errors INCORRECT_PICKUP_ADDRESS_FID or INCORRECT_PICKUP_ADDRESS_COMPANY the situation is the same as in the event of validation of generateProtocolV1 and V2 methods.

Query results

XML response structure for the correctly generated protocol by generateProtocolWithDestinations method version V2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolsWithDestinationsV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
    ...(no changes to V1)...
    </ns2:generateProtocolsWithDestinationsV2Response>
  </S:Body>
</S:Envelope>
```

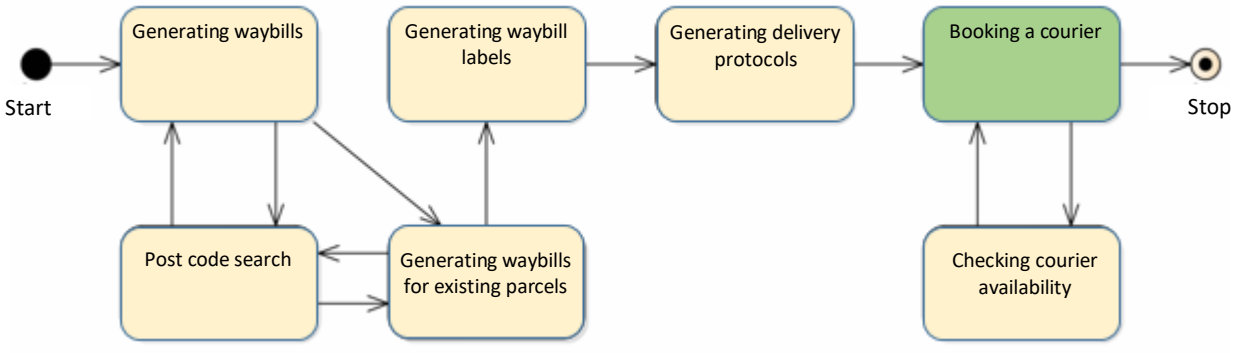
No changes in the returned document structure with respect to method version V1.

Method call

No changes to the way of method call as compared to version V1.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
    <S:Body>
      <ns2:generateProtocolsWithDestinationsV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      ...(no changes to V1)...
    </ns2:generateProtocolsWithDestinationsV2>
  </S:Body>
</S:Envelope>
```

1.2.3.5 Methods relating to booking a courier



Method name	packagesPickupCallV1
Signature	PackagesPickupCallResponseV1 packagesPickupCallV1(DPDPickupCallParamsV1 dpdPickupParamsV1, AuthDataV1 authDataV1)
Output	PackagesPickupCallResponseV1

Parameters:

dpdPickupParamsV1(DPDPickupCallParamsV1) – preferred date and time interval for collection

- policy(PolicyDPPEnumV1) – processing policy (required)
  - STOP_ON_FIRST_ERROR – processing stopped at the time of first error
  - IGNORE_ERRORS – processing will not be stopped in the event of an error and the incorrect parcel will be ignored
  - pickupAddress(PickupAddressDSPV1) – parcel collection address (required for generating the protocol) (FID field or at least one from the other fields is required). Detailed field structure is described in chapter on generateSpedLabelsV1 method.
  - contactInfo(ContactInfoDPPV1) – parcel and package session (required)
    - name(String) – First name and surname of the ordering person
    - company(String) – Company name of the ordering person
    - phone(String) – Phone number of the ordering person
    - email(String) – E-mail address
    - comments(String) – Comments of the ordering person
    - protocols(List<ProtocolDPPV1>) – Protocol list
      - documentId(String) – protocol ID
      - pickupDate(Date) – declared collection date
      - pickupTimeFrom(String) – lower time limit for collection
- pickupTimeTo(String) – upper time limit for collection

Validation

For packagesPickupCallV1:

Field name	Validation condition	Field required?	Additional information
dpdPickupParamsV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdPickupCallParamsV1 cannot be null"

For DPDPickupCallParamsV1 field:

Field name	Validation condition	Field required?	Additional information
Policy	Internal structure conditions met	Yes	
PickupAddress	Internal structure conditions met	No	
ContactInfo	Internal structure conditions met	No	
Protocols		Yes	
PickupDate	Value in the date format	Yes	
PickupTimeFrom	Maximum number of characters5.	Yes	
PickupTimeTo	Maximum number of characters5.	Yes	

For PickupAddress field:

Field name	Validation condition	Field required?	Additional information
FID	Integer value	No	
Name	Maximum number of characters100.	No	
Company	Maximum number of characters100.	No	
Address	Maximum number of characters100.	No	
City	Maximum number of characters50.	No	
CountryCode	Maximum number of characters2.	No	
PostalCode	Maximum number of characters10.	No	
Email	Maximum number of characters100.	No	
Phone	Maximum number of characters100.	No	

For ContactInfo field:

Field name	Validation condition	Field required?	Additional information
Name	Maximum number of characters100.	No	
Company	Maximum number of characters100.	No	
Email	Maximum number of characters100.	No	
Phone	Maximum number of characters100.	No	
Comments	Maximum number of characters100.	No	

For Protocols field:

Field name	Validation condition	Field required?	Additional information
Protocol	Zero or more occurrences	No	

For Protocol field:

Field name	Validation condition	Field required?	Additional information
DocumentId	Integer value	No	

For Policy field:

Field name	Validation condition	Field required?	Additional information
STOP_ON_FIRST_ERROR	One of the values allowed	No	
IGNORE_ERRORS			

Query results

XML response structure for the correctly generated courier booking in version V1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <prototocols>
          <documentId>274971</documentId>
          <statusInfo>
            <status>OK</status>
          </statusInfo>
        </prototocols>
      </return>
    </ns2:packagesPickupCallV1Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as packagesPickupCallV1Response and be PackagesPickupCallResponseV1 type.

The response returns the following elements:

PackagesPickupCallResponseV1

- orderNumber(String) – order number in the DPD system
- protocols(List<ProtocolPCRV1>) – list of protocols with status
  - documentId(String) – parcel ID
  - statusInfo(StatusInfoPCRV1) – information regarding parcel processing status
    - status(StatusPCREnumV1) – processing status
    - description(String) – additional description

StatusInfoPCRV1

- OK – object was processed correctly
- NOT_FOUND – object not found
- NOT_PROCESSED – object not processed
- INCORRECT_PICKUP_ADDRESS_FID – FID number (PickupAddress/FID) is not allowed or incorrect
- INCORRECT_PICKUP_ADDRESS_NAME - name (PickupAddress/Name) is incorrect (does not comply with the protocol)
- INCORRECT_PICKUP_ADDRESS_COMPANY – company name (PickupAddress/Company) is incorrect (does not comply with the protocol)

- INCORRECT_PICKUP_ADDRESS_ADDRESS - address (PickupAddress/Address) is incorrect (does not comply with the protocol)
- INCORRECT_PICKUP_ADDRESS_CITY - city (PickupAddress/City) is incorrect (does not comply with the protocol)
- INCORRECT_PICKUP_ADDRESS_COUNTRY – country code (PickupAddress/Country) is incorrect (does not comply with the protocol)
- INCORRECT_PICKUP_ADDRESS_POSTAL_CODE – post code (PickupAddress/PostalCode) is incorrect (does not comply with the protocol)
- INCORRECT_PICKUP_ADDRESS_EMAIL – e-mail address (PickupAddress/Email) is incorrect (does not comply with the protocol)
- INCORRECT_PICKUP_ADDRESS_PHONE – phone number (PickupAddress/Phone)
- ACCESS_DENOD_FOR_FID – the selected parcels have FID number which is not allowed for the user
- UNKNOWN_ERROR – unknown error
- DB_ERROR – database error
- RANGE_NOT_AVAILABLE_TEMPORARILY -

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsV1>
        <contactInfo>
          <email>moj@adrespoczty.pl</email>
          <name>Jan Kowalski</name>
        </contactInfo>
        <pickupAddress>
          <fid>1495</fid>
        </pickupAddress>
        <pickupDate>2018-01-01</pickupDate>
        <pickupTimeFrom>09:00</pickupTimeFrom>
        <pickupTimeTo>12:00</pickupTimeTo>
        <policy/>
      <protocols>
        <documentId>274971</documentId>
      </protocols>
    </dpdPickupParamsV1>
    <authDataV1>
      <login>userlogin</login>
      <masterFid>1495</masterFid>
      <password>xxxxxxxxx</password>
    </authDataV1>
  </ns2:packagesPickupCallV1>
</S:Body>
</S:Envelope>
```

Method name	packagesPickupCallV2
Signature	PackagesPickupCallResponseV2 packagesPickupCallV2(DPDPickupCallParamsV2 dpdPickupParamsV2, AuthDataV1 authDataV1)
Output	PackagesPickupCallResponseV2

Parameters:

DPDPickupCallParamsV2 dpdPickupParamsV2 – preferred collection date and time interval

- operationType(PickupCallOperationTypeDPPEnumV1) – operation type
  - INSERT – creating a new order
  - UPDATE – order update
  - CANCEL – cancelling the order
  - updateMode(PickupCallUpdateModeDPPEnumV1) – update mode. Parameter important only for the type of operation order update (OperationType = UPDATE)
    - DONT_CREATE_NEW_IF_CLOSED – if the updated order is already closed, a new one will not be created
    - CREATE_NEW_IF_CLOSED - if the updated order is already closed, a new one will be created
    - orderNumber(String) – order number. Parameter important only for the type of operation order update (OperationType = UPDATE)
    - pickupDate(String in the date format: yyyy-MM-dd) – order date
    - pickupTimeFrom(String in the time format: HH:mm) – lower collection time limit
    - pickupTimeTo(String in the time format: HH:mm) – upper collection time limit
    - orderType(PickupCallOrderTypeDPPEnumV1) – order type. Specifies if the order concerns domestic or international parcels
      - DOMESTIC – domestic parcel
      - INTERNATIONAL – international parcel
      - EXPRESS – international air freight parcel
      - waybillsReady(Boolean) – are the waybills ready>
      - pickupCallSimplifiedDetails(PickupCallSimplifiedDetailsDPPV1) – section with simplified order data
        - pickupPayer(PickupPayerDPPV1) – payer details
        - pickupCustomer(PickupCustomerDPPV1) – ordering party details
        - pickupSender(PickupSenderDPPV1) – sender details
- packagesParams(PickupPackagesParamsDPPV1) - parameters of the parcels

Validation

For packagesPickupCallV2:

Field name	Validation condition	Field required?	Additional information
dpdPickupParamsV2	No field	Yes	Lack thereof results in the message: "Parameter dpdPickupCallParamsV2 cannot be null"

For DPDPickupCallParamsV2 field:

Field name	Validation condition	Field required?	Additional information
OperationType	Internal structure conditions met	Yes	
UpdateMode	Internal structure conditions met	No	
OrderNumber	Maximum number of characters20.	No	
PickupDate	Value in the date format	Yes	
PickupTimeFrom	Maximum number of characters5.	Yes	

PickupTimeTo	Maximum number of characters5.	Yes	
OrderType	Internal structure conditions met	Yes	
WaybillsReady	Boolean type value	Yes	
PickupCallSimplifiedDetails	Internal structure conditions met	Yes	

For OperationType field:

Field name	Validation condition	Field required?	Additional information
INSERT	One of the values allowed	No	
UPDATE			
CANCEL			

For UpdateMode field:

Field name	Validation condition	Field required?	Additional information
DONT_CREATE_NEW_IF_CLOSED	One of the values allowed	No	
CREATE_NEW_IF_CLOSED			

For OrderType field:

Field name	Validation condition	Field required?	Additional information
DOMESTIC	One of the values allowed	No	
INTERNATIONAL			

For PickupCallSimplifiedDetails field:

Field name	Validation condition	Field required?	Additional information
PickupPayer	Internal structure conditions met	Yes	
PickupCustomer	Internal structure conditions met	Yes	
PickupSender	Internal structure conditions met	Yes	
PackagesParams	Internal structure conditions met	Yes	

For PickupPayer field:

Field name	Validation condition	Field required?	Additional information
PayerNumber	Integer value	Yes	
PayerName	Maximum number of characters100.	Yes	
PayerCostCenter	Maximum number of characters50.	No	

For PickupCustomer field:

Field name	Validation condition	Field required?	Additional information
CustomerName	Maximum number of characters100.	No	
CustomerFullName	Maximum number of characters100.	No	
CustomerPhone	Maximum number of characters20.	No	

For PickupSender field:

Field name	Validation condition	Field required?	Additional information
SenderName	Maximum number of characters 100.	Yes	
SenderFullName	Maximum number of characters 100.	No	
SenderAddress	Maximum number of characters 100.	Yes	
SenderCity	Maximum number of characters 50.	Yes	
SenderPostalCode	Maximum number of characters 10.	Yes	
SenderPhone	Maximum number of characters 50.	Yes	

For PackagesParams field:

Field name	Validation condition	Field required?	Additional information

DOX	Boolean type value	Yes	
StandardParcel	Boolean type value	Yes	
Pallet	Boolean type value	Yes	
ParcelsCount	Integer value	Yes	
PalletsCount	Integer value	Yes	
DOXCount	Integer value	Yes	
ParcelsWeight	Floating point value	No	
ParcelMaxWeight	Floating point value	No	
ParcelMaxHeight	Floating point value	No	
ParcelMaxWidth	Floating point value	No	
ParcelMaxDepth	Floating point value	No	
PalletsWeight	Floating point value	No	
PalletMaxWeight	Floating point value	No	
PalletMaxHeight	Floating point value	No	

Query results

XML response structure for the correctly generated courier booking in version V2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <orderNumber>201712193</orderNumber>
        <statusInfo>
          <status>OK</status>
        </statusInfo>
      </return>
    </ns2:packagesPickupCallV2Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as packagesPickupCallV2Response and be PackagesPickupCallResponseV2 type.

The response returns the following elements:

PackagesPickupCallResponseV2

- orderNumber(String)) – order number in the DPD system
- statusInfo(StatusInfoPCRV2) – status information on parcel processing status
  - status(String) – processing status
  - errorDetails(List<ErrorDetailsPCRV2>) – section with the list of errors
    - code(String) – error code
    - description(String) – error description
    - fields(String) – list of incorrect fields

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
      <ns2:packagesPickupCallV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
        <dpdPickupParamsV2>
          <operationType>INSERT</operationType>
          <orderType>DOMESTIC</orderType>
          <pickupDate>2018-01-01</pickupDate>
          <pickupTimeFrom>01:00</pickupTimeFrom>
          <pickupTimeTo>20:00</pickupTimeTo>
          <pickupCallSimplifiedDetails>
            <pickupCustomer>
              <customerFullName>Jakub Dąbrowski</customerFullName>
              <customerName>Jakub</customerName>
              <customerPhone>222222222</customerPhone>
            </pickupCustomer>
            <packagesParams>
              <dox>false</dox>
              <doxCount>11</doxCount>
              <pallet>false</pallet>
              <palletMaxHeight>15.2</palletMaxHeight>
              <palletMaxWeight>15.2</palletMaxWeight>
              <palletsCount>5</palletsCount>
              <palletsWeight>15.2</palletsWeight>
              <parcelMaxDepth>21.0</parcelMaxDepth>
              <parcelMaxHeight>19.2</parcelMaxHeight>
              <parcelMaxWeight>10.2</parcelMaxWeight>
              <parcelMaxWidth>20.2</parcelMaxWidth>
              <parcelsCount>3</parcelsCount>
              <parcelsWeight>10004.2</parcelsWeight>
              <standardParcel>true</standardParcel>
            </packagesParams>
            <pickupPayer>
              <payerCostCenter>abcdef</payerCostCenter>
              <payerName>Szymon Dudek</payerName>
              <payerNumber>57888</payerNumber>
            </pickupPayer>
            <pickupSender>
              <senderAddress>Ul. Leśna 100</senderAddress>
              <senderCity>Warszawa</senderCity>
              <senderFullName>Jan Kowalski</senderFullName>
              <senderName>Jan Kowalski</senderName>
              <senderPhone>777777777</senderPhone>
              <senderPostalCode>02274</senderPostalCode>
            </pickupSender>
          </pickupCallSimplifiedDetails>
          <waybillsReady>true</waybillsReady>
        </dpdPickupParamsV2>
        <authDataV1>
           <login>userlogin</login>
           <masterFid>1495</masterFid>
           <password>xxxxxxxxx</password>
         </authDataV1>
       </ns2:packagesPickupCallV2>
    </S:Body>
</S:Envelope>
```

Method name	packagesPickupCallV3

Signature	PackagesPickupCallResponseV3 packagesPickupCallV3(DPDPickupCallParamsV dpdPickupParamsV3, AuthDataV1 authDataV1)
Output	PackagesPickupCallResponseV3

Parameters:

dpdPickupParamsV3(DPDPickupCallParamsV3) – preferred date and time interval of collection.
No changes in comparison to method version V2 for the parameters:

- operationType(PickupCallOperationTypeDPPEnumV1) – operation type
- updateMode(PickupCallUpdateModeDPPEnumV1) – update mode
- orderNumber(String) – order number
- pickupDate(String in data format: yyyy-MM-dd)
- pickupTimeFrom(String in time format: HH:mm) – lower time limit for collection
- pickupTimeTo(String in time format: HH:mm) – upper time limit for collection
- orderType(PickupCallOrderTypeDPPEnumV1) – order type
- waybillsReady(Boolean) – are the waybills ready
- pickupCallSimplifiedDetails(PickupCallSimplifiedDetailsDPPV1) – section with simplified order details.

New parameter:

checkSum(Integer) – order check sum. Parameter important for update/cancellation of the order operation type (OperationType = UPDATE/CANCEL)

Validation

Method version V3 has a new parameter type DPDPickupCallParamsV3, which has been assigned two new fields in comparison to version DPDPickupCallParamsV2.

Validation is similar for INSERT type operation as in version V2. The only difference is the new allowed type for OrderType field:

Field name	Validation condition	Field required?	Additional information
DOMESTIC	One of the values allowed	No	
INTERNATIONAL			
EXPRESS			

For UPDATE type operations validation is similar to INSERT operation, the difference being that the DPDPickupCallParamsV3 parameter validates the new CheckSum field:

Field name	Validation condition	Field required?	Additional information
OperationType	Internal structure conditions met	Yes	

UpdateMode	Internal structure conditions met	No	
OrderNumber	Maximum number of characters20.	No	
CheckSum	Integer value	Yes	
PickupDate	Data format value	Yes	
PickupTimeFrom	Maximum number of characters5.	Yes	
PickupTimeTo	Maximum number of characters5.	Yes	
OrderType	Internal structure conditions met	Yes	
WaybillsReady	Boolean type value	Yes	
PickupCallSimplifiedDetails	Internal structure conditions met	Yes	

For CANCEL type operations validation of DPDPickupCallParamsV3 field has the following requirements:

Field name	Validation condition	Field required?	Additional information
OperationType	Internal structure conditions met	Yes	
OrderNumber	Maximum number of characters20.	Yes	
CheckSum	Integer value	Yes	

For OperationType field:

Field name	Validation condition	Field required?	Additional information
INSERT	One of the values allowed	No	
UPDATE			
CANCEL			

Query results
XML response structure for the correctly generated courier booking in version V3 with operationType parameter with INSERT value:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        <checkSum>61499130</checkSum>
        <orderNumber>2017122135</orderNumber>
        <statusInfo>
          <status>OK</status>
        </statusInfo>
      </return>
    </ns2:packagesPickupCallV3Response>
  </S:Body>
</S:Envelope>
```

XML response structure for the correctly generated courier booking in version V3 with operationType parameter with UPDATE or RESPONSE value:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
         <statusInfo>
           <status>OK</status>
         </statusInfo>
      </return>
    </ns2:packagesPickupCallV3Response>
  </S:Body>
</S:Envelope>
```

Correctly generated response should be labelled as packagesPickupCallV3Response and be PackagesPickupCallResponseV3 type.

No changes in relation to PackagesPickupCallResponseV2 type for parameters:

- orderNumber(String) – order number (optional)
- statusInfo(StatusInfoPCRV2) – parcel processing status information

New parameter:

checkSum(Integer) – order check sum (optional)

Method call

INSERT

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsV3>
        <operationType>INSERT</operationType>
        <orderType>DOMESTIC</orderType>
        <pickupDate>2018-01-01</pickupDate>
        <pickupTimeFrom>01:00</pickupTimeFrom>
        <pickupTimeTo>20:00</pickupTimeTo>
        <pickupCallSimplifiedDetails>
        ...(no structure changes to V2)...
        </pickupCallSimplifiedDetails>
        <waybillsReady>true</waybillsReady>
      </dpdPickupParamsV3>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:packagesPickupCallV3>
  </S:Body>
</S:Envelope>
```

UPDATE

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsV3>
        <operationType>UPDATE</operationType>
```

```
        <updateMode>DONT_CREATE_NEW_IF_CLOSED</updateMode>
        <orderType>DOMESTIC</orderType>
        <orderNumber>201712195</orderNumber>
        <checkSum>13185615</checkSum>
        <pickupDate>2018-01-01</pickupDate>
        <pickupTimeFrom>01:00</pickupTimeFrom>
        <pickupTimeTo>20:00</pickupTimeTo>
        <pickupCallSimplifiedDetails>
        ...(no structure changes to V2)...
        </pickupCallSimplifiedDetails>
        <waybillsReady>true</waybillsReady>
      </dpdPickupParamsV3>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:packagesPickupCallV3>
  </S:Body>
</S:Envelope>
```

CANCEL

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsV3>
        <operationType>CANCEL</operationType>
        <orderNumber>201712195</orderNumber>
        <checkSum>13185615</checkSum>
      </dpdPickupParamsV3>
     <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>>
    </ns2:packagesPickupCallV3>
  </S:Body>
</S:Envelope>
```

Method name	packagesPickupCallV4
Signature	PackagesPickupCallResponseV3 packagesPickupCallV4(DPDPickupCallParamsV3 dpdPickupParamsV3, AuthDataV1 authDataV1)
Output	PackagesPickupCallResponseV3

Parameters:

packagesPickupCallV4Response(PackagesPickupCallResponseV3) – preferred collection date and time interval.

No changes to the input parameters structure as compared to version V3.

Query results

XML response structure for the correctly generated courier booking in version V4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      ...(no changes to V3)...
    </ns2:packagesPickupCallV4Response>
  </S:Body>
</S:Envelope>
```

No changes to the returned result structure as compared to method version V3.

Method call

No changes to method call mode as compared to version V3.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      ...(no changes to V3)...
    </ns2:packagesPickupCallV4Response>
  </S:Body>
</S:Envelope>
```

1.2.4 Description of DPDPackageXMLService interface methods

1.2.4.1 Auxiliary XML methods

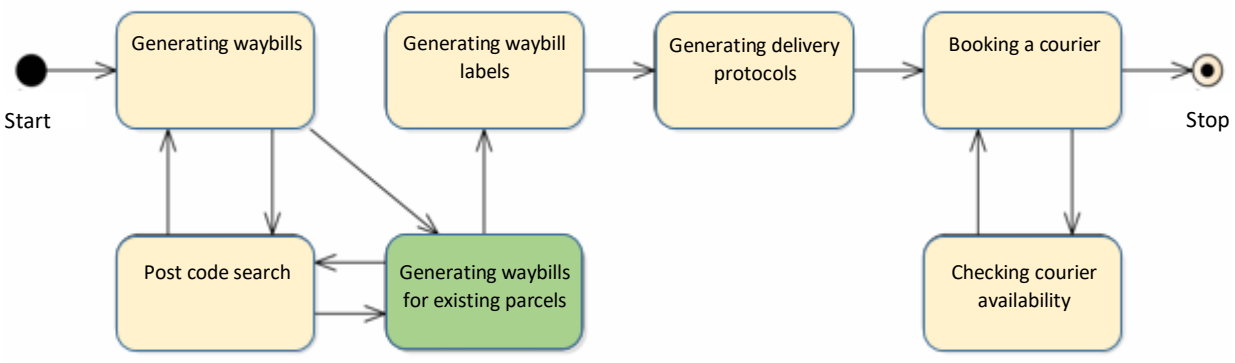
Method name	importDeliveryBusinessEventsV1
Signature	String importDeliveryBusinessEventsV1(String xmlDPDParcelBusinessEventV1, AuthDataV1 authDataV1)
Output	String

Parameters:

xmlDPDParcelBusinessEventV1 (String)

- CountryCode(String) – country code (ISO 3166-1 alpha-2)
- PostalCode(String) – post code (only meaningful characters without e.g. „-" for post codes)
- EventCode(String) – event code
- Waybill(String) – package waybill number
- EventTime(Date) – time and date of event (yyyy-MM-ddTHH-mm-ss)
- EventDataList(DPDParcelBusinessEventDataV1) – list of event's additional
    - Code(String) – code of event additional data
    - Value(String) – value of even additional data

Input parameters of the method are the equivalent of not encoded version.



Method name	appendParcelsToPackageCV1
Signature	byte[] appendParcelsToPackageCV1(byte[] parcelsAppend, AuthDataV1 authDataV1)
Output	byte[]
Method name	appendParcelsToPackageXV1
Signature	byte[] appendParcelsToPackageXV1(byte[] parcelsAppend, AuthDataV1 authDataV1)
Output	byte[]

The purpose of this method is to add packages to an existing parcel. During the attempt to do so, the possibility of adding packages is assessed, validated and if the possibility of such operation is confirmed, the packages are added and waybills are generated for new parcels.

Parameters:

parcelsAppend(byte[]) - (parameter encoded in base64 and in version CV1 previously compressed by ZIP)

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For appendParcelsToPackageXV1:

Field name	Validation condition	Field required?	Additional information
parcelsAppend	No field.	Yes	Lack thereof results in the message: "Parameter parcelsAppend cannot be null"

For appendParcelsToPackageCV1:

Field name	Validation condition	Field required?	Additional information

parcelsAppend	No field.	Yes	Lack thereof results in the message: "Parameter parcelsAppend cannot be null"

Validation of the parcelsAppend field type ParcelsAppendV1 is described in the chapter regarding version V1 of the method.

It is possible to encounter the following status field value during processing:

OK – the object was processed correctly

UNKNOWN_ERROR – unknown error occurred

PACKAGE_NOT_FOUND – parcel not found

NOT_PROCESSED – unprocessed data

BLOCKING_EVENT_EXISTS – there is a blocking status for at least one package in the parcel.

PROTOCOL_GENERATED – protocol was generated for the specific parcel.

INCORRECT_DATA – incorrect data. Details in InvalidFields section.

Query results

XML response structure for generating waybills for existing parcels in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:appendParcelsToPackageXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
PFBhcmNlbHNBcHBlbmRSZXNwb25zZVYxPjxTdGF0dXM+TOs8L1NOYXR1cz48UGFyY2Vscz48UGFyY2VscOFwcG
VuZD48U3RhdHVzPk9LPC9TdGF0dXM+PFBhcmNlbElkPjEwOTM2NzI8L1BhcmNlbElkPjxXYXliaWxsPjEzMDU5NjA
wMjMxMTgOPC9XYXliaWxsPjwvUGFyY2VscOFwcGVuZD48L1BhcmNlbHM+PC9QYXJjZWxzQXBwZW5kUmVzcG9
uc2VWMT4=
      </return>
    </ns2:appendParcelsToPackageXV1Response>
  </S:Body>
</S:Envelope>
```

The same for version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:appendParcelsToPackageCV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
UEsDBBQACAAIAChJlksAAAAAAAAAAAAAAAAEAAAAdGV4dLMJSCxKTsOpdiwoSM1LCUotLsjPKO4NM7SzCS5J
LCkttvP3ttGHMm2gauEMiCY8Sj1T7AwNLI3NzI1t9OEiNuGJlUmZOTl2hsYGppZmBgZGxoaGFqY2+jBxmFq4+fpwi/V
xORcAUEsHCApVC21lAAAAyAAAAFBLAQIUABQACAAIAChJlksKVQttZQAAAMgAAAAEAAAAAAAAAAAAAAAAAAA
AAAAABOZXhOUEsFBgAAAAABAAEAMgAAAJcAAAAAAA==
      </return>
    </ns2:appendParcelsToPackageCV1Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 (and compressed by ZIP for CV1) is as follows:

```
<ParcelsAppendResponseV1>
  <Status>OK</Status>
  <Parcels>
```

```
    <ParcelsAppend>
      <Status>OK</Status>
      <ParcelId>1093672</ParcelId>
      <Waybill>13059600231184</Waybill>
    </ParcelsAppend>
  </Parcels>
</ParcelsAppendResponseV1>
```

Method call

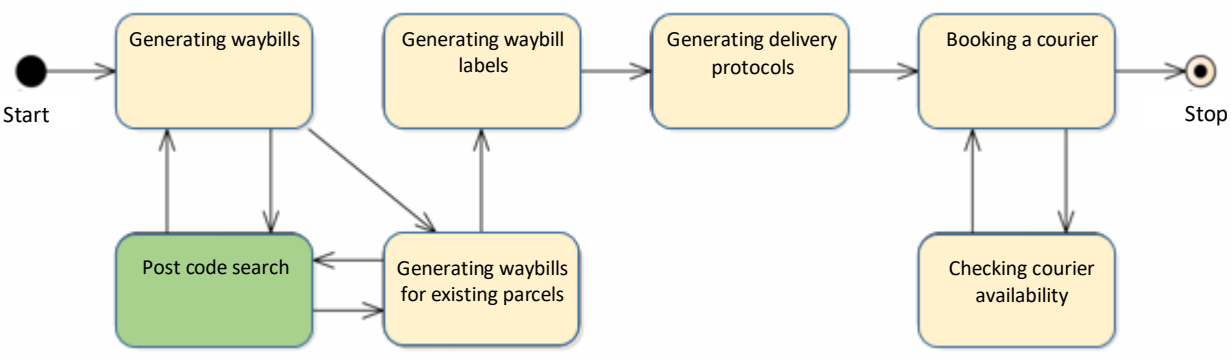
```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:appendParcelsToPackageXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <parcelsAppend>

PFBhcmNlbHNBcHBlbmRWMT48UGFja2FnZVNlYXJjaENyaXRlcmlhPjxQYWNrYWdlSWQ+NjQwNTI1PC9QYWNrY
WdlSWQ+PC9QYWNrYWdlU2VhcmNoQ3JpdGVyaWE+PFBhcmNlbHM+PFBhcmNlbD48Q29udGVudD5rYW1pZcW
EIGZpbG96b2ZpY3pueTwvQ29udGVudD48Q3VzdG9tZXJEYXRhMT5kYW5lIGtsaWVudGEgMTE8LON1c3RvbWVy
RGFOYTE+PEN1c3RvbWVyRGFOYTI+ZGFuZSBrbGllbnRhIDIyPC9DdXNOb21lckRhdGEyPjxDdXNOb21lckRhdGEz
PmRhbmUga2xpZW5OYSAzMzwvQ3VzdG9tZXJEYXRhMz48U2l6ZVg+MTwvU2l6ZVg+PFNpemVZPjM8L1NpemVZ
PjxTaXplWj4zPC9TaXplWj48V2VpZ2hOPjEuMjwvV2VpZ2hOPjwvUGFyY2VsPjwvUGFyY2Vscz48L1BhcmNlbHNBcH
BlbmRWMT4=
      </parcelsAppend>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:appendParcelsToPackageXV1>
  </S:Body>
</S:Envelope>
```

The same for version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:appendParcelsToPackageCV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <parcelsAppend>

UEsDBBQAAAAIAMtIlkstXunyvAAAAIgBAAAJAAAAaW5wdXQudHhOZZBLDoIwFEW3wgrEFnXUNDE4cWZC4m/W
wAMaoCWlDmTs1tyXT8rHxtl59572pWUnYVKou33bgsrOhGOQVqKABLAoYyMtGCnm+Jjx3Wa9pVsWLonj/OPT5RN
wFmtlQVleiUbC+xXksta9zmXaqycLxxa1R2d1A+YgrCA8EwqCqpZYiYAQ9H5r36a+TalvU9+OfDuKnL3ULJE9XDnuH
MDNN47eAG6+f2cH7AKyKCOnK1w8Mgvd62foZlq+/QNQSwECPwAUAAAACADLSJZLLV7p8rwAAACIAQAACQAkA
AAAAAAAACAAAAAAAAAAaW5wdXQudHhOCgAgAAAAAAABABgA4+P6wPt6OwExM/rA+3rTASFLwMD7etMBUEs
FBgAAAAABAAEAWwAAAOMAAAAAAA==
      </parcelsAppend>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:appendParcelsToPackageCV1>
  </S:Body>
</S:Envelope>
```

Where input data encoded in base64 (and compressed by ZIP for CV1) is as follows:

```
<ParcelsAppendV1>
  <PackageSearchCriteria>
    <PackageId>640525</PackageId>
  </PackageSearchCriteria>
  <Parcels>
    <Parcel>
      <Content>kamień filozoficzny</Content>
      <CustomerData1>dane klienta 11</CustomerData1>
      <CustomerData2>dane klienta 22</CustomerData2>
      <CustomerData3>dane klienta 33</CustomerData3>
      <SizeX>1</SizeX>
      <SizeY>3</SizeY>
      <SizeZ>3</SizeZ>
      <Weight>1.2</Weight>
    </Parcel>
  </Parcels>
</ParcelsAppendV1>
```



Method name	findPostalCodeXV1
Signature	byte[] findPostalCodeXV1(byte[] postalCodeV1, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

postalCodeV1(byte[]) - (parameter encoded in base64)

- CountryCode(String) – country code
- ZipCode(String) – post code (only significant characters e.g. without „-”)

Method’s input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For findPostalCodeXV1:

Field name	Validation condition	Field required?	Additional information
postalCodeV1	Internal structure conditions met	Yes	

For PostalCodeV1 field:

Field name	Validation condition	Field required?	Additional information
CountryCode		Yes	
ZipCode	Integer value	Yes	

Query results

XML response structure for checking the correctness of the post code in XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:findPostalCodeXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
<return>PEZpbmRQb3NOYWxDb2RlUmVzcG9uc2VWMT48U3RhdHVzPk9LPC9TdGFOdXM+PC9GaW5kUG9zdGFs
Q29kZVJlc3BvbnNlVjE+</return>
    </ns2:findPostalCodeXV1Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 is as follows:

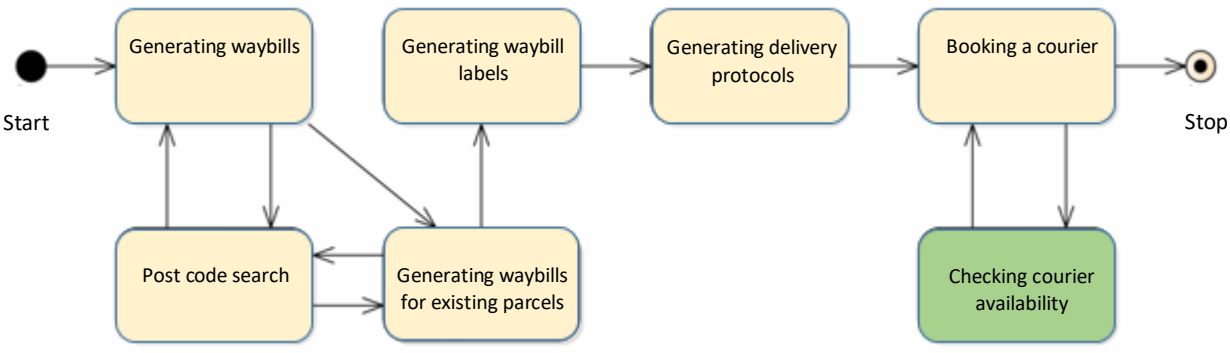
```
<FindPostalCodeResponseV1>
  <Status>OK</Status>
</FindPostalCodeResponseV1>
```

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:findPostalCodeXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <postalCodeV1>

PFBvc3RhbENvZGVWMT48Q291bnRyeUNvZGU+UEw8LONvdW5OcnlDb2RlPjxaaXBDb2RlPjAyMjc1PC9aaXBDb2
RlPjwvUG9zdGFsQ29kZVYxPg==
    </postalCodeV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:findPostalCodeXV1>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 is as follows:

```
<PostalCodeV1>
  <CountryCode>PL</CountryCode>
  <ZipCode>02275</ZipCode>
</PostalCodeV1>
```



Method name	getCourierOrderAvailabilityXV1
Signature	byte[] getCourierOrderAvailabilityXV1(byte[] senderPlaceV1, AuthDataV1 authDataV1)
Output	byte[]

Method which allows to book the courier.

Parameters:
senderPlaceV1(byte[])

- zcountryCode(String) – country code
- zipCode(String) – post code (only significant characters without e.g. „-”)

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For getCourierOrderAvailabilityXV1:

Field name	Validation condition	Field required?	Additional information
SenderPlaceV1	Internal structure conditions met	Yes	

Validation of the senderPlaceV1 field type SenderPlaceV1 is described in the chapter on version V1 of the method.

Query results

XML response structure for checking courier availability in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getCourierOrderAvailabilityXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PEdldENvdXJpZXJPcmRlckF2YWlsYWJpbGl0eVJlc3BvbnNlVjE+PFNOYXR1cz5PSzwvU3RhdHVzPjxSYW5nZXM+
PENvdXJpZXJPcmRlckF2YWlsYWJpbGl0eVJhbmdlVjE+PC9Db3VyaWVyT3JkZXJBdmFpbGFiaWxpdHlSYW5nZVY
xPjxDb3VyaWVyT3JkZXJBdmFpbGFiaWxpdHlSYW5nZVYxPjxSYW5nZT4xNDowMC0xNjowMDwvUmFuZ2U+PE9
mZnNldD4xMjA8LO9mZnNldD48LONvdXJpZXJPcmRlckF2YWlsYWJpbGl0eVJhbmdlVjE+PENvdXJpZXJPcmRlckF2
YWlsYWJpbGl0eVJhbmdlVjE+PFJhbmdlPjE1OjAwLTE3OjAwPC9SYW5nZT48T2Zmc2VOPjEyMDwvT2Zmc2VOPjwv
Q291cmllck9yZGVyQXZhaWxhYmlsaXR5UmFuZ2VWMT48Q291cmllck9yZGVyQXZhaWxhYmlsaXR5UmFuZ2VWM
T48UmFuZ2U+MTc6MDAtMTk6MDA8L1JhbmdlPjxPZmZzZXQ+PC9PZmZzZXQ+PC9Db3VyaWVyT3JkZXJBd
mFpbGFiaWxpdHlSYW5nZVYxPjwvUmFuZ2VzPjwvR2VOQ291cmllck9yZGVyQXZhaWxhYmlsaXR5UmVzcG9uc2V
WMT4=
      </return>
    </ns2:getCourierOrderAvailabilityXV1Response>
  </S:Body>
</S:Envelope>
```

Where output data encoded in base64 is as follows:

```
<GetCourierOrderAvailabilityResponseV1>
  <Status>OK</Status>
  <Ranges>
    <CourierOrderAvailabilityRangeV1></CourierOrderAvailabilityRangeV1>
    <CourierOrderAvailabilityRangeV1>
      <Range>14:00-16:00</Range>
      <Offset>120</Offset>
    </CourierOrderAvailabilityRangeV1>
    <CourierOrderAvailabilityRangeV1>
      <Range>15:00-17:00</Range>
      <Offset>120</Offset>
    </CourierOrderAvailabilityRangeV1>
    <CourierOrderAvailabilityRangeV1>
      <Range>17:00-19:00</Range>
      <Offset>120</Offset>
    </CourierOrderAvailabilityRangeV1>
  </Ranges>
</GetCourierOrderAvailabilityResponseV1>
```

Method call

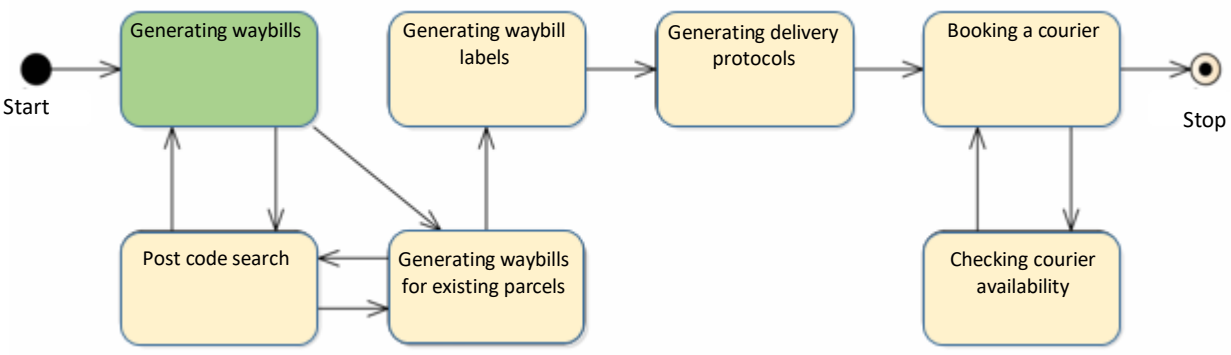
```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:getCourierOrderAvailabilityXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <senderPlaceV1>

PFNlbmRlclBsYWNlVjE+PENvdW5OcnlDb2RlPlBMPC9Db3VudHJ5Q29kZT48WmlwQ29kZT4wMjI3NTwvWmlwQ29k
ZT48L1NlbmRlclBsYWNlVjE+
      </senderPlaceV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
```

```
    </ns2:getCourierOrderAvailabilityXV1>
  </S:Body>
</S:Envelope>
```

Where input data encoded in base64 is as follows:

```
<SenderPlaceV1>
  <CountryCode>PL</CountryCode>
  <ZipCode>02275</ZipCode>
</SenderPlaceV1>
```

1.2.4.2 XML methods for generating waybills



Method name	generatePackagesNumbersCV1
Signature	byte[] generatePackagesNumbersCV1(byte[] openUMLCV1, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, AuthDataV1 authDataV1)
Output	byte{}

Method name	generatePackagesNumbersXV1
Signature	byte[] generatePackagesNumbersXV1(byte[] openUMLXV1, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, AuthDataV1 authDataV1)
Output	byte{}

Parameters

openUMLXV1(byte[]) and openUMLCV1(byte[]) – parameters responsible for data in connection with parcels, selected services and other (parameter encoded in base64 and in version CV1 additionally compressed by ZIP)

pkgNumsGenerationPolicyV1(PkgNumsGenerationPolicyV1) – glossary parameter determining service policy regarding reactions to errors occurred during execution of method call

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For generatePackagesNumbersXV1:

Field name	Validation condition	Field required?	Additional information
openUMLXV1	No field.	Yes	Lack thereof results in the message: "Parameter openUMLV1 cannot be null"

For generatePackagesNumbersCV1:

Field name	Validation condition	Field required?	Additional information
openUMLCV1	No field.	Yes	Lack thereof results in the message: "Parameter openUMLV1 cannot be null"

Validation of openUMLXV1 and openUMLCV1 fields type OpenUMLV1 is described in the chapter on version V2 of the method.

Query results

XML response structure for generating waybills in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PFBhY2thZ2VzR2VuZXJhdGlvblJlc3BvbnNlVjEPFNlc3Npb25JZD42MTAzMTU8L1Nlc3Npb25JZD48U3RhdHVzPk9L
PC9TdGF0dXM
        …(data cut due to length)
      </return>
    </ns2:generatePackagesNumbersXV1Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersCV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

UEsDBBQACAAIAG4LjEsAAAAAAAAAAAAAAAAEAAAAdGV4dH2QTQvCMAyGf1Lafckg9KiMHRwO9By3IMXSyTo
F/72d2g5kmEvePIcnJ
        …(data cut due to length)
      </return>
    </ns2:generatePackagesNumbersCV1Response>
  </S:Body>
</S:Envelope>
```

Where the return field includes the response encoded in base64 (and in version CV1 compressed by ZIP):

```
<PackagesGenerationResponseV1>
  <SessionId>610315</SessionId>
  <Status>OK</Status>
  <Packages>
    <Package>
      <PackageId>640490</PackageId>
      <Status>OK</Status>
      <InvalidFields/>
      <Parcels>
        <Parcel>
          <ParcelId>1093626</ParcelId>
          <Waybill>0000001052511S</Waybill>
          <Status>OK</Status>
        </Parcel>
      </Parcels>
    </Package>
  </Packages>
</PackagesGenerationResponseV1>
```

Method call

XML structure for generating waybills in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLXV1>

PFBhY2thZ2VzPgOKCTxQYWNrYWdlPgOKCQk8UGFyY2Vscz4NCgkJCTxQYXJjZWw+DQoJCQkJPENvbnRlbnQ+Q
29udGVudDwvQ29udGVudD4NCgkJCQk8Q3VzdG9tZXJEY
…(data cut due to length)
      </openUMLXV1>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersXV1>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersCV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLCV1>

UEsDBBQAAAAIAEkLjEsthTdVgEAAAcEAAAIAAAAZ2VuZXJhdGVQYWNrYWdlcO51bWJlcnNDVjFfYmFzZTY0Lnht
bKVTS2rDMBBdt9A79ARVbKeLgBgoSQ
…(data cut due to length)
      </openUMLCV1>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
```

```
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersCV1>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV1 compressed by ZIP) is as follows:

```
<Packages>
  <Package>
    <Parcels>
      <Parcel>
        <Content>kamień filozoficzny</Content>
        <CustomerData1>dane klienta 11</CustomerData1>
        <CustomerData2>dane klienta 22</CustomerData2>
        <CustomerData3>dane klienta 33</CustomerData3>
        <SizeX>1</SizeX>
        <SizeY>2</SizeY>
        <SizeZ>3</SizeZ>
        <Weight>1.2</Weight>
      </Parcel>
    </Parcels>
    <PayerType>SENDER</PayerType>
    <Receiver>
      <FID>1495</FID>
      <Company>XYZ</Company>
      <Name>Antoni Nowak</Name>
      <Address>Ul. Polna 200</Address>
      <City>Radom</City>
      <CountryCode>PL</CountryCode>
      <PostalCode>26605</PostalCode>
      <Phone>999999999</Phone>
      <Email>ktos@pocztowy.pl</Email>
    </Receiver>
    <Ref1>Nr ref. 111/01</Ref1>
    <Ref2>Nr ref. 111/02</Ref2>
    <Ref3>FVAT 17/03</Ref3>
    <Sender>
      <FID>1495</FID>
      <Company>Sklep ABC</Company>
      <Name>Jan Kowalski</Name>
      <Address>Ul. Leśna 100</Address>
      <City>Warszawa</City>
      <CountryCode>PL</CountryCode>
      <PostalCode>02274</PostalCode>
      <Phone>777777777</Phone>
      <Email>pan@chcepaczke.pl</Email>
    </Sender>
    <Services>
      <COD>
        <Amount>123.00</Amount>
        <Currency>PLN</Currency>
      </COD>
    </Services>
  </Package>
</Packages>
```

Method name	generatePackagesNumbersCV2
Signature	byte[] generatePackagesNumbersCV2(byte[] openUMLCV1, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, String langCode, AuthDataV1 authDataV1)
Output	byte{}

Method name	generatePackagesNumbersXV2
Signature	byte[] generatePackagesNumbersXV2(byte[] openUMLXV1, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, String langCode, AuthDataV1 authDataV1)
Output	byte{}

Parameters

openUMLXV1(byte[]) and openUMLCV1(byte[]) – parameters responsible for data regarding parcels, selected services and other (parameter encoded in base64 and in version CV1 additionally compressed by ZIP)

pkgNumsGenerationPolicyV1(PkgNumsGenerationPolicyV1) – glossary parameter determining the service policy regarding reaction to errors encountered during method call execution

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For generatePackagesNumbersXV2:

Field name	Validation condition	Field required?	Additional information
openUMLXV1	No field.	Yes	Lack thereof results in the message: "Parameter openUMLV1 cannot be null"

For generatePackagesNumbersCV2:

Field name	Validation condition	Field required?	Additional information
openUMLCV1	No field.	Yes	Lack thereof results in the message: "Parameter openUMLV1 cannot be null"

Validation of openUMLXV1 and openUMLCV1 fields type OpenUMLV1 is described in the chapter on version V2 of the method.

Query results

XML response structure for generating waybills in version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersXV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PFBhY2thZ2VzR2VuZXJhdGlvblJlc3BvbnNlVjI+PFNOYXR1cz5PSzwvU3RhdHVzPjxTZXNzaW9uSWQ+NjEwMzE5P
C9TZXNzaW9
        ...(data cut due to length)
      </return>
    </ns2:generatePackagesNumbersXV2Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersCV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure change to V1)...
      </return>
    </ns2:generatePackagesNumbersCV2Response>
  </S:Body>
</S:Envelope>
```

Method call

XML structure for generating wybills in version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersXV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLXV1>
        ...(no structure changes to V1)...
      </openUMLXV1>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersXV2>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
```

```
    <ns2:generatePackagesNumbersCV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLCV1>
        ...(no structure changes to V1)...
      </openUMLCV1>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersCV2>
  </S:Body>
</S:Envelope>
```

Where input data encoded in base64 have the same structure as in wersion V1.

Method name	generatePackagesNumbersCV3
Signature	byte[] generatePackagesNumbersCV3(byte[] openUMLFeCV2, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, String langCode, AuthDataV1 authDataV1)
Output	byte[]

Method name	generatePackagesNumbersXV3
Signature	byte[] generatePackagesNumbersXV3(byte[] openUMLFeCV2, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, String langCode, AuthDataV1 authDataV1)
Output	byte[]

Parameters

openUMLFeCV2(byte[])

pkgNumsGenerationPolicyV1(PkgNumsGenerationPolicyV1)

langCode(String)

Validation

For generatePackagesNumbersXV3:

Field name	Validation condition	Field required?	Additional information

openUMLFeCV2	No field.	Yes	Lack thereof results in the message: "Parameter openUMLFeV2 cannot be null"

For generatePackagesNumbersCV3:

Field name	Validation condition	Field required?	Additional information
openUMLFeCV2	No field.	Yes	Lack thereof results in the message: "Parameter openUMLFeV2 cannot be null"

Validation of openUMLFeCV2 and openUMLFeCV2 fields type OpenUMLFeV2 is described in the chapter on version V3 of the method.

Query results

XML response structure for generating waybills in version XV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersCV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

UEsDBBQACAAIAHwVjEsAAAAAAAAAAAAAAAAEAAAAdGV4dHWQywrCMBBFPykvWyoM2Yq4UAzoOtZBiiEtnbjw
7OOkSV3UrC7nXk5g4G
        ...(data cut due to length)
      </return>
    </ns2:generatePackagesNumbersCV3Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersCV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

UEsDBBQACAAIAHwVjEsAAAAAAAAAAAAAAAAEAAAAdGV4dHWQywrCMBBFPykvWyoM2Yq4UAzoOtZBiiEtnbjw
7OOkSV3UrC7nXk5g4G
        ...(no structure changes to V1)...
      </return>
    </ns2:generatePackagesNumbersCV3Response>
  </S:Body>
</S:Envelope>
```

Where the return field contains the response encoded in base64 (and in version CV1 compressed by ZIP):

```
<PackagesGenerationResponseV2>
  <Status>OK</Status>
  <SessionId>610326</SessionId>
  <Packages>
    <Package>
      <Status>OK</Status>
      <PackageId>640500</PackageId>
      <Parcels>
        <Parcel>
          <Status>OK</Status>
```

```
          <ParcelId>1093636</ParcelId>
          <Waybill>0000001052521S</Waybill>
        </Parcel>
      </Parcels>
    </Package>
  </Packages>
</PackagesGenerationResponseV2>
```

Method call

XML structure for generating waybills in version XV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersXV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLFeCV2>

PFBhY2thZ2VzPgOKCTxQYWNrYWdlPgOKCQk8UGFyY2Vscz4NCgkJCTxQYXJjZWw+DQoJCQkJPENvbnRlbnQ
        ...(no structure changes to V1)...
      </openUMLFeCV2>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersXV3>
  </S:Body>
</S:Envelope>
```

Similar to version CV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersCV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLFeCV2>

UEsDBBQAAAAIADQVjEvdX4rkfwEAAGEEAAAiAAAAZ2VuZXJhdGVQYWNrYWdlcO51bWJlcnNDVjNfcmVxLnR4dK
VU2WrDMBB8b
        ...(no structure changes to V1)...
      </openUMLFeCV2>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersCV3>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV3 additionally compressed by ZIP) have the similar form to the previous version of the method.

Method name	generatePackagesNumbersCV4
Signature	byte[] generatePackagesNumbersCV4(byte[] openUMLFeCV3, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, String langCode, AuthDataV1 authDataV1)
Output	byte[]

Method name	generatePackagesNumbersXV4
Signature	byte[] generatePackagesNumbersXV4(byte[] openUMLFeCV3, PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1, String langCode, AuthDataV1 authDataV1)
Output	byte[]

Parameters

byte[] openUMLFeCV3,

PkgNumsGenerationPolicyV1 pkgNumsGenerationPolicyV1,

String langCode,

Validation

For generatePackagesNumbersXV4:

Field name	Validation condition	Field required?	Additional information
openUMLFeCV3	No field.	Yes	Lack thereof results in the message: "Parameter openUMLF cannot be null"

For generatePackagesNumbersCV4

Field name	Validation condition	Field required?	Additional information
openUMLFeCV3	No field.	Yes	Lack thereof results in the message: "Parameter openUMLF cannot be null"

Validation of openUMLFeCV3 and openUMLFeCV3 fields type OpenUMLFeV3 is described in the chapter on version V4 of the method.

Query results

XML response structure for generating waybills in version XV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersXV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PFBhY2thZ2VzR2VuZXJhdGlvblJlc3BvbnNlVjIPFNOYXR1cz5JTkNPUlJFQ1RfREFUQTwvU3RhdHVzPjxQYWNrYW
dlcz48UGFja2FnZT4
        ...(data cut due to length)
      </return>
    </ns2:generatePackagesNumbersXV4Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generatePackagesNumbersCV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
UEsDBBQACAAIALoajEsAAAAAAAAAAAAAAAAEAAAAdGV4dO1WXOvDMBD
        ...(data cut due to length)
      </return>
    </ns2:generatePackagesNumbersCV4Response>
  </S:Body>
</S:Envelope>
```

Where the return field contains the response encoded in base64 (and in version CV4 compressed by ZIP):

Method call

XML structure for generating waybills in version XV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersXV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLFeCV3>

PFBhY2thZ2VzPgOKCTxQYWNrYWdlPgOKCQk8UGFyY2Vscz4NCgkJCTxQYXJjZWw+DQoJCQkJPENvbnRlbnQ
        ...(dane zostały obcięte ze względu na długość)
      </openUMLFeCV3>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersXV4>
  </S:Body>
</S:Envelope>
```

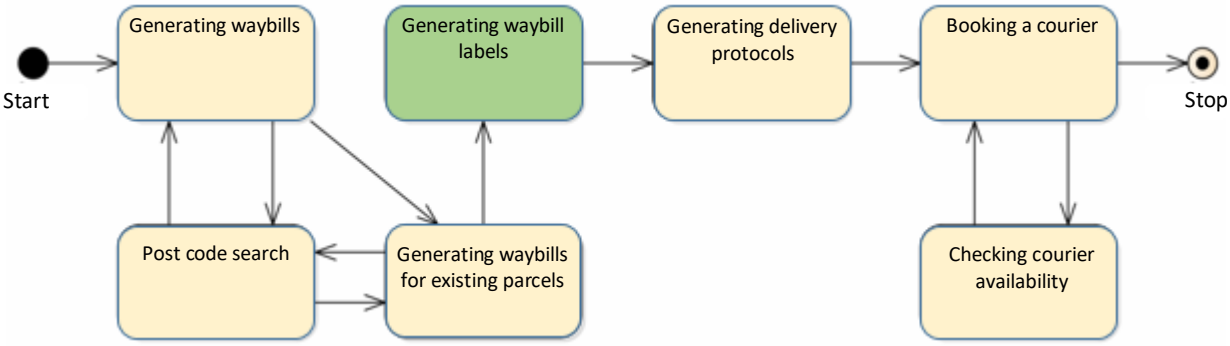
Similar to version CV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generatePackagesNumbersCV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <openUMLFeV3>
```

```
UEsDBBQAAAAIAEUajEtIyDDDogEAAMwEAAAiAAAAZ2VuZXJhdGVQYWNrYWdlc051bWJlcnNYVjRfcmVxLnR4dK
VUy2rDMBA8t
        ...(data cut due to length)
      </openUMLFeV3>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generatePackagesNumbersCV4>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV3 compressed by ZIP) have the similar form to the previous version of the method.

1.2.4.3 XML methods for generating labels



Method name	generateSpedLabelsCV1
Signature	byte[] generateSpedLabelsCV1(byte[] dpdServicesParamsCV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateSpedLabelsXV1
Signature	byte[] generateSpedLabelsXV1(byte[] dpdServicesParamsXV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

dpdServicesParamsXV1(byte[]) and dpdServicesParamsCV1(byte[]) – reference list for parcels and processing policy (parameter encoded in base64 and in version CV1 additionally compressed by ZIP)

outputDocFormatV1(OutputDocFormatDSPEnumV1) – format of the returned document

outputDocPageFormatV1(OutputDocPageFormatDSPEnumV1) - format of the document page

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For generateSpedLabelsXV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsXV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV1 cannot be null"

For generateSpedLabelsCV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsCV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV1 cannot be null"

Validation of DPDServicesParamsV1 field (for dpdServicesParamsXV1 and dpdServicesParamsCV1) is described in the chapter on version V1 of the method.

Query results

XML response structure for generating labels in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PERvY3VtZW5OR2VuZXJhdGlvbiJlc3BvbnNlVjE+PERvY3VtZW5ORGF0YT5KVkJFUmkweExqUUtKYXFyckswSO5D
QXdJRzlpYWdvOFBBb3ZVSEp2WkhWalpYSWdLRUZ3WVdOb1pTQkdU …(data cut due to length)
    </return>
    </ns2:generateSpedLabelsXV1Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsCV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
```

```
UEsDBBQACAAIAAUejEsAAAAAAAAAAAAAAAAEAAAAdGV4dIy819KrzJYteK9X4QLhIeJER+C9t+IO7OHCw9M361
9VtXd1ne4+K9YXkkgyczLtGEnC/+KmbBuKcRWLsZiTtZlGp1i+O7gUAfR//a//bOWSNfm
…(data cut due to length)
      </return>
    </ns2:generateSpedLabelsCV1Response>
  </S:Body>
</S:Envelope>
```

Where the return field contains the response encoded in base64 (and in version CV1 compressed by ZIP):

```
<DocumentGenerationResponseV1>
  <DocumentData>
JVBERi0xLjQKJaqrrK0KNCAwIG9iago8PAovUHJvZHVjZXIgKEFwYWNoZSBGT1AgVmVyc2lvbiBTVk4gdGFncy9mb
3AtMV8wKQovQ3JlYXRpb25EYXRlIChEOjIwMTcxMjIyMTIyNTQwKzAxJzAwJykK
    ...(dane zostały obcięte ze względu na długość)
  </DocumentData>
  <Session>
    <SessionId>610311</SessionId>
    <StatusInfo>
      <Status>OK</Status>
    </StatusInfo>
    <Packages>
      <Package>
      <PackageId>640485</PackageId>
      <Reference>##640485</Reference>
      <StatusInfo>
        <Status>OK</Status>
      </StatusInfo>
      <Parcels>
        <Parcel>
          <ParcelId>1093618</ParcelId>
          <Reference>##1093618</Reference>
          <WaybillI>00000010525O3S</WaybillI>
          <StatusInfo>
            <Status>OK</Status>
          </StatusInfo>
          </Parcel>
        </Parcels>
      </Package>
    </Packages>
  </Session>
</DocumentGenerationResponseV1>
```

Where output data encoded in base64 have the binary form and represent a PDF file.

Method call

XML structure for generating labels in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsXV1>

PERQRFNlcnZpY2VzUGFyYW1zVjE+DQoJPFBvbGljeT5JRO5PUkVfRVJST1JTPC9Qb2xpY3k+DQoJPFNlc3Npb24
+DQoJCTxTZXNzaW9uVHlwZT5ETO1FU1RJQzwvU2Vzc2lvblR5cGU+DQoJCTxTZXNzaW9uSWQ+NjEwMzExPC9
TZXNzaW9uSWQ+DQoJPC9TZXNzaW9uPgOKPC9EUERTZXJ2aWNlc1BhcmFtc1YxPg==
      </dpdServicesParamsXV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
```

```
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsXV1>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsCV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>

UEsDBBQAAAAIAModjEvn7NO3cQAAAK8AAAAdAAAAZ2VuZXJhdGVTcGVkTGFiZWxzWFYxX3JlcS5OeHRtjrkKhT
AUROv3wG+JQbC6pDFBUmhCIrYimiLghgHBv3fDDexmDodhgEqqzTjZyjhZjmXrcky8/w9k39hqJjxOhWIFUOooDeigu
6CNc7bvtnyVbB4MoSJhOuMRoAd9abwmIfYDjFflJPsmukcBfT1bAFBLAQI/ABQAAAAIAModjEvn7NO3cQAAAK8AAA
AdACQAAAAAAAAAIAAAAAAAAABnZW5lcmFOZVNwZWRMYWJlbHNYVjFfcmVxLnR4dAoAIAAAAAAAAQAYALSx
W2PzctMBuD5bY/NyOwEMwlNj83LTAVBLBQYAAAAAAQABAG8AAACsAAAAAAA=
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsCV1>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV1 compressed by ZIP) is as follows:

```
<DPDServicesParamsV1>
 <Policy>IGNORE_ERRORS</Policy>
 <Session>
  <SessionType>DOMESTIC</SessionType>
  <SessionId>610311</SessionId>
 </Session>
</DPDServicesParamsV1>
```

Method name	generateSpedLabelsCV2
Signature	byte[] generateSpedLabelsCV2(byte[] dpdServicesParamsCV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, String outputLabelTypeV2, AuthDataV1 authDataV1)

Output	byte[]

Method name	generateSpedLabelsXV2
Signature	byte[] generateSpedLabelsXV2(byte[] dpdServicesParamsXV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, String outputLabelTypeV2, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

No changes to the input parameter structure in comparison with method version XV1.

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V2 of the method.

Validation

No changes to validation rules in comparison with versions XV1 and CV1.

Query results

XML response structure for generating labels in version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsXV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
    </return>
    </ns2:generateSpedLabelsXV2Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsCV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
         ...(no structure changes to V1)...
      </return>
    </ns2:generateSpedLabelsCV2Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 (and in version CV2 compressed by ZIP) have the binary form and represent a PDF file.

Method call

XML structure for generating labels in version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsXV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsXV1>
        ...(no changes to V1)...
      </dpdServicesParamsXV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsXV2>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsCV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>
        ...(no changes to V1)...
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsCV2>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV2 compressed by ZIP) have the same form as version XV1 and CV1.

Method name	generateSpedLabelsCV3
Signature	byte[] generateSpedLabelsCV3(byte[] dpdServicesParamsCV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, OutputLabelTypeEnumV1 outputLabelType, String labelVariant, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateSpedLabelsXV3
Signature	byte[] generateSpedLabelsXV3(byte[] dpdServicesParamsXV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, OutputLabelTypeEnumV1 outputLabelType, String labelVariant, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

No changes to input parameter structure in comparison with method version XV1.

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V3 of the method.

Validation

No changes to validation rules in comparison to version XV1 and CV1.

Query results

XML response structure for generating labels in version XV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsXV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
      </return>
    </ns2:generateSpedLabelsXV3Response>
```

```
  </S:Body>
</S:Envelope>
```

Similar to version CV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsCV3Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
      </return>
    </ns2:generateSpedLabelsCV3Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 (and in version CV3 compressed by ZIP) have the binary form and represent a PDF file.

Method call

XML structure for generating labels in version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsXV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsXV1>
        ...(no changes to V1)...
      </dpdServicesParamsXV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <labelVariant/>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsXV3>
  </S:Body>
</S:Envelope>
```

Similar to version CV3:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsCV3 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>
        ...(no changes to V1)...
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <labelVariant/>
   <authDataV1>
        <login>userlogin</login>
```

```
    <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsCV3>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV3 compressed by ZIP) is as follows:

```
<DPDServicesParamsV1>
  <Policy>IGNORE_ERRORS</Policy>
  <Session>
    <SessionType>DOMESTIC</SessionType>
    <SessionId>610311</SessionId>
  </Session>
</DPDServicesParamsV1>
```

Method name	generateSpedLabelsCV4
Signature	byte[] generateSpedLabelsCV4(byte[] dpdServicesParamsCV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, OutputLabelTypeEnumV1 outputLabelType, String labelVariant, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateSpedLabelsXV4
Signature	byte[] generateSpedLabelsXV4(byte[] dpdServicesParamsXV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, OutputLabelTypeEnumV1 outputLabelType, String labelVariant, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

No changes to parameter dpdServicesParamsXV1 in comparison with method version XV1.

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V3 of the method.

No changes to validation rules in comparison with versions XV1 and CV1.

Query results

XML response structure for generating labels in version XV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsXV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
      </return>
    </ns2:generateSpedLabelsXV4Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateSpedLabelsCV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
      </return>
    </ns2:generateSpedLabelsCV4Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 (and in version CV4 compressed by ZIP) have the binary form and represent a PDF file.

Method call

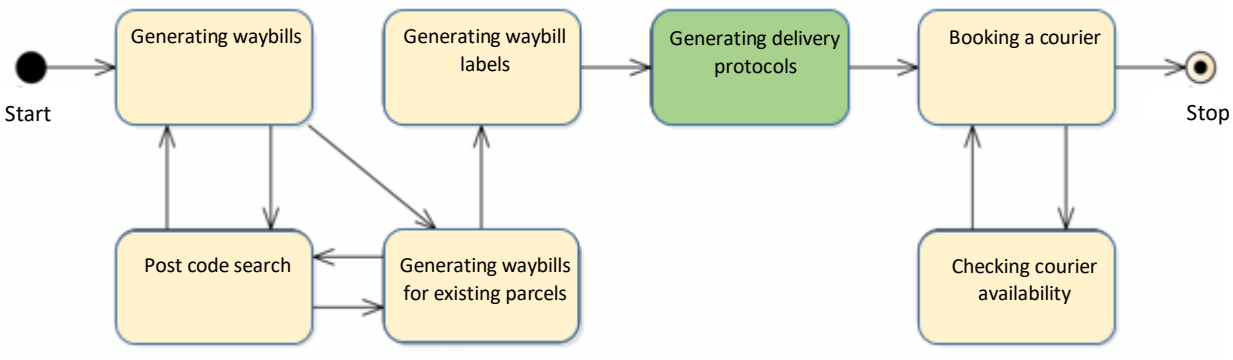
```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsXV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsXV1>
        ...(no changes to V1)...
      </dpdServicesParamsXV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <labelVariant/>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
        </authDataV1>
    </ns2:generateSpedLabelsXV4>
  </S:Body>
</S:Envelope>
```

Similar to version CV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateSpedLabelsCV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>
        ...(no changes to V1)...
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <outputLabelType>BIC3</outputLabelType>
      <labelVariant/>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateSpedLabelsCV4>
  </S:Body>
</S:Envelope>
```

Where input data encoded in base64 (and in version CV4 compressed by ZIP) have the same form as in version XV1 and CV1.

1.2.4.4 XML methods for generating protocols



Method name	generateProtocolCV1
Signature	byte[] generateProtocolCV1(byte[] dpdServicesParamsCV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateProtocolXV1
Signature	byte[] generateProtocolXV1(byte[] dpdServicesParamsXV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

dpdServicesParamsXV1(byte[]) and dpdServicesParamsCV1(byte[]) – list of parcel references and processing policy (parameter encoded in base64)

dpdServicesParamsXV1(byte[]) and dpdServicesParamsCV1(byte[]) – list of parcel references and processing policy (parameter compressed by ZIP and encoded in base64)

outputDocFormatV1(OutputDocFormatDSPEnumV1) - format of returned document

outputDocPageFormatV1(OutputDocPageFormatDSPEnumV1) - format of document page

Input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on generateSpedLabelsV1 method.

Validation

For generateProtocolXV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsXV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV1 cannot be null"

For generateProtocolCV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsCV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV1 cannot be null"

Validation for DPDServicesParamsV1 (dpdServicesParamsXV1 and dpdServicesParamsCV1) field has been described in the chapter on generateSpedLabelsXV1 and generateSpedLabelsCV1 methods.

In an error INCORRECT_PICKUP_ADDRESS_FID or INCORRECT_PICKUP_ADDRESS_COMPANY occurs, the situation is the same as in the case of validation of generateProtocolV1 and V2 methods in DPDPackageObjServices interface.

Query results

XML response structure for generating protocols by „generateProtocol” method version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

vPERvY3VtZW5OR2VuZXJhdGlvblJlc3BvbnNlVjEPERvY3VtZW5ORGFOYT5KVkJFUmkweExqUUtKYXFyckswSO5D
QXdJRzlpYWdvOFBBb3ZVSEp2WkhWalpYSWdLRUZ3WVdOb1pTQkdUMUFnVm1WeWMybHZiaUJUClZrNGdkRO
ZuY3k5bWIzQXRNVjh3S1FvdlEzSmxZWFJwYjI1RVlYUmxJQ2hFT2
       …(data cut due to length)
      </return>
    </ns2:generateProtocolXV1Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolCV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

UEsDBBQACAAIAGOgjEsAAAAAAAAAAAAAAAAEAAAAdGV4dIy8x7KkTLMlOs9XYYAmwaytzdBaa2aoRIsEEvXOT
X11xH/69O3bZXtb5iaCCA8PF2sFTvOPbsp/QzluYjmWS7o1O+iU6zyNaxnA//N//Hsrl27p/1QChnca6NRaW1XS77Kok
Gqw9CGLVJNWE2nRO…(data cut due to length)
      </return>
    </ns2:generateProtocolCV1Response>
  </S:Body>
</S:Envelope>
```

Where the return field contains the response encoded in base64 (and in version CV1 compressed by ZIP):

```
<DocumentGenerationResponseV1>
  <DocumentData>
JVBERiOxLjQKJaqrrKOKNCAwIG9iago8PAovUHJvZHVjZXIgKEFwYWNoZSBGT1AgVmVyc2lvbiBTVk4gdGFncy9mb
3AtMV8wKQovQ3JlYXRpb25EYXRlIChEOjIwMTcxMjIyMTI1NjM4KzAxJzAwJykKPj4KZW5kb2JqCjUgMCBvYmoKPD
wKICAvTiAzCiAgLOxlbmdOaCAxMSAwIFIKICAvRmlsdGVyIC9G
    …(data cut due to length)
  </DocumentData>
  <Session>
    <SessionId>610311</SessionId>
    <StatusInfo>
      <Status>OK</Status>
    </StatusInfo>
    <Packages>
      <Package>
        <PackageId>640485</PackageId>
        <Reference>##640485</Reference>
        <StatusInfo>
          <Status>OK</Status>
        </StatusInfo>
        <Parcels>
          <Parcel>
            <ParcelId>1093618</ParcelId>
            <Reference>##1093618</Reference>
            <WaybillO000001052503S</WaybillO>
          <StatusInfo>
            <Status>OK</Status>
          </StatusInfo>
        </Parcel>
      </Parcels>
```

```
    </Package>
  </Packages>
</Session>
<DocumentId>274977</DocumentId>
```

Where the output data encoded in base64 have the binary form and represent a PDF file.

Method call

XML structure for generating protocols by generateProtocol method version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>

PERQRFNlcnZpY2VzUGFyYW1zVjE+DQoJPFBvbGljeT5JRO5PUkVfRVJST1JTPC9Qb2xpY3k+DQoJPFNlc3Npb24
+DQoJCTxTZXNzaW9uVHlwZT5ETO1FU1RJQzwvU2Vzc2lvblR5cGU+DQoJCTxTZXNzaW9uSWQ+NjEwMzExPC9
TZXNzaW9uSWQ+DQoJPC9TZXNzaW9uPgOKPC9EUERTZXJ2aWNlc1BhcmFtc1YxPg==
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolXV1>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolCV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>

UEsDBBQAAAAIAEggjEvn7NO3cQAAAK8AAAAbAAAAZ2VuZXJhdGVQcm9Ob2NvbENWMV9yZXEudHh0bY65CoU
wFETr98BviUGwuqQxQVJoQiK2Ipoi4IYBwb93ww3sZg6HYYBKqsO42co4WY5l63JMvP8PZN/YaiY8ToViBVNKKA3o
oLugjXO277Z8lWweDKEiYTrjEaAHfWm8JiH2A4xX5ST7JrpHAXO9WwBQSwECPwAUAAAACABIIIxL5+zdN3EAAAC
vAAAAGwAkAAAAAAAAACAAAAAAAAAAZ2VuZXJhdGVQcm9Ob2NvbENWMV9yZXEudHh0CgAgAAAAAAABABg
AbBnmnPVyOwF8puWc9XLTAYMD3pz1ctMBUEsFBgAAAAABAAEAbQAAAKoAAAAAA==
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolCV1>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV1 compressed by ZIP) is as follows:

```
<DPDServicesParamsV1>
  <Policy>IGNORE_ERRORS</Policy>
    <Session>
      <SessionType>DOMESTIC</SessionType>
    <SessionId>610311</SessionId>
  </Session>
</DPDServicesParamsV1>
```

Method name	generateProtocolCV2
Signature	byte[] generateProtocolCV2(byte[] dpdServicesParamsCV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateProtocolXV2
Signature	byte[] generateProtocolXV2(byte[] dpdServicesParamsXV1, OutputDocFormatDSPEnumV1 outputDocFormatV1, OutputDocPageFormatDSPEnumV1 outputDocPageFormatV1, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

The list of accepted parameters in relation to method version V1 has not changed. Detailed description of the above parameters can be found in the chapter on generateSpedLabelsV1 method.

Validation

No changes to validation rules in comparison with versions XV1 and CV1.

Query results

XML response structure for generation protocol by generateProtocol method version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolXV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
      </return>
```

```
    </ns2:generateProtocolXV2Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
   <S:Body>
    <ns2:generateProtocolCV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
     <return>
      …(no structure changes to V1)…
     </return>
    <ns2:generateProtocolCV2Response>
   </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 (and in version CV2 compressed by ZIP) have the binary form and represent a PDF file.

Method call

XML structure for generating protocols by „generateProtocol" method in version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolXV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>
        ...(brak zmian względem V1)...
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolXV2>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolCV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV1>
        ...(no changes to V1)...
      </dpdServicesParamsCV1>
      <outputDocFormatV1>PDF</outputDocFormatV1>
      <outputDocPageFormatV1>LBL_PRINTER</outputDocPageFormatV1>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolCV2>
```

```
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV2 compressed by ZIP) have the same form as in version XV1 and CV1.

Method name	generateProtocolsWithDestinationsCV1
Signature	byte[] generateProtocolsWithDestinationsCV1(byte[] dpdServicesParamsCV2, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateProtocolsWithDestinationsXV1
Signature	byte[] generateProtocolsWithDestinationsXV1(byte[] dpdServicesParamsXV2, AuthDataV1 authDataV1)
Output	byte[]

Comments:

With respect to generateProtocols methods, the generateProtocolsWithDestinations method enables generating protocols together with description of destinations into which the protocol will be divided. This functionality is carried out by an additional parameter being the list of DeliveryDestination type objects.

Parameters:

dpdServicesParamsXV2(byte[]) – (parameter encoded in base64)

dpdServicesParamsCV2(byte[]) – (parameter compressed by ZIP and encoded in base64)

Validation

For generateProtocolsWithDestinationsXV1:

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsXV2	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV2 cannot be null"

For generateProtocolsWithDestinationsCV1

Field name	Validation condition	Field required?	Additional information
dpdServicesParamsCV2	No field.	Yes	Lack thereof results in the message: "Parameter dpdServicesParamsV2 cannot be null"

Validation of the dpdServicesParamsXV2 and generateProtocolsWithDestinationsCV1 fields type DPDServicesParamsV2 is described in the chapter on version V1 of the method.

Query results:

XML response structure for generating protocols by generateProtocolWithDestinations method in versionXV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolsWithDestinationsXV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

vY3VtZW5OR2VuZXJhdGlvblJlc3BvbnNlVjIPERvY3VtZW5ORGFOYT5KVkJFUmkweExqUUtKYXFyckswSO5DQXdJR
zlpYWdvOFBBb3ZVSEp2WkhWalpYSWdLRUZ3WVdOb1pTQkdUMUFnVm1WeWMybHZiaUJUClZrNGdkROZuY3k5
bWIzQXRNVjh3S1FvdlEzSmxZWFJwYjI1RVlYUmxJQ2hFT2pJd01UY3hNakU1TURFd01ERXdLekF4SnpBdOp5aOsK
UGoOS1pXNW
        …(data cut due to length)
      </return>
    </ns2:generateProtocolsWithDestinationsXV1Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolsWithDestinationsCV1Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

UEsDBBQACAAIAHcIkOsAAAAAAAAAAAAAAAAEAAAAdGV4dIy7166OyrIlfF+vwgWeAqnVEt4V3nOHKbwrPDx9861
19t7r/Kf1q6fmVFEzITOJjBgxRhL8L27Mtv47rOJ3+M7JWo+D/V2mcVi+PvK//9e
        …(dane zostały obcięte ze względu na długość)
      </return>
    </ns2:generateProtocolsWithDestinationsCV1Response>
  </S:Body>
</S:Envelope>
```

Where the return field contains the response encoded in base64 (and in version CV1 compressed by ZIP):

```
<DocumentGenerationResponseV2>
<DocumentData>JVBERiOxLjQKJaqrrKOKNCAwIG9iago8PAovUHJvZHVjZXIgKEFwYWNoZSBGT1AgVmVyc2lvbiBT
4gdGFncy9mb3AtMV8wKQovQ3JlYXRpb25EYXRlIChEOjIwMTcxMjIyMTMzMDE5KzAxJzAwJykK
    …(dane zostały obcięte ze względu na długość)
  </DocumentData>
  <Session>
    <StatusInfo>
      <Status>OK</Status>
    </StatusInfo>
    <Packages></Packages>
  </Session>
  <DestinationDataList>
    <pl.com.dpd.dpdservicesbeans.to.docgenresp.v2.DestinationsData>
      <DocumentId>1991</DocumentId>
      <DestinationName></DestinationName>
      <Domestic>true</Domestic>
    </pl.com.dpd.dpdservicesbeans.to.docgenresp.v2.DestinationsData>
  </DestinationDataList>
</DocumentGenerationResponseV2>
```

Where the output data encoded in base64 have the binary form and represent a PDF file.

Method call

XML structure for generating protocols by „generateProtocolWithDestinations” method version XV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolsWithDestinationsXV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsXV2>

PERQRFNlcnZpY2VzUGFyYW1zVjI+PFBvbGljeT5TVE9QX09OX0ZJUlNUX0VSUk9SPC9Qb2xpY3k+PFNlc3Npb24
…(data cut due to length)
        </dpdServicesParamsXV2>
        <authDataV1>
          <login>userlogin</login>
          <masterFid>1495</masterFid>
          <password>xxxxxxxxx</password>
        </authDataV1>
    </ns2:generateProtocolsWithDestinationsXV1>
  </S:Body>
</S:Envelope>
```

Similar to version CV1:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolsWithDestinationsCV1 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV2>

UEsDBBQAAAAIAFkIkOsq1gTBSwEAAPoCAAAsAAAAZ2VuZXJhdGVQcm9Ob2NvbHNXaXRoRGVzdGluYXRpb25z
Q1YxX3JlcS5OeHRtUmFrwjAQ
        …(data cut due to length)
      </dpdServicesParamsCV2>
    <authDataV1>
      <login>userlogin</login>
      <masterFid>1495</masterFid>
      <password>xxxxxxxxx</password>
    </authDataV1>
  </ns2:generateProtocolsWithDestinationsCV1>
</S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV1 compressed by ZIP) is as follows:

```
<DPDServicesParamsV2>
  <Policy>STOP_ON_FIRST_ERROR</Policy>
  <Session>
    <SessionId>610311</SessionId>
    <Packages>
      <Package>
        <PackageId>640485</PackageId>
        <Parcels>
          <Parcel>
            <ParcelId>1093618</ParcelId>
          </Parcel>
        </Parcels>
      </Package>
```

```
    </Packages>
  </Session>
  <PickupAddress>
    <Address>Słoneczna 11/22</Address>
    <City>Kraków</City>
    <Company>Sklep u Ewy</Company>
    <CountryCode>PL</CountryCode>
    <Email>email@skrzynkapickup.pl</Email>
    <FID>1495</FID>
    <Name>Sklep u Ewy</Name>
    <Phone>PPhone</Phone>
    <PostalCode>30001</PostalCode>
  </PickupAddress>
  <DeliveryDestinations>
    <DeliveryDestination>
      <DestinationName>nazwa kierunku</DestinationName>
      <DepotList>
          <ProtocolDepot>
            <Number>1305</Number>
          </ProtocolDepot>
        </DepotList>
      </DeliveryDestination>
    </DeliveryDestinations>
  <GenProtForNonMatching>true</GenProtForNonMatching>
</DPDServicesParamsV2>
```

Method name	generateProtocolsWithDestinationsCV2
Signature	byte[] generateProtocolsWithDestinationsCV2(byte[] dpdServicesParamsCV2, AuthDataV1 authDataV1)
Output	byte[]

Method name	generateProtocolsWithDestinationsXV2
Signature	byte[] generateProtocolsWithDestinationsXV2(byte[] dpdServicesParamsXV2, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

byte[] dpdServicesParamsXV2 – (encoded in base64)

byte[] dpdServicesParamsCV2 – (parameter compressed by ZIP and encoded in base64)

Validation

No changes to validation rules in comparison to version XV1 and CV1.

Query results

XML response structure for generating protocols by „generateProtocolWithDestinations" method version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolsWithDestinationsXV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...(no structure changes to V1)...
      </return>
    </ns2:generateProtocolsWithDestinationsXV2Response>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:generateProtocolsWithDestinationsCV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...( no structure changes to V1)...
      </return>
    </ns2:generateProtocolsWithDestinationsCV2Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 (and in version CV2 compressed by ZIP) have the binary form and represent a PDF file.

Method call

XML structure for generating protocols by generateProtocolWithDestinations method version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolsWithDestinationsXV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsXV2>
        ...(no changes to V1)...
      </dpdServicesParamsXV2>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolsWithDestinationsXV2>
  </S:Body>
</S:Envelope>
```

Similar to version CV2:

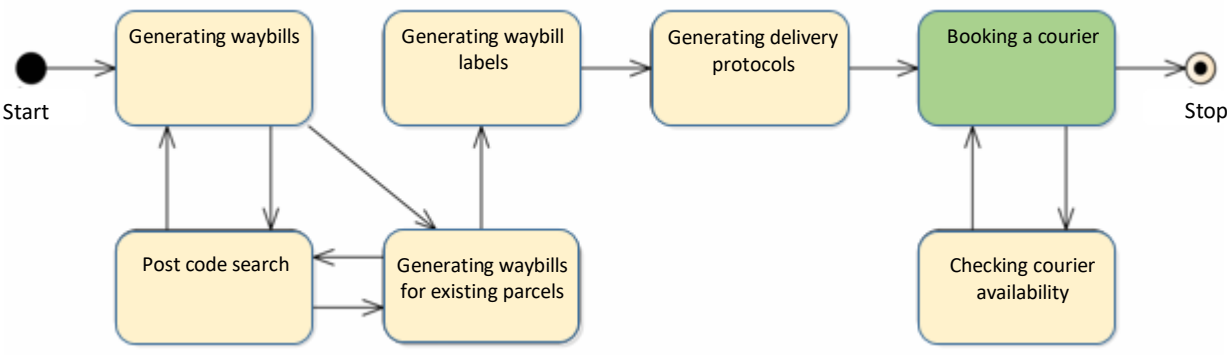
```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:generateProtocolsWithDestinationsCV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdServicesParamsCV2>
```

```
        ...(brak zmian względem V1)...
      </dpdServicesParamsCV2>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:generateProtocolsWithDestinationsCV2>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 (and in version CV2 compressed by ZIP) have the same form as in versions XV1 and CV1.

When errors INCORRECT_PICKUP_ADDRESS_FID or INCORRECT_PICKUP_ADDRESS_COMPANY occur

1.2.4.5 XML methods for booking a courier



Method name	packagesPickupCallXV1
Signature	byte[] packagesPickupCallXV1(byte[] dpdPickupParamsXV1, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

dpdPickupParamsXV1(byte[]) – preferred pickup (parameter encoded in base64)

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V1 of the method.

Validation

For packagesPickupCallXV1:

Field name	Validation condition	Field required?	Additional information
dpdPickupParamsXV1	No field.	Yes	Lack thereof results in the message: "Parameter dpdPickupCallParamsV1 cannot be null"

Validation of the dpdPickupParamsXV1 field type DPDPickupCallParamsV1 is described in the chapter on version V1 of the method.

Method name	packagesPickupCallXV2
Signature	byte[] packagesPickupCallXV2(byte[] dpdPickupParamsXV2, AuthDataV1 authDataV1)
Output	byte[]

Parameters: dpdPickupParamsXV2(byte[]) – preferred pickup date and time interval (parameter encoded in base64)

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V2 of the method.

Validation

For packagesPickupCallXV2:

Field name	Validation condition	Field required?	Additional information
dpdPickupParamsXV2	No field.	Yes	Lack thereof results in the message: "Parameter dpdPickupCallParamsV2 cannot be null"

Validation of the dpdPickupParamsXV2 field type DPDPickupCallParamsV1 is described in the chapter on version V1 of the method.

Query results

XML response structure for booking a courier in version XV2:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallXV2Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PFBhY2thZ2VzUGlja3VwQ2FsbFJlc3BvbnNlVjI+PE9yZGVyTnVtYmVyPjIwMTcxMjIyMjwvT3JkZXJOdW1iZXI+PFNO
YXR1cOluZm8+PFNOYXR1cz5PSzwvU3RhdHVzPjxFcnJvckRldGFpbHM+PC9FcnJvckRldGFpbHM+PC9TdGFOdXN
JbmZvPjwvUGFja2FnZXNQaWNrdXBDYWxsUmVzcG9uc2VWMj4=
      </return>
    </ns2:packagesPickupCallXV2Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 are as follows:

```
<PackagesPickupCallResponseV2>
  <OrderNumber>201712222</OrderNumber>
  <StatusInfo>
    <Status>OK</Status>
    <ErrorDetails></ErrorDetails>
  </StatusInfo>
</PackagesPickupCallResponseV2>
```

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallXV2 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsXV2>

PERQRFBpY2t1cENhbGxQYXJhbXNWMj48T3BlcmFOaW9uVHlwZT5JTlNFUlQ8LO9wZXJhdGlvblR5cGU+PE9yZG
VyVHlwZT5ETO1FU1RJQzwvT3JkZXJUeXBlPjxQaWNrdXBEYXRlPjIwMTgtMDEtMDE8L1BpY2t1cERhdGU…
    (data cut due to length)</dpdPickupParamsXV2>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:packagesPickupCallXV2>
  </S:Body>
</S:Envelope>
```

Where the intput data encoded in base64 are as follows:

```
<DPDPickupCallParamsV2>
  <OperationType>INSERT</OperationType>
  <OrderType>DOMESTIC</OrderType>
  <PickupDate>2018-01-01</PickupDate>
  <PickupTimeFrom>01:00</PickupTimeFrom>
  <PickupTimeTo>20:00</PickupTimeTo>
  <PickupCallSimplifiedDetails>
    <PickupCustomer>
      <CustomerFullName>Jakub Dąbrowski</CustomerFullName>
      <CustomerName>Jakub</CustomerName>
      <CustomerPhone>222222222</CustomerPhone>
    </PickupCustomer>
    <PackagesParams>
      <DOX>false</DOX>
      <DOXCount>11</DOXCount>
      <Pallet>false</Pallet>
      <PalletMaxHeight>15.2</PalletMaxHeight>
      <PalletMaxWeight>15.2</PalletMaxWeight>
      <PalletsCount>5</PalletsCount>
      <PalletsWeight>15.2</PalletsWeight>
      <ParcelMaxDepth>21.0</ParcelMaxDepth>
      <ParcelMaxHeight>19.2</ParcelMaxHeight>
      <ParcelMaxWeight>10.2</ParcelMaxWeight>
      <ParcelMaxWidth>20.2</ParcelMaxWidth>
      <ParcelsCount>3</ParcelsCount>
      <ParcelsWeight>10004.2</ParcelsWeight>
      <StandardParcel>true</StandardParcel>
    </PackagesParams>
    <PickupPayer>
      <PayerCostCenter>abcdef</PayerCostCenter>
      <PayerName>Szymon Dudek</PayerName>
      <PayerNumber>57888</PayerNumber>
    </PickupPayer>
    <PickupSender>
      <SenderAddress>Ul. Leśna 100</SenderAddress>
      <SenderCity>Warszawa</SenderCity>
      <SenderFullName>Jan Kowalski</SenderFullName>
      <SenderName>Jan Kowalski</SenderName>
      <SenderPhone>777777777</SenderPhone>
```

```
      <SenderPostalCode>02274</SenderPostalCode>
    </PickupSender>
  </PickupCallSimplifiedDetails>
<WaybillsReady>true</WaybillsReady>
</DPDPickupCallParamsV2>
```

Method name	packagesPickupCallXV3
Signature	byte[] packagesPickupCallXV3(byte[] dpdPickupParamsXV3, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

dpdPickupParamsXV3(byte[]) – preferred pickup date and time interval (parameter encoded in base64)

Method's input parameters are the equivalent of the uncoded version. Their detailed description can be found in the chapter on version V3 of the method.

Validation

For packagesPickupCallXV3:

Field name	Validation condition	Field required?	Additional information
dpdPickupParamsXV3	No field.	Yes	Lack thereof results in the message: "Parameter dpdPickupCallParamsV3 cannot be null"

Validation of the dpdPickupParamsXV3 field type DPDPickupCallParamsV3 is described in the chapter on version V3 of the method.

Query results

INSERT

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallXV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>

PFBhY2thZ2VzUGlja3VwQ2FsbFJlc3BvbnNlVjM+PE9yZGVyTnVtYmVyPjIwMTcxMjE5OTwvT3JkZXJOdW1iZXI+PF
NOYXR1cOluZm8+PFNOYXR1cz5PSzwvU3RhdHVzPjxFcnJvckRldGFpbHM+PC9FcnJvckRldGFpbHM+PC9TdGF0d
XNJbmZvPjxDaGVja1N1bT45NjEONjMyMzwvQ2hlY2tTdWO+PC9QYWNrYWdlc1BpY2t1cENhbGxSZXNwb25zZVYz
Pg==
      </return>
    </ns2:packagesPickupCallXV4Response>
  </S:Body>
</S:Envelope>
```

Where the output data encoded in base64 are as follows:

INSERT

```
<PackagesPickupCallResponseV3>
  <OrderNumber>201712199</OrderNumber>
    <StatusInfo>
      <Status>OK</Status>
      <ErrorDetails></ErrorDetails>
    </StatusInfo>
  <CheckSum>96146323</CheckSum>
</PackagesPickupCallResponseV3>
```

UPDATE AND CANCEL

```
<PackagesPickupCallResponseV3>
  <StatusInfo>
    <Status>OK</Status>
    <ErrorDetails></ErrorDetails>
  </StatusInfo>
</PackagesPickupCallResponseV3>
```

Method call

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallXV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsXV3>

PERQRFBpY2t1cENhbGxQYXJhbXNWMz48T3BlcmFOaW9uVHlwZT5JTlNFUlQ8L09wZXJhdGlvblR5cGU+PE9yZG
VyVHlwZT5ET01FU1RJQzwvT3JkZXJUeXBlPjxQaWNrdXBEYXRlPjIwMTgtMDEtMDE8L1BpY2t1cERhdGU…
      (data cut due to length)
    </dpdPickupParamsXV3>
    <authDataV1>
      <login>userlogin</login>
      <masterFid>1495</masterFid>
      <password>xxxxxxxxx</password>
    </authDataV1>
    </ns2:packagesPickupCallXV4>
  </S:Body>
</S:Envelope>
```

Where the input data encoded in base64 are as follows:

INSERT

```
<DPDPickupCallParamsV3>
  <OperationType>INSERT</OperationType>
  <OrderType>DOMESTIC</OrderType>
  <PickupDate>2018-01-01</PickupDate>
  <PickupTimeFrom>01:00</PickupTimeFrom>
  <PickupTimeTo>20:00</PickupTimeTo>
  <PickupCallSimplifiedDetails>
    ...(no structure changes to V2)...
  </PickupCallSimplifiedDetails>
  <WaybillsReady>true</WaybillsReady>
</DPDPickupCallParamsV3>
```

UPDATE

```
<DPDPickupCallParamsV3>
  <OperationType>UPDATE</OperationType>
  <UpdateMode>DONT_CREATE_NEW_IF_CLOSED</UpdateMode>
  <OrderType>DOMESTIC</OrderType>
  <OrderNumber>201712199</OrderNumber>
  <CheckSum>96146323</CheckSum>
  <PickupDate>2018-01-01</PickupDate>
  <PickupTimeFrom>01:00</PickupTimeFrom>
  <PickupTimeTo>20:00</PickupTimeTo>
  <PickupCallSimplifiedDetails>
    ...(brak zmian w strukturze względem V2)...
  </PickupCallSimplifiedDetails>
  <WaybillsReady>true</WaybillsReady>
</DPDPickupCallParamsV3>
```

CANCEL

```
<DPDPickupCallParamsV3>
  <OperationType>CANCEL</OperationType>
  <OrderNumber>201712199</OrderNumber>
  <CheckSum>96146323</CheckSum>
</DPDPickupCallParamsV3>
```

Method name	packagesPickupCallXV4
Signature	byte[] packagesPickupCallXV4(byte[] dpdPickupParamsXV3, AuthDataV1 authDataV1)
Output	byte[]

Parameters:

dpdPickupParamsXV3(byte[]) – preferred pickup date and time interval (parameter encoded in base64)

No changes to the input parameter structure in comparison to method version XV3.

Validation

No changes to validation rules in comparison to version XV3.

Query results

XML response structure for the correctly generated courier booking in version XV4:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:packagesPickupCallXV4Response xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <return>
        ...( no changes to XV3)...
```

```
      </return>
    </ns2:packagesPickupCallXV4Response>
  </S:Body>
</S:Envelope>
```

No changes to the returned result structure in comparison to version XV3.

Method call

No changes to method call in comparison to version XV3.

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:packagesPickupCallXV4 xmlns:ns2="http://dpdservices.dpd.com.pl/">
      <dpdPickupParamsXV3>
        ...(no changes to XV3)...
      </dpdPickupParamsXV3>
      <authDataV1>
        <login>userlogin</login>
        <masterFid>1495</masterFid>
        <password>xxxxxxxxx</password>
      </authDataV1>
    </ns2:packagesPickupCallXV4>
  </S:Body>
</S:Envelope>
```

1.2.5 DPD services

Within the waybill generation it is possible to add services which modify various aspects in connection with parcel delivery. The list will vary depending on the method used and the delivery type (domestic or international). Some of the services require specific conditions which must occur in order for them to happen. Validation of specific services can also exclude one another.

1.2.5.1 Examples of services

The services are used as a fragment of different versions of the Open UMLF parameter during waybill generation.

Regardless of the method version, the services are always located in the service section of the query.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:dpd="http://dpdservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <dpd:generatePackagesNumbersV4>
      <openUMLFeV3>
        <packages>
          <parcels>
            <sizeX>4</sizeX>
            <sizeY>10</sizeY>
            <sizeZ>10</sizeZ>
            <weight>30</weight>
          </parcels>
          <payerType>SENDER</payerType>
          <receiver>
            <countryCode>PL</countryCode>
```

```
            <email>jan.kowalski@gmail.com</email>
            <name>Jan Kowalski</name>
            <phone>111111111</phone>
            <address>Szeroka 3</address>
            <city>Toruń</city>
            <company/>
            <postalCode>87100</postalCode>
          </receiver>
          <ref1/>
          <ref2/>
          <ref3/>
          <reference/>
          <sender>
            <address>Mineralna 15</address>
            <city>Warszawa</city>
            <company>TestCom</company>
            <countryCode>PL</countryCode>
            <email>testCom@test.pl</email>
            <fid>1495</fid>
            <name>ZbigNow Nowak</name>
            <phone>2222222222</phone>
            <postalCode>02743</postalCode>
          </sender>
          <services>
            <cod>
              <amount>10</amount>
              <currency>PLN</currency>
            </cod>
            <guarantee>
              <type>TIMEFIXED</type>
              <value>10:45</value>
            </guarantee>
          </services>
        </packages>
      </openUMLFeV3>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <langCode>PL</langCode>
      <authDataV1>
        <login>XXXXXXX</login>
        <masterFid>0001</masterFid>
        <password>XXXXXXX</password>
      </authDataV1>
    </dpd:generatePackagesNumbersV4>
  </soapenv:Body>
</soapenv:Envelope>
```

1.2.5.2 Domestic services

List of domestic services

Name of the domestic service	Code of the domestic service
guarantee	Guaranteed parcel delivery for specified conditions.
declaredValue	Declared parcel value – additional cover.
cod	Cash on delivery.
cud	Return parcel.
rod	Return documents.
inPers	Delivery to a specific person.
selfCol	Collection from a DPD depot.
privPers	Delivery to a private address.
carryIn	Carry in service.
Palette	Pallet.
Dox	Documents.
dedicatedDelivery	Dedicated delivery.
Tires	Tyres.
dpdPickup	Delivery to a DPD Pickup point.

Services for generatePackagesNumbersV1 in domestic service

ServicesOpenUMLFeV2 type is responsible for the set of services available in version V1 of the method responsible for waybill generation. It consists of the following elements:

Guarantee(type: ServiceGuaranteeOpenUMLFeV1)

Description

Guarantee service is responsible for guaranteed delivery in accordance with specified rules and differs depending on the selected service subtype.

Parameters

The service has two parameters:

Parameter name	Parameter type	Parameter description
Type	ServiceGuaranteeTypeEnumOpenUMLFeV1	Guarantee service type. Exists in the form of various glossary values determining the specifics of this service. See description below.
Value	String	Parameter used for some types of the service. Specifies the type parameter and can assume different values depending on the selected Type glossary value.

Glossary of Guarantee service type:

The glossary value column contains the glossary value code used for defining the specific version of guarantee service. In the Value of the Value Parameter column it is described how the value parameter for Guarantee service must be filled out for the associated glossary value.

Glossary value	Description of the glossary value	Value of the Value parameter
TIME0930	Guarantee of attempt of delivery before 9:30, on the first working day following the dispatch day. In the period from 1st November until the last day of February, the delivery time is moved to 10:30.	Not used
TIME1200	Guarantee of attempt of delivery before 12:00, on the first working day following the dispatch day.	Not used
B2C	Premium client - for certain post codes it is possible to select precise guaranteed delivery time interval.	Defining the delivery time interval is required. Available intervals: 12:00-15:00, 15:00-18:00, 18:00-21:00
TIMEFIXED	Guarantee of attempt of delivery at the time specified by the sender(in the Value parameter below), on the first working day following the dispatch day. The time specified in the Value parameter will be considered to be the delivery time (+/- 20 minutes). If the selected timeframe is from 10:30 – 12:00, the cost of the additional service is increased by the cost of delivery before 12:00. (TIME1200).	Defining the specific time of delivery attempt is required. The allowed time format is HH:MM (hours and minutes). The time provided should be between 10:30 - 16:00.
SATURDAY	Guarantee of attempt of delivery on Saturday between 8:00 -17:00.	Not used
DPDNEXTDAY	Guarantee of attempt of delivery on the next working day following the dispatch.	Not used

Validation

Guarantee service cannot be used in conjunction with any other guarantee service. It is not possible to use guarantee services in conjunction with tires, dedicatedDelivery or dpdPickup (collection from a DPD Pickup point) services.

Example of the use

An example of Guarantee service premium client call for generatePackagesNumbersV4 method

```
<services>
  <guarantee>
    <type>B2C</type>
```

```
    <value>12:00-15:00</value>
  </guarantee>
</services>
```

Description

Declared value of the parcel. An additional cover is automatically added for value declarations over PLN 1 000.00 PLN.

Parameters

The service has two parameters:

Parameter name	Parameter type	Parameter description
Amount	String	Declared value for a currency defined in the Currency parameter.
Currency	ServiceCurrencyEnum	Currency in which the declared value was specified. The currency assumes glossary values.

Currency glossary for ServiceCurrencyEnum type:

Glossary value	Description
PLN	Polish Zloty
EUR	Euro
USD	US Dollar
CHF	Swiss Frank
SEK	Swedish Krona
NOK	Norwegian Krone
CZK	Czech Koruna
RON	Romanian Leu
HUF	Hungarian Forint
HRK	Croatian Kuna
BGN	Bulgarian Lew
DKK	Danish Krone
GBP	British Pound
RSD	Serbian Dinar
RUB	Russian Rubel
TRY	Turkish Lira

UNKNOWN	Currency unknown

Validation

It is not possible to combine the additional cover with dedicatedDelivery service. The Amount parameter assumes numeric value from 0 – 500 000.00 PLN. Currency for domestic service must be set at PLN value.

Example of the use

An example of use of the service in the structure of generatePackagesNumbersV4 method.

```
<services>
  <declaredValue>
    <amount>10</amount>
    <currency>PLN</currency>
  </declaredValue>
</services>
```

Cod (type: ServiceCODOpenUMLFeV1)

Description

Cash on delivery. Once the service is added the recipient is obliged to pay for the parcel in accordance with the amount declared in service parameters. The amount collected by the courier is delivered to the Client. The maximum amount collectable by the courier is the PLN equivalent of EUR 3000.00 calculated on the basis of the mean exchange rate published by the National Bank of Poland on the last working day before the day of dispatch. If the amount exceeds 1000.00 PLN it is necessary to chargé an additional insurance. The insurance amount is determined on the basis of the current standard price list.

Parameters

The service has two parameters:

Parameter name	Parameter type	Parameter description
Amount	String	The amount to be paid by the recipient to the courier upon delivery.
Currency	ServiceCurrencyEnum	Currency in which the amount to be paid is entered. Glossary value is similar to declared value service (the same type to determine the currency)

Validation

For domestic services the currency used in the Currency parameter must be PLN. The amount must be between PLN 0.01-15000. COD service is available for the payer/recipient who is not blocked for that service. It is not possible to select this service in conjunction with the dedicatedDelivery service. It is possible to use the dpdPickup service (delivery to a DPD Pickup point) but it is only available for selected Pickup points.

```
<services>
  <cod>
    <amount>10</amount>
    <currency>PLN</currency>
  </cod>
</services>
```

Cud (type: ServiceCUDOpenUMLeFV1)

Description

Return parcel service. At the time of releasing the parcel to the recipient, the DPD collects another parcel from the recipient, addressed to the sender of the primary parcel. The service is performed within 2 working days from the day of delivery.

Validation

It is not possible to use this service in conjunction with the dedicatedDelivery service or the dpdPickup service if the selected DPD Pickup point does not provide such option.

Example of the use

```
<services>
  <cud/>
</services>
```

Rod (type: ServiceRODOpenUMLFeV1)

Description

Return documents service. Similar to return parcel, the delivered parcel is returned to the sender once the recipient's signature has been obtained. Returns are carried out in weekly cycles.

Validation

It is not possible to use this service in conjunction with the dedicatedDelivery service or the dpdPickup service if the selected DPD Pickup point does not provide such option.

Example of the use

```
<services>
  <rod/>
</services>
```

inPers (type: ServiceInPersOpenUMLFeV1)

Description

Delivery to a specific person provided on the delivery confirmation. The recipient's identity must be verified by an ID document (national ID card, passport or driving licence)

Validation

It is not possible to use this service in conjunction with the dedicatedDelivery service or the dpdPickup service. First name and surname of the recipient is required.

Example of the use

```
<services>
  <inPers/>
</services>
```

selfCol (type: ServiceSelfColOpenUMLFeV1)

Description

Personal collection of the parcel from a depot. The recipients identity must be verified by an ID (a private person) or accompany stamp (a non-private person).

Parameters

Personal collection service has a recipient type parameter:

Parameter name	Parameter type	Parameter description
Receiver	ServiceSelfColReceiverTypeEnumOpenUMLFeV1	Type of recipient

Glossary values of ServiceSelfColReceiverTypeEnumOpenUMLFeV1 type:

Glossary value	Value description
PRIV	Private recipient
COMP	Company recipient

Validation

It is not possible to use this service in conjunction with the dedicatedDelivery service, carryIn or the dpdPickup service. Receiver parameter (i.e. recipient type) is required.

Example of the use

```
<services>
  <selfCol>
    <receiver>PRIV</receiver>
  </selfCol>
</services>
```

privPers (type: ServicePrivPersOpenUMLFeV1)

Description

Delivery to a private person.

Validation

It is not possible to use this service in conjunction with the dedicatedDelivery service or the dpdPickup service. First name and surname of the recipient is required.

Example of the use

```
<services>
  <privPers/>
</services>
```

carryIn (type: ServiceCarryInOpenUMLFeV1)

Description

Carry in service. Available only to selected clients who signed the agreement before 1st July 2013. The service is not developed anymore.

Validation

The service is available only for selected clients (selected client numbers). It is only for parcels consisting of one package only. The actual weight of the parcel cannot exceed 31.5 – 150 kg. Volumetric value of such parcel cannot exceed 312 kg. This service is available only for selected recipient post codes. It is not possible to use this service in conjunction with dedicatedDelivery or selfCol (personal collection).

Example of the use

```
<services>
  <carryIn/>
</services>
```

palette (type: ServicePalletOpenUMLFeV1)

Description

Service for sending goods on pallets.

Validation

Maximum allowed parcel weight is 700 kg. Maximum parcel height is 180 cm. Maximum parcel length is 120 cm. Maximum parcel width is 80 cm. Dimensions provided (order of x, y, z must be maintained) must be a positive integer.

Example of the use

```
<services>
  <palette/>
</services>
```

Dox (type: ServicePalletOpenUMLFeV1)

Description

Service for handling documents.

Validation

Maximum weight of a parcel is 0.5 kg. The service cannot be used in conjunction with tires and dedicatedDelivery services.

Example of the use

```
<services>
  <dox/>
</services>
```

dedicatedDelivery (type: ServiceDedicatedDeliveryOpenUMLFeV1)

Description

Dedicated delivery. Service available only for selected clients. Once it is selected and the client number is verified the parcel is flagged with an appropriate code.

Validation

It is not possible to use in conjunction with cod, guarantee, inPers, colSelf, dox, carryIn, tires , dpdPickup services. This service is available only for selected clients.

Example of the use

```
<services>
  <dedicatedDelivery/>
</services>
```

Tires (type: ServiceTiresOpenUMLFeV1)

Description

An additional service for transporting tyres as a parcel. An additional fee is charged if the tyres were not specified in the dispatch confirmation. At the moment the amount is 0 PLN in the domestic service.

Validation

This service cannot be used in conjunction with guarantee, dox, dpdPickup, dedicatedDelivery services.

Example of the use

```
<services>
  <tires/>
</services>
```

Services for generatePackagesNumbersV2 in domestic service

As in generatePackagesNumbersV1 service, the method uses ServicesOpenUMLFeV2 type to specify the services available to the users. All provisions regarding services available and described for the generatePackagesNumbersV1 method shall apply.

Services for generatePackagesNumbersV3 in domestic service

Together with the change of input parameter model the data type responsible for services in the generatePackagesNumbersV3 method also changes. Services are described by ServicesOpenUMLFeV3 type. One service has been added to it (in comparison to the previous version):

dpdPickup (type: ServiceDpdPickupOpenUMLFeV1)

Description

Collection of a parcel from a Pickup point.

Parameters

This service has one parameter:

Parameter name	Parameter type	Parameter description
Pudo	String	Pickup point code

Validation

The service is available only to private persons. Recipient's first name, surname and phone number or e-mail address is required. Maximum parcel weight is 20 kg, the longest side cannot exceed 100 cm. Furthermore the sum of base circumference and height cannot exceed 250 cm. It is not possible to use the service in conjunction with guarantee, inPers, selfCol, carryIn, dedicatedDelivery, cod, dox, tires services. This service is available only to selected clients.

Example of the use

```
<services>
  <dpdPickup>
    <pudo>PL00001</pudo>
  </dpdPickup>
</services>
```

Services for generatePackagesNumbersV4 in domestic service

This method operates on an extended input parameter model (including services) and ServicesOpenUMLFeV4 is the type for handling the services. There is no change to domestic services in comparison to version V3.

1.2.5.3 International services

List of international services

International service name	International service code
guarantee	Guaranteed delivery on the next working day.
declaredValue	Declared value – additional insurance cover.
cod	Cash on delivery.
privPers	Delivery to a private person.
duty	Customs clearance.
palette	Pallet.
tiresExport	Tyres.
dpdExpress	International air freight.
documentsInternational	International documents.

Services for generatePackagesNumbersV1 in international service

ServicesOpenUMLFeV2 type is responsible for a set of services available in version V1 of the method used for generating waybills. It consists of the following elements:

Guarantee(type: ServiceGuaranteeOpenUMLFeV1)

Description

Guarantee service is responsible for guaranteed parcel delivery in accordance with strictly set rules and differs in accordance with the selected service subtype.

Parameters

The service has two parameters:

Parameter name	Parameter type	Parameter description
Type	ServiceGuaranteeTypeEnumOpenUMLFeV1	Guarantee type service. There are different glossary values describing its specifics. See description below.
Value	String	Parameter used for some types of this service. It specifies the type parameter and can assume different values depending on selected Type glossary value.

Guarantee type service glossary:

Glossary value column contains the glossary value code used for defining the specific version of Guarantee service. The Value of the Value column describes how to fill in the Guarantee service value parameter for the assigned glossary value.

Glossary value	Description of the glossary value	Value of the Value parameter
INTER	Guaranteed day of delivery. Available only as international parcel.	Not used

Validation

When glossary types for domestic service are used a message of service unavailability for the selected service will be displayed. The service cannot be used in conjunction with palette, tires, privPers services. The service must be available for the recipient country and specific post code.

Example of the use

An example of Guarantee INTER service call for generatePackagesNumbersV4 method

```
<services>
  <guarantee>
    <type>INTER</type>
  </guarantee>
</services>
```

DeclaredValue(type: ServiceDeclaredValueOpenUMLFeV1)

Description

Declared parcel value for additional insurance cover.

Parameters

The service has two parameters:

Parameter name	Parameter type	Parameter description
Amount	String	Declared value amount for the currency defined in the Currency parameter.
Currency	ServiceCurrencyEnum	Currency for which the declared parcel value was provided. The currency assumes glossary values.

Currency value for ServiceCurrencyEnum type:

Glossary value	Description
PLN	Polish Zloty
EUR	Euro
USD	US Dollar
CHF	Swiss Frank
SEK	Swedish Krona
NOK	Norwegian Krone
CZK	Czech Koruna
RON	Romanian Leu
HUF	Hungarian Forint
HRK	Croatian Kuna
BGN	Bulgarian Lew
DKK	Danish Krone
GBP	British Pound
RSD	Serbian Dinar
RUB	Russian Ruble
TRY	Turkish Lira
UNKNOWN	Unknown currency

Validation

Depending on the currency there are maximum limiting values which cannot be exceeded. The parcel value must be a positive number. PLN is the default currency.

Limiting values for assigned currencies:

Currency code	Maximum threshold
PLN	500 000.00
CHF	135 135.00
EUR	108 696.00
NOK	833 333.00
SEK	1 000 000.00
USD	142 857.00

Other currencies specified in the glossary, but without limiting values, are not allowed in this service.

Example of the use

An example of the use of the service in generatePackagesNumbersV4 method structure.

```
<services>
  <declaredValue>
    <amount>10</amount>
    <currency>PLN</currency>
  </declaredValue>
</services>
```

Cod (type: ServiceCODOpenUMLFeV1)

Description

Cash on delivery. Once the service is added the recipient is obliged to pay for the parcel in accordance with the amount declared in service parameters. The amount collected by the courier is delivered to the Client. The maximum amount collectable by the courier is the PLN equivalent of EUR 3000.00 calculated on the basis of the mean exchange rate published by the National Bank of Poland on the last working day before the day of dispatch. If the amount exceeds 1000.00 PLN it is necessary to chargé an additional insurance. The insurance amount is determined on the basis of the current standard price list.

Parameters

The service has two parameters:

Parameter name	Parameter type	Parameter description
Amount	String	Amount to be paid to the client upon receipt of the parcel.
Currency	ServiceCurrencyEnum	Currency in which the amount to be paid is denominated. Glossary value is the same as for declared value service (the same currency determination type)

Validation

Currency selected at payment must be consistent with the selected country. Maximum limiting amount to be paid is:

Currency code	Maximum threshold
SEK	27 000.00
EUR	3 000.00
CZK	75 000.00
CHF	3 000.00
NOK	25 000.00
RON	12 000.00
HUF	860 000.00
HRK	20 800.00
BGN	5 500.00
DKK	21 000.00
GBP	2 400.00
RSD	350 000.00
RUB	175 000.00
TRY	11 000.00

Some currencies are available only for selected clients. The service does not apply to some of the countries. The service excludes dox service.

Example of the use

```
<services>
 <cod>
   <amount>10</amount>
  <currency>PLN</currency>
 </cod>
</services>
```

privPers (type: ServicePrivPersOpenUMLFeV1)

Description

Delivery to an individual.

Example of the use

```
<services>
  <privPers/>
</services>
```

duty (type: ServiceDutyOpenUMLeFV1)

Description

Customs clearance for parcels dispatched outside of the EU common customs zone.

Validation

Service available for selected countries and post codes.

Example of the use

```
<services>
  <duty/>
</services>
```

pallet (type: ServicePalletOpenUMLFeV1)

Description

Service enables sending goods on pallets.

Validation

Maximum allowed parcel weight is 700 kg. Maximum parcel height is 175 cm. Maximum parcel length is 175 cm. Maximum parcel width is 175 cm. Dimensions provided (order of x, y, z must be maintained) must be a positive integer. The sum of base circumference and height cannot exceed 300 cm.

Example of the use

```
<services>
  <palette/>
</services>
```

tiresExport (type: ServiceTiresExportOpenUMLFeV1)

Description

Tyres delivery as export service.

Validation

The service cannot be used in conjunction with guarantee, palette, privPers services.

Example of the use

```
<services>
  <tiresExport/>
</services>
```

Services for generatePackagesNumbersV2 in international service

Similar to generatePackagesNumbersV1 service this method uses ServicesOpenUMLFeV2 type to specify the services available to the users. All provisions regarding services available and described for the generatePackagesNumbersV1 shall apply.

Services for generatePackagesNumbersV3 in international service

Together with the change of input parameter model the data type responsible for services in the generatePackagesNumbersV3 method also changes. Services are described by ServicesOpenUMLFeV3 type. One service has been added to it (in comparison to the previous version):

dpdPickup (type: ServiceDpdPickupOpenUMLFeV1)

Description

Service of parcel collection from a pickup point.

Parameters

The service has one parameter:

Parameter name	Parameter type	Parameter description
Pudo	String	Pickup point code

Validation

The service is not available for some countries. The recipient must be a private person. The longest dimension is 100 cm. The sum of base circumference and height cannot exceed 250 cm. Recipient's first name, surname and phone number or e-mail address is required. Maximum parcel weight is 20 kg. It is not possible to use the service in conjunction with guarantee, tires, palette, privPers services.

Example of the use

```
<services>
  <dpdPickup>
    <pudo>PL00001</pudo>
  </dpdPickup>
</services>
```

Services for generatePackagesNumbersV4 in international service

This method operates on an extended input parameter model (including services) and ServicesOpenUMLFeV4 is the type for handling the services. In comparison to version V3, the following has been added or modified:

duty (type: ServiceDutyOpenUMLeFV2)

Description

Customs clearance for parcels dispatched outside of the EU common customs zone.

Parameters

The previous method has been extended by the following parameters:

Parameter name	Parameter type	Parameter description

Amount	String	Declared amount for customs clearance.
Currency	ServiceCurrencyEnum	Declared currency for customs clearance. The currency parameter is described in detail in declaredValue service and it assumes the same values.

Validation

Service available for selected countries and post codes. The amount must be a positive non-zero value. The currency should correspond to the dispatch country. Customs clearance is not available for freight within the EU.

Example of the use

```
<services>
  <duty>
    <amount>10</amount>
    <currency>PLN</currency>
  </duty>
</services>
```

dpdExpress (type: ServiceFlagOpenUMLF)

Description

International air freight service.

Validation

Maximum actual weight cannot exceed 70 kg. The parcel can consist of maximum 10 packages (not containing documents). The parcel may contain no more than 1 document. Maximum weight of a package in the parcel is 31.5 kg. Maximum length of a side of the parcel is 175 cm. The sum of circumference and height cannot exceed 300 cm.

Example of the use

```
<services>
  <dpdExpress/>
</services>
```

documentsInternational (type: ServiceFlagOpenUMLF)

Description

Delivery of documents outside of the sender’s country.

Validation

Maximum parcel weight for international documents is 31.5 kg. Selecting documents excludes a multi-package parcel. It is not possible to use the duty service (customs clearance) in conjunction with international documents.

Example of the use

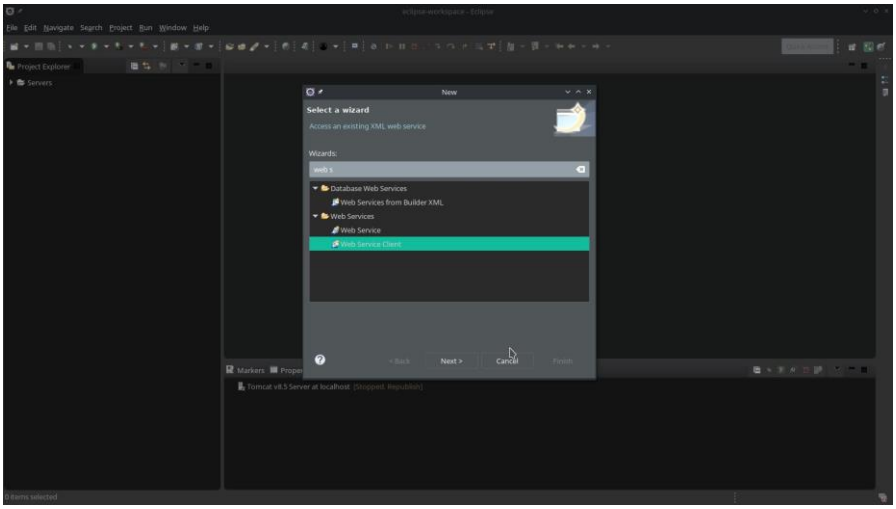
```
<services>
  <documentsInternational/>
</services>
```

1.3 DPDServices service in C#, PHP, JAVA

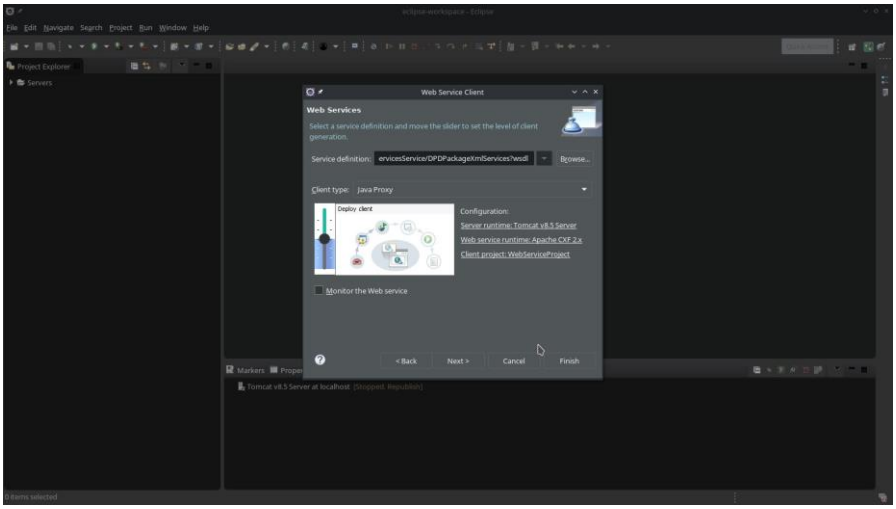
Example of the use of DPDServices in Java

Generating with Eclipse

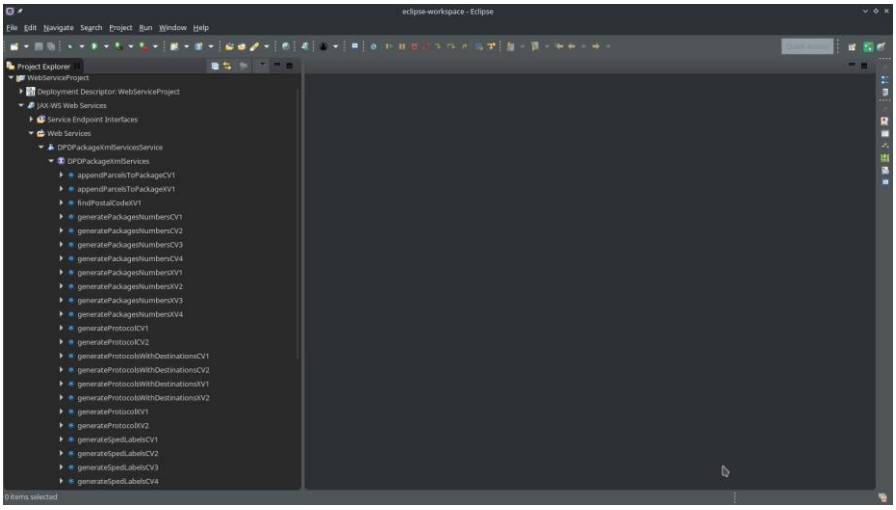
Setting up the new Web Service Client project:



Then paste the wsdl address in the Service Definition field. Select server in the configuration (Tomcat 8.5 in this particular example) and CXF2 as the environment to generate, then click Next:

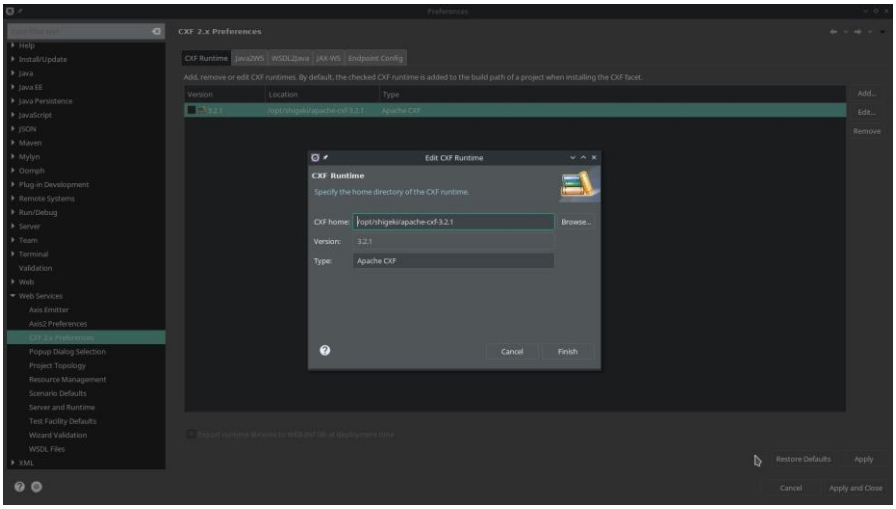


Setup output directory and click finish. After the generation process is completed it is possible to view the generated codes:



Setting CXF in eclipse

Enter the Preference menu, select CXF and add the entire folder where the downloaded CXF is located:



Generating with wsimport tool.

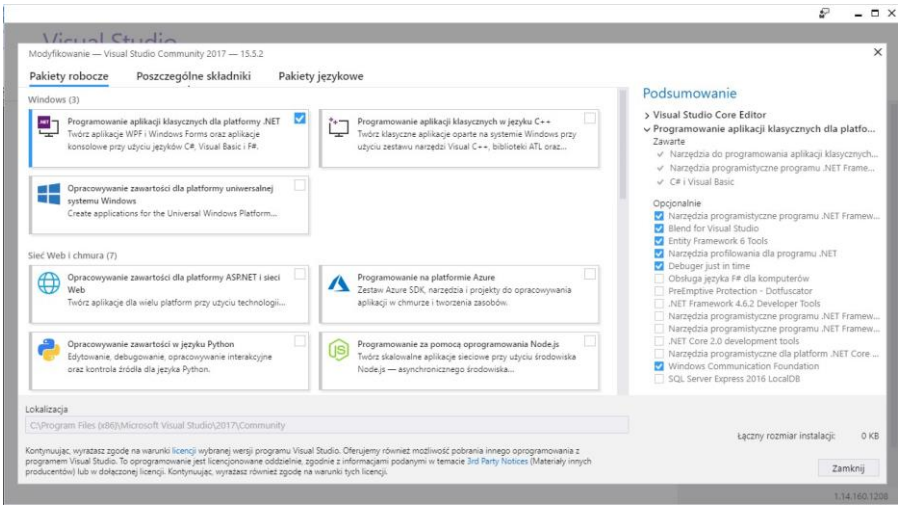
Generating codes with that tool is based on enabling the following command:

```
wsimport -keep -verbose
https://dpdservicesdemo.dpd.com.pl/DPDPackageXmlServicesService/DPDPackageXmlServices?wsdl
```

Example of the use DPDServices w C#

WCF (Windows Communication Foundation) needs to be installed to use SOAP. The WCF installation screen for Visual Studio 2017 is shown below:

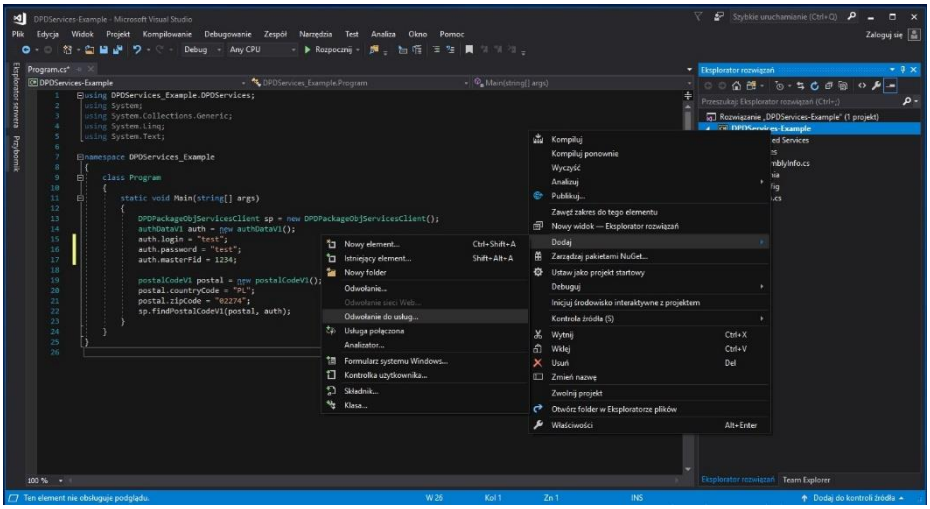
Once the Visual Studio Installer is open, please select Modify under the installed version of Visual Studio.



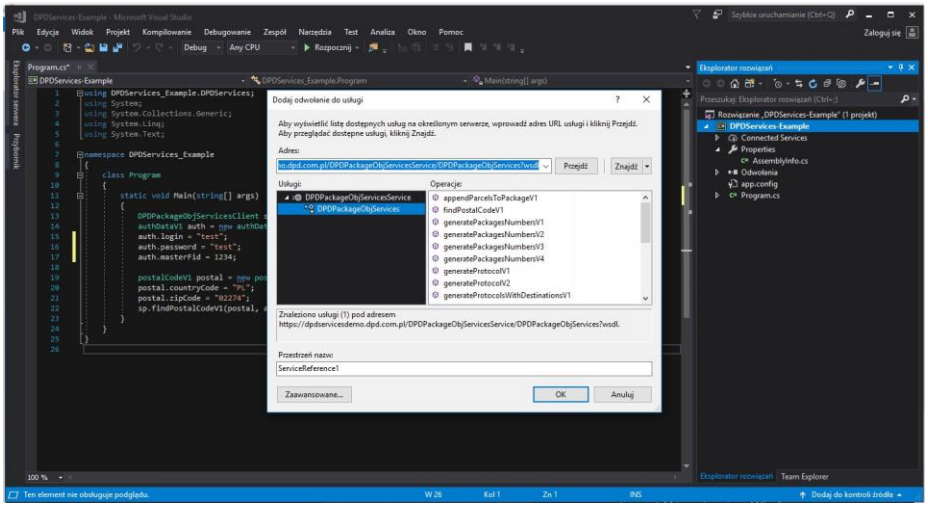
Then on the right hand side in Summary please select Windows Communication Foundation and click Modify button.

Generating

To add a web service to the project, please click the project name, then select add → services reference:



Then we provide the address for the wsdl file and the name of the service:



1.3.1 Implementation examples

Implementation examples are attached to this documentation

1.4 Good practice

The way of transmitting sender, recipient and payer details

Our experience shows that sender, recipient and payer details are best transmitted in the following way:

- sender and recipient – data given directly, without the FID number
- payer – 'third party' as a payer, even if the sender pays for the service

```
<PayerType>THIRD_PARTY</PayerType>
<ThirdPartyFID>xxxxxxxxx</ThirdPartyFID>
    <Sender>
        <CountryCode>PL</CountryCode>
        <PostalCode>54436</PostalCode>
        <Company>CompanyName</Company>
        <Name>Adam Nowak</Name>
        <Address>Mokotowska 2</Address>
        <City>Warszawa</City>
        <Phone>+48126237122</Phone>
        <Email>aaa@cbbb.com.pl</Email>
    </Sender>
    <Receiver>
        <CountryCode>PL</CountryCode>
        <Company>CompanyName</Company>
        <Name>Adam Kowalski</Name>
        <Address>Prosta 17</Address>
        <City>Szczecin</City>
        <Phone>+22121110600</Phone>
        <Email>eee.xxx@cccc.xx</Email>
    </Receiver>
```

Reference values used

In the method call returning the parcel number, the reference field can be used only once, e.g. <Reference>F-VAT 1234</Reference>. It means that in the event of a mistake in e.g. address details, the 'F-VAT 1234' value will not be accepted in the repeated call.

1.5 Appendix 1. OpenUMLF description

This appendix describes the OpenUMLF format in versions V1 – V3.

Structure

Package – package structure, containing sender and recipient details as well as the service and parcel list;

```
<Packages>
        <Package>
        ...
        </Package>
</Packages>
```

Reference – a unique key enabling identification of the parcel, e.g. id number from the client's warehouse system;

Receiver – recipient data;

```
<Receiver>
        <FID>Recipient client number</FID>
        <Company>Company name</Company>
        <Name>Recipient name</Name>
        <Address>Recipient address</Address>
        <City>Recipient city</City>
        <CountryCode>Country code</CountryCode>
        <PostalCode>Recipient post code</PostalCode>
        <Phone>Phone number</Phone>
        <Email>E-mail address</Email>
</Receiver>
```

Sender – sender data;

```
<Sender>
        <FID>Sender client number</FID>
        <Company>Company name</Company>
        <Name>Sender name</Name>
```

<Address>Sender address</Address>

<City>Sender city</City>

<CountryCode>Country code</CountryCode>

<PostalCode>Sender post code</PostalCode>

<Phone>Phone number</Phone>

<Email>E-mail address</Email>

</Sender>

PayerType – payer – the party who pays DPD the cost of parcel handling (payer types in Glossary Types section);

ThirdPartyFID – third party client number – available only in OpenUMLFV1;

Ref1 – Reference field

Ref2 – Reference field

Ref3 – Reference field

Services – the list of services for the parcel, description is provided in the Services section.

Parcels – the list of parcels, each parcel contains the following fields:

- Reference – unique key enabling identification of the parcel;
- Weight – parcel weight
- SizeX – dimension X in cm
- SizeY– dimension Y in cm
- SizeZ– dimension Z in cm
- Content – Description of parcel content
- CustomerData1 – Notes for delivery
- CustomerData2 - User data
- CustomerData3 - User data
- Guarantee – Guaranteed delivery of a parcel to the recipient at the specified time:

Services

◦ DPD 9:30 – express delivery by 9.30 Monday to Friday. From 1st November until 1st March the service is guaranteed at 10.30.

◦ DPD 12:00 express delivery by 12.00 Monday to Friday.

◦ DPD at the hour – delivery on the next working day at the specified time (+/- 20 minutes) within the time intervals: 10:30 – 16:00, 18:00 – 8:00.

Cost of the service as per the price list;

Available attribute types in Glossary Types section, for TIMEFIXED and B2C sections there is an additional field Attr1 for providing the expected delivery time in the hh:mm format (TIMEFIXED) or the time interval in the hh:mm-hh:mm(B2C) format;

- Guaranteed 9.30 delivery (but also TIME1200, SATURDAY):

<Guarantee type="TIME9300"/>

  - Guaranteed delivery within the specified time interval:

<Guarantee type="B2C">

<Attr1>12:00-18:00</Attr1>

</Guarantee>

  - Guaranteed delivery at the agreed time:

<Guarantee type="TIMEFIXED">

- <Attr1>15:45</Attr1>

</Guarantee>

- DeclaredValue – declared value of the parcel, contains the obligatory fields: Amount – amount collected, field format 0.00 and Currency – currency specification. If the currency is not specified the system assumes the default value which is PLN. Description of available currencies in the Glossary Types section;
- COD – cash on delivery, contains the obligatory fields: Amount – amount collected, field format 0.00 and Currency – currency specification. If the currency is not specified the system assumes the default value which is PLN. Description of available currencies in the Glossary Types section;
- CUD – at the time of releasing the parcel to the recipient, the DPD collects another parcel from the recipient, addressed to the sender of the primary parcel;
- ROD – the recipient's signature on the document attached to the parcel is obtained and such document is then returned to the sender;
- InPers – the parcel must be collected by the person named on the label. The courier verifies the recipient's identity on the basis of an ID, e.g. national identity card;
- SelfCol – personal collection from a DPD Depot, description of available values in the Glossary Types section;
- PrivPers – delivery to a private address;
- CarryIn – the courier carries the parcels into the place shown by the client;
- Duty – customs clearance – Note! Amount and Currency fields available only in OpenUMLFV3;
- Pallet – international service delivered on a pallet (for domestic pallet parcels it is enough to provide its actual dimensions and weight);
- DOX – an envelope up to 0,5 kg;
- DedicatedDelivery – dedicated delivery. Available only for selected clients;
- Tires - parcel containing tyres. Only for domestic parcels;
- TiresExport – parcel containing tyres. Only for international parcels;
- DpdPickup – delivery of the parcel to a DPD Pickup point, pickup point ID comes from the DPD pickup points database – please note that the service is available in OpenUMLFV2

Field size

Pole	Description	Type/maximum size	Example of a value	Default value
Packages	List of parcels			
Package	Parcel details			
Reference	A unique key for identification of the parcel	Text(150)	RE23433	Empty
Receiver	Recipient data section			
FID	Recipient client number	Number	123123	
Company	Recipient company name	Text(100)	Przykładowa sp. z o.o.	
Name	Recipient name	Text(100)	Jan Kowalski	
Address	Recipient address	Text(100)	Ul. Cicha 5 m 8	
City	Recipient city	Text(50)	Warszawa	
CountryCode	Recipient country code	Text(2)	PL	PL
PostalCode	Recipient post code	Text(10)	02878	
Phone	Recipient phone number	Text(100)	+48 22213143	
Email	Recipient e-mail	Text(100)	abc@poczta.pl	
Sender	Sender data section			
FID	Sender client number	Number	1222	
Company	Sender company name	Text(100)	Przykładowa sp. z o.o.	
Name	Sender name	Text(100)	Jan Kowalski	
Address	Sender address	Text(100)	Ul. Cicha 5 m 8	
City	Sender city	Text(50)	Warszawa	
CountryCode	Sender country code	Text(2)	PL	PL
PostalCode	Sender post code	Text(10)	40122	
Phone	Sender phone number	Text(100)	+48 22213143	
Email	Sender e-mail	Text(100)	abc@poczta.pl	
ThirdPartyFID	Third party client number	Number	12345	
Ref1	Reference field	Text(100)	12456790	Empty
Ref2	Reference field	Text(100)	FV001	Empty
Ref3	Reference field	Text(100)	documentation	Empty
Services	Additional services section	Details in chapter on openUMLF services		
Parcels	Number of packages in a parcel			
Parcel	Parcel data section			

Reference	A unique key for identification of the parcel	Text(150)	RE23433	Empty
Weight	Parcel weight in kg	Number	12,50	0
SizeX	Dimension X in cm	Number	110	0
SizeY	Dimension Y in cm	Number	100	0
SizeZ	Dimension Z in cm	Number	55	0
Content	Parcel content	Text(300)	modems	Empty
CustomerData1	User details	Text(200)	abc	Empty
CustomerData2	User details	Text(200)		Empty
CustomerData3	User details	Text(200)		Empty

Glossary types

a) PayerType – available payer types:

- RECEIVER – paid by the recipient;
- SENDER – paid by the sender;
- THIRD_PARTY – paid by a third party who is neither the recipient nor the sender;

b) Guarantee – available types:

- TIME0930 – Delivery by 9.30;
- TIME1200 – Delivery by 12.00;
- B2C – Premium client – for certain post codes it is possible to select precise guaranteed delivery time interval.
- TIMEFIXED – Delivery at the specific time hh:mm;
- SATURDAY,- Delivery on Saturday;
- INTER - guaranteed day of delivery;
- DPDNEXTDAY – guaranteed delivery attempt on the next working day following dispatch;

c) Currency – available currencies:

Glossary value	Description

PLN	Polish Zloty
EUR	Euro
USD	US Dollar
CHF	Swiss Frank
SEK	Swedish Krona
NOK	Norwegian Krone
CZK	Czech Koruna
RON	Romanian Leu
HUF	Hungarian Forint
HRK	Croatian Kuna
BGN	Bulgarian Lew
DKK	Danish Krone
GBP	Pound Sterling
RSD	Serbian Dinar
RUB	Russian Rubel
TRY	Turkish Lira
UNKNOWN	Unknown currency

d) SelfCol – recipient type:

- PRIV – private person
- COMP – company

Examples

OpenUMLFV1

> Expand

```
<Packages>
<Package>
<Reference>Reference</Reference>
<Receiver>
<FID>1495</FID>
<Company>Company name</Company>
<Name>Recipient name</Name>
<Address>Recipient address</Address>
<City>Recipient city</City>
<CountryCode>Country code</CountryCode>
<PostalCode>Recipient post code</PostalCode>
```

```
<Phone>Phone number</Phone>
<Email>E-mail address</Email>
</Receiver>
<Sender>
<FID>123456</FID>
<Company>Company name</Company>
<Name>Sender name</Name>
<Address>Sender address</Address>
<City>Sender city</City>
<CountryCode>Country code</CountryCode>
<PostalCode>Sender post code</PostalCode>
<Phone>Phone number</Phone>
<Email>E-mail address</Email>
</Sender>
<PayerType>RECEIVER</PayerType>
<ThirdPartyFID>123123</ThirdPartyFID>
<Ref1>Reference 1</Ref1>
<Ref2>Reference 2</Ref2>
<Ref3>Reference 3</Ref3>
<Services>
<Guarantee type="TIMEFIXED">
<Attr1>15:45</Attr1>
</Guarantee>
<DeclaredValue>
<Amount>200.00</Amount>
<Currency>PLN</Currency>
</DeclaredValue>
<COD>
<Amount>200.00</Amount>
<Currency>PLN</Currency>
</COD>
<CUD/>
<ROD/>
<InPers/>
<SelfCol receiver="PRIV"/>
<PrivPers/>
<CarryIn/>
<Duty/>
<Pallet/>
<DOX/>
<DedicatedDelivery/>
<Tires/>
<TiresExport/>
</Services>
<Parcels>
<Parcel>
<Reference>Parcel reference</Reference>
<Weight>12.3</Weight>
<SizeX>12</SizeX>
<SizeY>7</SizeY>
<SizeZ>15</SizeZ>
<Content>Parcel content</Content>
<CustomerData1>Additional client data</CustomerData1>
<CustomerData2>Additional client data</CustomerData2>
<CustomerData3>Additional client data</CustomerData3>
</Parcel>
</Parcels>
</Package>
<Packages>
<Package>
<Reference>Reference</Reference>
<Receiver>
```

```
<FID>123456</FID>
<Company>Company name</Company>
<Name>Recipient name</Name>
<Address>Recipient address</Address>
<City>Recipient city</City>
<CountryCode>Country code</CountryCode>
<PostalCode>Recipient post code</PostalCode>
<Phone>Phone number</Phone>
<Email>E-mail address</Email>
</Receiver>
<Sender>
<FID>123456</FID>
<Company>Company name</Company>
<Name>Sender name</Name>
<Address>Sender address</Address>
<City>Sender city</City>
<CountryCode>Country code</CountryCode>
<PostalCode>Sender post code</PostalCode>
<Phone>Phone number</Phone>
<Email>E-mail address</Email>
</Sender>
<PayerType>RECEIVER</PayerType>
<Ref1>Additional reference fields</Ref1>
<Ref2>Additional reference fields </Ref2>
<Ref3>Additional reference fields </Ref3>
<Services>
<Guarantee type="B2C">
<Attr1>12:00-18:00</Attr1>
</Guarantee>
<DeclaredValue>
<Amount>200.00</Amount>
<Currency>PLN</Currency>
</DeclaredValue>
<COD>
<Amount>200.00</Amount>
<Currency>PLN</Currency>
</COD>
<CUD/>
<ROD/>
<InPers/>
<SelfCol receiver="PRIV"/>
<PrivPers/>
<CarryIn/>
<Duty/>
<Pallet/>
<DOX/>
<DedicatedDelivery/>
<Tires/>
<TiresExport/>
<DpdPickup>
<pudo>PL123456</pudo>
</DpdPickup>
</Services>
<Parcels>
<Parcel>
<Reference>Parcel reference</Reference>
<Weight>12.3</Weight>
<SizeX>12</SizeX>
<SizeY>7</SizeY>
<SizeZ>15</SizeZ>
<Content>Parcel content</Content>
<CustomerData1>Additional client data</CustomerData1>
```

```
<CustomerData2>Additional client data </CustomerData2>
<CustomerData3>Additional client data </CustomerData3>
</Parcel>
</Parcels>
</Package>
</Packages>
```

OpenUMLFV3

> Expand

```
<Packages>
<Package>
<Reference>Reference</Reference>
<Receiver>
<FID>123456</FID>
<Company>Company name</Company>
<Name>Recipient name</Name>
<Address>Recipient address</Address>
<City>Recipient city</City>
<CountryCode>Country code</CountryCode>
<PostalCode>Recipient post code</PostalCode>
<Phone>Phone number</Phone>
<Email>E-mail address</Email>
</Receiver>
<Sender>
<FID>123456</FID>
<Company>Company name</Company>
<Name>Sender name</Name>
<Address>Sender address</Address>
<City>Sender city</City>
<CountryCode>Country code</CountryCode>
<PostalCode>Sender post code</PostalCode>
<Phone>Phone number</Phone>
<Email>E-mail address</Email>
</Sender>
<PayerType>RECEIVER</PayerType>
<Ref1>Additional reference field</Ref1>
<Ref2>Additional reference field </Ref2>
<Ref3>Additional reference field </Ref3>
<Services>
<Guarantee type="TIME1200"/>
<DeclaredValue>
<Amount>200.00</Amount>
<Currency>PLN</Currency>
</DeclaredValue>
<COD>
<Amount>550.00</Amount>
<Currency>PLN</Currency>
</COD>
<CUD/>
<ROD/>
<InPers/>
<SelfCol receiver="PRIV"/>
<PrivPers/>
<CarryIn/>
<Duty>
<Amount>123.00</Amount>
<Currency>PLN</Currency>
</Duty>
<Pallet/>
```

```
<DOX/>
<DedicatedDelivery/>
<Tires/>
<TiresExport/>
<DpdPickup>
<pudo>PL12345</pudo>
</DpdPickup>
<dpdExpress/>
<documentsInternational/>
</Services>
<Parcels>
<Parcel>
<Reference>Parcel reference</Reference>
<Weight>12.3</Weight>
<SizeX>12</SizeX>
<SizeY>7</SizeY>
<SizeZ>15</SizeZ>
<Content>Parcel content</Content>
<CustomerData1>Additional client data</CustomerData1>
<CustomerData2>Additional client data</CustomerData2>
<CustomerData3>Additional client data</CustomerData3>
</Parcel>
</Parcels>
</Package>
</Packages>
```