



dpd

Table of contents

1	Introduction	4
1.1	Purpose of the document	4
1.2	Glossary	4
1.3	Service addresses	5
1.4	Safety standard and permissions	5
2	Business description	6
3	Service structure description	6
3.1	Permissions	6
3.2	Description of interface methods	7
3.2.1	DPDCRXmlServices interface	7
3.3	Error codes	38
3.4	Comments and suggestion	38
4	Attachments	39

CLIENT DOCUMENTATION

1 Introduction

1.1 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 Web Serwisu DPD AppServices methods used for transfer of Collection Request (CR) to DPD systems. The description concerns DPD AppServices version 1.10.3.

Collection request CR types:

- CR-IN – CR requests, for which the sender's address (collection of the parcel from the sender) is within the territory of Poland;
- CR-OUT – CR requests, for which the sender's address is outside of the territory of Poland;

Order type is determined by the parameter of the country code provided in the senders parameters.

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 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.

1.2 Glossary

A list of definitions of terms which are not commonly used, but are referred to herein.

Term	Definition
notification	Delivery of information regarding the parcel being handed over

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. The format has a number of versions marked with 'VX' suffix, where 'X' relates to the number of the next version e.g. OpenUMLFV2. The format also has variations of dedicated purposes which are marked with 'e' suffix – external use for DPD client e.g. OpenUMLFeV2 or 'i' – internal use for DPD e.g. OpenUMLFiV2.

1.3 Service addresses

DPDCRXmlServices interface

PROD version:

<https://dpdappservices.dpd.com.pl/DPDCRXmlServicesService/DPDCRXmlServices?wsdl>

DEMO version:

<https://dpdappservicesdemo.dpd.com.pl/DPDCRXmlServicesService/DPDCRXmlServices?wsdl>

Login details to the DEMO version

```
<login>DAS test</login>
<masterFid>1495</masterFid>
<password>123_DAS test_456</password>
```

1.4 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 performed 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.

2 Business description

DPD AppServices is used for recording collection requests CR in DPD systems and for their status updates. CR orders can be generated on the basis of requests from DPD Client systems. Depending on the request type CR-IN or CR-OUT, it is recorded by AppServices in dedicated databases. As the CR order is recorded it is also validated for proper syntax and business context. In response to request for recording the request the system returns the status of record completion as well as information regarding any errors during recording.

3 Service structure description

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
- Validation- concerns the feedback in the event the form which makes the query has been filled in incorrectly
- Method call – examples of ready requests in different parameter configurations
- • Request results – different feedback variants

3.1 Permissions

In their structure, all methods described in this chapter have the `authDataV1` (type `AuthDataV1`) parameter which stores the authentication 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 (open text) assigned to the login;
- `masterFid` (Integer) – a unique client catalogue number provided by the service provider. Also known as `numkat`;

This 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	

3.2 Description of interface methods

Two variations exist for some available DPDAppServices interface methods:

- Containing prefix „X” in the version number, which receive and return data coded in the Base64 form;
- Containing prefix „C” in the version number, which receive and return data compressed by ZIP and then coded in the Base64 form;

3.2.1 DPDCRXmlServices interface

The DPDCRXmlServices interface provides the methods for transferring CR orders from external systems to the DPD system. Two methods - importPackagesXV1 and importPackagesCV1 are made available within the interface. Information regarding CR orders, including information on the parcels covered by the orders, are provided as part of input parameters. Detailed method description is provided further in the document.

3.2.1.1 importPackagesXV1 method

3.2.1.1.1 Signature

Method name	importPackagesXV1
Input signature	byte[] importPackagesXV1(byte[] openUMLFXV2, PkgImportPolicyV1 pkgNumsGenerationPolicyV1, AuthDataV1 authDataV1)
Input parameters	openUMLFXV2 – data (XML coded as Base64, in the form of byte table) regarding parcels in the OpenUMLFV2 format;
	pkgNumsGenerationPolicyV1 – policy of operation in the event of data import errors, calculation type PkgImportPolicyV1;
	authDataV1 – data used for user authorisation, type AuthDataV1;

Output	byte[] – XML in the importPackagesXV1Response format, coded as Base64, in the form of a byte table;
---------------	---

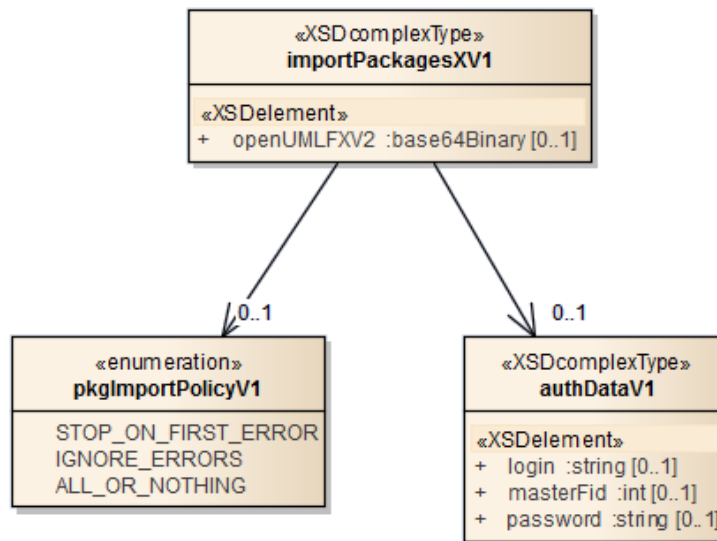
3.2.1.1.2 Parameters and data validation

For importPackagesXV1:

Field name	Type	Validation condition	Mandator y	Additional information
openUMLFXV2	byte[]	No field	Yes	Parameter, which contains data regarding the parcel, selected services and other key information which indicate the mode of processing parcel information. Data is provided in the form of a byte table which contains XML in the OpenUMLFV2 format, coded in Base64. More information regarding the OpenUMLFV2 format is provided in the attachment describing all version of the OpenUMLF format. The above mentioned attachment is entitled OpenUMLF description.

pkgNumsGenerationPolicyV1	PkgNumsGenerationPolicyV1	No validation	No	<p>Glossary parameter describing the service policy regarding the mode of reaction to errors appearing during the execution of the method call. Depending on the selected option the service will be conducted in a different way when errors occur. The possible values are:</p> <p>STOP_ON_FIRST_ERROR – means that the service will stop the operation when the first error occurs for any of the parcels. The import of first parcels will be successful;</p> <p>IGNORE_ERRORS – results in ignoring a general error for the specified record / parcel. This means that the wrong parcel will be ignored, but the other parcels with correct data will be correctly issued by the service;</p> <p>ALL_OR_NOTHING – stops the import of parcels for any error. All parcels must be positively verified for the import to take place. The default value is ALL_OR_NOTHING.</p>
authDataV1	AuthDataV1	No field	Yes	Lack of field results in the message "AuthData is null".

3.2.1.1.3 Query structure



An example of a correct XML query for the `importPackagesXV1` method is provided below. The `openUMLFXV2` element includes an XML in the `OpenUMLFV2` format coded as `Base64`.

Query for the importPackagesXV1 method for DPDCRXmlServices interface:

```
<?xml version='1.0' encoding='UTF-8'?><soapenv:Envelope xmlns:cr="http://schemas.xmlsoap.org/soap/envelope/" xmlns:cr="http://cr.dpdappservices.dpd.com.pl"><soapenv:Header/><soapenv:Body><cr:importPackagesXV1><openUMLFXV2>PFBhY2thZ2VzPg0KCTxQYWNRyWdIPg0KICAgICAglCA8U2VuZGVyPg0KICAgICAglCAglCAglCAgPENvbXBhbmkPC9Db21wYW55Pg0KICAgICAglCAglCAglCAgPE5hbWU+cHJ6ZXN5bGthIHogYXBwlHNlcncZpY2VzPC9OYW1lPg0KICAgICAglCAglCAglCAgPEFkZHJlc3M+VGVyZW5janVzemEgMTI8L0FkZHJlc3M+DQoglCAglCAglCAglCA8Q2I0eT5PbHN6dHluPC9DaXR5Pg0KICAgICAglCAglCAglCAgPENvdW50cmlDb2RlPiBMPC9Db3VudHJ5Q29kZT4NCiAgICAglCAglCAglCAglDxB3N0YWxDb2RlPiEwMTc4PC9Qb3N0YWxDb2RlPg0KICAgICAglCAglCAglCAgPFBob25lPjwwUGhvbmU+DQoglCAglCAglCAglCA8RW1haWw+PC9FbWFpbD4NCiAgICAglCAglCAgPC9TZW5kZXI+DQoglCAglCAglCAglDxSZWNlaXZlcj4NCiAgICAglCAglCAglDxB3N0YW55PjwwQ29tcGFueT4NCiAgICAglCAglCAglCAglDxOYW1lPIRlc3QgUGllY2R6aWVzaWF0eSBUCnplY2k8L05hbWU+DQoglCAglCAglCAglCA8QRWRkcmVzc5DYXJza2EgMzl8L0FkZHJlc3M+DQoglCAglCAglCAglCA8Q2I0eT5XYXJzemF3YTtwvQ2I0eT4NCiAgICAglCAglCAglCAglDxB3VudHJ5Q29kZT5QTDRwvQ29lbnRyeUNvZGU+DQoglCAglCAglCAglCA8UG9zdGFsQ29kZT4wMjI3MjwwUG9zdGFsQ29kZT4NCiAgICAglCAglCAglCAglDxQaG9uZT48L1B0b25lPg0KICAgICAglCAglCAglCAgPEvtYWIsPjwvRW1haWw+DQoglCAglCAglCAglDwvUmVjZWl2ZXI+DQoJCCTxQYXIlciR5cGU+VEhJUkrfUEFSVFk8L1BheWVyVHlwZT4NCgkJPFROaXJkUGFydHIHSUQ+MTQ0NTtwvVGhpcmRQYXJ0eUZJRDR4NCgkJPFJIZjE+MTExMTExPC9SZWYxPg0KCQk8UmVmMj48L1JlZjl+DQoJCCTxDdXN0b21lcj4NCgkJCTxGSUQ+MTQ0NTtwvRkIEPg0KCQk8L0N1c3RvbWVvPg0KCQk8U2VydmljZXI+DQoJCQk8R3VhcmFudGVIIHR5cGU9IIRJTUUwOTMwli8+DQoJCQk8SW5QZXJzLz4NCgkJCTxDYXJjeUlULz4NCgkJPC9TZXJ2aWNlcj4NCgkJPF BhcmNlbHM+DQoJCQk8UGFyY2VsPg0KICAgICAglCAglCAglCAglCAglDxxZWlnaH
```

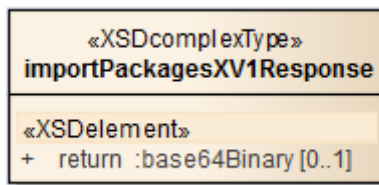


```

<Customer>
  <FID>1495</FID>
</Customer>
<Services>
  <Guarantee type="TIME0930"/>
  <InPers/>
<CarryIn/>
</Services>
<Parcels>
  <Parcel>
    <Weight>35</Weight>
    <SizeX>100</SizeX>
    <SizeY>40</SizeY>
    <SizeZ>40</SizeZ>
    <Content>test</Content>
    <CustomerData1>customer data 1</CustomerData1>
    <CustomerData2>customer data 2</CustomerData2>
    <CustomerData3>customer data 3</CustomerData3>
  </Parcel>
</Parcels>
</Package>
</Packages>

```

3.2.1.1.4 Response structure



An example of a response to the correct XML query of importPackagesXV1 method is provided below.

A response for the correct query of the importPackagesXV1 method for DPDCRXmlServices interface:

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:importPackagesXV1Response xmlns:ns2="http://cr.dpdappservices.dpd.com.pl/">
      <return>
        PFBhY2thZ2VzSW1wb3J0UmVzcG9uc2U+PFN0YXR1c0luZm8+PFN0YXR1cz5PSzwwU3RhdHVzPjwv
        vU3RhdHVzSW5mbz48UGFja2FnZXNM+PFBhY2thZ2U+PFBhY2thZ2VJZD45NzY5MDgzPC9QYWNRyY
        WdlSWQ+PFN0YXR1c0luZm8+PFN0YXR1cz5PSzwwU3RhdHVzPjwvU3RhdHVzSW5mbz48UGFyY2
        Vscz48UGFyY2VsPjxQYXJjZWxJZD45NjkyMTYyPC9QYXJjZWxJZD48V2F5YmIsbD4wMDAwMDA5
        NjA5MzYyVDwvV2F5YmIsbD48L1BhcmNlbD48L1BhcmNlbHM+PE9yZGVyTnVtYmVYPkFQUC9DUk
        IOLzk3NjkwODM8L09yZGVyTnVtYmVYPjwvUGFja2FnZT48L1BhY2thZ2VzPjwvUGFja2FnZXNjbXB
        vcnRSZXNwb25zZT4=
      </return>
    </ns2:importPackagesXV1Response>
  </S:Body>
</S:Envelope>

```

The response type for XML query for importPackagesXV1 method for DPDCRXmlServices interface with wrong user authentication data.

Response to a query with wrong user authentication data:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>S:Server</faultcode>
      <faultstring>Login failed</faultstring>
      <detail>
        <ns2:DPDAppAuthenticationException xmlns:ns2="http://cr.dpdappservices.dpd.com.pl/">
          <login>pmarat</login>
          <message>Login failed</message>
        </ns2:DPDAppAuthenticationException>
        <ns2:exception
class="pl.com.dpd.dpdappservices.exception.DPDAppAuthenticationException" note="To disable
this feature, set com.sun.xml.ws.fault.SOAPFaultBuilder.disableCaptureStackTrace
system property to false" xmlns:ns2="http://jax-ws.dev.java.net/">
          <message>Login failed</message>
          <ns2:stackTrace>
            <!--Zawartość StackTrace-->
          </ns2:stackTrace>
        </ns2:exception>
      </detail>
    </S:Fault>
  </S:Body>
</S:Envelope>
```

The return element of the response for importPackagesXV1 contains an XML in the PackagesImportResponse format coded as Base64. It returns the parcel import status. An example of correctly performed import is provided below.

PackagesImportResponse structure - Status OK:

```
<PackagesImportResponse>
  <StatusInfo>
    <Status>OK</Status>
  </StatusInfo>
  <Packages>
    <Package>
      <PackageId>9769083</PackageId>
      <StatusInfo>
        <Status>OK</Status>
      </StatusInfo>
      <Parcels>
        <Parcel>
          <ParcelId>9692162</ParcelId>
          <Waybill>0000009609362T</Waybill>
        </Parcel>
      </Parcels>
      <OrderNumber>APP/CRIN/9769083</OrderNumber>
    </Package>
  </Packages>
</PackagesImportResponse>
```

Example of PackagesImportResponse for the XML query importPackagesXV1 method for DPDCRXmlServices interface with wrong sender post code.

PackagesImportResponse structure - Status INCORRECT_DATA:

```
<PackagesImportResponse>
  <StatusInfo>
    <Status>INCORRECT_DATA</Status>
  </StatusInfo>
  <Packages>
    <Package>
      <StatusInfo>
        <Status>INCORRECT_DATA</Status>
        <ErrorDetails>
          <Code>INCORRECT_SENDER_POSTAL_CODE</Code>
          <Description>Incorrect sender postal code (10178)</Description>
          <Fields>SenderPostalCode</Fields>
        </ErrorDetails>
      </StatusInfo>
      <Parcels>
        <Parcel></Parcel>
      </Parcels>
    </Package>
  </Packages>
</PackagesImportResponse>
```

Example of PackagesImportResponse for the XML query importPackagesXV1method for DPDCRXmlServices interface with wrong service configuration for the parcel.

PackagesImportResponse structure - Status INCORRECT_DATA:

```
<PackagesImportResponse>
  <StatusInfo>
    <Status>INCORRECT_DATA</Status>
  </StatusInfo>
  <Packages>
    <Package>
      <StatusInfo>
        <Status>INCORRECT_DATA</Status>
        <ErrorDetails>
          <Code>ERROR_INCORRECT_WEIGHT_FOR_DOX</Code>
          <Description>DOX service available for parcels up to 0.5 kg</Description>
          <Fields>Parcels/Parcel/Weight;Services/DOX;</Fields>
        </ErrorDetails>
      </StatusInfo>
      <Parcels>
        <Parcel></Parcel>
      </Parcels>
    </Package>
  </Packages>
</PackagesImportResponse>
```

3.2.1.2 importPackagesCV1 method

3.2.1.2.1 Signature

Method name	importPackagesCV1
Input signature	byte[] importPackagesCV1(byte[] openUMLFCV2, PkgImportPolicyV1 pkgNumsGenerationPolicyV1, AuthDataV1 authDataV1)
Input parameters	openUMLFCV2 – data (byte table – XML compressed by ZIP and then coded as Base64) regarding parcels in the OpenUMLFV2 format;
	pkgNumsGenerationPolicyV1 – policy of operation in the event of data import errors, calculation type PkgImportPolicyV1;
	authDataV1 – data used for user authentication, type AuthDataV1;
Output	byte[] – XML in the importPackagesCV1Response format, compressed by ZIP and then coded as Base64 in the form of a byte table;

3.2.1.2.2 Parameters and data validation

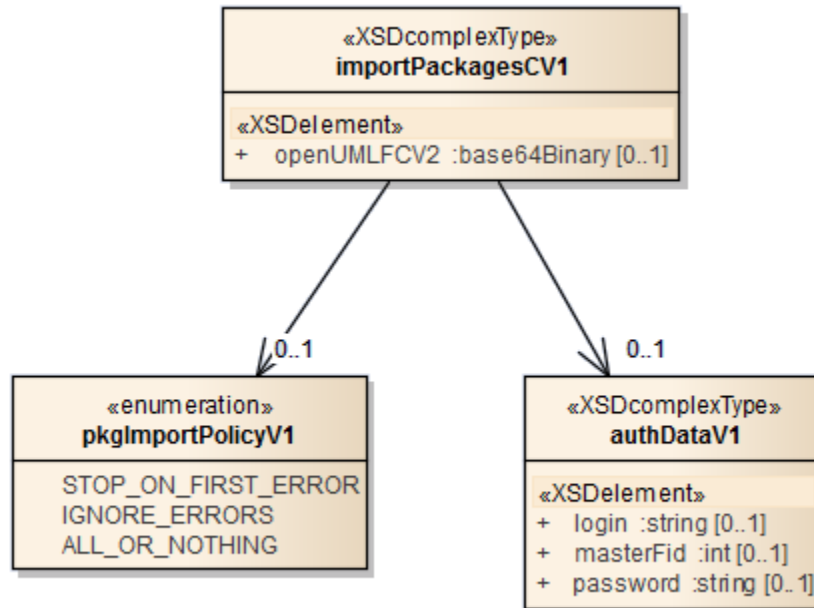
For importPackagesCV1:

Field name	Type	Validation condition	Mandatory	Additional information
------------	------	----------------------	-----------	------------------------

openUMLFCV2	byte[]	No field	Yes	Parameter, which contains data regarding the parcel, selected services and other key information which indicate the mode of processing parcel information. Data is provided in the form of a byte table which contains XML in the OpenUMLFV2 format compressed by ZIP and then coded in Base64. More information regarding the OpenUMLFV2 format is provided in the attachment describing all version of the OpenUMLF format. The above mentioned attachment is entitled OpenUMLF description.
-------------	--------	----------	-----	--

pkgNumsGenerationPolicyV1	PkgNumsGenerationPolicyV1	No validation	No	<p>Glossary parameter describing the service policy regarding the mode of reaction to errors appearing during the execution of the method call. Depending on the selected option the service will be conducted in a different way when errors occur. The possible values are:</p> <p>STOP_ON_FIRST_ERROR – means that the service will stop the operation when the first error occurs for any of the parcels. The import of first parcels will be successful;</p> <p>IGNORE_ERRORS – results in ignoring a general error for the specified record / parcel. This means that the wrong parcel will be ignored, but the other parcels with correct data will be correctly issued by the service;</p> <p>ALL_OR_NOTHING – stops the import of parcels for any error. All parcels must be positively verified for the import to take place. The default value is ALL_OR_NOTHING.</p>
authDataV1	AuthDataV1	No field	Yes	No field results in the message "AuthData is null".

3.2.1.2.3 Query structure



An example of a correct XML query for importPackagesCV1 method. The openUMLFCV2 element contains an XML in the OpenUMLFV2 format, compressed by ZIP and then coded as Base64. The nature of this element is the same as for the description of importPackagesXV1 method for DPDCRXmlServices interface.

Query for importPackagesCV1 method for DPDCRXmlServices interface:

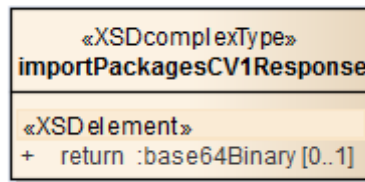
```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:cr="http://cr.dpdappservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <cr:importPackagesCV1 xmlns:ns2="http://cr.dpdappservices.dpd.com.pl/">
      <openUMLFCV2>
        UEsDBBQAAAAIAHtwbEwfZm5GqgIAAB4JAAAYAAAAQ1ltSU5fb3BibIVNTFhWMI9yZXEueG1s7V
        bbbtNAEH0uEv9g8h7WI5YWtLVU7BYiocZyTG8vaGVP2yWObe2uC/YP8BH8DvwX42vqpMUSKm/1
        Q7xn5szs7pyZyNRj4ZLdgLRfvtjpAK619qELSCIQraW3OukqY0lhU9KuNgmnBAV2JkqQRbxbkWqmx
        LNMkiDsegqSkcm+GHEWRACntAAQk4ddclkwzTEpa+9YRuCrseSxLVSSUVGj7kHmiROGkEdjeJ+S
        s8SbVS6Vice0xdGP/gJLess29TRPAm1fvLefxvEYnfX7fh3JViGpDyHwu38tbgBSaR6HMC05SM5Uo
        QVY75D/vboOEXfFsUYKe460kn1jT1IZ3TT3zf9Q2Xt13KI6uAARFBIW6OPMd794R35wiTk7c8MKbrml
        PCZUcTJzbWP37R4IA1tD8+Hasl36wX0Q9GYTj1K9eoNI+8cnlqYE/P7560fttBqnk0uVrprzIVxv2G9DB
        hS6aOekQa+m0ybwQ84ESxSAPvAeh5P3pjOpKJXvSCmBJzXf6frUOMBfShpTE0r62NbgQhgZAdEZi
        3Pok6wqHW1T119XCRpYO+tbGouCxT8lJIOdfk389URc/cJMjdZat5nl7RLf94vZ4kHqnZoAfG1k8aaa
        HvicOL5s7NJ5/YEv7tPx2kQxSzpoJuroqeyOAbVe+YX/RliHjlfKqtxtUUFEXAcPwE4/p6lYp0jizweLvOs
        u3iWRyle2jCt3b03IFSwu/WQO522fTLsDOzVEOLNAW7Na+vacw785hZloKRZPcRZ8BlusLnr92OEy5
        Zw+RjhqiVcPURwUuxE1F/hXxglHdom9pPjMsUMexlZ5DEtYgloOI8D71i4OQzHEgy8Y+HWMNwahu
        bEpBGg0bNcUH2xgUx9FFJDH1UFEN/lqWVpQPNlw9Zf/r0a2n/AVBLAQI/ABQAAAAIAHtwbEwfZm5
        GqgIAAB4JAAAYACQAAAAAAAAAAAAAAAAABDUi1JT19vcGVuVU1MWFYyX3JlcS54bWwKAC
      </openUMLFCV2>
    </cr:importPackagesCV1>
  </soapenv:Body>
</soapenv:Envelope>
  
```

AAAAAAAAEAGADC96aSArrTAS+KF0YvttMBL6gSRi+20wFQSwUGAAAAAAEAAQBqAAAA4AIAA
AAA

```
</openUMLFCV2>
<pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
<authDataV1>
  <login>pmarat</login>
  <masterFid>1495</masterFid>
  <password>****</password>
</authDataV1>
</cr:importPackagesCV1>
</soapenv:Body>
</soapenv:Envelope>
```

3.2.1.2.4 Response structure



Example of a response for the correct XML query for importPackagesCV1 method is provided below.

Response for correct query for importPackagesCV1 method for DPDCRXmlServices interface:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:importPackagesCV1Response xmlns:ns2="http://cr.dpdappservices.dpd.com.pl">
      <return>
        UEsDBBQACAAIAMhzbEwAAAAAAAAAAAAAAAAAAAAAAdGV4dJWRbwuCMBCHP9KZyZwwBtGr
        EahY0OupK8L5h22+6NtH5k2Jltqbe/jdeDjuWC6rRI6VFe3QG1coO/SdVZwdnXSjFd2IR+bZgcGMCK8
        +Ojx5EDVPYkljGjNYoj/lplLalmB9mkkSbqLJPCfsLO/ITWseTC8hAY1JeGKAOf79aqO/bNuPNvBTZq
        ZWJh3bUhm+y3PYFylFv4N1129klbvCt4M8AFBLBwgjiqNKogAAAKkBAABQSwECFAAUAAgACAD
        lc2xMI4qjSqlAAACpAQAABAAAAAAAAAAAAAAAAAAAAAAdGV4dFBLBQYAAAAAAAAQABADIA
        AADUAAAAAAAA=
      </return>
    </ns2:importPackagesCV1Response>
  </S:Body>
</S:Envelope>
```

The return element of the above message contains the XML in the PackagesImportResponse format compressed by ZIP and the coded as Base64. It returns the parcel import status. The nature of the PackagesImportResponse format is the same as for the response for DPDCRXmlServices method.

3.2.1.3 importPackagesXV2 method

3.2.1.3.1 Signature

Method name	importPackagesXV2
Input signature	byte[] importPackagesXV2(byte[] openUMLFXV2, PkgImportPolicyV1 pkgNumsGenerationPolicyV1, AuthDataV1 authDataV1)
Input parameters	openUMLFXV2 – data (XML coded as Base64, in the form of byte table) regarding parcels in the OpenUMLFV2 format;
	pkgNumsGenerationPolicyV1 – policy of operation in the event of data import errors, calculation type PkgImportPolicyV1;
	authDataV1 – data used for user authorisation, type AuthDataV1;
Output	byte[] – XML in the importPackagesXV2Response format, coded as Base64, in the form of a byte table;

3.2.1.3.2 Parameters and data validation

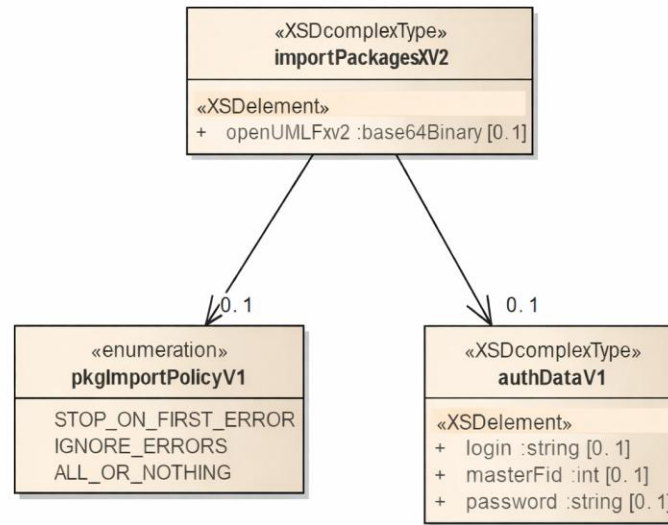
For importPackagesXV2:

Field name	Type	Validation condition	Mandatory	Additional information
------------	------	----------------------	-----------	------------------------

openUMLFXV2	byte[]	No field	Yes	Parameter, which contains data regarding the parcel, selected services and other key information which indicate the mode of processing parcel information. Data is provided in the form of a byte table which contains XML in the OpenUMLFV2 format, coded in Base64. More information regarding the OpenUMLFV2 format is provided in the attachment describing all version of the OpenUMLF format. The above mentioned attachment is entitled OpenUMLF description.
-------------	--------	----------	-----	--

pkgNumsGenerationPolicyV1	PkgNumsGenerationPolicyV1	No validation	No	<p>Glossary parameter describing the service policy regarding the mode of reaction to errors appearing during the execution of the method call. Depending on the selected option the service will be conducted in a different way when errors occur. The possible values are:</p> <p>STOP_ON_FIRST_ERROR – means that the service will stop the operation when the first error occurs for any of the parcels. The import of first parcels will be successful;</p> <p>IGNORE_ERRORS – results in ignoring a general error for the specified record / parcel. This means that the wrong parcel will be ignored, but the other parcels with correct data will be correctly issued by the service;</p> <p>ALL_OR_NOTHING – stops the import of parcels for any error. All parcels must be positively verified for the import to take place. The default value is ALL_OR_NOTHING.</p>
authDataV1	AuthDataV1	No field	Yes	Lack of field results in the message "AuthData is null".

3.2.1.3.3 Query structure



An example of a correct XML query for the `importPackagesXV2` method is provided below. The `openUMLFXV2` element includes an XML in the `OpenUMLFV2` format coded as `Base64`.

Query for the importPackagesXV2 method for DPDCRXmlServices interface:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:cr="http://cr.dpdappservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <cr:importPackagesXV2>
      <openUMLFXV2>
```

[illegible]


```

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

```

The response type for XML query for importPackagesXV2 method for DPDCRXmlServices interface with wrong user authentication data.

Response to a query with wrong user authentication data:

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>S:Server</faultcode>
      <faultstring>Login failed</faultstring>
      <detail>
        <ns2:DPDAppAuthenticationException xmlns:ns2="http://cr.dpdappservices.dpd.com.pl/">
          <login>pmarat</login>
          <message>Login failed</message>
        </ns2:DPDAppAuthenticationException>
        <ns2:exception
class="pl.com.dpd.dpdappservices.exception.DPDAppAuthenticationException" note="To disable
this feature, set com.sun.xml.ws.fault.SOAPFaultBuilder.disableCaptureStackTrace
system property to false" xmlns:ns2="http://jax-ws.dev.java.net/">
          <message>Login failed</message>
          <ns2:stackTrace>
            <!--Zawartość StackTrace-->
          </ns2:stackTrace>
        </ns2:exception>
      </detail>
    </S:Fault>
  </S:Body>
</S:Envelope>

```

The return element of the response for importPackagesXV2 contains an XML in the PackagesImportResponse format coded as Base64. It returns the parcel import status. An example of correctly performed import is provided below.

PackagesImportResponse structure - Status OK:

```

<PackagesImportResponse>
  <StatusInfo>
    <Status>OK</Status>
  </StatusInfo>
  <Packages>
    <Package>
      <PackageId>9769083</PackageId>
      <StatusInfo>
        <Status>OK</Status>
      </StatusInfo>
      <Parcels>
        <Parcel>
          <ParcelId>9692162</ParcelId>
          <Waybill>0000009609362T</Waybill>
        </Parcel>
      </Parcels>
    </Package>
  </Packages>
  <OrderNumber>APP/CRIN/9769083</OrderNumber>

```

```

    </Package>
  </Packages>
</PackagesImportResponse>

```

Example of PackagesImportResponse for the XML query importPackagesXV2 method for DPDCRXmlServices interface with wrong sender post code.

PackagesImportResponse structure - Status INCORRECT_DATA:

```

<PackagesImportResponse>
  <StatusInfo>
    <Status>INCORRECT_DATA</Status>
  </StatusInfo>
  <Packages>
    <Package>
      <StatusInfo>
        <Status>INCORRECT_DATA</Status>
        <ErrorDetails>
          <Code>INCORRECT_SENDER_POSTAL_CODE</Code>
          <Description>Incorrect sender postal code (10178)</Description>
          <Fields>SenderPostalCode</Fields>
        </ErrorDetails>
      </StatusInfo>
      <Parcels>
        <Parcel></Parcel>
      </Parcels>
    </Package>
  </Packages>
</PackagesImportResponse>

```

Example of PackagesImportResponse for the XML query importPackagesXV2 method for DPDCRXmlServices interface with wrong service configuration for the parcel.

PackagesImportResponse structure - Status INCORRECT_DATA:

```

<PackagesImportResponse>
  <StatusInfo>
    <Status>INCORRECT_DATA</Status>
  </StatusInfo>
  <Packages>
    <Package>
      <StatusInfo>
        <Status>INCORRECT_DATA</Status>
        <ErrorDetails>
          <Code>ERROR_INCORRECT_WEIGHT_FOR_DOX</Code>
          <Description>DOX service available for parcels up to 0.5 kg</Description>
          <Fields>Parcels/Parcel/Weight;Services/DOX;</Fields>
        </ErrorDetails>
      </StatusInfo>
      <Parcels>
        <Parcel></Parcel>
      </Parcels>
    </Package>
  </Packages>
</PackagesImportResponse>

```

3.2.1.4 importPackagesCV2 method

3.2.1.4.1 Signature

Method name	importPackagesCV2
Input signature	byte[] importPackagesCV2(byte[] openUMLFCV2, PkgImportPolicyV1 pkgNumsGenerationPolicyV1, AuthDataV1 authDataV1)
Input parameters	openUMLFCV2 – data (byte table – XML compressed by ZIP and then coded as Base64) regarding parcels in the OpenUMLFV2 format;
	pkgNumsGenerationPolicyV1 – policy of operation in the event of data import errors, calculation type PkgImportPolicyV1;
	authDataV1 – data used for user authentication, type AuthDataV1;
Output	byte[] – XML in the importPackagesCV2Response format, compressed by ZIP and then coded as Base64 in the form of a byte table;

3.2.1.4.2 Parameters and data validation

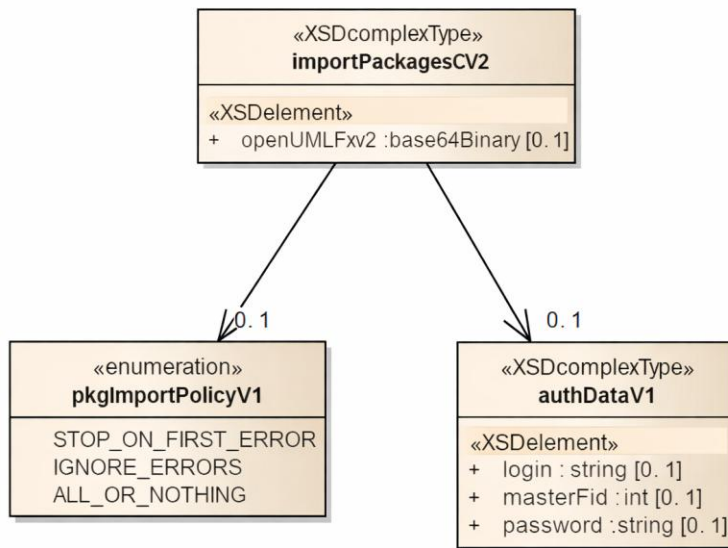
For importPackagesCV2:

Field name	Type	Validation condition	Mandatory	Additional information
------------	------	----------------------	-----------	------------------------

openUMLFCV2	byte[]	No field	Yes	Parameter, which contains data regarding the parcel, selected services and other key information which indicate the mode of processing parcel information. Data is provided in the form of a byte table which contains XML in the OpenUMLFV2 format compressed by ZIP and then coded in Base64. More information regarding the OpenUMLFV2 format is provided in the attachment describing all version of the OpenUMLF format. The above mentioned attachment is entitled OpenUMLF description.
-------------	--------	----------	-----	--

pkgNumsGenerationPolicyV1	PkgNumsGenerationPolicyV1	No validation	No	<p>Glossary parameter describing the service policy regarding the mode of reaction to errors appearing during the execution of the method call. Depending on the selected option the service will be conducted in a different way when errors occur. The possible values are:</p> <p>STOP_ON_FIRST_ERROR – means that the service will stop the operation when the first error occurs for any of the parcels. The import of first parcels will be successful;</p> <p>IGNORE_ERRORS – results in ignoring a general error for the specified record / parcel. This means that the wrong parcel will be ignored, but the other parcels with correct data will be correctly issued by the service;</p> <p>ALL_OR_NOTHING – stops the import of parcels for any error. All parcels must be positively verified for the import to take place. The default value is ALL_OR_NOTHING.</p>
authDataV1	AuthDataV1	No field	Yes	No field results in the message "AuthData is null".

3.2.1.4.3 Query structure



An example of a correct XML query for importPackagesCV2 method. The openUMLFV2 element contains an XML in the OpenUMLFV2 format, compressed by ZIP and then coded as Base64. The nature of this element is the same as for the description of importPackagesXV2 method for DPDCRXmlServices interface.

Query for importPackagesCV2 method for DPDCRXmlServices interface:

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:cr="http://cr.dpdappservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <cr:importPackagesCV2 xmlns:ns2="http://cr.dpdappservices.dpd.com.pl/">
      <openUMLFV2>
        UEsDBBQACAgIAE9NTFwAAAAAAAAAAAAAAAAAAAAAdGV4dN2V327aMBTG7yf1HfICJYIDoEiR
        NQSDvRfNEWXq6M3kJqdgkdjINq3C09f5AyQmNL3YxVQulvs7v5Nz9B2bBCGJNmQFEI99sfQvqPbV
        tpDugcUgakqhTni6JSzD453iKVEQWyUX2lelklCLpNBCF7KBjuNYgJT4J2UgSMKI5fqBfVDNRqjK8A
        MRck9eia6eb8963TElsgmPAYc/8g5PewMNUVQkKSIOQsN+YNcUk11zBthFXt8fDG9GGi0Eg7pNCU2
        wAqncr/E27kU87W2TwC71ms/2mdHBHCKg5fN/6jZh/e8b/dM7DO5V/yZWDOe7Nln7Tt63/cZT7lge
        P/pfED5Pio2/jRzXDg9z3kdhrf4XuLy/oeZCAW2Rbw4vvdvPo3HM8XS13oKNfyxZqKOCRCZd/uptjtj/R
        hbWqNkT67OH/kdfWyGUJ5CBUhVA9NdILP8vwgnAoaZeyWDH2TxQuNQBqn7lzUjUeQmCNvzqBA
        WojgAehqrbDuqVq1MPd0D39ypFxcIJbYd0pkeQI5PCKPbciEMwVMYSc/e+Wyyjaq8mhJFXPz7layotU
        mopsnJyDLYkY1w8Wxmoa4sr8jymlmeOWu71fIPDKL3X43C/ZSjOEj5nQmmkOT/J5nOBYwcNLh20L
        WrX9oIVF9c+/TJPa4lfgNQSwcIXRTeyvMBAACZBwAAUESBAhQAFAAICAgAT01MXF0U3srzAQAA
        mQcAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAHRleHRQSwUGAAAAAAEAAQAYAAAAJQIAAAAA
      </openUMLFV2>
      <pkgNumsGenerationPolicyV1>ALL_OR_NOTHING</pkgNumsGenerationPolicyV1>
      <authDataV1>
  
```

```

<login>pmarat</login>
<masterFid>1495</masterFid>
<password>****</password>
</authDataV1>
</cr:importPackagesCV2>
</soapenv:Body>
</soapenv:Envelope>

```

3.2.1.4.4 Response structure



Example of a response for the correct XML query for importPackagesCV2 method is provided below.

Response for correct query for importPackagesCV2 method for DPDCRXmlServices interface:

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:importPackagesCV2Response xmlns:ns2="http://cr.dpdappservices.dpd.com.pl/">
      <return>
        UEsDBBQACAgIAJxNTFwAAAAAAAAAAAAAAAAAAAAAAdGV4dJWRUQuCMBDHP9JcLm0wBtHT
        CFQs6Hnqimg62eZD3z4sb0oU0V7ux//Gj+OOfbK+yYtyou2N9aVyvemc4uzgpR+c6M4GmOd7hiYEe
        PXBESiAaDjdpDHGK4bm6E+5rZV2AFBHC0olxfFonhJ2kvfqjWPno8mlOBofWQlcvj71UZ+2ZKPNh
        SmzG2jbDa0IbJ8WxRoV4oMhR0su2EjM7kFvh3kAVBLBwh378tpogAAAKkBAABQSwECFAAUAAgl
        CACcTUxcd+/LaaIAAACpAQAABAAAAAAAAAAAAAAAAAAAAAAdGV4dFBLBQYAAAAAAQAB
        ADIAAADUAAAAAA=
      </return>
    </ns2:importPackagesCV2Response>
  </S:Body>
</S:Envelope>

```

The return element of the above message contains the XML in the PackagesImportResponse format compressed by ZIP and the coded as Base64. It returns the parcel import status. The nature of the PackagesImportResponse format is the same as for the response for DPDCRXmlServices method.

3.2.1.5 Parcel services

In the DPDAppServices, within the queries for the interface, it is possible to select services relating to parcel delivery. Such services are shown in the Services element as new elements. The list of service will vary for domestic and international parcels. Some of the services require specific conditions which must be met in order to use such service. Some of the services may be found mutually exclusive during the validation. Forbidden configurations will result in an appropriate message being returned in response to a query (examples are shown in the subsection regarding response structure). The tables below present services available on the basis whether the dispatch and collection occurs in Poland (domestic service) or the dispatch or collection occurs outside of Poland (international service).

3.2.1.5.1 Domestic service

Domestic service:

- Guarantee – the service relates to the guarantee of delivery of the parcel in accordance with service configuration. The configuration is performed by setting a selected value for the attribute type and any additional elements. The available configurations are:
 - 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. Set by selecting the value of TIME0930 for the type attribute;
 - TIME1200 – Guarantee of attempt of delivery before 9:30, on the first working day following the dispatch day. Set by selecting the value of TIME0930 for the type attribute;
 - TIMEFIXED – Guarantee of attempt of delivery at the time specified by the sender, on the first working day following the dispatch day. The specified time 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). Set by selecting the TIMEFIXED value for the type attribute and the child element Attr1, which provides the specific time in the hh:mm format. The selected time should be between 10:30 - 16:00;
 - B2C – service provided by DPD to selected clients - for certain post codes it is possible to select precise guaranteed delivery time interval, when parcel delivery is guaranteed. Set by selecting B2C value for the type attribute and child element Attr1, which provides the time interval in the hh:mm-hh:mm format. Available time intervals: 10:00-13:00, 13:00-16:00, 16:00-19:00;
 - SATURDAY – Guarantee of attempt of delivery on Saturday between 8:00 - 17:00. Set by selecting the SATURDAY value for the type attribute;
 - DPDNEXTDAY – Guarantee of attempt of delivery on the next working day following the dispatch. Set by selecting the DPDNEXTDAY value for the type attribute.

Examples of use:

```
<Services>  
<Guarantee type="TIME0930"/>
```

</Services>

<Services>
<Guarantee type="TIME1200"/>
</Services>

<Services>
<Guarantee type="TIMEFIXED">
<Attr1>15:45</Attr1>
</Guarantee>
</Services>

<Services>
<Guarantee type="B2C">
<Attr1>10:00-13:00</Attr1>
</Guarantee>
</Services>

<Services>
<Guarantee type="SATURDAY"/>
</Services>

<Services>
<Guarantee type="DPDNEXTDAY"/>
</Services>

- DeclaredValue – declared value of the parcel contains mandatory elements: Amount – the amount provided in decimal format of 0.00 and Currency – the currency. If the currency is not specified the system will assume the default value of PLN. The values available for currency:
 - PLN
 - EUR
 - USD

Example of use:

<Services>
<DeclaredValue>
<Amount>100.5</Amount>
<Currency>PLN</Currency>
</DeclaredValue>
</Services>

- COD – 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 charge an additional insurance. The insurance amount is determined on the basis of the current standard price list.

The service contains mandatory elements: Amount – the COD amount provided in decimal format 0.00 and Currency – the currency. If the currency is not specified the system will assume the default value of PLN. The values available for currency:

- PLN
- EUR

- USD

For the domestic service, the currency used in the Currency parameter must be set at the PLN value. The amount must be within the 0.01-15000 interval. COD is available for the Payer/Recipient who has not been blocked for this service.

Example of use:

```
<Services>
  <COD>
    <Amount>78.35</Amount>
    <Currency>EUR</Currency>
  </COD>
</Services>
```

- CUD – return parcel service. At the time of delivery the recipient hands over a return parcel sent to the address of the sender of the initial parcel. The service is performed within 2 working days from the day of delivery. Example of use:

```
<Services>
  <CUD/>
</Services>
```

- DOX – an envelope weighing up to 0.5 kg. The weight specified in the Wight element cannot exceed 0.5 kg.

Example of use:

```
<Services>
  <DOX/>
</Services>
```

- ROD – service consisting in collecting the recipients signature on the document attached to the parcel and returning that document to the sender.

Example of use:

```
<Services>
  <ROD/>
</Services>
```

- InPers – delivery to a specific person. The parcel must be collected by the person specified on the label. The courier verifies the recipient's identity on the basis of an ID document, such as national ID. First name and surname of the addressee is required.

Example of use:

```
<Services>
  <InPers/>
</Services>
```

- SelfCol – personal collection from a DPD depot. The recipient must be verified by an ID document – for an individual or by a company stamp – for legal entities. The receiver attribute contains the recipient type. Available values:

- PRIV – individual;
- COMP – company;

Example of use:

```
<Services>
  <SelfCol receiver="PRIV"/>
</Services>
```

- PrivPers – delivery to an individual to a private address.

Example of use:

```
<Services>
  <PrivPers/>
```

</Services>

- CarryIn – the parcel is carried in by the courier to the place shown by the client. 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.

Example of use:

<Services>

<CarryIn/>

</Services>

- Tires – parcel containing tires.

Example of use:

<Services>

<Tires/>

</Services>

3.2.1.5.2 International service

International service:

- Guarantee – the service relates to the guarantee of delivery of the parcel in accordance with service configuration. The configuration is performed by setting a selected value for the attribute type and any additional elements. The available configurations are:

- INTER – guaranteed day of delivery, this service excludes Palette and TiresExport services;

Example of use:

<Services>

<Guarantee type="INTER"/>

</Services>

- DeclaredValue – declared value of the parcel contains mandatory elements: Amount – the amount provided in decimal format of 0.00 and Currency – the currency. If the currency is not specified the system will assume the default value of PLN. The values available for currency:

- PLN
- EUR
- USD

Example of use:

<Services>

<DeclaredValue>

<Amount>100.5</Amount>

<Currency>PLN</Currency>

</DeclaredValue>

</Services>

- COD – 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 charge an additional insurance. The insurance amount is determined on the basis of the current standard price list.

The service contains mandatory elements: Amount – the amount provided in decimal format of 0.00 and Currency – the currency. If the currency is not specified the system will assume the default value of PLN. The values available for currency:

- PLN
- EUR
- USD

For the domestic service, the currency used in the Currency parameter must be set at the PLN value. The service is available only to the recipients in Poland. The amount must be within the 0.01-15000 interval. COD is available for the Payer/Recipient who has not been blocked for this service.

Example of use:

```
<Services>
  <COD>
    <Amount>78.35</Amount>
    <Currency>EUR</Currency>
  </COD>
</Services>
```

- CUD – return parcel service. At the time of delivery the recipient hands over a return parcel sent to the address of the sender of the initial parcel. The service is performed within 2 working days from the day of delivery. Service available only to recipients in Poland.

Example of use:

```
<Services>
  <CUD/>
</Services>
```

- DOX – an envelope weighing up to 0.5 kg. The weight specified in the Wight element cannot exceed 0.5 kg. Service available only to recipients in Poland.

Example of use:

```
<Services>
  <DOX/>
</Services>
```

- ROD – service consisting in collecting the recipients signature on the document attached to the parcel and returning that document to the sender. Service available only to recipients in Poland.

Example of use:

```
<Services>
  <ROD/>
</Services>
```

- PrivPers – delivery to an individual to a private address. Service available only to recipients in Poland.

Example of use:

```
<Services>
  <PrivPers/>
</Services>
```

- Duty – customs clearance for parcels sent outside the EU common customs territory. The service is available for selected countries and post codes.
Example of use:

```
<Services>
  <Duty/>
</Services>
```
- Pallet – Service enables sending goods on pallets. Maximum allowed weight is 700 kg. Maximum parcel height is 180cm. Maximum parcel length is 120cm. Maximum width is 80cm. Dimensions provided (order of x, y, z must be maintained) must be a positive integer.
It is not possible to combine this service with: Guarantee INTER, and TiresExport services.
Example of use:

```
<Services>
  <Pallet/>
</Services>
```
- TiresExport – transport of tires as an export service. It is not possible to combine this service with Guarantee INTER and Palette services.
Example of use:

```
<Services>
  <TiresExport/>
</Services>
```

3.3 Error codes

The full list of possible errors together with IDs and descriptions is provided in the attachment entitled Error Codes.

3.4 Comments and suggestion

Permission levels:

When the user attempts to perform operations unavailable for his permission levels, he/she will receive a message: „User has no privileges to execute this operation”.

OpenUMLF format error:

There are a number of cases where OpenUMLF format error can occur. It usually means a wrong interpretation of the parameter syntax. The exact contents of the format error are preceded by the message: „Incorrect OpenUMLFe format: ”

Field with the collection date:

For the importPackagesV2, XV2, and CV2 methods, the <DeliveryDate>2026-02-11</DeliveryDate> field has been defined, which accepts a date from 1 to 3 business

days in advance and is required only when the sender's country is PL. In other cases, the field is ignored.

Unknown error:

In the event of service's external problem an „Unknown error” message can appear. It relates to all non-classified errors in the service. If such error appears, it will be necessary to contact the service provider.

4 Attachments

1. OpenUMLF description
2. Error codes