bpost address services

Version 1.6

Table of Contents

[1. Introduction 3](#_Toc475368347)

[Required Knowledge 3](#_Toc475368348)

[bpost Address Validation Services 3](#_Toc475368349)

[Address rendition 4](#_Toc475368350)

[Available interfaces 5](#_Toc475368351)

[Address formatting 5](#_Toc475368352)

[Address validation 6](#_Toc475368353)

[Exact match 7](#_Toc475368354)

[Match but anomalies found 7](#_Toc475368355)

[No match found 7](#_Toc475368356)

[Partial match found 7](#_Toc475368357)

[Multiple matches found 8](#_Toc475368358)

[No match found but suggestions returned 8](#_Toc475368359)

[2. Input addresses 9](#_Toc475368360)

[Address formatting service 9](#_Toc475368361)

[Fictitious address examples 10](#_Toc475368362)

[Address validation service 13](#_Toc475368363)

[Address feedback service 17](#_Toc475368364)

[3. Calling the webservice via SOAP UI 17](#_Toc475368365)

[Creating a new project 17](#_Toc475368366)

[Generating a valid request 21](#_Toc475368367)

[4. Warnings and Errors 23](#_Toc475368368)

[5. Use cases and example of implementation 26](#_Toc475368369)

[Capturing and submitting an address 26](#_Toc475368370)

[Receiving a response from the address validation & formatting webservice 27](#_Toc475368371)

[6. Web Services 34](#_Toc475368372)

[Web Services Implementation 34](#_Toc475368373)

[REST 34](#_Toc475368374)

[SOAP 34](#_Toc475368375)

[Protocol 34](#_Toc475368376)

[Endpoint 34](#_Toc475368377)

[Versioning 34](#_Toc475368378)

[XML Validation 35](#_Toc475368379)

[JSON Schema Validation 35](#_Toc475368380)

[Security 35](#_Toc475368381)

[Operations 35](#_Toc475368382)

[ANNEX 40](#_Toc475368383)

1. Introduction

This document describes in detail the bpost address validation services and how to set them up for use in your business.

Required Knowledge

In order to use this manual you need knowledge of one of the following topics:

* JSON, REST web services

or

* XML, SOAP web services

JSON stands for JavaScript Object Notation. For an introduction, go to the W3 Schools website <https://www.w3schools.com/js/js_json_intro.asp>

XML stands for eXtensible Markup Language. For an introduction, go the W3 Schools XML Tutorial at <http://w3schools.com/xml/default.asp>.

bpost Address Validation Services

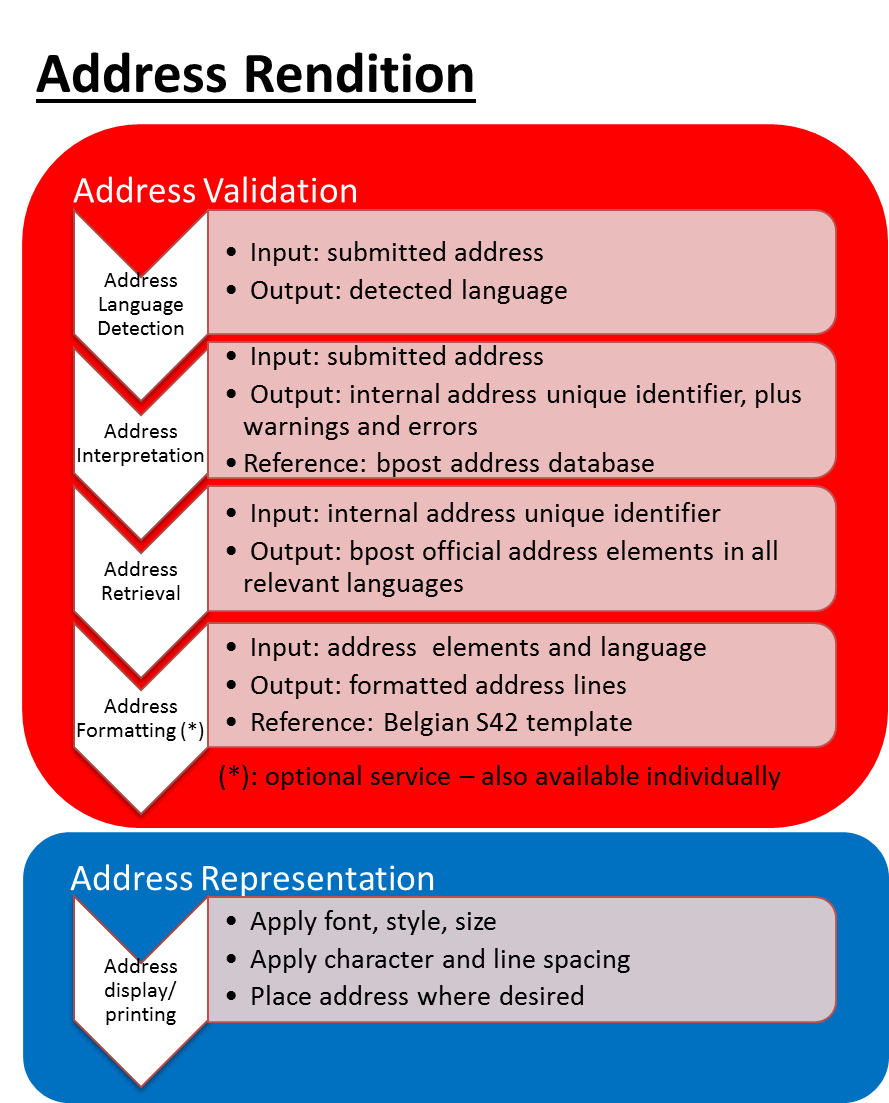
Address standards for Belgium have been published (<http://bpost.be/adressage>, <http://bpost.be/adressering> and <http://www.upu.int/fileadmin/documentsFiles/activities/addressingUnit/belEn.pdf>) and individual address validation is available as an interactive service (<http://bpost.be/validationadresse> and <http://bpost.be/adressvalidatie>). With this webservice API, bpost introduces a machine-to-machine interface to allow automated validation and formatting services.

We offer both REST and SOAP webservice, but please be aware that **future developments and improvements will only be supported by the REST service**.

For these webservices, an **API** **key** is required. Please request your personalAPI key to [addressvalidation@bpost.be](mailto:addressvalidation@bpost.be) API keys should be added in the header ‘x-api-key’.

Address rendition

Address rendition is the process of properly displaying or printing a valid address. This includes several concepts that may seem trivial, yet become critical when one enters an address into a GPS, dictates an address to emergency services, or delivers a package or letter to an address. Every detail ends up counting: the address content, the language, the postal code, the spelling of the street and the city, the house number and the letterbox number, the name of the recipient, the structure and sequence of the lines, the way the address is printed, etc… This rendition process includes several sub-processes:



* address validation: sequence of bpost processes all executed when called
  1. language detection: process to detect the language of the submitted address. This is useful to return address in proper language when the municipality is multi-lingual (Brussels plus facilities).
  2. address interpretation: process to recognize the submitted address via bpost proprietary algorithms, using bpost master address database as reference, and detect anomalies
  3. official bpost address retrieval: process to fetch the official bpost version of the address in its reference database
  4. address formatting: process to properly assemble and format the address lines according to the Belgian Addressing Standard registered with UPU (Universal Postal Union) and CEN (Centre Européen de Normalisation). Only this process is also available separately.
* address representation: client process to apply to the address lines: font, style, size, character and line spacing, placement on screen page, letter, envelope, etc

Available interfaces

bpost currently offers the following Address Validation services:

* formatAddresses: this interface takes address components as input, and returns address lines as output. Both Structured or Semi-structured inputfields are allowed.
* validateAddresses: this interface takes addresses (in a structured or semi-structured or address block lines) as input, and returns feedback about each address submitted: whether or not it was recognized, warnings and errors if detected, and a corrected version of the address if available

Address formatting

This service is available either packaged in the validation process, or individually to properly order and format the components of a previously validated address. It may be used either for individual addresses, or in batches of up to 100 addresses at a time[[1]](#footnote-1).

Formatting seems again to be a negligible aspect, but it is essential in proper addressing. Each country, including Belgium, has its own standard to facilitate the efficient processing of mail and parcels by machines and postmen. For example, the standard in Belgium specifies that the house number comes after the street name (whereas in France for example, the house number precedes the street name), and that an apartment building letterbox number is preceded by the key words ‘bus’ or ‘bte’ (and not the punctuation signs ‘/’, ‘#’ or letter ‘b’, etc…). Also, the address starts with the most specific information (addressee name) and ends with the most general information (postcode and town for domestic mail, or country name for cross-border mail).

The address formatting webservice expects as input all the individual components of an address. It returns properly assembled address lines that can be used on a label, an envelope or a screen.

Example:

|  |  |  |
| --- | --- | --- |
| **Request** | **Address Formatting** | **Response** |
| *Individual address fields* |  | *Properly ordered and formatted address lines according to standards* |
| Durand SA |  |  |
| Monsieur |
| Dupont | Monsieur Alain Dupont |
| Alain | Durand SA |
| 7C | Rue du Vivier 7C bte 5 |
| Rue du Vivier | 1000 Bruxelles |
| 5 |  |
| Bruxelles |
| 1000 |

Address validation

This service is available either for individual addresses, or in batches of up to 100 addresses at a time[[2]](#footnote-2). It validates whether an address is precise and unambiguous.

bpost has compiled a database of all the addresses in Belgium where mail or parcels may be delivered. This database does not include roads without postal delivery point. Also, this database is constantly updated according to what bpost employees can observe in the field, and this database may vary from time to time or be different from other public or private databases.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **Request** | **Address Validation and Formatting** | **Response** | **Errors** |
| *Individual address fields* |  | *Corrected fields* | *Message* |
| Charles | *People and companies are not validated by service* | |
| Michel |
| Avenue de la Loi | Rue de la Loi | anomaly |
| 16 | 16 |  |
|  | 1000 | missing |
| Bruxeles | Bruxelles | anomaly |
|  |  |  |
| *Properly ordered and formatted address lines according to standards* | |
|  | |
| Charles Michel | |
| Rue de la Loi 16 | |
| 1000 Bruxelles | |

Submitted addresses are validated against this reference database using a library of sophisticated algorithms. The validation process is tolerant of some imperfections. Below are different validation use cases. You may also try them out interactively at <http://bpost.be/validationadresse> or <http://bpost.be/adresvalidatie>

Exact match

Address submitted: Rue de la Loi 16

1000 Bruxelles

Feedback : no warnings or errors are returned as the submitted address is perfect

Match but anomalies found

Address submitted: Avenue de la Loi 16

1000 Bruxelles

Feedback : warning returned on the street name field (‘Avenue’ is not correct) and bpost’s version of the address is returned: Rue de la Loi 16

1000 Bruxelles

To ensure that bpost machines and postman can best handle postal items, it would be ideal for the client to integrate this feedback to update the source database with the bpost address returned.

IMPORTANT NOTE: In some cases, the address submitted is so ambiguous that the interpretation does not provide the expected result. This interpretation depends on the system’s ability to resolve the ambiguities. It most often works well, but not always. The feedback returned by the bpost service should be reviewed before being integrated.

No match found

Address submitted: Zwaluwenlaan 1

2610 Namur

Feedback: there are too many incoherences with this address for the system to interpret any address, or to pinpoint any particular cause.

These addresses require the most attention, because no bpost machine, and probably no human either, will be able to resolve them. They would most likely result in a ‘return to sender’, if not corrected by the client before mailing.

Partial match found

Address submitted: Rue du Faleau 12

6200 Châtelet

Feedback: error returned on the house number field as number 12 does not exist in this street, according to the bpost reference data.

These addresses should also require attention, because even though they may attempt to deliver the item, postmen may encounter much difficulty finding the right mailbox. No correction or suggestion can be provided.

Multiple matches found

Address submitted: Kerkstraat 6

Dilbeek

Feedback: error returned on the missing postal code. Multiple alternatives are returned as they all are possible solutions for the submitted address:

Kerkstraat 6

1700 Sint-Ulriks-Kapelle

Kerkstraat 6

1701 Itterbeek

Kerkstraat 6

1703 Schepdaal

As even a human could not arbitrate between these multiple valid solutions, these addresses should be reviewed and adjusted.

No match found but suggestions returned

Address submitted: Bredabaan 1

1800 Vilvoorde

Feedback: the address cannot be resolved. The system returns a number of possible suggestions with a pertinence level calculated by the algorithm, e.g.: Arendlaan, Medialaan, Streekbaan, etc…

These suggestions may or may not be relevant, and the addresses should be reviewed and adjusted.

2. Input addresses

Address formatting service

Only addresses intended for Belgium are supported currently, whether the mail originates from Belgium or not. The webservice expects input structured as seen below. Calling applications with a different structure should map their available fields to these appropriate elements:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **National Elements** | **CEN/UPU Address Elements** | | |
|  | **Address Element** | **Title** | **Code** | **Comment** |
| **Addressee Individual Identification** | Greeting | Form of Address | 10.05.0.0.0 | Mandatory if addressed mail |
| First Name | Given Name | 10.06.0.0.0 |
| Last Name | Surname | 10.08.0.0.0 |
| Customer Number | Suppl Dispatch Info | 30.33.0.0.0 | Only if bpack 24/7 |
| **Mailee Individual Identification** |  | Role Descriptor | 20.11.0.0.0 | If applicable |
|  | Form of Address | 20.05.0.0.0 |
|  | Given Name | 20.06.0.0.0 |
|  | Surname | 20.08.0.0.0 |
| **Mailee Organization Identification** | Title | Function | 20.03.0.0.0 | If applicable |
| Department | Organizational Unit | 20.02.0.0.0 |
| Company Name | Organization Name | 20.00.0.0.0 |
|  | Legal Status | 20.01.0.0.0 |
| **Mail Recipient Dispatching Information** | Building | Wing Type | 30.29.0.0.1 | Optional, but preferable for Registered mail and Parcels |
| Wing Indicator | 30.29.0.0.2 |
| Stairwell Type | 30.30.0.0.1 |
| Stairwell Indicator | 30.30.0.0.2 |
| Floor Type | 30.31.0.0.1 |
| Floor Indicator | 30.31.0.0.2 |
| Door Type | 30.32.0.0.1 |
| Door Indicator | 30.32.0.0.2 |
|  | Building/Construction Level 1 | 30.26.1.0.0 | Complex of buildings -- used to reference industrial zones |
| **Other Delivery Information** | PO Box Number  or  “bpack 24/7” name | Delivery Service Type | 40.19.0.0.1 | Possible values: ‘Postbus’ or ‘Boite Postale’ or ‘PB’ or ‘BP’ or ‘bpack’ |
| Delivery Service Indicator | 40.19.0.0.2 | PO Box number or bpack station name |
| **Delivery Point Location** | Street Name | Thoroughfare Name | 40.21.0.0.0 | Mandatory |
| House Number | Street Number or Plot | 40.24.0.0.0 |  |
| Box Number | Extension Designation | 40.28.0.0.0 | If not null, and must be preceded in representation by literal ‘bus’ or ‘bte’, or ‘box’ |
| **Postcode**  **/**  **Town** | Postal Code | Postcode | 40.13.0.0.0 | Mandatory |
| City | Town | 40.16.0.0.0 |
| Delivery Service Qualifier | 40.35.0.0.0 | When PO Box |
| **Country** | Country Name | Country | 40.14.0.0.0 | Only for cross-border mail |
| **Encoding** |  | Address Parameter Script | 50.50.0.0.0 | Always ‘Latn’ |
| **Language** |  | Address Parameter Language | 50.51.0.0.0 | Possible values: ‘de’, ‘en’, ‘fr’, ‘nl’ |
| **Despatching Country** | Country of origin |  | 50.53.0.0.0 | If not ‘BE’, will force ‘BELGIUM’ in Country Name |
| **Delivering Country** |  |  | 50.54.0.0.0 | Always ‘BE’ |

Fictitious address examples

|  |  |  |
| --- | --- | --- |
| *EXAMPLE 1* | *Private address of a person living in a house*  *Formatted address*  Monsieur Alain Dupont  Rue du Vivier 7  1000 Bruxelles | *Address elements*  10.05.0.0.0 Monsieur  10.06.0.0.0 Alain  10.08.0.0.0 Dupont  40.21.0.0.0 Rue du Vivier  40.24.0.0.0 7  40.13.0.0.0 1000  40.16.0.0.0 Bruxelles |
| *EXAMPLE 2* | *Private address, care of a person living in a house*  *Formatted address*  Monsieur Alain Dupont  Chez Madame Charlotte Durant  Rue du Vivier 15  1000 Bruxelles | *Address elements*  10.05.0.0.0 Monsieur  10.06.0.0.0 Alain  10.08.0.0.0 Dupont  20.11.0.0.0 Chez  20.05.0.0.0 Madame  20.06.0.0.0 Charlotte  20.08.0.0.0 Durant  40.21.0.0.0 Rue du Vivier  40.24.0.0.0 15  40.13.0.0.0 1000  40.16.0.0.0 Bruxelles |
| *EXAMPLE 3* | *Private address of a person living in a block of flats*  *Formatted address*  Dhr Paul Janssens  Kortijkstraat 37 bus 1  9800 Deinze | *Address elements*  10.05.0.0.0 Dhr  10.06.0.0.0 Paul  10.08.0.0.0 Janssens  40.21.0.0.0 Kortijkstraat  40.24.0.0.0 37  40.28.0.0.0 1  40.13.0.0.0 9800  40.16.0.0.0 Deinze |
| *EXAMPLE 4* | *Private address with delivery to a PO Box*  *Formatted address*  Dhr Paul Janssens  Postbus 24  9000 Gent Centrum | *Address elements*  10.05.0.0.0 Dhr  10.06.0.0.0 Paul  10.08.0.0.0 Janssens  40.19.0.0.1 Postbus  40.19.0.0.2 24  40.13.0.0.0 9000  40.35.0.0.0 Gent Centrum |
| *EXAMPLE 5* | *“bpack 24/7” address*  *Formatted address*  Mr Alain Dupont RC 123456789  bpack De Brouckère  1000 Bruxelles | *Address elements*  10.05.0.0.0 Mr  10.06.0.0.0 Alain  10.08.0.0.0 Dupont  30.33.0.0.0 RC 123-456-789  40.19.0.0.1 bpack  40.19.0.0.2 De Brouckère  40.13.0.0.0 1000  40.16.0.0.0 Bruxelles |
| *EXAMPLE 6* | *Private address of a person living in a block of flats with spatial information*  *Formatted address*  Dhr Paul Janssens  Gebouw A - Verdieping 3 - Kamer 8  Volklorenlaan 81-83 bus 15  2610 Wilrijk | *Address elements*  10.05.0.0.0 Dhr  10.06.0.0.0 Paul  10.08.0.0.0 Janssens  30.29.0.0.1 Gebouw  30.29.0.0.2 A  30.31.0.0.1 Verdieping  30.31.0.0.2 3  30.32.0.0.1 Kamer  30.32.0.0.2 8  40.21.0.0.0 Volklorenlaan  40.24.0.0.0 81-83  40.28.0.0.0 15  40.13.0.0.0 2610  40.16.0.0.0 Wilrijk |
| *EXAMPLE 7* | *Company address*  *Formatted address*  Monsieur Alain Dupont  Service Achats  Durand SA  Rue du Vivier 7C bte 5  1000 Bruxelles | *Address elements*  10.05.0.0.0 Monsieur  10.06.0.0.0 Alain  10.08.0.0.0 Dupont  20.02.0.0.0 Service Achats  20.00.0.0.0 Durand  20.01.0.0.0 SA  40.21.0.0.0 Rue du Vivier  40.24.0.0.0 7C  40.28.0.0.0 5  40.13.0.0.0 1000  40.16.0.0.0 Bruxelles |
| *EXAMPLE 8* | *Company address with function & department, and with industrial zone*  *Formatted address*  Monsieur Alain Dupont  Manager Service Achats  Durand SA  Parc d’Entreprises B  Rue du Vivier 7C bte 5  1000 Bruxelles | *Address elements*  10.05.0.0.0 Monsieur  10.06.0.0.0 Alain  10.08.0.0.0 Dupont  20.03.0.0.0 Manager  20.02.0.0.0 Service Achats  20.00.0.0.0 Durand  20.01.0.0.0 SA  30.26.1.0.0 Parc d’Entreprises B  40.21.0.0.0 Rue du Vivier  40.24.0.0.0 7C  40.28.0.0.0 5  40.13.0.0.0 1000  40.16.0.0.0 Bruxelles |
| *EXAMPLE 9* | *Company address with spatial information*  *Formatted address*  Dhr Paul Janssens  Accounting  Acme NV  Building West - Verdieping 4  Bloemendalelaan 62/3 bus 47  9990 Maldegem | *Address elements*  10.05.0.0.0 Dhr  10.06.0.0.0 Paul  10.08.0.0.0 Janssens  20.02.0.0.0 Accounting  20.00.0.0.0 Acme  20.01.0.0.0 NV  30.29.0.0.1 Building  30.29.0.0.2 West  30.31.0.0.1 Verdieping  30.31.0.0.2 4  40.21.0.0.0 Bloemendalelaan  40.24.0.0.0 62/3  40.28.0.0.0 47  40.13.0.0.0 9990  40.16.0.0.0 Maldegem |

Address validation service

Only addresses in Belgium can currently be validated.

1. Structured Address Input fields

Just like the address formatting service, this webservice expects input structured as seen below. Calling applications with a different structure should map their available fields to these appropriate elements:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **National Elements** | **CEN/UPU Address Elements** | | |
|  | **Address Element** | **Title** | **Code** | **Comment** |
| **Addressee Individual Identification** | Greeting | Form of Address | 10.05.0.0.0 | Mandatory if addressed mail |
| First Name | Given Name | 10.06.0.0.0 |
| Last Name | Surname | 10.08.0.0.0 |
| Customer Number | Suppl Dispatch Info | 30.33.0.0.0 | Only if bpack 24/7 |
| **Mailee Individual Identification** |  | Role Descriptor | 20.11.0.0.0 | If applicable |
|  | Form of Address | 20.05.0.0.0 |
|  | Given Name | 20.06.0.0.0 |
|  | Surname | 20.08.0.0.0 |
| **Mailee Organization Identification** | Title | Function | 20.03.0.0.0 | If applicable |
| Department | Organizational Unit | 20.02.0.0.0 |
| Company Name | Organization Name | 20.00.0.0.0 |
|  | Legal Status | 20.01.0.0.0 |
| **Mail Recipient Dispatching Information** | Building | Wing Type | 30.29.0.0.1 | Optional, but preferable for Registered mail and Parcels |
| Wing Indicator | 30.29.0.0.2 |
| Stairwell Type | 30.30.0.0.1 |
| Stairwell Indicator | 30.30.0.0.2 |
| Floor Type | 30.31.0.0.1 |
| Floor Indicator | 30.31.0.0.2 |
| Door Type | 30.32.0.0.1 |
| Door Indicator | 30.32.0.0.2 |
|  | Building/Construction Level 1 | 30.26.1.0.0 | Complex of buildings -- used to reference industrial zones |
| **Other Delivery Information** | PO Box Number  or  “bpack 24/7” name | Delivery Service Type | 40.19.0.0.1 | Possible values: ‘Postbus’ or ‘Boite Postale’ or ‘PB’ or ‘BP’ or ‘bpack’ |
| Delivery Service Indicator | 40.19.0.0.2 | PO Box number or bpack station name |
| **Delivery Point Location** | Street Name | Thoroughfare Name | 40.21.0.0.0 | Mandatory |
| House Number | Street Number or Plot | 40.24.0.0.0 |  |
| Box Number | Extension Designation | 40.28.0.0.0 | If not null, and must be preceded in representation by literal ‘bus’ or ‘bte’, or ‘box’ |
| **Postcode**  **/**  **Town** | Postal Code | Postcode | 40.13.0.0.0 | Mandatory |
| City | Town | 40.16.0.0.0 |
| Delivery Service Qualifier | 40.35.0.0.0 | When PO Box |
| **Country** | Country Name | Country | 40.14.0.0.0 | Only for cross-border mail |
| **Encoding** |  | Address Parameter Script | 50.50.0.0.0 | Always ‘Latn’ |
| **Language** |  | Address Parameter Language | 50.51.0.0.0 | Possible values: ‘de’, ‘en’, ‘fr’, ‘nl’ |
| **Despatching Country** | Country of origin |  | 50.53.0.0.0 | Not used for validation. If not ‘BE’, will force ‘BELGIUM’ in Country Name |
| **Delivering Country** |  |  | 50.54.0.0.0 | Always ‘BE’ |

2. Semi-structured Address input-fields:

Alternatively, addresses may be submitted in a semi-structured way where, for address each line, several elements are combined. While it is authorized to submit some lines structured and others semi-structured, it is not authorized to submit some elements of a line structured while other elements are semi-structured.

|  |  |  |
| --- | --- | --- |
|  | **Title** | **Comment** |
| **Addressee Individual Identification** | Form of Address +Given Name + Surname | Mandatory if addressed mail |
| **Mailee Individual Identification** | Role Descriptor + Form of Address +  Given Name + Surname | Not used for validation |
| **Mailee Organization Identification** | Organization Name + Legal Status | If applicable |
| **Mail Recipient Dispatching Information** | Wing Type + Wing Indicator + Stairwell Type + Stairwell Indicator + Floor Type + Floor Indicator + Door Type + Door Indicator | Optional, but preferable for Registered mail and Parcels. Each pair separated by ‘ – ‘ |
| Building/Construction Level 1 | Complex of buildings -- used to reference industrial zones |
| **Other Delivery Information** | Delivery Service Type + Delivery Service Indicator | Possible values: ‘Postbus’ or ‘Boite Postale’ or ‘PB’ or ‘BP’ or ‘bpack’ +  PO Box number or bpack station name |
| **Delivery Point Location** | Thoroughfare Name + Street Number or Plot + Extension Designation | If Extension Designation, must be preceded by literal ‘bus’ or ‘bte’, or ‘box’ |
| **Postcode**  **/**  **Town** | Postcode + Town + (in case of PO Box) Delivery Service Qualifier |  |
| **Country** | Country | Only for cross-border mail |
| **Encoding** | Address Parameter Script | Always ‘Latn’ |
| **Language** | Address Parameter Language | Possible values: ‘de’, ‘en’, ‘fr’, ‘nl’ |
| **Despatching Country** |  | Not used for validation |
| **Delivering Country** |  | Always ‘BE’ |

3. Address block input lines:

Addresses may also be submitted as a block of undefined lines. Our system will attempt to parse the lines into elements that can be validated. Up to 7 lines can be submitted in one request.

|  |  |  |
| --- | --- | --- |
|  | **Title** | **Comment** |
| **L1** | Contains a part of the address or is empty |  |
| **L2** | Contains a part of the address is empty |  |
| **L3** | Contains a part of the address is empty |  |
| **L4** | Contains a part of the address is empty |  |
| **L5** | Contains a part of the address is empty |  |
| **L6** | Contains a part of the address is empty |  |
| **L7** | Contains a part of the address is empty |  |
| **Encoding** | Address Parameter Script | Always ‘Latn’ |
| **Language** | Address Parameter Language | Possible values: ‘de’, ‘en’, ‘fr’, ‘nl’ |
| **Despatching Country** |  | Not used for validation |
| **Delivering Country** |  | Always ‘BE’ |

4. Other input fields

In all 3 cases, whether you submit an address in a structured , semi-structured or address block request, extra service and information can be returned by the address validation service , when asked for.

Following flags (true/false) can be submitted together within the request.

* Formatting flag : when the flag is true, the system will also return in the XML response the formatted address lines of the validated address.
* Formattedsubmitted address flag: this flag indicates whether one wants his submitted address components to be sent back in correctly formatted address lines. This is not supposed to be a valid address, it is just the result of the formatting webservice on your submitted address components.

When you do not provide this information in your request, the system will make following suppositions:

* Formatting flag: is by default true, the system shall always return the formatted address lines when no formatting flag was indicated in the request.
* Formattedsubmitted address flag : is by default false, the system shall never return the formatted-submitted-address lines when no flag was indicated in the request.

For the REST service, there are additional options that can be used. We will not describe them here but the options can be found in the yaml document attached.

5. Output fields / response of the address validation webservice

* Validated address components

The validated address components will be returned by the system in a structured way, indicating the type of validated address component (eg. Streetname, housenumber, boxnumber, Postalcode, Locality, …)

* Warnings and Errors

Errors and warnings will be returned by the system (when applicable), indicating the address component on which the error or warning was applicable. For more information on Errors and Warnings we would like to refer to chapter 4.Warnings and Errors.

* Formatted address lines

The validated address can be returned as formatted address lines. The formatted address lines are returned in the order in which the address information should be displayed on a letter or on a screen.

* Formattedsubmitted address

When a user wants his originally submitted address components to be formatted, independently of the outcome of the address validation of its address components, he can set this flag to true. The system will format the submitted address qnd return the formatted lines in the response.

* Detected language

The language that was detected from the address components will be retured in the response.

* AddressLanguage

The language in which the address components should be used will be returned in the response (e.g. “Rue des hirondelles” should be called “zwaluwenlaan” in the Flemish region)

* Transaction ID

ID which identifies the address validation response (on address level). The transaction ID is used to link the feedback you want to provide on an address as result of the address validation operation. The feedback you want to provide can be provided using another webservice as explained in the next topic: Address feedback service.

Address feedback service

As one can see on the bpost website when validating an address (<http://bpost.be/validationadresse> and <http://bpost.be/adressvalidatie>) , that a feedbackmessage (comment) can be sent to bpost regarding the outcome of the address validation by bpost on the submitted address. The same functionality is provided when calling the Address feedback service, where a feedback message can be provided (free text) by submitting this “free text” into the request indicating the Transaction ID (see above). Our system will than store the feedback that was provided on this address, and will be used for study and internal improvement purposes.

3. Warnings and Errors

A ‘validated address’ does not mean that the address is valid. It merely means that the address submitted has gone through the process of validation, whether the process was successful or not.

The webservice may return technical or functional messages.

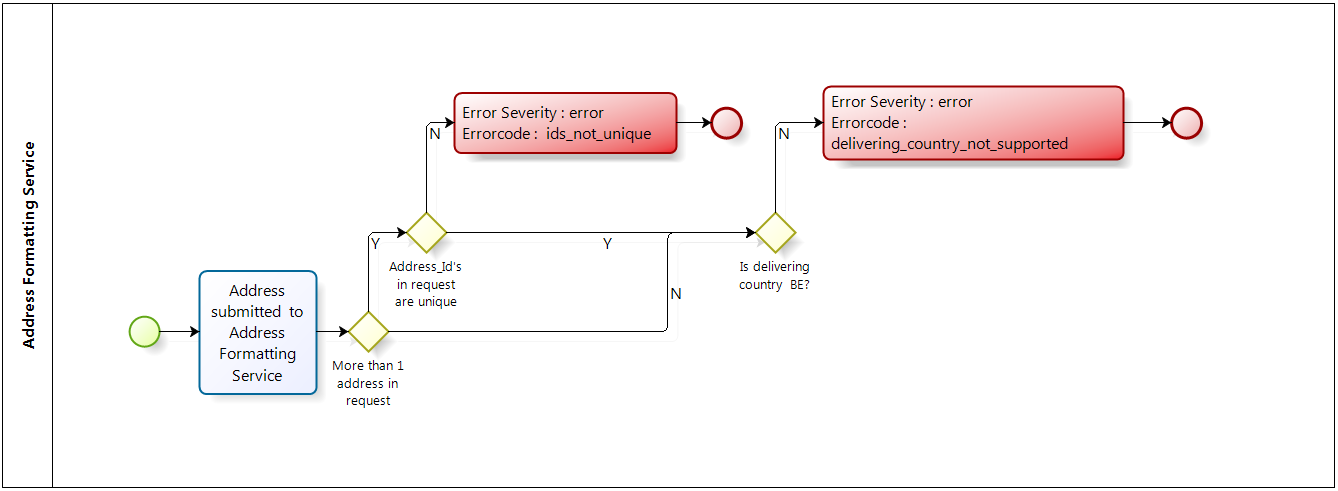
Severity ‘warning’ means that a match could be found even though anomalies were founds or some information was missing. The impacted component(s) is(are) identified and a corrected version of the address is provided in the feedback.

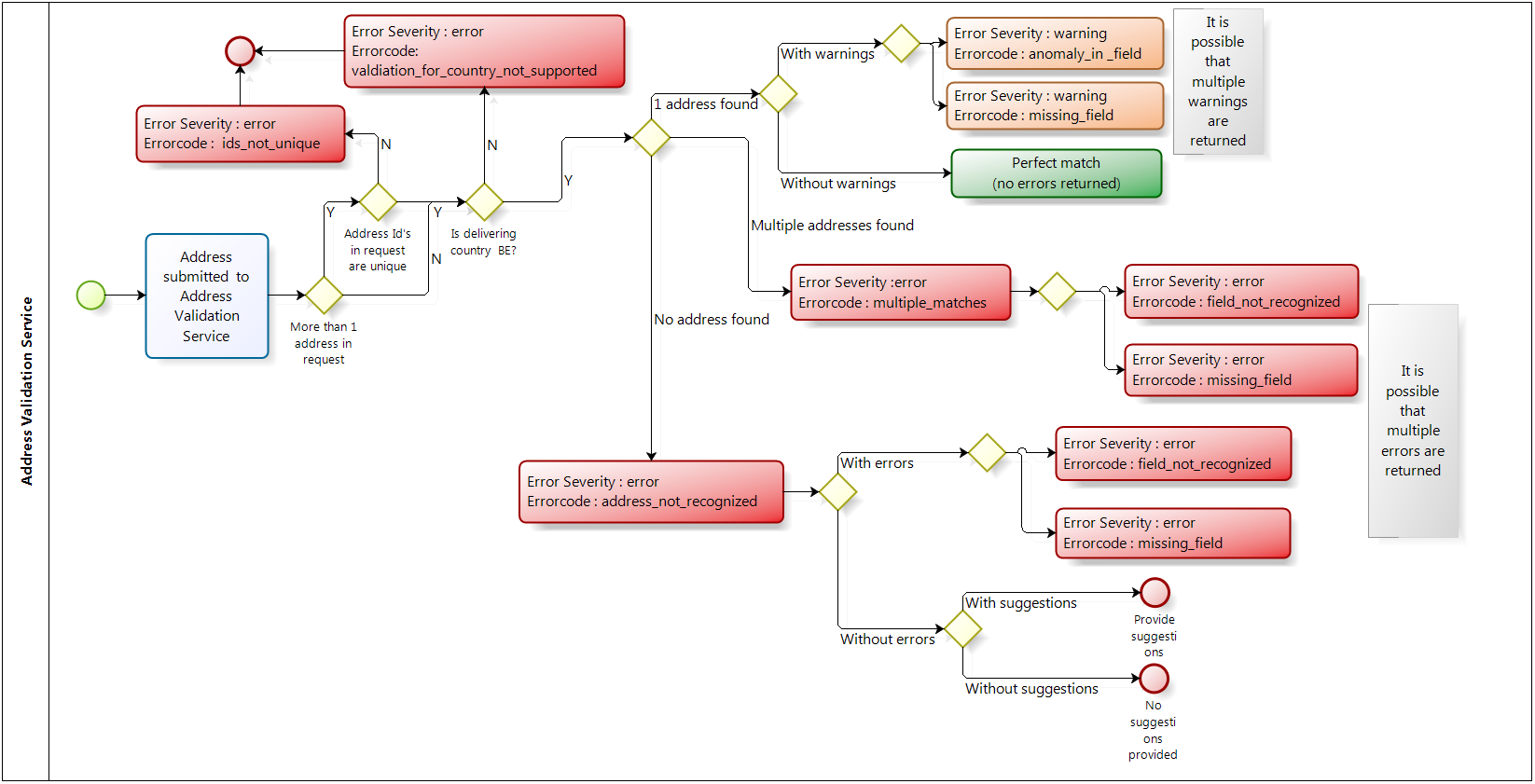
Severity ‘error’ means that issues in the input prevented the interpretation of the submitted address. If the cause of the error could be identified, the component is indicated.

The table below lists the different functional errors that may be returned as part of the webservice response:



And the following flow charts describe the message generation logic.





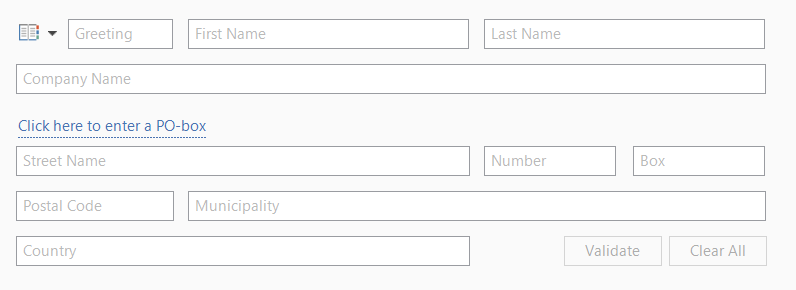
4. Use cases and example of implementation

This section shows a theoretical example of how to integrate the bpost address validation webservice within your business process, such as an e-commerce checkout or a personal information capture. The following uses cases and screen mockups are only indicative. Each implementation will require rigorous analysis to integrate the functions and fit your precise business needs.

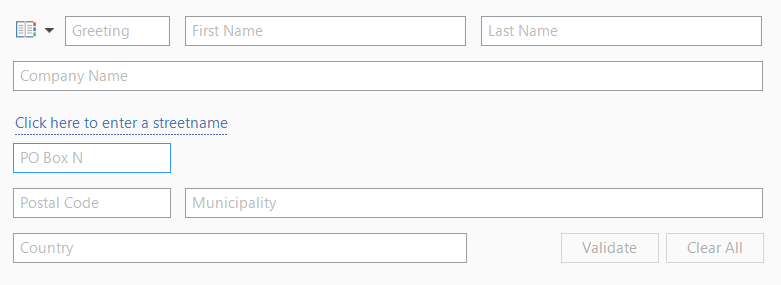
Capturing and submitting an address

To request bpost feedback on the address entered, you have first to capture it, then submit it for validation.

*USE CASE 1.1 – Capture a structured address with a streetname*



*USE CASE 1.2 – Capture a structured address with a PO box number*



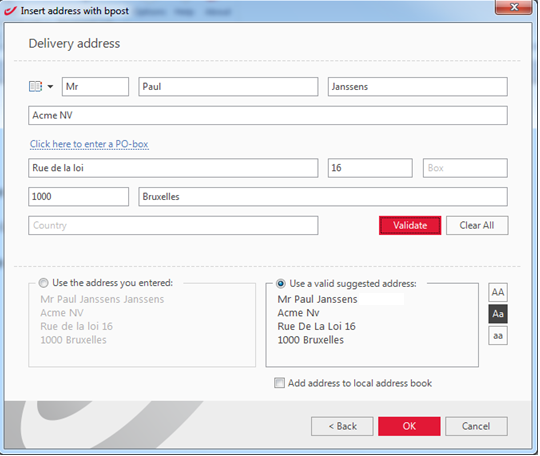
When the user clicks on the ‘Clear’ button, all input fields are emptied. When the user clicks on the the ‘Validate’ button, the input is mapped according to the specification herein (see the elements mapping table [here](#Elements_Mapping)) and submitted to the bpost webservice.

Receiving a response from the address validation & formatting webservice

The responses of the address validation webservice can be classified in one of the following use cases. The processflow and logic describing the use cases and the interactions with the system can be found at the end of this chapter.

*USE CASE 2.1 – The submitted address is valid, and no warning or error was detected*

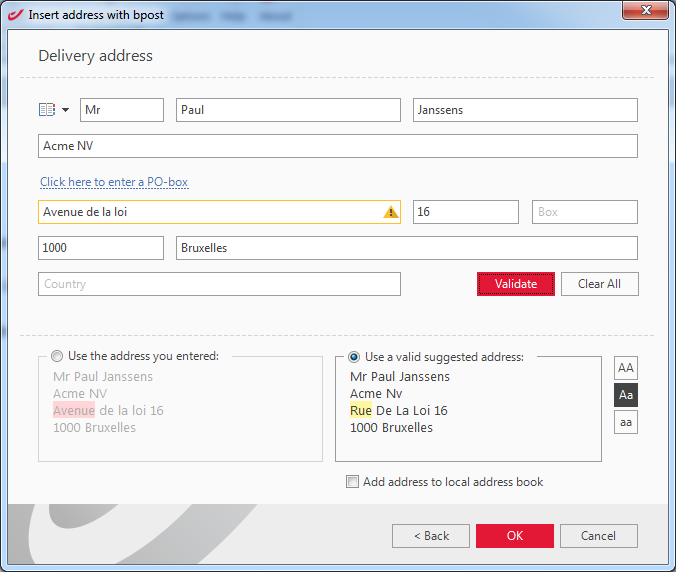
The system retrieves the webservice reponse and displays it to the user. The address submitted perfectly matches an address in the reference database of bpost.



The validated address returned by bpost will always be in UPPERCASE. You could foresee buttons to let the user reformat the returned address in UpperLower or in lowercase. You could also let the user choose to use his original address (possibly formatted through the formatting webservice), though this option defeats a little bit the purpose of using the validation service. But it provides maximum flexibility in case the user strongly disagrees with the output of the validation service.

*USE CASE 2.2 – The submitted address is valid, but one or more anomalies were detected.*

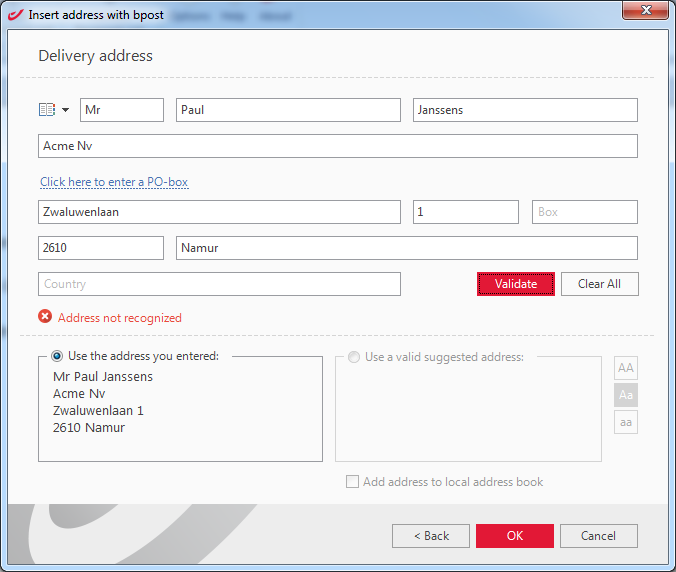
In this case, the bpost validation service was able to find a match for the submitted address, but proposes another version of the address. Anomalies could be found in one or more fields of the address, and it is recommended to use the bpost version when using the address on mail and parcels for optimal processing by bpost. It may be interesting to display a signal of a non-fatal anomaly (*, possibly with a tooltip providing more information and guidance) next to the appropriate field, as indicated by the webservice response, to draw the user’s attention to the fact that an anomaly was detected.



The validated address returned by bpost will always be in UPPERCASE. You could foresee buttons to let the user reformat the returned address in UpperLower or in lowercase. You could also let the user choose to use his original address (possibly formatted through the formatting webservice), though this option defeats a little bit the purpose of using the validation service. But it provides maximum flexibility in case the user strongly disagrees with the output of the validation service.

*USE CASE 2.3 – The submitted address cannot be recognized, and bpost has no alternative nor suggestion to propose.*

The submitted address is incoherent when compared to the addresses in bpost’s reference database. As a result, the service is not able to return the field(s) causing the confusion.

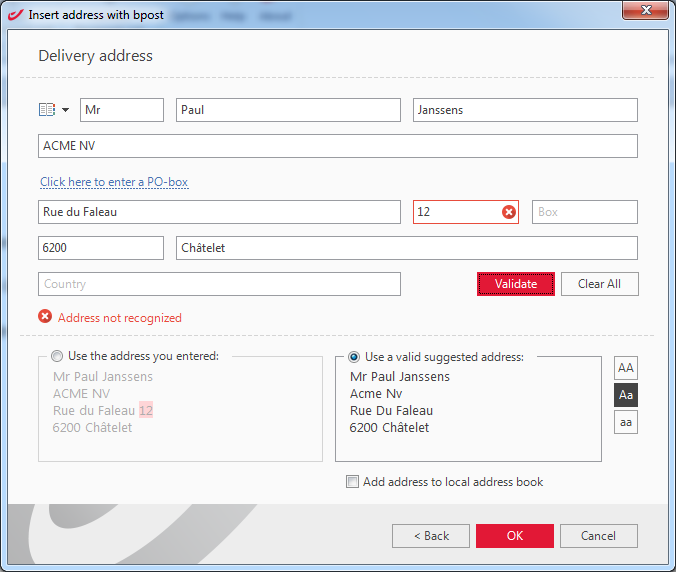


It is possible that the address is missing from bpost’s reference database, maybe because it is totally new. For such extreme cases, the system should still let the user choose to use the original address.

*USE CASE 2.4 – The submitted address cannot be matched fully*

The service can match part of the address, but not all of it.

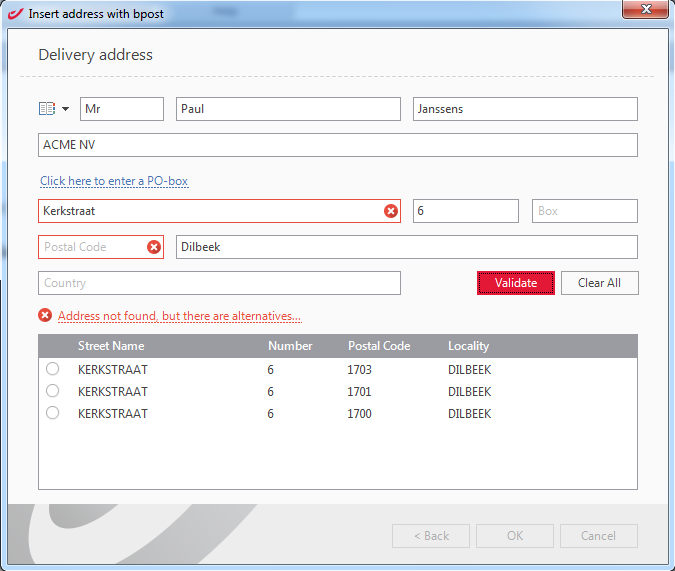
It may be interesting to display a signal of a fatal anomaly (* , possibly with more information and guidance) next to the appropriate field, as indicated by the webservice response, to draw the user’s attention to the fact that an anomaly was detected.

**

*USE CASE 2.5 – The submitted address can be matched with multiple valid addresses*

The issue here is that there is an ambiguity preventing the service from choosing one and only one valid solution for the submitted address. The input needs to be adapted by the user in order to clear this ambiguity.

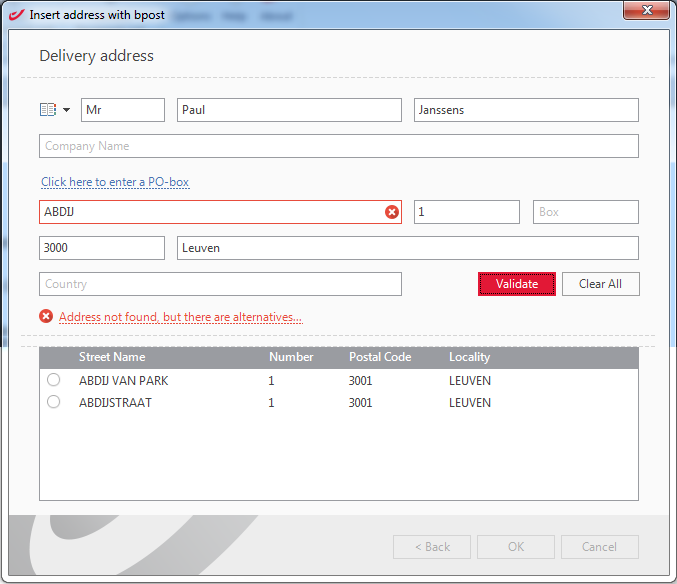
The service returns all these possible solutions. The system should display them to the user, so the appropriate one is chosen.



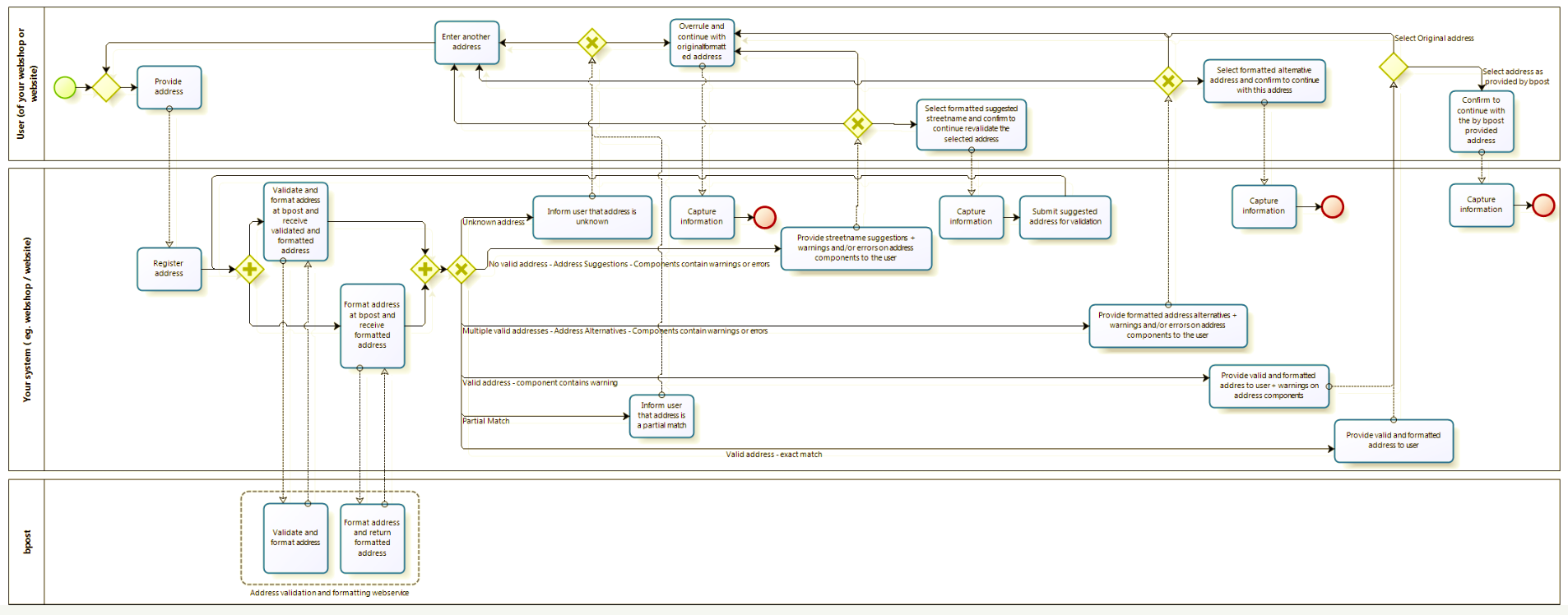
The validated address returned by bpost will always be in UPPERCASE. You could foresee buttons to let the user reformat the returned address in UpperLower or in lowercase. You could also let the user choose to use his original address (possibly formatted through the formatting webservice), though this option defeats a little bit the purpose of using the validation service. But it provides maximum flexibility in case the user strongly disagrees with the output of the validation service.

*USE CASE 2.6 – The submitted address cannot be recognized, but bpost found some suggestions to help transform the original address into a valid address*

The user may retry with one of the suggestions, or clear and enter a totally different address. The system goes back to a use case above.



Annex to chapter 6 – Processflow and logic describing the use cases and the interactions between the different actors:



5. Web Services

This chapter describes the Address Validation Web Services in detail, showing how to write XML code for the specific interface. The examples will show you how to use the required tags and how to fill in their values.

Rest Web Services Implementation

REST

A REST protocol is available for this service, and the documentation can be found in the ANNEX at the end of this document.

SOAP

SOAP, originally defined as Simple Object Access Protocol, is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks. It is used to expose Address Validation resources as services to the external parties of bpost.

Protocol

Web Services offered by Address Validation are then implemented by sending and/or receiving XML SOAP messages over the HTTP(s) Protocol.

Endpoint

The Address Formatting and Validation Web Service is accessible at the following URL:

* SOAP: <https://api.mailops.bpost.cloud/roa-info/externalMailingAddressProofingCS>
* REST Validation: <https://api.mailops.bpost.cloud/roa-info/externalMailingAddressProofingRest/validateAddresses>
* REST Formatting: <https://api.mailops.bpost.cloud/roa-info/externalMailingAddressProofingRest/formatAddresses>

This URL will be referenced as **<service-endpoint>** in the rest of the document.

Please contact [addressvalidation@bpost.be](mailto:addressvalidation@bpost.be) to get an API key for your application.

XML Validation

The structure of the XML request and response messages must be validated against a schema definition (XSD). The used XSD files are referenced in the provided WSDL.

For a request to be executed, the XML provided as input must be well formed and valid against the XSD.

JSON Schema Validation

The structure of the REST JSON request and response messages must be validated against a JSON Schema draft v4 (found in the ANNEX below).

Security

The Address Formatting and Validation service is an anonymous Web Service, which requires an API key for authentication.

Operations

When you want to format ou validate an address, you need to send the information to the server using the HTTP POST operation on the URI

Client REST Request

Use the REST client request to send to the server.

The following listing shows a request example:

{

"FormatAddressesRequest": {

"AddressToFormatList": {

"AddressToFormat": [

{

"@id": "1",

"MaileeAndAddressee": {

"StructuredAddresseeIndividualIdentification": {

"AddresseeFormOfAddress": "Mr",

"AddresseeGivenName": "Charles",

"AddresseeSurname": "Michel"

},

"StructuredMaileeOrganizationIdentification": {

"MaileeOrganizationOrganizationalName": "Federal Government"

}

},

"PostalAddress": {

"StructuredDeliveryPointLocation": {

"StreetName": {

"@locale": "fr",

"\*body": "Rue de la Loi"

},

"StreetNumber": "16",

"BoxNumber": "1"

},

"StructuredPostalCodeMunicipality": {

"PostalCode": "1000",

"MunicipalityName": {

"@locale": "fr",

"\*body": "Brussels"

} },

"CountryName": {

"@locale": "en",

"\*body": "BELGIUM"

}

},

"AddressLanguage": "nl",

"DispatchingCountryISOCode": "CN",

"DeliveringCountryISOCode": "BE"

}

]

},

"CallerIdentification": {

"CallerName": "China Post"

}

}

}

The request only contains the required parameters. The other parameters are described in the JSON Schema and in this document.

Server REST Response

If your request is successful, the server will respond with the following:

{"FormatAddressesResponse": {"FormattedAddressResultList": {"FormattedAddressResult": [{

"@id": "1",

"Label": {"Line": [

"Mr Charles Michel",

"Federal Government",

"Rue de la Loi 16 bus 1",

"1000 Brussels",

"BELGIUM"

]}

}]}}}

Client SOAP Request

Use the SOAP client request to send to the server.

The following listing shows a request example:

<!--copy paste all text below into the XML request which is send to the webservice:-->

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:v001="http://schema.bpost.be/services/common/address/ExternalMailingAddressProofingCSMessages/v001">

<soapenv:Header/>

<soapenv:Body>

<v001:ValidateAddressesRequest>

<v001:AddressToValidateList>

<!--1 to 100 repetitions:-->

<v001:AddressToValidate id="1">

<v001:MaileeAndAddressee>

<v001:AddresseeIndividualIdentification>

<v001:StructuredAddresseeIndividualIdentification>

<v001:AddresseeFormOfAddress>Mr</v001:AddresseeFormOfAddress>

<v001:AddresseeGivenName>Jean</v001:AddresseeGivenName>

<v001:AddresseeSurname>Dupont</v001:AddresseeSurname>

</v001:StructuredAddresseeIndividualIdentification>

</v001:AddresseeIndividualIdentification>

<v001:MaileeIndividualIdentification>

<v001:StructuredMaileeIndividualIdentification>

<v001:MaileeRole/>

</v001:StructuredMaileeIndividualIdentification>

</v001:MaileeIndividualIdentification>

<v001:MaileeOrganizationIdentification> <v001:StructuredMaileeOrganizationIdentification></v001:StructuredMaileeOrganizationIdentification>

</v001:MaileeOrganizationIdentification>

<v001:MailRecipientDispatchingInformation>

<v001:StructuredMailRecipientDispatchingInformation>

<v001:Wing></v001:Wing>

<v001:Stairwell></v001:Stairwell>

<v001:Floor></v001:Floor>

<v001:Door></v001:Door>

</v001:StructuredMailRecipientDispatchingInformation>

</v001:MailRecipientDispatchingInformation>

</v001:MaileeAndAddressee>

<v001:PostalAddress>

<v001:OtherDeliveryInformation> <v001:StructuredOtherDeliveryInformation></v001:StructuredOtherDeliveryInformation>

</v001:OtherDeliveryInformation>

<v001:DeliveryPointLocation>

<v001:StructuredDeliveryPointLocation>

<v001:StreetName>Avenue Brugmann</v001:StreetName>

<v001:StreetNumber>587</v001:StreetNumber>

<v001:BoxNumber>2</v001:BoxNumber>

</v001:StructuredDeliveryPointLocation>

</v001:DeliveryPointLocation>

<v001:PostalCodeMunicipality>

<v001:StructuredPostalCodeMunicipality>

<v001:PostalCode>1180</v001:PostalCode>

<v001:MunicipalityName>Uccle</v001:MunicipalityName>

</v001:StructuredPostalCodeMunicipality>

</v001:PostalCodeMunicipality>

</v001:PostalAddress>

<v001:DispatchingCountryISOCode>BE</v001:DispatchingCountryISOCode>

<v001:DeliveringCountryISOCode>BE</v001:DeliveringCountryISOCode>

</v001:AddressToValidate>

</v001:AddressToValidateList>

<v001:ValidateAddressOptions>

<v001:IncludeFormatting>true</v001:IncludeFormatting>

<v001:IncludeSuggestions>true</v001:IncludeSuggestions>

<v001:IncludeSubmittedAddress>false</v001:IncludeSubmittedAddress>

</v001:ValidateAddressOptions>

<v001:CallerIdentification>

<v001:CallerName>customername</v001:CallerName>

</v001:CallerIdentification>

</v001:ValidateAddressesRequest>

</soapenv:Body>

</soapenv:Envelope>

The request only contains the required parameters. The other parameters are described in the XSD and above in this document.

Server SOAP Response

If your request is successful, the server will respond with the following:

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

<soapenv:Body>

<msg:ValidateAddressesResponse xmlns:msg="http://schema.bpost.be/services/common/address/ExternalMailingAddressProofingCSMessages/v001">

<msg:ValidatedAddressResultList>

<msg:ValidatedAddressResult id="1">

<msg:MaileeAndAddressee>

<msg:AddresseeIndividualIdentification>

<msg:StructuredAddresseeIndividualIdentification>

<msg:AddresseeFormOfAddress>Mr</msg:AddresseeFormOfAddress>

<msg:AddresseeGivenName>Jean</msg:AddresseeGivenName>

<msg:AddresseeSurname>Dupont</msg:AddresseeSurname>

</msg:StructuredAddresseeIndividualIdentification>

</msg:AddresseeIndividualIdentification>

<msg:MaileeIndividualIdentification>

<msg:StructuredMaileeIndividualIdentification>

<msg:MaileeRole/>

</msg:StructuredMaileeIndividualIdentification>

</msg:MaileeIndividualIdentification>

</msg:MaileeAndAddressee>

<msg:ValidatedAddressList>

<msg:ValidatedAddress>

<msg:PostalAddress>

<msg:StructuredDeliveryPointLocation>

<msg:StreetName>AVENUE BRUGMANN</msg:StreetName>

<msg:StreetNumber>587</msg:StreetNumber>

<msg:BoxNumber>2</msg:BoxNumber>

</msg:StructuredDeliveryPointLocation>

<msg:StructuredPostalCodeMunicipality>

<msg:PostalCode>1180</msg:PostalCode>

<msg:MunicipalityName>UCCLE</msg:MunicipalityName>

</msg:StructuredPostalCodeMunicipality>

</msg:PostalAddress>

<msg:AddressLanguage>fr</msg:AddressLanguage>

<msg:Label>

<msg:Line>Mr Jean Dupont</msg:Line>

<msg:Line>AVENUE BRUGMANN 587 bte 2</msg:Line>

<msg:Line>1180 UCCLE</msg:Line>

</msg:Label>

</msg:ValidatedAddress>

</msg:ValidatedAddressList> <msg:DetectedInputAddressLanguage>fr</msg:DetectedInputAddressLanguage>

<msg:TransactionID>dc1a0434-d9d7-4378-92c6-bea82f9f81e4</msg:TransactionID>

</msg:ValidatedAddressResult>

</msg:ValidatedAddressResultList>

</msg:ValidateAddressesResponse>

</soapenv:Body>

</soapenv:Envelope>

### ANNEX

The following zip file includes the JSON Schema documentation (REST service) for the Address Validation webservice.



The following zip file includes the JSON Schema documentation (REST service) for the Address Formatting webservice.



OpenAPI specifications with the latest developments included have been created for the Validate Address REST service and can be found here:



For reference, here is the XSD, which can also be found in the WSDL



The following document provides the technical documentation for the XSD



6. Calling the webservice via SOAP UI

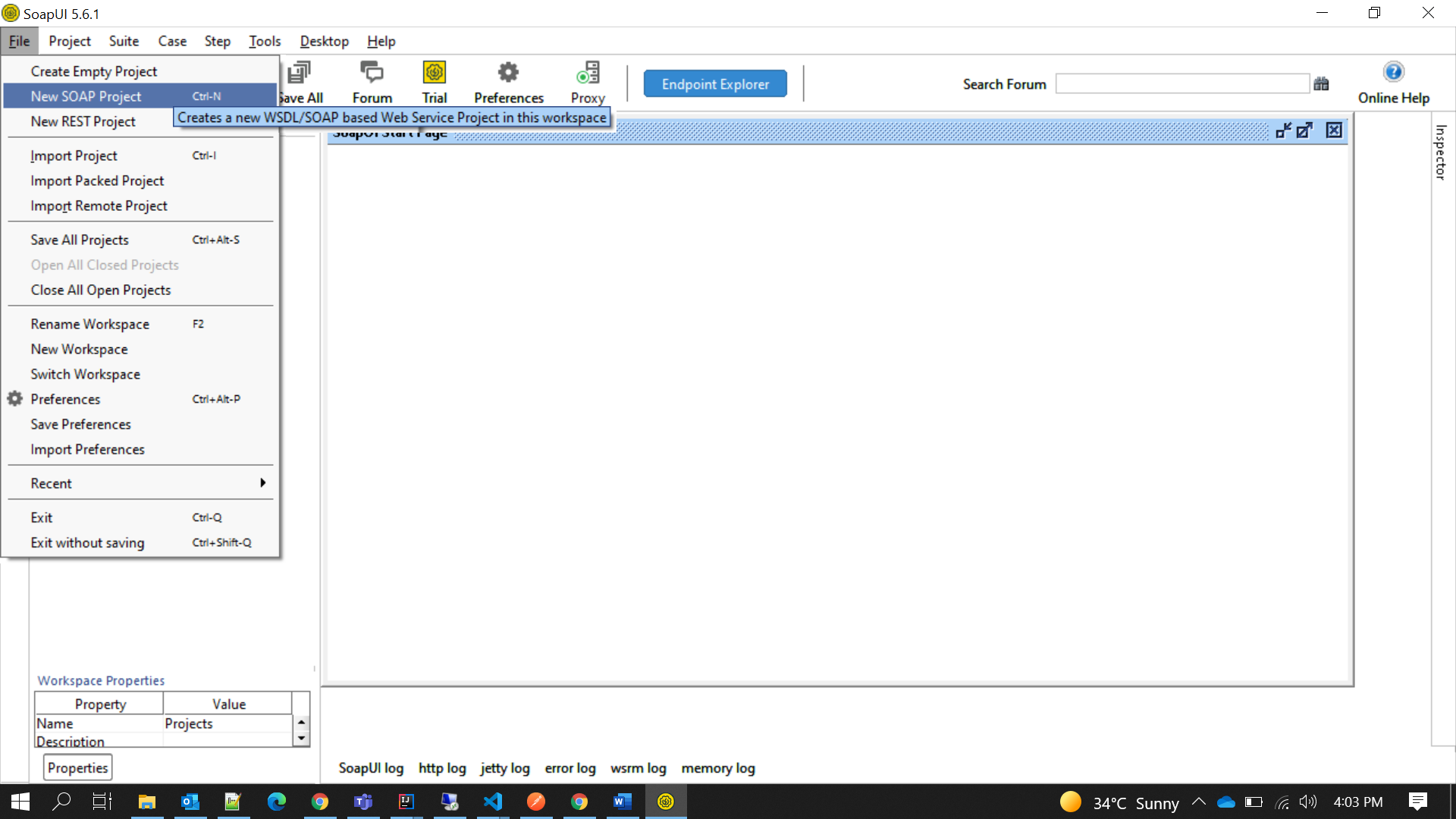
To facilitate testing and validation of the SOAP web services integration, it is possible to use a simple tool i.e. SOAP UI. SoapUI is a free and open source cross-platform Functional Testing solution. For more information on SoapUI we refer to: <http://www.soapui.org/About-SoapUI/what-is-soapui.html>

Download SoapUI from <http://sourceforge.net/projects/soapui/files/> and install it.

Creating a new project

Once you have successfully installed SoapUI you must create a new project as shown below:

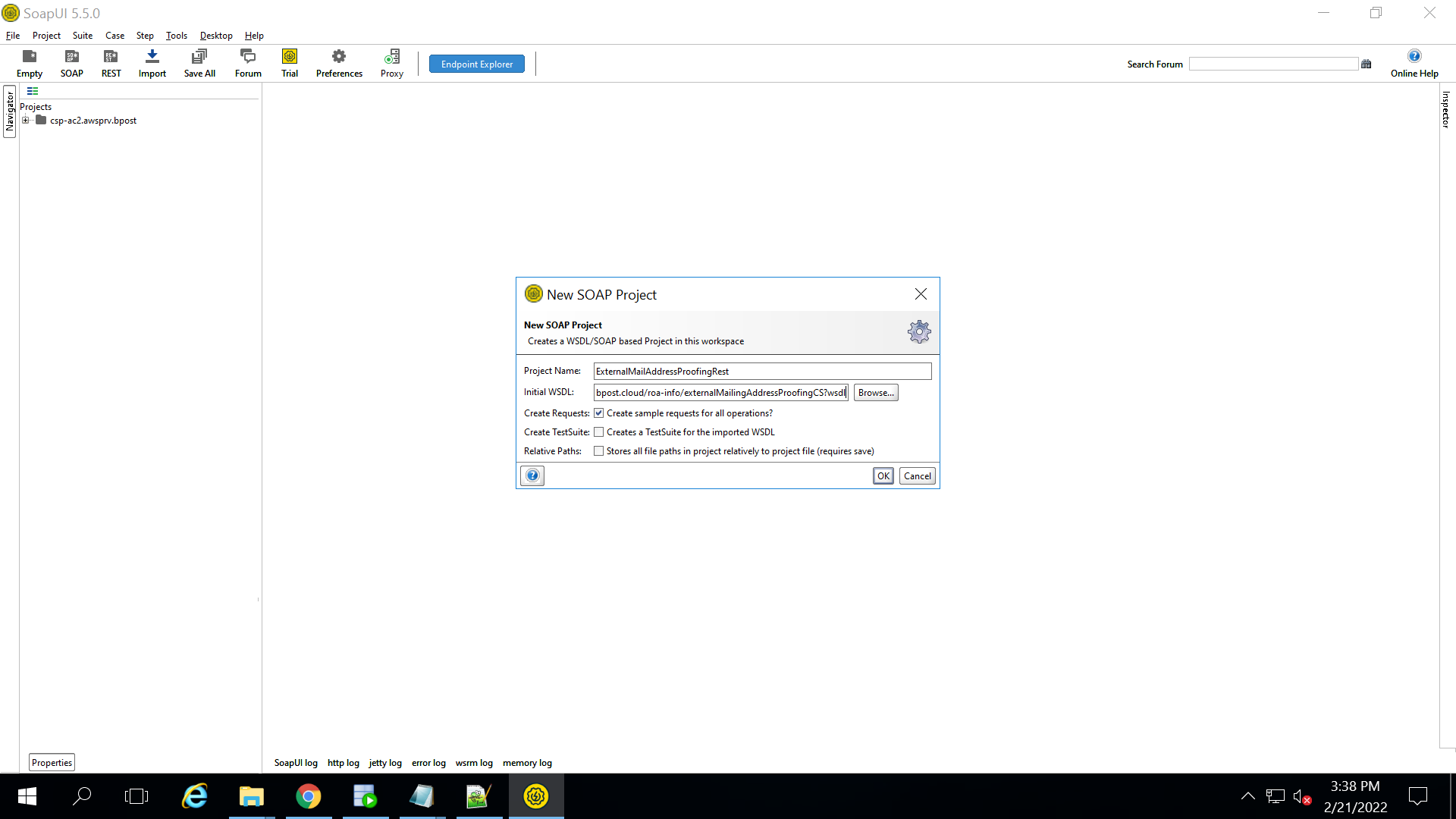
*1. Select : File\New SOAP Project Or you can just click on SOAP.*



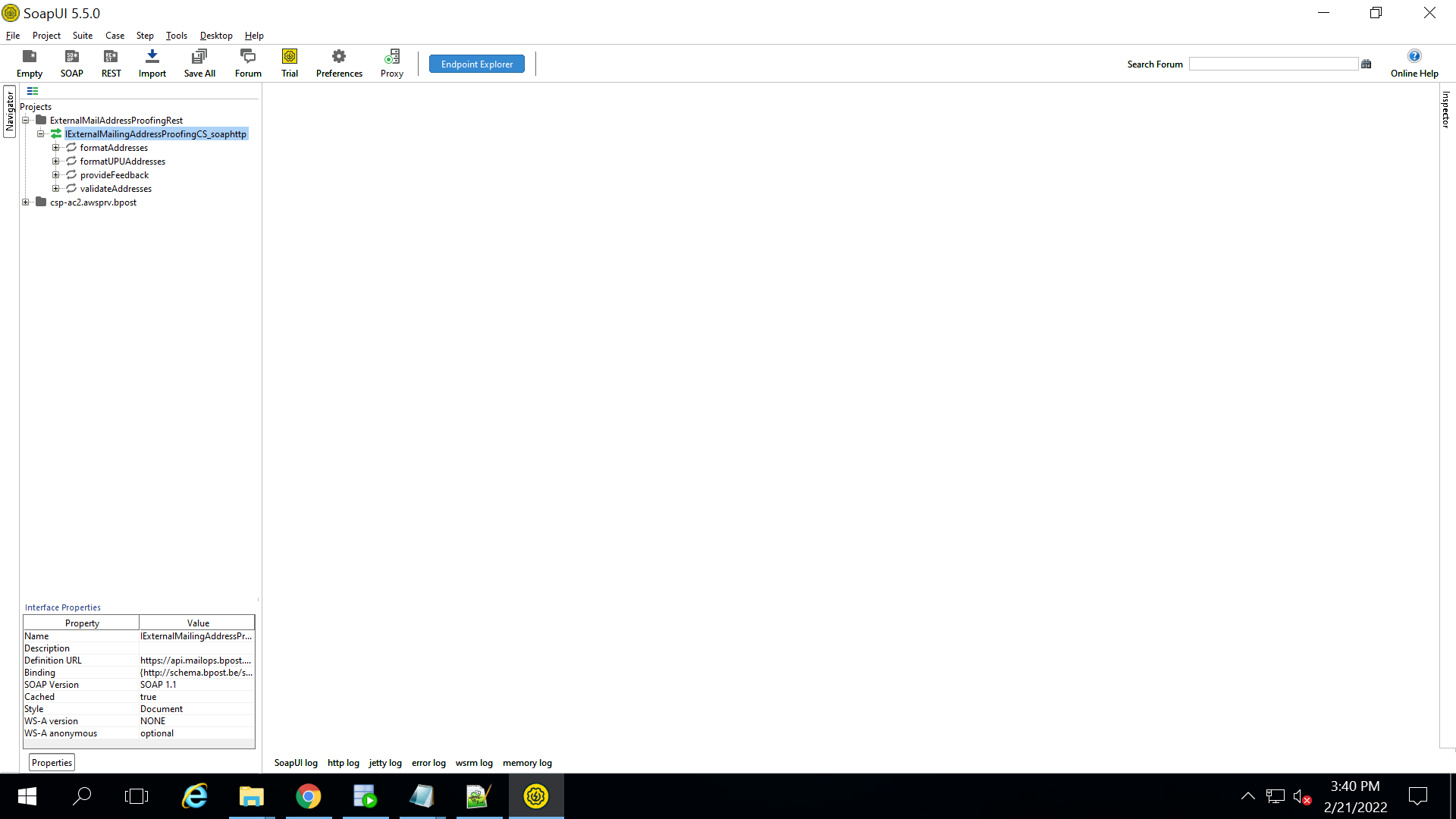
*2. Fill in a Project Name field(e.g. ‘bpost Address Validation Services’), and the ‘wsdl’ into the Initial WSDL/WADL field.*

***Production URL :***

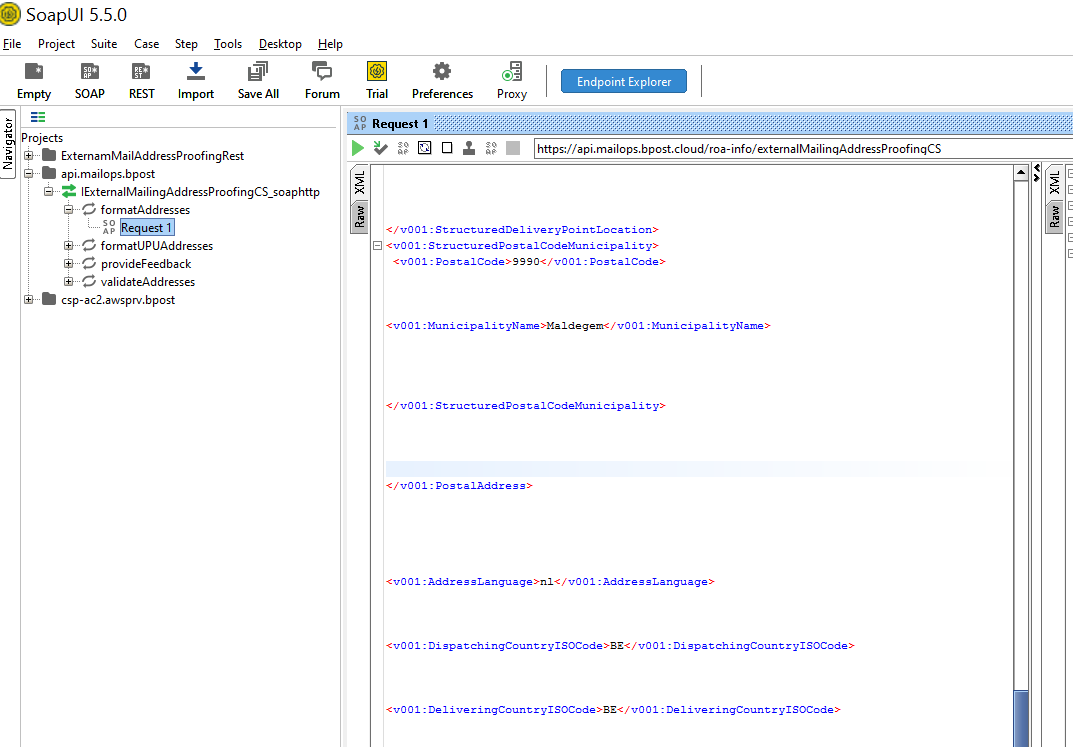
<https://api.mailops.bpost.cloud/roa-info/externalMailingAddressProofingCS?wsdl>



*3. After clicking OK in the previous step, and being connected to the Internet, the following structure will be created, and two sample requests will be generated.*



4. O*pen Request 1 within the formatAddresses service by double clicking on it, and maximize the window.*



! Verify that the information in the red rectangle is equal to the endpoint as mentioned in 3.2.If this is not the case, edit it by clicking on it.

(https://api.mailops.bpost.cloud/roa-info/externalMailingAddressProofingCS).

A sample request is filled with question marks in the place where a value should appear (mandatory = value / not mandatory = value *or* empty). Verify that a request you send out does not contain any question mark as value. You’ll find below how to generate a valid request.

The same is applicable for Request 1 in the validateAddresses webservice.

Generating a valid request

Here are 3 basic XML tools to help create testcases by generating the XML requestfile, for respectively:

1. formatAddress webservice call



2. validateAddress webservice call (structured input fields)  

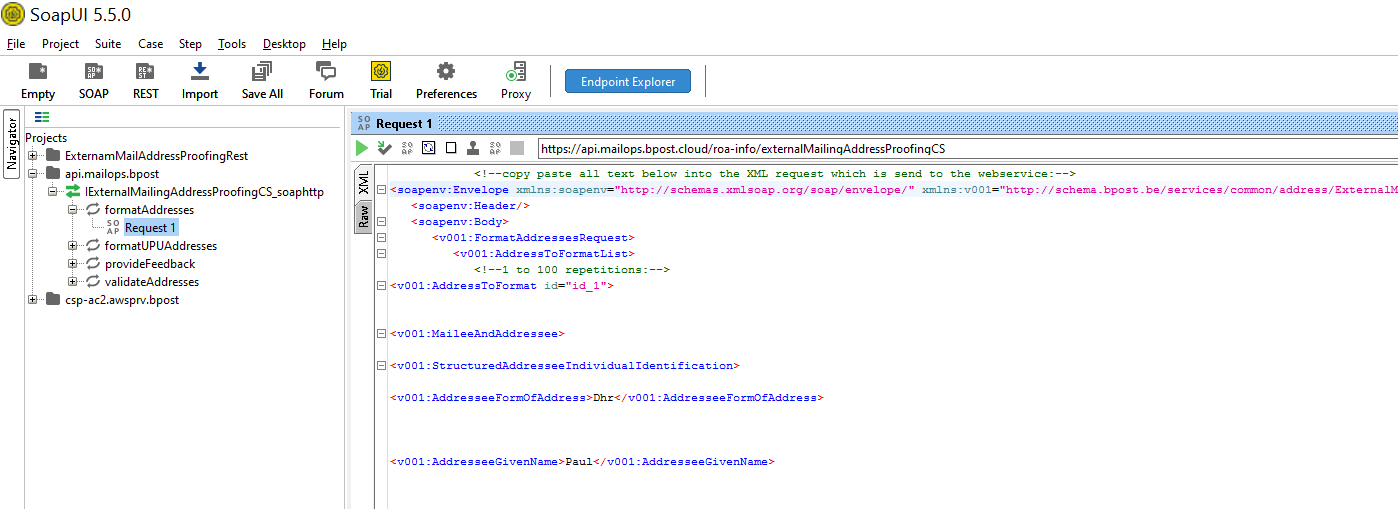

3. validateAddress webservice call (semi-structured input fields)



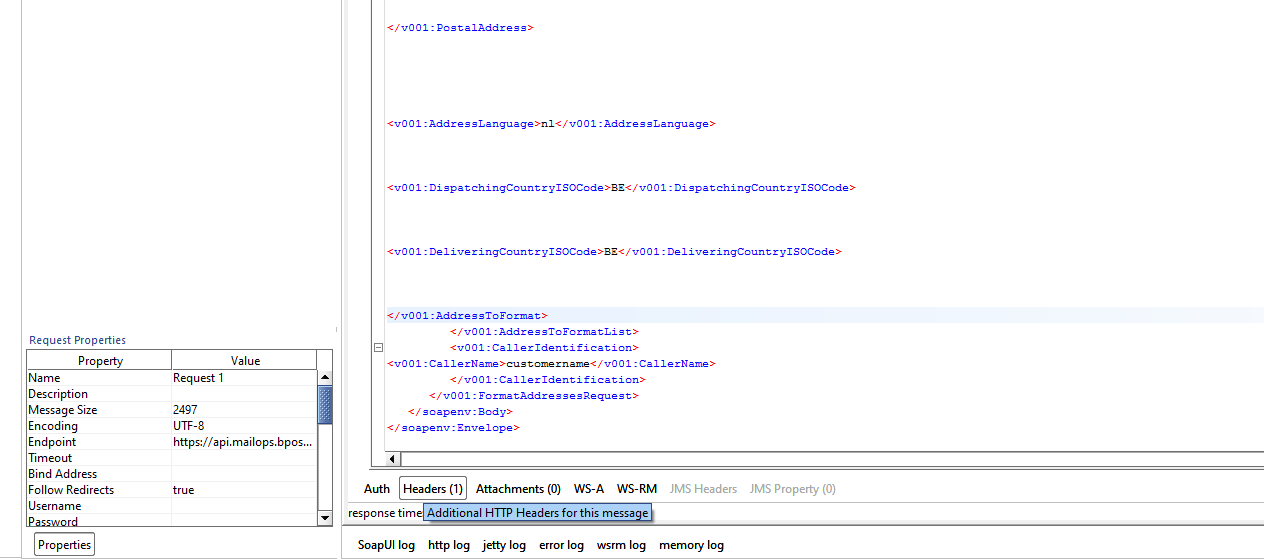
4. validateAddress webservice call (address block)



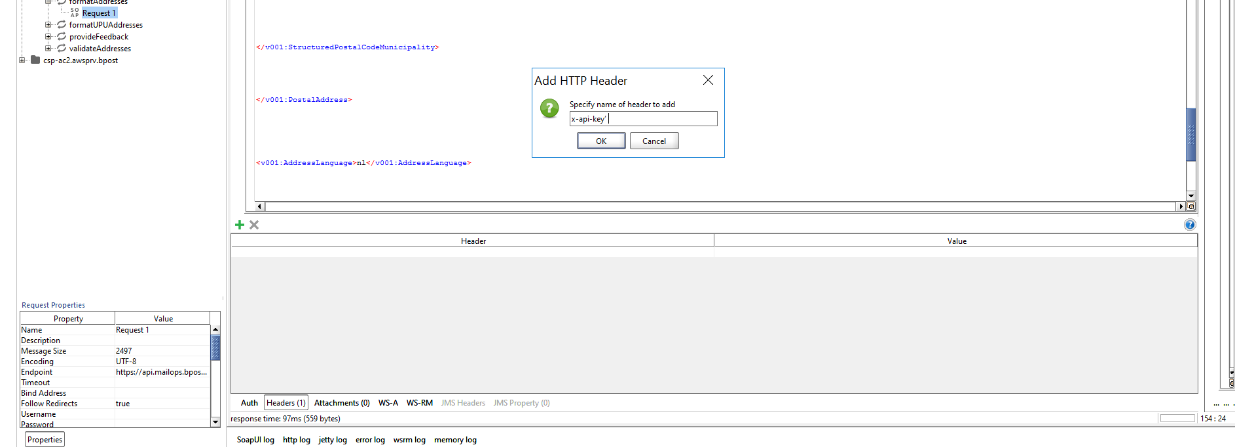
Just fill in the red fields as desired. Then click on the button *“copy XML to clipboard”* and paste into the left pane of the SoapUI request window in place of previous entry.

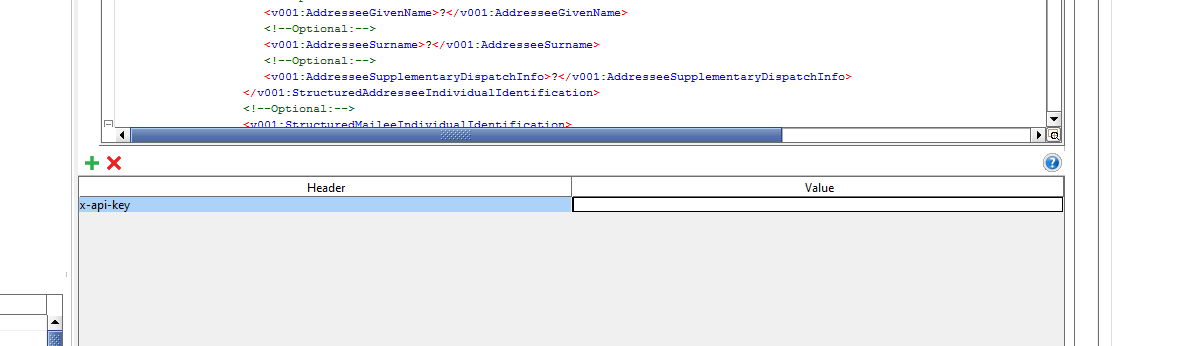


Click on Header located at the bottom of the page.



Click on ‘+’ Symbol and add ‘x-api-key’ in the box as shown below.

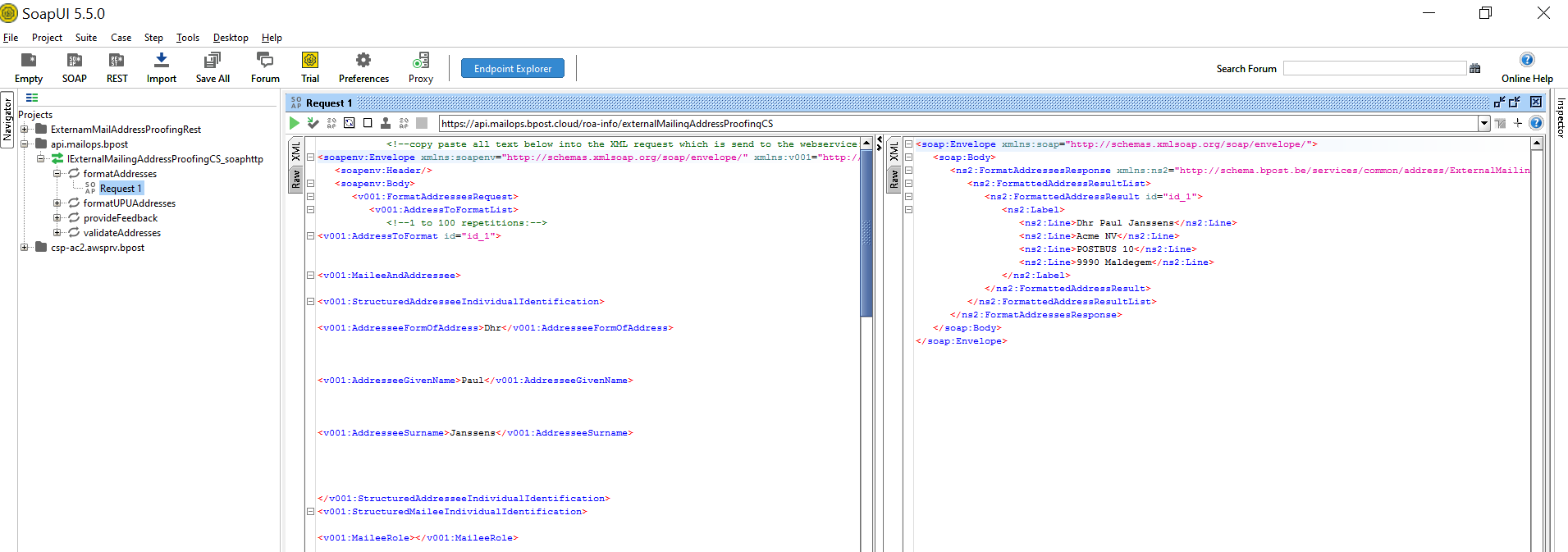




Validate the request against the XSD prior to submission, by pressing Alt+V

Then run the request by clicking the green arrow 

This should run the request across the Internet to the bpost system which will return a response such as:



1. The single-address process is optimized for time-critical applications and have a higher processing priority than batch submissions. If an abuse is detected on the single-address submissions (by submitting large numbers of consecutive single-addresses), the service may be shut down for the originating system. [↑](#footnote-ref-1)
2. The single-address process is optimized for time-critical applications and have a higher processing priority than batch submissions. If an abuse is detected on the single-address submissions (by submitting large numbers of consecutive single-addresses), the service may be shut down for the originating system. [↑](#footnote-ref-2)