Deploying the Address Verification REST Interface

Abstract

This document describes the architecture of the Informatica Address Verification REST interface and tells you how to deploy the interface in a networked environment.

Supported Versions

- Informatica Address Verification 5.10.0 and later

Table of Contents

Introduction

The Address Verification REST interface converts web service job requests into SOAP requests that Address Verification can process in the Informatica Cloud. Address Verification processes the request and returns the result in a SOAP response. The REST interface converts the SOAP response to a REST-style response to return the result to the user who submitted the request.

Use the interface to map the different types of address information in your data set to appropriate address elements in Address Verification. The interface parameters have the same default settings as the Address Verification engine, and the Address Verification output values, including status codes, match the outputs that the engine generates. You can use REST aliases to avoid conflicts when input and output elements have the same name.

The REST interface uses verbs to submit an address to Address Verification and to fetch the verified addresses. You can use JSON and XML data objects to organize the input and output addresses.

For more information about parameter definitions, see the Address Verification documentation.

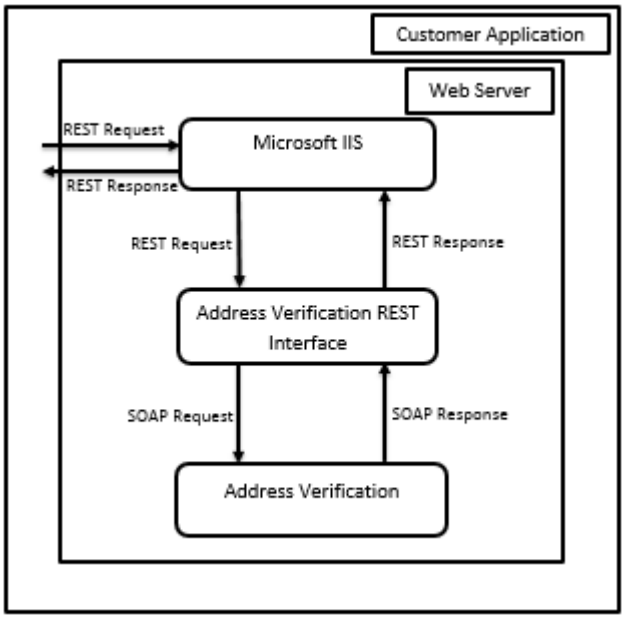
Address Verification REST Interface Deployment Architecture

You can deploy the REST interface on a single-server or in an environment that includes multiple servers.

Deploy the interface in a Windows environment. The Address Verification REST interface requires Windows Server, Microsoft IIS, and Microsoft .Net framework version 4.5.

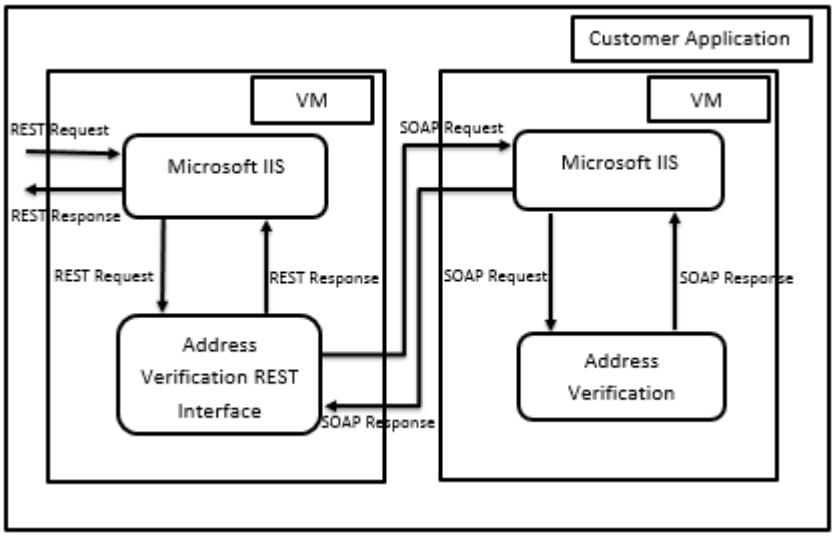
Architecture on a single server

The following image shows the architecture of the REST interface deployment on a single server:



Architecture on multiple servers

The following image shows the architecture of the REST interface deployment on multiple servers:



REST Interface Deployment

You can deploy the REST interface in an architecture that you define on a single server or on multiple servers.

To deploy the interface, complete the following steps:

- Install the interface files on a Microsoft IIS machine. You find the files in the `ADCloudRestAPI.zip` file.
- Configure Microsoft IIS to communicate with the interface.

Installing the REST Interface Files

To install the interface files, extract the `ADCloudRestAPI.zip` file to the Microsoft IIS machine.

1. Open **Windows Explorer** on the Microsoft IIS machine.
2. Create a directory that Microsoft IIS can read from, for example `C:\ADCloudRestAPI`.
3. Extract the `ADCloudRestAPI.zip` file into the directory that you created in step 2.

REST API File Set

The `ADCloudRestAPI.zip` file contains the `BIN` and the `Web References` directories.

The `BIN` directory contains the following files:

- `AddressDoctorRestAPI.dll`
- `AddressDoctorRestAPI.XmlSerializers.dll`
- `Log4net.dll`
- `NewtonsoftJson.dll`
- `System.Net.HTTP.Formatting.dll`
- `System.Web.Cors.dll`
- `System.Web.Http.dll`
- `System.Web.Http.Webhost.dll`

The `Web References` directory contains the following files and directory:

- `Global.asax`
- `Packages.config`
- `Web.config`
- `ADCloud` directory

  The `ADCloud` directory contains the `Reference.map` file and the `Response.datasource` file.

Note: The `NewtonsoftJson.dll` file is an open-source component and free to use.

Configuring Microsoft IIS for the REST Interface

You can host the REST API on the Microsoft IIS server.

1. Open **IIS Manager** and expand the `server` directory.
2. Right-click the `Sites` directory and select **Add Website**.

   The **Add Website** dialog box appears.
3. Configure the dialog box options.

The following table lists the options that you must configure in the **Add Website** dialog box:

Options	Sample Value	Description
Site name	`prod.adcloudrest.com`	Matches the host name.
Physical path	`C:\ADCloudRestApi`	The name of the directory to which you extracted the `ADCloudRestApi.zip` file.
Host name	`prod.adcloudrest.com`	The full name of the Microsoft IIS host machine.

Click **OK**.

4. Double-click the **Application Pools** tree view object.
5. Find and double-click the application pool entry for the site name that you created in step 3.

   The **Edit Application Pool** dialog box appears.
6. In the **Edit Application Pool** dialog box, verify the following options:
   - The Microsoft .Net CLR version is `4.0.30319` or later.
   - The managed pipeline mode is `Integrated`.

   Click **OK**.

Address Verification REST Interface Endpoint

The Address Verification REST interface exposes a single REST endpoint:

https://rest-adcloud.informaticadaas.com/adcloud/VerifyAddress

The endpoint supports the following verbs:

- GET. Use to validate a single address with the QueryString property.
- POST. Use to validate up to 1,000 addresses in a single request.

The following URL includes the endpoint and an address for verification:

https://rest-adcloud.informaticadaas.com/adcloud/VerifyAddress?login=102562&password=85H5F4Ra&street=15501%20Weston%20Pkwy&postalcode=27513&locality=Cary&province=NC&country=USA&format=xml

The REST API supports either JSON or XML output. Default is JSON.

The following table lists the parameters that you can set for each format:

Format	Parameter
JSON	`format=json`
XML	`format=xml`

GET and POST Verbs

Use the GET and POST verbs of the REST interface to submit an address to Address Verification and to verify the input addresses.

GET

You can use the GET verb to verify a single address.

A valid GET request requires the following parts:

- Credentials
- Input Address

Credentials

Enter the following credentials:

- Login. Your Address Verification login name.
- Password. Your Address Verification password.

Always specify the credentials with `?login=<userid>&password=<password>`.

Input Address

Specify the input address as parameter key value pairs.

For example, the following URL describes a fully-fielded GET request:

http://rest-adcloud.informaticadaas.com/adcloud/verifyaddress?login=<userid>&password=<password>&street=15501 Weston Pkwy&Locality=Cary&province=NC&postalcode=27513&country=usa

The following URL describes a fully-fielded GET request that requests XML output:

http://rest-adcloud.informaticadaas.com/adcloud/verifyaddress?login=<userid>&password=<password>&street=15501 Weston Pkwy&Locality=Cary&province=NC&postalcode=27513&country=usa&format=xml

POST

You can use the POST verb to submit up to 1,000 addresses at a time.

Consider the following guidelines for the POST verb:

- You cannot use the POST verb directly from a browser.

  The POST verb requires a JSON data object for input.
- The POST verb supports `application/json` and `application/xml` in the header, and the format that you select dictates the output type.
- When you submit a large number of addresses, the return objects can grow correspondingly large.
- The Boolean type uses the values *true*, *false*, *on*, and *off* to be consistent with Boolean standards and Address Verification definitions.
- Sample Angular JS Code is available for the POST verb. For information about the sample code, contact Informatica Global Customer Support.

REST Aliases

Use REST aliases to resolve naming conflicts, for example when an input and output parameter have the same name. REST aliases can also reduce the length of the REST requests.

Consider the following rules and guidelines for REST aliases:

- The REST aliases are applicable to the GET verb. The POST verb requires a data object, which eliminates the need for aliases.
- The primary use case for aliases is the Output option. The format of the alias is `AO.[element_name]`.

  For example, `AO.Organization` specifies the `Organization` output element, while the corresponding input is `Organization`.
- Key alias values are not case-sensitive.
- If the input and output parameters do not have naming conflicts, do not create aliases.

The REST interface supports the following shortcuts for the GET verb:

OutputOptions

Use to shorten the configuration of setting of multiple output options.

For example, `"OutputOptions=Organization,Contact,Building,SubBuilding,Street,HouseNumber,Locality"` will only return the Organization, Contact, Building, SubBuilding, Street, HouseNumber, and Locality output variables.

You can also specify the output variables as follows: `"AO.Organization=ON&AO.Contact=ON&AO.Building=ON&AO.Subbuilding=ON&AO.Street=ON&AO.HouseNumber=ON&AO.Locality=ON"`.

Enrichment

Use to shorten enrichment requests.

For example, you can use the shortcut `"enrichment=CAMEO,CASS,GEOCODING"` instead of `"Type=CAMEO&Type=CASS&Type=GEOCODING"`.

For more information about enrichments, see the Address Verification documentation.

Address Verification WSDL to REST API Element Mappings

The Address Verification web service elements maps to REST interface elements. In many cases, the web service element name and the corresponding REST interface element name are the same.

Process Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the Process object:

WSDL Element	REST Alias
Login	Login
Password	Password
ProcessMode	ProcessMode

ServiceParameters Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the ServiceParameters object:

WSDL Element	REST Alias
JobToken	JobToken
CampaignId	CampaignId
ReservedXml	ReservedXml
UseTransactionPool	UseTransactionPool

ValidationParameters Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the ValidationParameters object:

WSDL Element	REST Alias
FormatType	FormatType
FormatDelimiter	FormatDelimiter
DefaultCountryISO3	DefaultCountryISO3
ForceCountryISO3	ForceCountryISO3
CountryType	CountryType
CountryOfOriginISO3	CountryOfOriginISO3
StreetWithNumber	StreetWithNumber
FormatWithCountry	FormatWithCountry
ElementAbbreviation	ElementAbbreviation
PreferredScript	PreferredScript
PreferredLanguage	PreferredLanguage
AliasStreet	AliasStreet
AliasLocality	AliasLocality
GlobalCasing	GlobalCasing
GlobalMaxLength	GlobalMaxLength
GlobalPreferredDescriptor	GlobalPreferredDescriptor
MatchingScope	MatchingScope

WSDL Element	REST Alias
MaxResultCount	MaxResultCount
DualAddressPriority	DualAddressPriority
StandardizeInvalidAddresses	StandardizeInvalidAddresses
RangesToExpand	RangesToExpand
FlexibleRangeExpansion	FlexibleRangeExpansion
GeoCodingType	GeoCodingType
MatchingAlternatives	MatchingAlternatives
MatchingExtendedArchive	MatchingExtendedArchive
DisableCertifiedModeISO3	DisableCertifiedModeISO3
FormatAddressComplete	FormatAddressComplete
FormatMaxLines	FormatMaxLines

ValidationParameters.Standardizations Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the ValidationParameters.Standardizations object:

WSDL Element	REST Alias
Element	Element
Casing	Casing
MaxLength	MaxLength
MaxItemCount	MaxItemCount

ValidationParameters.AdditionalInformation Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the ValidationParameters.AdditionalInformation object:

WSDL Element	REST Alias
Name	Name
Value	Value

ValidationParameters.OutputOptions Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the ValidationParameters.OutputOptions object:

WSDL Element	REST Alias
RecordId	AO.RecordId
Organization	AO.Organization
Department	AO.Department
Contact	AO.Contact
Email	AO.Email
Building	AO.Building
SubBuilding	AO.SubBuilding
Street	AO.Street
HouseNumber	AO.HouseNumber
DeliveryService	AO.DeliveryService
Locality	AO.Locality
PreferredLocality	AO.PreferredLocality
PostalCode	AO.PostalCode
Province	AO.Province
Country	AO.Country
Residue	AO.Residue
RecipientLines	AO.RecipientLines
DeliveryAddressLines	AO.DeliveryAddressLines
CountrySpecificLocalityLine	AO.CountrySpecificLocalityLine
FormattedAddress	AO.FormattedAddress
AddressComplete	AO.AddressComplete
AddressDetailed	AO.AddressDetailed

Note: The arguments with AO.* indicate aliases.

Enrichments Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the Enrichments object:

WSDL Element	REST Alias
Type	Type

Address Object Mappings

The following table lists the Address Verification web service elements and the corresponding REST interface elements in the Address object:

WSDL Element	REST Alias
RecordId	RecordId
Organization	Organization
Department	Department
Contact	Contact
Email	Email
Building	Building
SubBuilding	SubBuilding
Street	Street
HouseNumber	HouseNumber
DeliveryService	DeliveryService
Locality	Locality
PreferredLocality	PreferredLocality
PostalCode	PostalCode
Province	Province
Country	Country
Residue	Residue
RecipientLines	RecipientLines
DeliveryAddressLines	DeliveryAddressLines
CountrySpecificLocalityLine	CountrySpecificLocalityLine
FormattedAddress	FormattedAddress

WSDL Element	REST Alias
AddressComplete	AddressComplete
AddressDetailed	AddressDetailed

POST Data Object structure

You must understand the data structure to configure the data elements efficiently.

Full JSON data object structure for POST

Use the following data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters:
   {
      ProcessMode: "",
      ValidationParameters:
      {
         FormatType: "",
         FormatDelimiter: "",
         DefaultCountryISO3: "",
         ForceCountryISO3: "",
         CountryType: "",
         CountryOfOriginISO3: "",
         StreetWithNumber: <true/false/yes/no>,
         FormatWithCountry: <true/false/yes/no>,
         ElementAbbreviation: <true/false/yes/no>,
         PreferredScript: "",
         PreferredLanguage: "",
         AliasStreet: "",
         AliasLocality: "",
         GlobalCasing: "",
         GlobalMaxLength: <number>,
         GlobalPreferredDescriptor: "",
         MatchingScope: "",
         MaxResultCount: <number>,
         DualAddressPriority: "",
         StandardizeInvalidAddresses: <true/false/yes/no>,
         RangesToExpand: "",
         FlexibleRangeExpansion: <true/false/yes/no>,
         GeoCodingType: "",
         MatchingAlternatives: "",
         MatchingExtendedArchive: <true/false/yes/no>,
         DisableCertifiedModeISO3: "",
         FormatAddressComplete: "",
         FormatMaxLines: <number>,
         Standardizations:
         [
            {
               Element: "",
               Casing: "",
               MaxLength: <number>,
               MaxItemCount:  <number>
            }
         ],
         AdditionalInformationSet:
         [
            {
               Name: ""
```

```
            }
         ],
   OutputOptions:
         {
            RecordId: "",
            Organization: "",
            Department: "",
            Contact: "",
            Email: "",
            Building: "",
            SubBuilding: "",
            Street: "",
            HouseNumber: "",
            DeliveryService: "",
            Locality: "",
            PreferredLocality: "",
            PostalCode: "",
            Province: "",
            Country: "",
            Residue: "",
            RecipientLines: "",
            DeliveryAddressLines: "",
            CountrySpecificLocalityLine: "",
            FormattedAddress: "",
            AddressComplete: "",
            AddressDetailed: "",
         },
      },
   },
   enrichments:
   [
      {
         Type: ""
      }
   ],
   addresses:
   [
      {
         RecordId: [""],
         Organization: [""],
         Department: [""],
         Contact: [""],
         Email: [""],
         Building: [""],
         SubBuilding: [""],
         Street: [""],
         HouseNumber: [""],
         DeliveryService: [""],
         Locality: [""],
         PreferredLocality: [""],
         PostalCode: [""],
         Province: [""],
         Country: [""],
         Residue: [""],
         RecipientLines: [""],
         DeliveryAddressLines: "",
         CountrySpecificLocalityLine: [""],
         FormattedAddress: [""],
         AdditionalAddressInformation: [""],
         AddressComplete: [""],
         AddressCode: [""],
         AddressDetailed: [""],
      }
   ]
};
```

Minimum JSON data object structure for POST - Fully Fielded Input

Use the following data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters: {ProcessMode: ""}
   addresses:
   [
      {
         Street: [""],
         Locality: [""],
         PostalCode: [""],
         Province: [""],
         Country: [""],
      }
   ]
};
```

The following example shows the single-address data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters: {ProcessMode: ""}
   addresses:
   [
      {
         Street: ["501 W Williams St"],
         Locality: ["Apex"],
         PostalCode: ["27502"],
         Province: ["NC"],
         Country: ["USA"],
      }
   ]
};
```

The following example shows the multiple-address data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters: {ProcessMode: ""}
   addresses:
   [
      {
         Street: ["501 W Williams St"],
         Locality: ["Apex"],
         PostalCode: ["27502"],
         Province: ["NC"],
         Country: ["USA"],
                        },
                        {
                           Street: ["15501 Weston Parkway", "Suite 150"],
                           Locality: ["Cary"],
                           PostalCode: ["27513"],
                           Province: ["NC"],
                           Country: ["USA"],
                        },
                       {
                          Street: ["2100 Seaport Blvd"],
                          Locality: ["Redwood City"],
                          PostalCode: ["94063"],
                          Province: ["CA"],
                           Country: ["USA"],
      }
```

```
   ]
};
```

Minimum JSON data object structure for POST - Unfielded Input

Use the following data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters: {ProcessMode: ""}
   addresses:
   [
      {
         FormattedAddress: [""],
         Country: [""],       }
   ]
};
```

The following example shows the single-address data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters: {ProcessMode: ""}
   addresses:
   [
      {
         FormattedAddress: ["501 W Williams St", "Apex", "NC", "27502"],
         Country: ["USA"],
       }
  ]
};
```

The following example shows the multiple-address data structure for the POST verb:

```
var dataObj =
{
   login: "<your-login>",
   password: "<your-password>",
   parameters: {ProcessMode: ""}
   addresses:
   [
      {
         FormattedAddress: ["501 W Williams St", "Apex", "NC", "27502"],
         Country: ["USA"],
      },
      {
         FormattedAddress: ["15501 Weston Parkway", "Suite 150", "Cary", "NC", "27513"],
         Country: ["USA"],
      },
      {
         FormattedAddress: ["2100 Seaport Blvd", "Redwood City", "CA", "94063"],
         Country: ["USA"],
       }
  ]
};
```

Author

Shahani Natalia Mendonca
Documentation Trainee

Acknowledgements

The author would like to thank Daniel Paner and David Handy for their assistance.