Web Services Implementation Guide

Table of Contents

1 Introduction

1.1 What are Web Services?

TowerData's Real-Time Data Services are based on web service software technology, which is a collection of protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to easily exchange data over computer networks like the Internet.

TowerData Real-Time Email Validation and Telephone Validation Web Services can be integrated seamlessly into your system regardless of the platform you use. What's more, it can be done without any costly upgrades or time consuming installations!

1.2 How Do Web Services Work?

TowerData Web Services work very simply. You send us a processing request and we do all the work! TowerData will process your request by cleaning and standardizing the data, validating and verifying the information, and appending additional information as needed. We then return it all back to you in the same format in which you sent us your request.

All communication between TowerData and yourself is done in Real-Time and independent of the platform you use. TowerData Web Services make this possible by using XML and SOAP, two open standards that are platform independent and vendor neutral:

SOAP (Simple Object Access Protocol) is a protocol for exchanging XML-based messages over a computer network.

XML (Extensible Markup Language) Is a flexible way to create common information formats and share both the format and the data on the World Wide Web, intranets, and elsewhere. XML is a formal recommendation from the World Wide Web Consortium (W3C).

The data for your Web Service request is formatted in XML and sent to TowerData using SOAP. TowerData processes the request and creates an XML response which is returned to you in a SOAP object. An example of a Request

and Response can be found in Appendix A: Sample Request and Appendix B: Sample Response.

1.3 Why Should I use Web Services?

TowerData's Real-Time Telephone and Email Web Services can benefit you in numerous ways including reducing costs, eliminating lost opportunities, and increasing communication with customers. TowerData Web Services are fast and reliable and are easy to integrate into your existing systems. A few key benefits of TowerData's Web Services include:

Reduce Costs
TowerData's Real-Time Web Services will save you time and money by providing the ability to verify information as it is received. The costs of post-collection data correction can be eliminated.

Eliminate Lost Opportunities
Bounced emails and invalid phone numbers are missed opportunities to communicate with new clients as well as a costly effort, particularly if the results are fruitless. With TowerData's Email and Telephone Web Services you will no longer have to worry about these lost opportunities.

Increase Communication
TowerData's Web Services will increase the ability of your customer service call centers and sales force to communicate with both new customers and existing customers, which will result in increased conversion rates and higher customer satisfaction levels.

Fast and Reliable
TowerData's Web Services are backed by a cluster of load-balanced servers. The technology and hardware ensures that we provide you with fast and efficient results with a 99.9% uptime.

Easy to Integrate
TowerData's Web Services will accommodate you regardless of what system and platform you have in place. And with no software installation or costly upgrades required, you can quickly begin using the service.

1.4 Getting Started

The first step in using TowerData's Email or Telephone Validation & Verification Web Services is to register at https://client.towerdata.com/reg/register.cgi. The

email address and password you register with will serve as your Web Services UserID and Password. This information must be submitted along with each web service request.

2 TowerData Web Services

2.1 Web Services

TowerData provides both Email and Telephone Validation Web Services. These services can be used independently of each other in separate requests or together in single requests. Below is a brief description of the services and the information you can gain from the service.

2.2 Email Validation

The Email Validation Web Service will detect general format and syntax errors in email addresses, validate email address domains, and verify if those domains can receive emails. The Email Validation Web Service will:

- Ensure addresses use a valid top-level domain (e.g. .com, .net, .biz, etc.).
- Detect improper email address formats for common domains such as Hotmail and AOL.
- Verify whether the domains of email addresses exist
- Confirm that the domains can receive email
- Determine whether the mailbox can receive email at the domain.

2.3 Telephone Validation

The Telephone Validation Web Service will verify telephone numbers ensuring that they are not only syntactically correct but that they are also valid for all telephone numbers in the United States, Canada and 15 other countries. The Telephone Validation service will:

- Verify if the Area Code is valid and whether the Exchange, the first three digits of a seven digit number, is valid for that Area Code.
- Provide detailed regional information for the Telephone number. Including: Time Zones, Day Light Saving times, Country, State, County, City, Longitude, and Latitude.
- Recognize and provide new Area Codes for numbers which are incorrectly stating old Area Codes for their region

3 Web Service Requests

3.1 Request Tags

Every Web Service request is submitted using XML. Below is an example of a Web Service request with a single Email and Telephone record.

```
<Envelope>
 <Body>
  <Validate test="true" password="password" version="1.2" login="DoeJohn">
   <Parameters>
    <email validationLevel="2" correct="true" maxCorrect="3" />
    <phone detailLevel="2" />
   </Parameters>
   <Records>
     <Record rid="1">
       <email>JohnDoe@towerdata.com</email>
     </Record>
     <Record rid="2">
       <phone>
         <number>866-377-3630</number>
       </phone>
     </Record>
   </Records>
  </Validate>
 </Body>
</Envelope>
```

Every request that is submitted to TowerData must contain an Envelope tag, Body tag, and then a Validate tag.

```
<Validate test="true" password="password" version="1.2" login="DoeJohn">
```

The Validate tag will contain four attributes:

Attribute	Description
Login	The username you used when registering.
Password	The password you used when registering.
Test	Takes the value true if you wish to validate the XML strictly. False otherwise. TowerData recommends you set this value to true.
Version	The version of the TowerData WSDL that you are using.

Nested within the Validation tag will be a Parameters tag.

```
<Parameters>
   <email validationLevel="2" correct="true" maxCorrect="3" />
   <phone detailLevel="2" />
   <timeout>10</timeout>
</Parameters>
```

The Parameters tag is where you will set any parameters for the web services that you are invoking in this request. These parameters are optional and if the XML request does not include them the Web Service will use the default values. These tags are described further in sections 3.2 Email Request Tags, 3.3 Telephone Request Tags, and 3.4 Other Request Tags.

After the Parameters tag you will then include the records you want processed. Each of these records should be contained within a set of Record tags and all Record tags should be contained within a Parent Records tag.

```
<Records>
  <Record rid="1">
     <email>JohnDoe@TowerData.com</email>
  </Record>
</Records>
```

Each Record tag can have an optional RID attribute, which is a unique identifier for that record. If you do not assign an RID then TowerData will do so automatically. Nested within each Record tag will be the data for either an email request (in an email tag), telephone request (in a phone tag), or both. You may include any number of Record Tags in a request but only one email or phone tag within a record tag. Below is an example of how you can include multiple records.

```
<Records>
  <Record rid="1">
     <email>JohnDoe@TowerData.com</email>
  </Record>
  <Record rid="2">
     <phone>
        <number>866-377-3630</number>
     </phone>
  </Record>
  <Record rid="3">
     <email>JaneDoe@TowerData.com</email>
     <phone>
        <number>(866)377-3630</number>
     </phone>
  </Record>
</Records>
```

After the final Record tag and the closing Records tag, the closing Validate tag will be followed by the closing Body and Envelope Tags. A full request example can be viewed in Appendix A: Sample Request.

3.2 Email Request Tags

3.2.1 Email Tags within Parameter Tag

```
<Parameters>
    <email validationLevel="2" correct="true" maxCorrect="3" />
</Parameters>
```

The Email tag within the Parameters tag contains the three attributes; Correct, MaxCorrect, and Validation Level.

Attribute	Description
Correct	TowerData can attempt to provide alternative correct email addresses if the one provided is determined to be invalid. Specify True to use this feature.
MaxCorrect	Specify the number of correct email addresses to return if above attribute is set to True.
ValidationLevel	Specifies the level of validation to perform on the Email records submitted.

There are 5 levels of validation listed as follows:

Level	Description
1	Check email syntax only
2	Check email syntax and check whether the domain is in the Domain Database (described below)
3	Check email syntax and the domain in the Domain Database. If the domain is not in the Domain Database, then check in real-time if the domain exists
4	Check email syntax and the domain in the Domain Database. If the domain is not in the Domain Database, then check in real-time if the domain exists and if it can receive email
5	Mailbox check – In addition to the checks above, does the user exist at that domain.

TowerData's proprietary Domain Database determines with certainty whether a domain is alive or dead. Other solutions will check the existence of a domain in real-time, which is prone to machine or network failures. Only TowerData checks the status of a domain multiple times over several days to make a conclusive evaluation. For greatest accuracy and speed, rely on the Domain Database to determine if a domain is valid or not.

3.2.2 Email Tags within Record Tags

If you have subscribed to the Email Validation Web Service, then you may include Email records within the Record tags.

```
<email>JohnDoe@TowerData.com</email>
```

The Email tag will contain within it the email address that you wish to be processed.

3.3 Telephone Request Tags

3.3.1 Telephone Tags within Parameter Tag

```
<Parameters>
    <phone detailLevel="2" />
</Parameters>
```

The Phone tag within the Parameters tag contains the single attribute, DetailLevel. This attribute specifies the amount of information to be returned for a valid telephone number. There are three different detail levels. Each level will return a:

- Standardized Telephone Number
- Standardized Extension (if applicable)
- Validation Message (and warnings if applicable)

Each detail level contains the information in the level above it. The three detail levels are defined as follows:

Level	Description	Return Fields
1	Verification & Correction	• New Area Code
2	Time	• Time Zone • Daylight Savings Time
3	Location	• Country • State • City • Longitude • Latitude • County

3.3.2 Telephone Tags within Record Tags

If you have subscribed to the Telephone Validation Web Service, then you may include Phone records within the Record tags.

```
<phone>
    <number>873-928-3983</number>
    <extension>4567</extension>
</phone>
```

The Phone tag will have nested within it a Number Tag and an optional Extension Tag.

Tag	Description
Number	A complete telephone number including area code. The format of the number is irrelevant, as TowerData will standardize the information for you, however the area code should appear first.
Extension (optional)	An extension number for the telephone number. Alternatively the extension can also be included at the end of the Telephone number and be included within the Number Tag. Example below.

```
<phone>
    <number>873-928-3983x4567</number>
</phone>
```

A sample of the Response output can be viewed in Appendix B: Sample Response.

3.4 Other Request Tags

An optional timeout tag may be specified within the Parameters tag which can take a floating point value. This value specifies the maximum number of seconds the service should spend processing a record.

```
<Parameters>
    <timeout>5.5</timeout>
</Parameters>
```

If the number of seconds specified in the timeout elapses while processing a record, the service will stop processing and return a timeout status code for the information that was not completely validated.

4 Web Service Responses

Every Web Service response to a request will be sent enclosed within an Envelope and Body Tag. Nested within the Body tag will be a ValidateResponse tag and within that a Records tag. Inside the Records tag there will be a Record tag for each record that was submitted to TowerData. Each of these records can be identified by the RID either supplied by you or generated by TowerData.

4.1 Email Response

For an email record the result information will be returned within an Email tag. Below is an example of a successful email response.

```
<email ok="true" validationLevel="2">
    <address>JohnDoe@TowerData.com</address>
    <username>JohnDoe</username>
    <domain>TowerData.com</domain>
    <status code="10">Syntax OK</status>
    <corrections />
</email>
```

The Email tag will contain two attributes: Ok and ValidationLevel.

Attribute	Description
Ok	This will take the value of 'true' if the email was valid and 'false' if it was not.
ValidationLevel	This is the level of validation actually performed on the email record and may be different than the level requested.

Nested within the Email tags will be the following tags: Address, Username, Domain, Status Code, and Corrections.

Tag	Description
Address	The full email address.
Username	The username portion of the email address.
Domain	The email address domain.
Status Code	The status code contains an attribute, a numeric Status Code, and the tag holds the corresponding status message for the validation performed. For a detailed list of Status Codes refer to Appendix C: Email Status Codes & Messages
Corrections	For an incorrect email address, when possible, an array of Address tags with possible correct email address will be returned nested within the Corrections tag. Only present if requested in the request parameters.

Here is another example, this time of an unsuccessful response.

```
<email ok="false" validationLevel="2">
    <address>John;Doe@hot_mail.com</address>
    <username />
    <domain />
    <status code="115">Invalid domain syntax</status>
    <corrections>
        <address cid="1">John.Doe@hotmail.com</address>
        <address cid="2">JohnDoe@hotmail.com</address>
    </corrections>
</email>
```

4.2 Telephone Response

For a telephone record the result information will be returned within a Phone tag.

```
<phone ok="true" detailLevel="1">
    <number>4012107829</phone>
    <extension />
    <status code="10">Successfully Parsed and Standardized, Area Code and Exchange
    Match</status>
    <messages>
        <status code="140">Extension is greater than 4 digits in length</status>
    </messages>
</phone>
```

The Phone tag will contain two attributes: Ok and DetailLevel.

Attribute	Description
Ok	This will take the value of 'true' if the telephone number was valid and 'false' if it was not.
DetailLevel	This is the requested amount of information you want returned.

Nested within the Phone tag will be the following tags:

Tag	Description
Number	The standardized telephone number.
Extension	The standardized extension, if applicable.
Status Code	The status code contains an attribute, a numeric Status Code, and the tag holds the corresponding status message for the validation check. For a detailed list of Status Codes refer to Appendix D: Email Status Codes & Messages
Messages	The Message tag may contain nested within it more status code tags with

	further information, for example, information on a parsed extension number submitted.
new_npa	The new area code for the Area Code / Exchange combination. During the permissive dialing period, the old Area Code may still be displayed and returned as valid.
All Above fields are included in Detail Level 1.	
Time Zone	The time zone the telephone number belongs in.
observes_dst	The day light savings time. 1 = Observes DST (Daylight Saving Time) during the normal DST observance period, 0 = Does not observe DST at all.
All Above fields are included in Detail Level 2.	
Country	Two character International Standard ISO 3166-1 Country Code for the Area Code / Exchange Wire Center location. US - United States CA - Canada BS - Bahamas BB - Barbados AI - Anguilla AG - Antigua and Barbuda VG - Virgin Islands, British KY - Cayman Islands BM - Bermuda GD - Grenada TC - Turks and Caicos Islands MS - Montserrat LC - Saint Lucia DM - Dominica VC - Saint Vincent and the Grenadines DO - Dominican Republic TT - Trinidad and Tobago KN - Saint Kitts and Nevis JM - Jamaica
State	Two letter USPS state abbreviation (US, Canada and US territories) for the Area Code / Exchange Wire Center location.
County	County name of the Area Code / Exchange Wire Center.
City	Name or general location of the Area Code / Exchange Wire Center.
Latitude	Latitude in decimal degrees of the center of the Area Code / Exchange Wire Center.
Longitude	Longitude in decimal degrees of the center of the Area Code / Exchange Wire Center.
All Above fields are included in Detail Level 3.	

Here is another example, this time of an unsuccessful response.

```
<phone ok="false" detailLevel="2">
    <number />
    <extension />
    <status code="130">Error - Invalid Telephone number – too many digits</status>
</phone>
```

5 Appendix A: Sample Request

```
<Envelope>
   <Body>
      <Validate test="true" password="password" version="1.2" login="DoeJohn">
         <Parameters>
            <email validationLevel="2" correct="true" maxCorrect="3" />
            <phone detailLevel="2" />
         </Parameters>
         <Records>
           <Record rid="1">
              <email>johndoe@yahoo.com</email>
           </Record>
           <Record rid="2">
              <email>johndoe@hotmail</email>
              <phone>
                 <number>4012107829ext78123</number>
              </phone>
           </Record>
         </Records>
      </Validate>
   </Body>
</Envelope>
```

6 Appendix B: Sample Response

```
<Envelope>
 <Body>
  <ValidateResponse>
   <Records>
    <Record rid="1">
     <email ok="true" validationLevel="2">
      <address>johndoe@yahoo.com</address>
      <username>johndoe</username>
      <domain>yahoo.com</domain>
      <status code="10">Syntax OK</status>
      <corrections />
     </email>
    </Record>
    <Record rid="2">
     <email ok="false" validationLevel="2">
      <address>johndoe@hotmail</address>
      <username />
      <domain />
      <status code="115">Invalid domain syntax</status>
      <corrections>
        <address cid="1">johndoe@hotmail.com</address>
        <address cid="2"> johndoe@hotmail.net</address>
        <address cid="3"> johndoe@hotmail.org</address>
      </corrections>
     </email>
     <phone ok="true" detailLevel="2">
      <number>4012107829</phone>
      <extension />
      <status code="10">Successfully Parsed and Standardized, Area Code and Exchange
Match</status>
      <messages>
       <status code="140">Extension is greater than 4 digits in length</status>
      </messages>
      <timezone>-5</timezone>
      <observes_dst>1</observes_dst>
      <new_npa />
     </phone>
    </Record>
   </Records>
  </ValidateResponse>
 </Body>
</Envelope>
```

7 Appendix C: Email Status Codes & Messages

Code	Description
5	Validation Timeout
10	Syntax OK
20	Syntax OK and domain valid according to the domain database
30	Syntax OK and domain exists
40	Syntax OK, domain exists, and domain can receive email
50	Syntax OK, domain exists, and mailbox does not reject mail
100	General syntax error
110	Invalid character in address
115	Invalid domain syntax
120	Invalid username syntax
125	Invalid username syntax for that domain
130	Address is too long
135	Address has unbalanced parentheses, brackets, or quotes
140	Address does not have a username
145	Address does not have a domain
150	Address does not have an @ sign
155	Address has more than one @ sign
200	Invalid top-level-domain (TLD) in address
210	Domain is an invalid IP address
215	Unquoted spaces are not allowed in email addresses
310	Domain does not exist
315	Domain does not have a valid IP address
325	Domain can not receive email
400	The mailbox is invalid or the username does not exist at the domain
410	Mailbox is full and can not receive email at this time
420	Mail is not accepted for this domain
500	Addresses with that username are not allowed
505	Addresses with that domain are not allowed
510	The email address is suppressed and not allowed

8 Appendix D: Phone Status Codes & Messages

Code	Description
5	Validation Timeout
10	Successfully Parsed and Standardized, Area Code and Exchange Match
100	Successfully Parsed and Standardized, Valid Area Code, Invalid Exchange for specified Area Code
110	Successfully Parsed and Standardized, Invalid Area Code and Exchange
120	Error - Invalid Telephone number – too few digits.
130	Error - Invalid Telephone number – too many digits.
140	Warning – Extension is greater than 4 digits in length.
150	Warning – Toll free number
160	Warning – 900 toll number

9 Appendix E: Web Service Fault Codes

Code	Description
600	Bad Request
610	Unauthorized
615	Blocked IP
620	Method Not Allowed
700	Internal Server Error
710	Service Unavailable
720	Interface Version Not Supported
730	Configuration File Error
800	Error Connecting to the Database
810	Error Logging