Overview

This technical implementation guide provides information and instructions on consuming the COA Validation Web Service. The main purpose of this web service is to provide Yale system and application owners a way in which to validate Chart of Accounts (COA) segments. Access to and use of the web service will require approval and access (via a service account).

Who can consume the Web Service?

COA Validation service can be consumed by:

- Yale Internal Clients (System and Applications).
- External vendors and B2B trading partners.

Basic authentication will be utilized as the security mechanism that will help enforce access control to the web service/resources. Consumers will need to pass and be authenticated by a username and password. In addition, external vendors/users to Yale will need to provide their invoking server/machine IP address so that firewall access can be granted before consuming the web service.

Following are the steps that will need to be taken, in order to obtain access to, and to successfully invoke the COA Validation Service:

1. An e-mail (in a specific template) to be submitted to fss@yale.edu with a cc: to your ISR PoC requesting access and use of the COA Validation Service. The Finance Support Services team will review the request and approve (or deny with follow-up questions).
2. Once approved a Service now ticket will be created by Finance Support Services team (on requestor’s behalf) to the BSG Finance ITS group. Once the ticket is completed, requestor will be provided with access information.
   a. For any new applications / systems, a service account (following standardized naming conventions) will need to be created as part of this request and will reside in Active Directory. A Service Now subtask will be initiated to ITS Helpdesk for the account to be created by Client Accounts. A 2-3 day turnaround time should be expected.

Note: Web service access will not be granted to Individual user Net IDs

Request/Response Formats

The COA Validation Web Service can be invoked via the following protocols and formats:

- Representational State Transfer (REST) (** Preferred Option **)  
  - REST permits many different data formats. Formats supported will include XML or JSON.
  - Additional information as to the use of REST services include
    - REST allows better support for browser clients due to it’s support for JSON.
Implementation Guide: COA Validator Web Service - DRAFT

- REST has better performance and scalability.
- REST reads can be cached, SOAP based reads cannot be cached.

- Simple Object Access Protocol (SOAP).
  - To support different transport protocols, such as HTTP and SMTP
  - Response Format will be XML

Data Availability for the Web Service

Due to the volumes projected against the COA Validation Service, the validation being performed will be going against a COA Integration Hub (SQL DB). Unfortunately, performance concerns were a primary reason that the service was not designed to go against Workday directly. Currently the COA integration hub is being updated once per day (early AM). All efforts are to ensure that the COA Validation Service is available for consumption by service consumers 24x7.

Defaulting of COA Segments

The COA Validation service will only validate the segments that are provided as input to the service. If a partial COA segment structure is provided, the Validator service will not default to the full COA segment structure. However, certain COA segments were configured in Workday to have default worktags (or segments), these will be returned in the response. For example, inputting ‘Grant code’ in the request will return default segments, ‘Cost Center’, ‘Program’ and ‘Project’ in the response.

Note: It is possible that even though a COA segment structure is validated by the service, Workday might do additional validations and consider the COA structure invalid.

Web Service Contract

Below is the link to the COA Validator web service contract. The document has information on request/response details, validation errors etc. for the web service.


Restful Service Implementation
URL (Test Environment) | https://gw-tst.its.yale.edu/rest/coavalidator
---|---
Methods | GET

**QueryString**
- Transaction_Date=
- Company_code=
- Grant_code=
- Gift_code=
- Yale_Designated_code=
- Cost_Center_code=
- Program_code=
- Project_code=
- Assignee_code=
- Location_code=
- Revenue_Category_code=
- Spend_Category_code=
- Ledger_Account_code=

**Returns**
- XML
- (JSON – **Work in progress**)

---

**Below is an example of a request/response that has no validation errors:**

**Sample Request (no validation errors):**


**Sample Response (no validation errors):**

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
      <Transaction_Date>12/01/2016</Transaction_Date>
      <Transaction_Date_Message>W004 – Current date assumed</Transaction_Date_Message>
      <Company>CO01</Company>
      <Company_Description>Yale University</Company_Description>
      <Company_Message xsi:nil="true"/>
      <Grant xsi:nil="true"/>
    </ser-root:validateCOAResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
Below is an example of a request/response that has validation errors:

For help contact: Service Now request to Finance and Procure-to-Pay Systems/Financial Systems

Last Revised: 11/29/2016
Sample REST Request (with validation errors):

https://gw-tst.its.yale.edu/rest/coavalidator?Company_Code=CO01

Sample REST Response (with validation errors):

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
      <Transaction_Date>12/01/2016</Transaction_Date>
      <Transaction_Date_Message>W004 – Current date assumed</Transaction_Date_Message>
      <Company xsi:nil="true"/>
      <Company_Description xsi:nil="true"/>
      <Company_Message xsi:nil="true"/>
      <Grant xsi:nil="true"/>
      <Grant_Description xsi:nil="true"/>
      <Grant_Message xsi:nil="true"/>
      <Gift xsi:nil="true"/>
      <Gift_Description xsi:nil="true"/>
      <Gift_Message xsi:nil="true"/>
      <Yale_Designated xsi:nil="true"/>
      <Yale_Designated_Description xsi:nil="true"/>
      <Yale_Designated_Message xsi:nil="true"/>
      <Fund xsi:nil="true"/>
      <Fund_Description xsi:nil="true"/>
      <Cost_Center xsi:nil="true"/>
      <Cost_Center_Description xsi:nil="true"/>
      <Cost_Center_Message xsi:nil="true"/>
      <Program xsi:nil="true"/>
      <Program_Description xsi:nil="true"/>
      <Program_Message xsi:nil="true"/>
      <Project xsi:nil="true"/>
      <Project_Description xsi:nil="true"/>
      <Project_Message xsi:nil="true"/>
      <Assignee xsi:nil="true"/>
      <Assignee_Description xsi:nil="true"/>
      <Assignee_Message xsi:nil="true"/>
      <Location xsi:nil="true"/>
      <Location_Description xsi:nil="true"/>
      <Location_Message xsi:nil="true"/>
      <Revenue_Category xsi:nil="true"/>
      <Revenue_Category_Description xsi:nil="true"/>
    </ser-root:validateCOAResponse>
  </soapenv:Body>
</soapenv:Envelope>
SOAP Service Implementation

Web Service Definition (WSDL)

The web service definition (WSDL) to invoke COA Validator service (In Test Environment) is below:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="validateCOA" targetNamespace="esb/YaleWDFDM.ws:validateCOA"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:tns="esb/YaleWDFDM.ws:validateCOA"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
<wsdl:types>
<xsd:schema targetNamespace="esb/YaleWDFDM.ws:validateCOA"
xmlns:tns="esb/YaleWDFDM.ws:validateCOA" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="validateCOA" type="tns:validateCOA"/>
  <xsd:element name="validateCOAResponse" type="tns:validateCOAResponse"/>
  <xsd:complexType name="validateCOA">
    <xsd:sequence>
      <xsd:element name="Mode" nillable="true" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
</wsdl:types>
</wsdl:definitions>
```
<xsd:element name="Assignee" nillable="true" type="xsd:string"/>
<xsd:element name="Assignee_Description" nillable="true" type="xsd:string"/>
<xsd:element name="Assignee_Message" nillable="true" type="xsd:string"/>
<xsd:element name="Location" nillable="true" type="xsd:string"/>
<xsd:element name="Location_Description" nillable="true" type="xsd:string"/>
<xsd:element name="Location_Message" nillable="true" type="xsd:string"/>
<xsd:element name="Revenue_Category" nillable="true" type="xsd:string"/>
<xsd:element name="Revenue_Category_Description" nillable="true" type="xsd:string"/>
<xsd:element name="Revenue_Category_Message" nillable="true" type="xsd:string"/>
<xsd:element name="Spend_Category" nillable="true" type="xsd:string"/>
<xsd:element name="Spend_Category_Description" nillable="true" type="xsd:string"/>
<xsd:element name="Spend_Category_Message" nillable="true" type="xsd:string"/>
<xsd:element name="Ledger_Account" nillable="true" type="xsd:string"/>
<xsd:element name="Ledger_Account_Description" nillable="true" type="xsd:string"/>
<xsd:element name="Ledger_Account_Message" nillable="true" type="xsd:string"/>
<xsd:element name="ErrorMessage" nillable="true" type="xsd:string"/>
<xsd:element maxOccurs="unbounded" name="ErrorMessageList" nillable="true" type="xsd:string"/>
<xsd:element name="Number_Of_Errors" nillable="true" type="xsd:string"/>
<xsd:element name="Number_Of_Warnings" nillable="true" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>

<wsdl:message name="validateCOA_PortType_validateCOA">
  <wsdl:part name="parameters" element="tns:validateCOA"/>
</wsdl:message>

<wsdl:message name="validateCOA_PortType_validateCOAResponse">
  <wsdl:part name="parameters" element="tns:validateCOAResponse"/>
</wsdl:message>

<wsdl:portType name="validateCOA_PortType">
  <wsdl:operation name="validateCOA">
    <wsdl:input message="tns:validateCOA_PortType_validateCOA"/>
    <wsdl:output message="tns:validateCOA_PortType_validateCOAResponse"/>
  </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="YaleWDFDM_ws_validateCOA_Binder" type="tns:validateCOA_PortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="validateCOA"/>
<soap:operation soapAction="YaleWDFDM_ws_validateCOA_Binder_validateCOA" style="document"/>
<wsdl:input>
  <soap:body parts="parameters" use="literal"/>
</wsdl:input>
<wsdl:output>
  <soap:body parts="parameters" use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="YaleWDFDM.ws.validateCOA">
  <wsdl:port name="YaleWDFDM_ws_validateCOA_Port" binding="tns:YaleWDFDM_ws_validateCOA_Binder">
    <soap:address location="https://gw-tst.its.yale.edu/soap/coavalidator"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

Below is an example of a SOAP request/response that has no validation errors:

**Sample SOAP Request (no validation errors):**

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:yal="esb/YaleWDFDM.ws:validateCOA">
  <soapenv:Header/>
  <soapenv:Body>
    <yal:validateCOA>
      <Mode/>
      <Transaction_Date/>
      <Company_Code>CO01</Company_Code>
      <Grant_Code/>
      <Gift_Code>GE006415</Gift_Code>
      <Yale_Designated_Code/>
      <Cost_Center_Code>CC0835</Cost_Center_Code>
      <Program_Code>PG00041</Program_Code>
      <Project_Code/>
      <Assignee_Code/>
      <Location_Code/>
      <Revenue_Category_Code/>
      <Spend_Category_Code>SC372</Spend_Category_Code>
      <Ledger_Account_Code/>
    </yal:validateCOA>
  </soapenv:Body>
</soapenv:Envelope>
```
Sample SOAP Response (no validation errors):

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
      <Transaction_Date>12/01/2016</Transaction_Date>
      <Transaction_Date_Message>W004 – Current date assumed</Transaction_Date_Message>
      <Company>CO01</Company>
      <Company_Description>Yale University</Company_Description>
      <Company_Message xsi:nil="true"/>
      <Grant xsi:nil="true"/>
      <Grant_Description xsi:nil="true"/>
      <Grant_Message xsi:nil="true"/>
      <Gift>GE006415</Gift>
      <Gift_Description>Dupuy Harry W &amp; Chas M #1 06415</Gift_Description>
      <Gift_Message xsi:nil="true"/>
      <Yale_Designated xsi:nil="true"/>
      <Yale_Designated_Description xsi:nil="true"/>
      <Yale_Designated_Message xsi:nil="true"/>
      <Fund xsi:nil="true"/>
      <Fund_Description xsi:nil="true"/>
      <Fund_Message xsi:nil="true"/>
      <Cost_Center>CC0835</Cost_Center>
      <Cost_Center_Description>MEDNEU Neurology - All</Cost_Center_Description>
      <Cost_Center_Message xsi:nil="true"/>
      <Program>PG00041</Program>
      <Program_Description>Clinical - Other Affiliated Hospital &amp; Contracts</Program_Description>
      <Program_Message xsi:nil="true"/>
      <Project xsi:nil="true"/>
      <Project_Message xsi:nil="true"/>
      <Assignee xsi:nil="true"/>
      <Assignee_Description xsi:nil="true"/>
      <Assignee_Message xsi:nil="true"/>
      <Location xsi:nil="true"/>
      <Location_Description xsi:nil="true"/>
      <Location_Message xsi:nil="true"/>
      <Revenue_Category xsi:nil="true"/>
      <Revenue_Category_Description xsi:nil="true"/>
      <Revenue_Category_Message xsi:nil="true"/>
      <Spend_Category>SC372</Spend_Category>
      <Spend_Category_Description>Assessment Expense - Affil Hosp (SC372)</Spend_Category_Description>
      <Spend_Category_Message xsi:nil="true"/>
      <Ledger_Account xsi:nil="true"/>
      <Ledger_Account_Description xsi:nil="true"/>
      <Ledger_Account_Message xsi:nil="true"/>
    </ser-root:validateCOAResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
Below is an example of a request/response that has validation errors:

**Sample SOAP Request (with validation errors):**

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:yal="esb/YaleWDFDM.ws:validateCOA">
  <soapenv:Header/>
  <soapenv:Body>
    <yal:validateCOA>
      <Mode>?</Mode>
      <Transaction_Date>07/31/2016</Transaction_Date>
      <Company_Code>CO01</Company_Code>
      <Grant_Code>?</Grant_Code>
      <Gift_Code>GE006415</Gift_Code>
      <Yale_Designated_Code>?</Yale_Designated_Code>
      <Cost.Center_Code>CC0835a</Cost.Center_Code>
      <Program_Code>PG00041a</Program_Code>
      <Project_Code>?</Project_Code>
      <Assignee_Code>?</Assignee_Code>
      <Location_Code>?</Location_Code>
      <Revenue_Category_Code>?</Revenue_Category_Code>
      <Spend_Category_Code>SC372</Spend_Category_Code>
      <Ledger_Account_Code>?</Ledger_Account_Code>
    </yal:validateCOA>
  </soapenv:Body>
</soapenv:Envelope>
```

**Sample SOAP Response (with validation errors):**

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ser-root="esb/YaleWDFDM.ws:validateCOA"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <ser-root:validateCOAResponse xmlns:ser-root="esb/YaleWDFDM.ws:validateCOA">
      <Transaction_Date>07/31/2016</Transaction_Date>
      <Transaction_Date_Message xsi:nil="true"/>
    </ser-root:validateCOAResponse>
  </soapenv:Body>
</soapenv:Envelope>
```
ErrorMessage>E009 - Gift and Yale Designated are not allowed together
E012 - Only one of Revenue Category, Spend Category is required</ErrorMessage>
<ErrorMessageList>E009 - Gift and Yale Designated are not allowed together
E012 - Only one of Revenue Category, Spend Category is required</ErrorMessageList>
</ser-root:validateCOAResponse>
</soapenv:Body>
</soapenv:Envelope>
## Validation Errors

The following is a full set of warnings/error messages that will be returned by the COA Validation Web service if the validation fails on one or more COA segments. These are the defined error/warning messages, they are always prefixed with “Ennn” or “Wnnn”.

<table>
<thead>
<tr>
<th>Input</th>
<th>Validation</th>
<th>Error/Warning Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company</td>
<td>Input is blank</td>
<td>E001 – Company is required</td>
</tr>
<tr>
<td>Company</td>
<td>Input is not a valid value</td>
<td>E002 – Company does not exist</td>
</tr>
<tr>
<td>Transaction Date</td>
<td>Tran date is blank then Tran date=current date</td>
<td>W004 – Current date assumed</td>
</tr>
<tr>
<td>Transaction Date</td>
<td>Transaction date format is not mm/dd/yyyy</td>
<td>E005 – Invalid transaction date format</td>
</tr>
<tr>
<td>Transaction Date</td>
<td>Transaction date is not a valid date (e.g. 02/29/2015)</td>
<td>E006 – Invalid transaction date</td>
</tr>
<tr>
<td>Cost Center</td>
<td>Input is blank and Grant is blank</td>
<td>E001 – Cost Center is required</td>
</tr>
<tr>
<td>Cost Center</td>
<td>Input is not a valid value</td>
<td>E002 – Cost Center does not exist</td>
</tr>
<tr>
<td>Cost Center</td>
<td>Cost Center status &lt;&gt; active</td>
<td>E003 - Cost Center not active</td>
</tr>
<tr>
<td>Program</td>
<td>Input is blank and Grant is blank</td>
<td>E001 – Program is required</td>
</tr>
<tr>
<td>Program</td>
<td>Input is not a valid value</td>
<td>E002 – Program does not exist</td>
</tr>
<tr>
<td>Program</td>
<td>Program status &lt;&gt; active</td>
<td>E003 – Program not active</td>
</tr>
<tr>
<td>Yale Designated</td>
<td>Input is blank and Gift is blank and Grant is blank</td>
<td>E010 – One of Grant, Gift or Yale Designated is required</td>
</tr>
<tr>
<td>Yale Designated</td>
<td>Input is not a valid value</td>
<td>E002 – Yale Designated does not exist</td>
</tr>
<tr>
<td>Yale Designated</td>
<td>Yale Designated status &lt;&gt; active</td>
<td>E003 – Yale Designated not active</td>
</tr>
<tr>
<td>Yale Designated</td>
<td>Yale Designated &lt;&gt; blank and Gift &lt;&gt; blank</td>
<td>E009 – Gift and Yale Designated are not allowed together</td>
</tr>
<tr>
<td>Gift</td>
<td>Input is blank and YD is blank and Grant is blank</td>
<td>E010 – One of Grant, Gift or Yale Designated is required</td>
</tr>
<tr>
<td>Gift</td>
<td>Input is not a valid value</td>
<td>E002 – Gift does not exist</td>
</tr>
<tr>
<td>Column</td>
<td>Description</td>
<td>Error Code</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>Gift</td>
<td>Inactive = ‘yes’</td>
<td>E003</td>
</tr>
<tr>
<td>Gift</td>
<td>Inactive = ‘no’ and Gift status &lt;&gt; ‘approved’</td>
<td>E004</td>
</tr>
<tr>
<td>Gift</td>
<td>Gift company &lt;&gt; Company value</td>
<td>E005</td>
</tr>
<tr>
<td>Gift</td>
<td>Yale Designated &lt;&gt; blank and Gift &lt;&gt; blank</td>
<td>E009</td>
</tr>
<tr>
<td>Grant</td>
<td>Input is blank and YD is blank and Gift is blank</td>
<td>E010</td>
</tr>
<tr>
<td>Grant</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Grant</td>
<td>Grant status &lt;&gt; active</td>
<td>E003</td>
</tr>
<tr>
<td>Grant</td>
<td>Grant status = active and Tran date not between Award Line Start Date and Award Line End Date</td>
<td>E007</td>
</tr>
<tr>
<td>Grant</td>
<td>Grant status = active and Award Line status not in (active, at risk)</td>
<td>E008</td>
</tr>
<tr>
<td>Revenue Category</td>
<td>Input is blank and Spend Category is blank and Ledger Account is blank</td>
<td>E012</td>
</tr>
<tr>
<td>Revenue Category</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Revenue Category</td>
<td>Revenue Category status&lt;&gt;active</td>
<td>E003</td>
</tr>
<tr>
<td>Revenue Category</td>
<td>If Revenue category &lt;&gt; blank and Spend category &lt;&gt; blank</td>
<td>E011</td>
</tr>
<tr>
<td>Spend Category</td>
<td>Input is blank and revenue Category is blank and Ledger Account is blank</td>
<td>E012</td>
</tr>
<tr>
<td>Spend Category</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Spend Category</td>
<td>Spend Category status &lt;&gt; active</td>
<td>E003</td>
</tr>
<tr>
<td>Spend Category</td>
<td>If Revenue category &lt;&gt; blank and Spend category &lt;&gt; blank</td>
<td>E011</td>
</tr>
<tr>
<td>Ledger Account</td>
<td>Input is blank and Spend Category is blank and Revenue Category is blank</td>
<td>E012</td>
</tr>
<tr>
<td>Ledger Account</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Category</td>
<td>Condition</td>
<td>Error Code</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Ledger Account</td>
<td>Ledger Account status &lt;&gt; active</td>
<td>E003</td>
</tr>
<tr>
<td>Project</td>
<td>Input is blank and Grant is blank and Cost Center is blank</td>
<td>E001</td>
</tr>
<tr>
<td>Project</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Project</td>
<td>Inactive = 'yes'</td>
<td>E003</td>
</tr>
<tr>
<td>Project</td>
<td>Inactive = 'no' and Tran date not between Project Start Date and Project End Date</td>
<td>E007</td>
</tr>
<tr>
<td>Assignee</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Assignee</td>
<td>Assignee status = inactive</td>
<td>E003</td>
</tr>
<tr>
<td>Location</td>
<td>Input is not a valid value</td>
<td>E002</td>
</tr>
<tr>
<td>Location</td>
<td>Location status &lt;&gt; active</td>
<td>E003</td>
</tr>
</tbody>
</table>

**Frequently Asked Questions (FAQ)**

- To be added here
Document Changes

<table>
<thead>
<tr>
<th>Date</th>
<th>Change Description</th>
<th>Who</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/2/2016</td>
<td>Initial Draft Version</td>
<td>Ramesh Gade</td>
</tr>
<tr>
<td>12/7/2016</td>
<td>Added Configuration Item, “COA Validator / COA Integration Hub” to the ServiceNow ticket that FSS should use when creating a ticket to BSG ITS team</td>
<td>Ramesh Gade</td>
</tr>
<tr>
<td>12/8/2016</td>
<td>Minor updates to segment validation info</td>
<td>Ramesh Gade</td>
</tr>
<tr>
<td>12/9/2016</td>
<td>Replaced In/Out parameter with a Service Contract</td>
<td>Ramesh Gade</td>
</tr>
<tr>
<td>12/14/2016</td>
<td>Updated content and validation error tables</td>
<td>Ramesh Gade</td>
</tr>
</tbody>
</table>

Appendix

- Tools for testing Web Services
  
  [https://dzone.com/articles/12-great-web-service-testing-tools](https://dzone.com/articles/12-great-web-service-testing-tools)