SDTM-ETL 4.0 User Manual and Tutorial: Using RESTful Web Services

Author: Jozef Aerts, XML4Pharma

Last update: 2021-04-20

RESTful web services are everywhere. Whether we order something on Amazon, book a flight, search for information, a RESTful web service is often behind the scene. RESTful web services become more and more important in medicine and pharma. An ever growing list can be found here.

RESTful web services is about machine-machine communication (software to software), and, as the name states, is about "services". Using RESTful web services enables computer programs to use services from other computers, either from within the own company, or using publicly available services such as from the "National Library of Medicine" (NLM) or even as provided by the FDA.

SDTM-ETL allows to directly use such services from within the mapping script.

A number of such services have been **pre-defined** in our scripting language, and are mostly related to the use of CDISC controlled terminology and to the use of LOINC coding. Others (e.g. company-internal ones) can easily be implemented.

Using SDTM-ETL pre-defined RESTful web services

SDTM-ETL comes with a number of pre-defined RESTful web service functions that can be used directly in mapping scripts. These are all related to the use of CDISC Controlled Terminology (CT) and to the use of LOINC, especially for lab tests. Keep in mind that the use of LOINC coding will soon become mandatory by the FDA in SDTM submissions!

The currently available RESTful Web Services functions are:

CDISC-CT and CDISC-SDTM functions:

A. **rws:testNameFromTestCode**(test-code, test-code-system):
returns the CDISC test name (--TEST value) from the CDISC test code (--TEST).

For example:
$lbtestname = rws:testNameFromTestCode(ALB', 'LBTESTCD'):

Will return "Albumin" and store it in the variable $lbtestname.

This function is especially interesting to automate the generation of LBTEST, VSTEST, …
It automatically uses the latest version of CDISC-CT available.

B. **rws:sdtmLabel**(sdtm-version, sdtm-variable):

returns the label of an SDTM variable given the variable name and the SDTM model version (1.2, 1.3,1.4, 1.7)

For example:
$label = rws:sdtmLabel('1.4', 'VISITNUM');

Will return "Visit Number" and store it in the variable $label.

LOINC functions:

The following LOINC functions are extremely useful to automate conversion from LOINC codes into CDISC variable values, and thus automate generation of e.g. LB datasets when the LOINC code is provided.

This does currently not include the "CDISC LOINC to SDTM-LB" mapping yet, which will immediately be implemented as a RESTful web service once the final version of the mapping is published by CDISC. This will **not** require an update of the software, only of a single file that comes with the software. Existing customers will obtain the file automatically and without any cost.

Already implemented LOINC services are:

A. **rws:loincClass**(LOINC-code):
   returns the LOINC class from the LOINC code.

   For example:
   $loincname = rws:loincClass('1751-7');

   Will return "CHEM" (chemistry) and store it in the variable $loincname.

B. **rws:loincComponent**(LOINC-code):
   returns the LOINC "component" (i.e. the chemical compound) from the LOINC code.

   For example:
   $component = rws:loincComponent('1751-7');

   Will return "Albumin" and store it in the variable $component.

C. **rws:loincProperty**(LOINC-code):
   returns the measured property from the LOINC code.

   For example:
   $property = rws:loincProperty('1751-7');

   Will return "MCnc" ("mass concentration") and store it in the variable $property.

D. **rws:loincSystem**(LOINC-code):
   returns the system from which the test is taken.

   For example:
   $system = rws:loincSystem('1751-7');

   Will return "Ser/Plas" (Serum / Plasma) and store it in the variable $system.

E. **rws:loincMethod**(LOINC-code):
   returns the method used (if provided and important to uniquely identify the test).

   For example:

$method = rws:loincMethod('5792-7');

Will return "Test strip" and store it in the variable $method.

F. **rws:loincScale**(LOINC-code):
returns the scale type of the test (quantitative, ordinal, nominal, …)

For example:
$scale = rws:loincScale('1751-7');

Will return "Qn" ("Quantitative") and store it in the variable $scale

G. **rws:loincTimeAspect**(LOINC-code):
returns the time aspect of the test.

For example:
$timeasp = **rws:loincTimeAspect('21482-5');**

Will return "24H" ('Protein concentratioin 24 hour urine) and store it in the variable $timeasp

H. **rws:partExplanation**(part-type, part-name):
returns the explanation (definition) of a LOINC part.

For example:
$explanation = **rws:partExplanation('method','BCG');**

returns the explanation for the LOINC method with LOINC name "BCG", which is:
"*Bromocresol green (BCG) dye binding method*"

New LOINC functions

Already now, FDA is requiring (for new studies) that LBLOINC (LOINC code for the laboratory test) is populated. As CDISC does not want to give up its "post-coordinated" approach for lab tests, it will still be required to provide values for LBTESTCD, LBTEST, LBSPEC, …

CDISC has been developing a mapping between the LOINC code of the most used lab tests in research (approximately 1400 tests), and the "classic" SDTM variables LBTESTCD, LBTEST, LBSPEC … We have extended these mappings considerably, and provide a public and free RESTful web service implementing this mapping. This is extremely useful to auto-populate LBTESTCD, LBTEST, LBSPEC and others when the LOINC code, received from the laboratory, is known.

See: http://xml4pharmaserver.com/WebServices/LOINC2CDISC_webservices.html

A separate tutorial "**Using the LOINC-SDTM-LB Mapping and similar functions**" is separately available on our website.

Unit conversion RESTful web services functions

When generating SDTM datasets, generating (correct) code for unit conversions can be a nightmare. Not so with SDTM-ETL. We developed a number of unit conversion functions that are based on the National Library of Medicine RESTful web services (see https://ucum.nlm.nih.gov/ucum-service.html).

A separate tutorial about using these functions "**Performing Unit Conversions in SDTM-ETL**" is also available from our website.

Creating your own RESTful web services functions

SDTM-ETL users can easily create their own functions to use RESTful web services that are either available from their company's intranet, or from publicly available RESTful web services such as these provided by the National Library of Medicine (NLM). The latter e.g. provides services for unit conversions based on the UCUM notation for units[1]. Unfortunately, CDISC still refuses to allow UCUM notation. If it would allow UCUM notation, the population of –STRESN variables would become much more easy, and could easily be automated using these RESTful web services.

In SDTM-ETL, it is very easy to create own reusable "functions" by adding some the code for it in the "functions.xsl" file (in the "stylesheets" folder). Using the same mechanism, it is also very easy to create own RESTful web services functions by extending the file "**functions_restful_web_services.xsl**".

Doing so requires some knowledge of XSLT, but every IT student nowadays knows XSLT very well. So, if you cannot do so yourself, just call your local IT department and ask for someone who knows XSLT.

The existing RESTful web service functions in the file "functions_restful_web_services.xsl" can easily be used as starting examples for your own RESTful web service function calls.
It is highly recommended to use the "http://www.xml4pharma.com/SDTM-ETL/RWS/ns" namespace and the "rws" prefix, as these are recognized by the SDTM-ETL software. When using another namespace and prefix, a warning will be given by the software, but one will be able to continue anyway.

Remark that currently, only "HTTP GET" can be used in SDTM-ETL functions and scripts. "HTTP POST" RESTful web service calls are currently not yet possible.

Calling a RESTful Web Service from within a mapping script

One can also pretty easily do a call to an existing RESTful web service, either from a public source such one of the many RESTful web services of the National Library of Medicine, or from the company's own intranet. It is also already possible to use the CDISC SHARE RESTful web services[2] (the API can be found at https://www.cdisc.org/share/api-user-documentation).

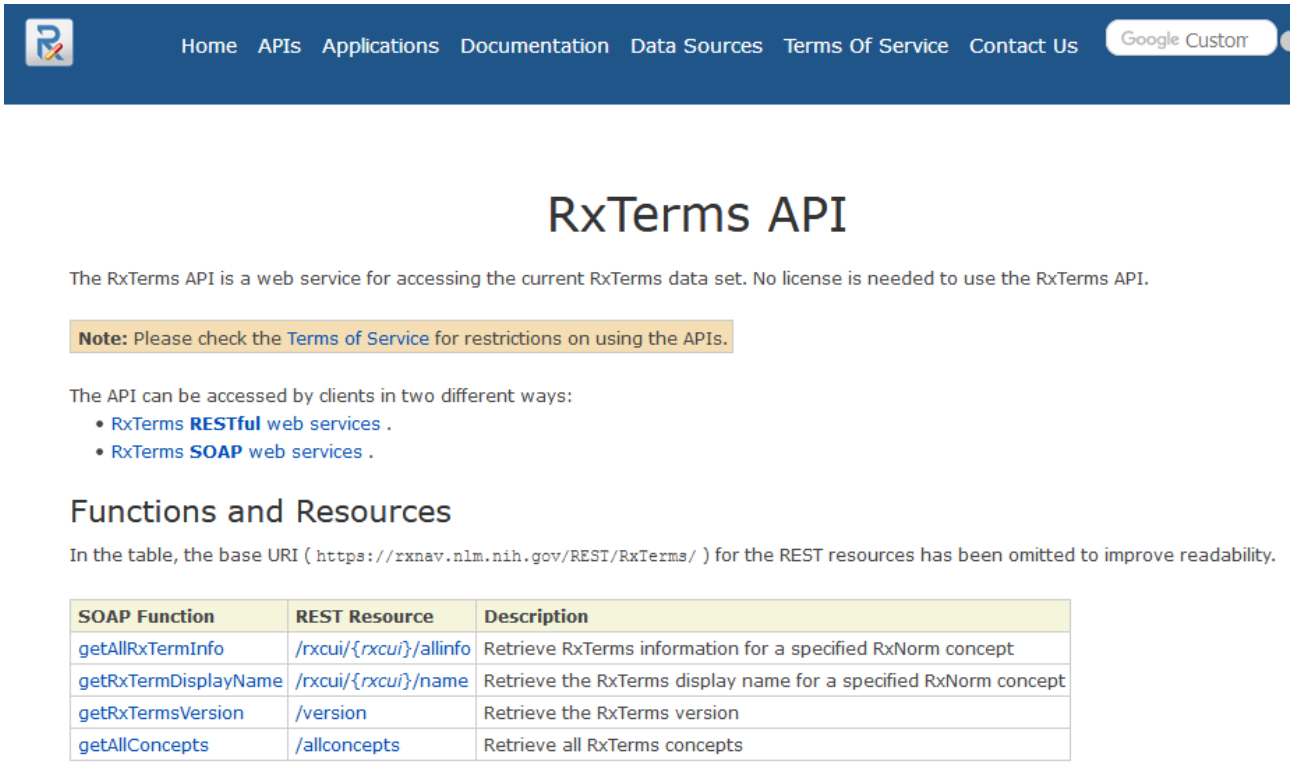
Often, it will be useful to create a variable as a "global" variable containing the "base" of the RESTful web service query string. As "global" variables can be referenced anywhere, this makes it easy to apply "write once, run many times".

---

[1] The "UCUM" RESTful web services were developed by us in cooperation with the University of Applied Sciences FH Joanneum, and have been donated to the National Library of Medicine.
[2] The SHARE API currently has not many functions yet on the variable or individual CT term yet. This will probably change in future.

Example: calling NLM RESTful web services for RxNorm

For example, suppose you would like to call the NLM RESTful web service to obtain information for a medication which has an RxNorm number. This function is described at: https://rxnav.nlm.nih.gov/RxTermsAPIs.html (there are many other functions).



The base of the function is: https://rxnav.nlm.nih.gov/REST/RxTerms and we store it in the "global" variable $RXTERM, i.e. the script is:

$RXTERMBASE = "https://rxnav.nlm.nih.gov/REST/RxTerms";

As we learn from the API, we can ask all info about a medication with a specific "cui" (identifier) using:

https://rxnav.nlm.nih.gov/REST/RxTerms/rxcui/{cui}/allinfo.xml

e.g. calling "https://rxnav.nlm.nih.gov/REST/RxTerms/rxcui/198440/allinfo.xml" results in a response:

```
<rxtermsdata>
  <rxtermsProperties>
    <brandName/>
    <displayName>Acetaminophen (Oral Pill)</displayName>
    <synonym>APAP</synonym>
    <fullName>Acetaminophen 500 MG Oral Tablet</fullName>
    <fullGenericName>Acetaminophen 500 MG Oral Tablet</fullGenericName>
    <strength>500 mg</strength>
    <rxtermsDoseForm>Tab</rxtermsDoseForm>
    <route>Oral Pill</route>
    <termType>SCD</termType>
    <rxcui>198440</rxcui>
    <genericRxcui>0</genericRxcui>
    <rxnormDoseForm>Oral Tablet</rxnormDoseForm>
    <suppress/>
  </rxtermsProperties>
</rxtermsdata>
```

In the specific SDTM-ETL mapping script for getting the "full name" of the medication, we then write e.g.

```
$rxcui = '198440';
# the RESTful web service as a string
$restquery = concat($RXTERMBASE, '/rxcui/', $rxcui, '/allinfo.xml');
# call the RESTful web service and retrieve the "full name" (XML element "fullName)
$fullname = doc($restquery)/rxtermsdata/rxtermsProperties/fullName;
```

The latter part (after doc($restquery)) being the path to be followed within the result XML.

IMPORTANT: all this is case sensitive!

And then using the value of $fullname further on in the script.

The result that is obtained is the string "Acetaminophen 500 MG Oral Tablet".

Using company-specific RESTful web services

RESTful web services have the advantage that they are easy to implement. SDTM-ETL supports any RESTful web service (currently limited to "GET" methods) for which the API (i.e. the list with method descriptions) is known by the user of the software. This of course includes company-internal RESTful web services.

Users can either use these services directly from within the mapping scripts (as described above) or can be "encapsulated" into company-internal SDTM-ETL functions by extending the file "**functions_restful_web_services.xsl**".

For the server side of the RESTful web service, if you need help or assistance, mail us. We have already developed dozens of RESTful web service for use in clinical research, including a number of free and public ones, for which the documentation can be found on our "RESTful Web Services web page".