PROV-Template Registry and Expansion Service

WORK PACKAGE 8 – Data Curation and Cataloging

LEADING BENEFICIARY: Umweltbundesamt Austria

Author(s):	Beneficiary/Institution
Doron Goldfarb	Umweltbundesamt GmbH (EAA)
Paul Martin	University of Amsterdam (UvA)

Document history:

Date	Version
27th September 2018	Initial Version
22th October 2018	Update for POST based expansion / storage + python example

A document of ENVRI plus project - www.envri.eu/envriplus

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant

TABLE OF CONTENTS

1 Introduction

This document provides a brief introduction to the EnvriPlus PROV-Template registry and expansion service prototype. Available on the Web at https://envriplus-provenance.test.fedcloud.eu/, it allows potential users to register and describe their own PROV-Templates (For the concept, see [Moreau, 2018] and the online specification[1]) in order to gather a selection of PROV related views on various aspects of the data life cycles of EnvriPlus related research infrastructures.

The individual contributions are made available to everyone via the Web interface in order to be reused and/or instantiated for own purposes. This way, the templates can be browsed and downloaded in various formats. A dedicated API allows the instantiation of registered templates into PROV documents via bindings as described in the PROV-Template specification.

[1] https://provenance.ecs.soton.ac.uk/prov-template/, retrieved September 26, 2018

The source code for the prototype is available via GitHub at https://github.com/EnvriPlus-PROV/EnvriProvTemplates, it uses a dedicated PROV-template expansion library also created in the context of Task 8.3 by DKRZ and EAA, available at https://github.com/EnvriPlus-PROV/EnvriProvTemplates.

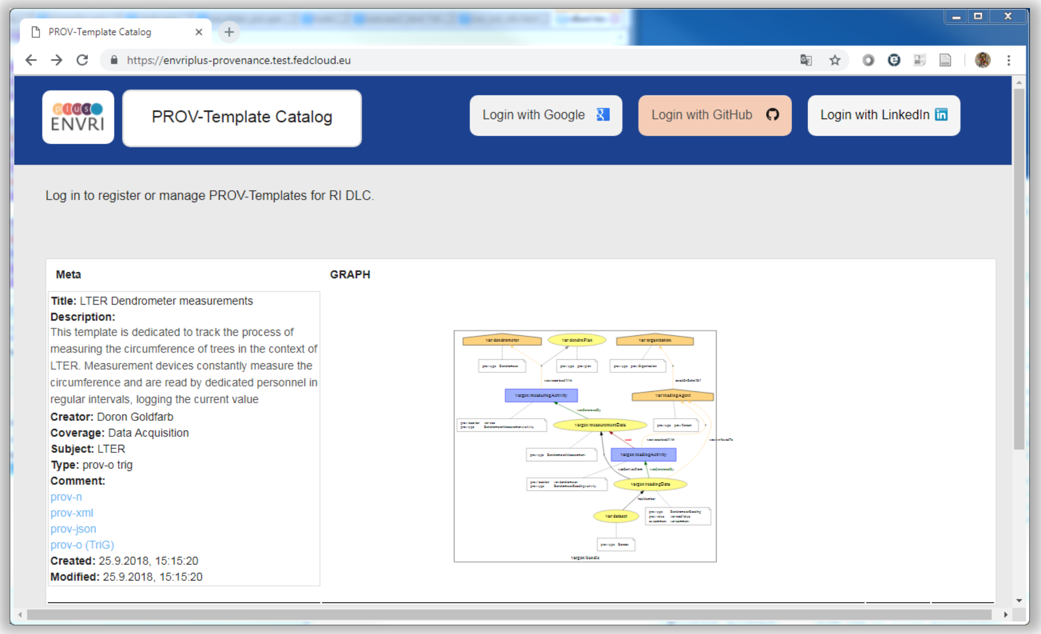
The reminder of this document is structured as follows: Section 2 describes the Web Interface for browsing and registering new templates, followed by Section 3 introducing the API functionality.

2 Web Interface

Figure 1 shows the Web Interface populated with one single template. It should run in Firefox, Safari and Chrome browsers, Internet Explorer is currently not supported. Every registered template and the related instantiation API (see below) is visible to and usable for the public. In order to register their own templates, however, users need to log in first in order to be able to modify or delete their contributions at a later stage.

User Login

Currently, it is possible to log in using existing oauth2 profiles provided by common Web services, at the moment, this is implemented for Google, GitHub and LinkedIn profiles only. Clicking on one of the three provided buttons in the upper right corner opens a separate window for providing the respective credentials and authorizing the retrieval of basic profile information.

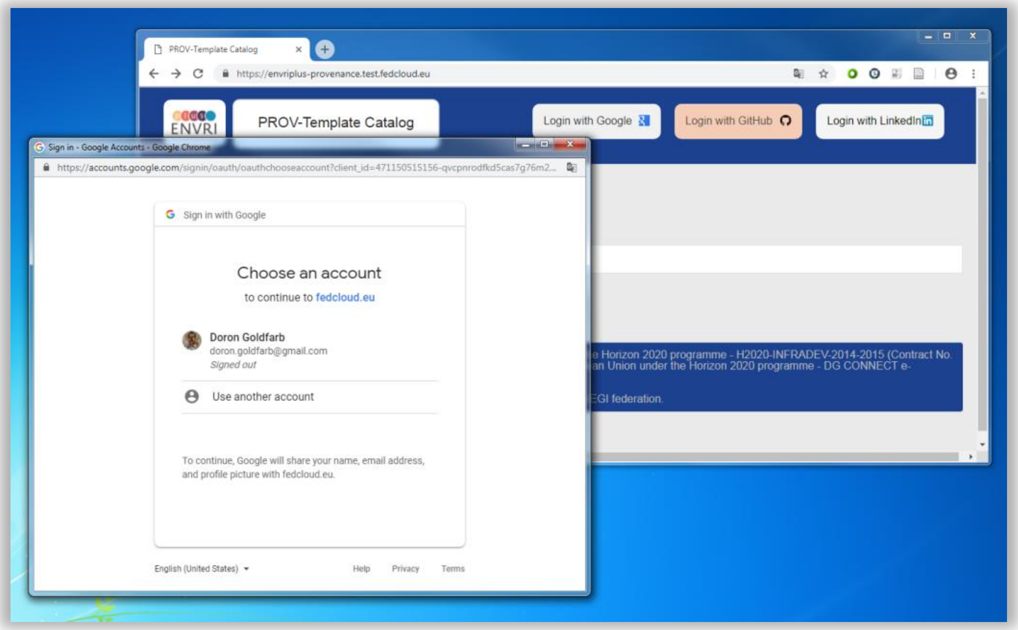


Log in to register or manage PROV-Templates for RI DLC.

Title: LTER Dendrometer measurements
Description:
This template is dedicated to track the process of measuring the circumference of trees in the context of LTER. Measurement devices constantly measure the circumference and are read by dedicated personnel in regular intervals, logging the current value
Creator: Doron Goldfarb
Coverage: Data Acquisition
Subject: LTER
Type: prov-o trig
Comment:
prov-n
prov-xml
prov-json
prov-o (TriG)
Created: 25.9.2018, 15:15:20
Modified: 25.9.2018, 15:15:20

Figure 1: Web interface for registering and browsing PROV-Templates

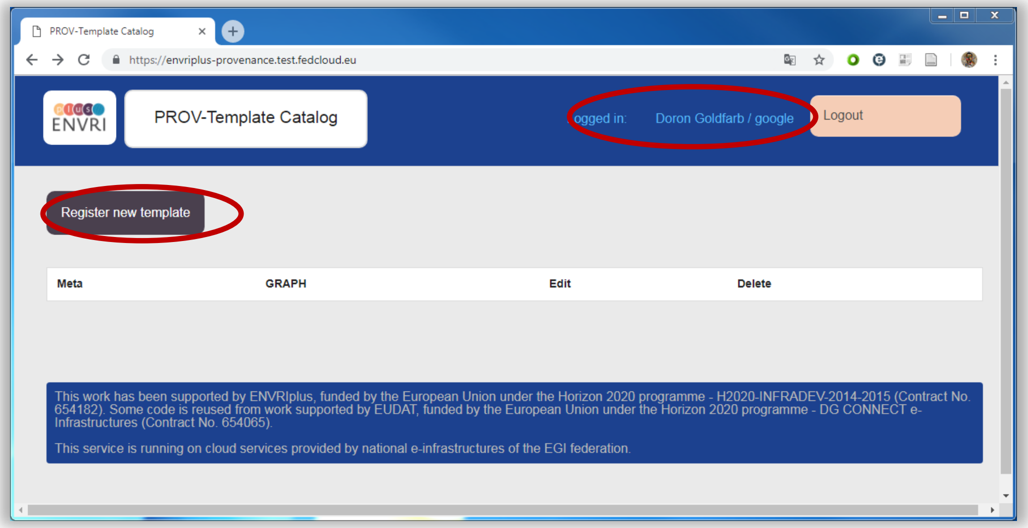
Figure 2 shows the separate login window for a Google account. Once the login procedure is completed, the window closes by itself.



To continue, Google will share your name, email address, and profile picture with fedcloud.eu.

Figure 2: Logging in via Google

As visible in Figure 3, the currently logged in user is visible in the upper right corner, including a button to log out again. Another button is activated, allowing the registration of new templates.



This work has been supported by ENVRIplus, funded by the European Union under the Horizon 2020 programme - H2020-INFRADEV-2014-2015 (Contract No. 654182). Some code is reused from work supported by EUDAT, funded by the European Union under the Horizon 2020 programme - DG CONNECT e-Infrastructures (Contract No. 654065).

This service is running on cloud services provided by national e-infrastructures of the EGI federation.

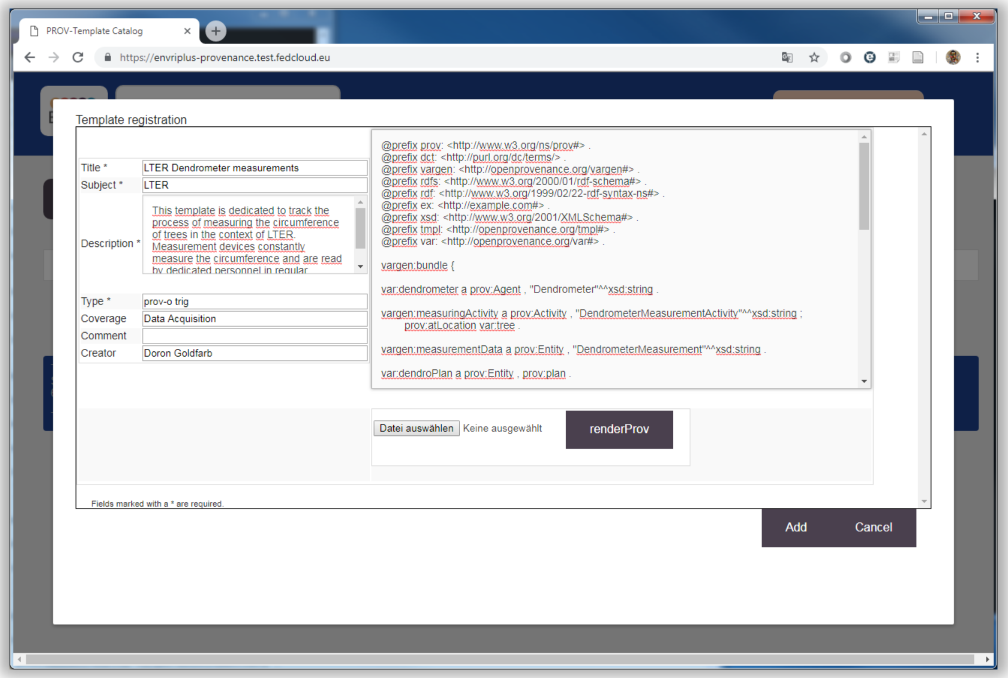
Figure 3: Registering new templates for logged in users

Registering templates

Once the “Register new template” button is clicked, an overlay window appears, providing a form to enter the information for the new template, shown in Figure 4.

On the upper right side, a text area is dedicated to hold the PROV-Template document to be stored, currently, it is possible to import PROV Templates as prov-xml, prov-json or as prov-o conforming RDF-TriG. They can either by copy pasted directly into the text area or imported from a local file via the dedicated button.

On the left side, a set of Dublin Core metadata fields serves to describe the new template. The *type* field should be used to specify the used PROV format, while the *coverage* field should be used to describe the DLC aspect covered by the template. While the former is realized as option menu, the coverage field is currently accepting free text and should eventually be limited to a set of controlled vocabulary terms.



Template registration

Title *	LTER Dendrometer measurements
Subject *	LTER
Description *	This template is dedicated to track the process of measuring the circumference of trees in the context of LTER. Measurement devices constantly measure the circumference and are read by dedicated personnel in regular
Type *	prov-o trig
Coverage	Data Acquisition
Comment	
Creator	Doron Goldfarb

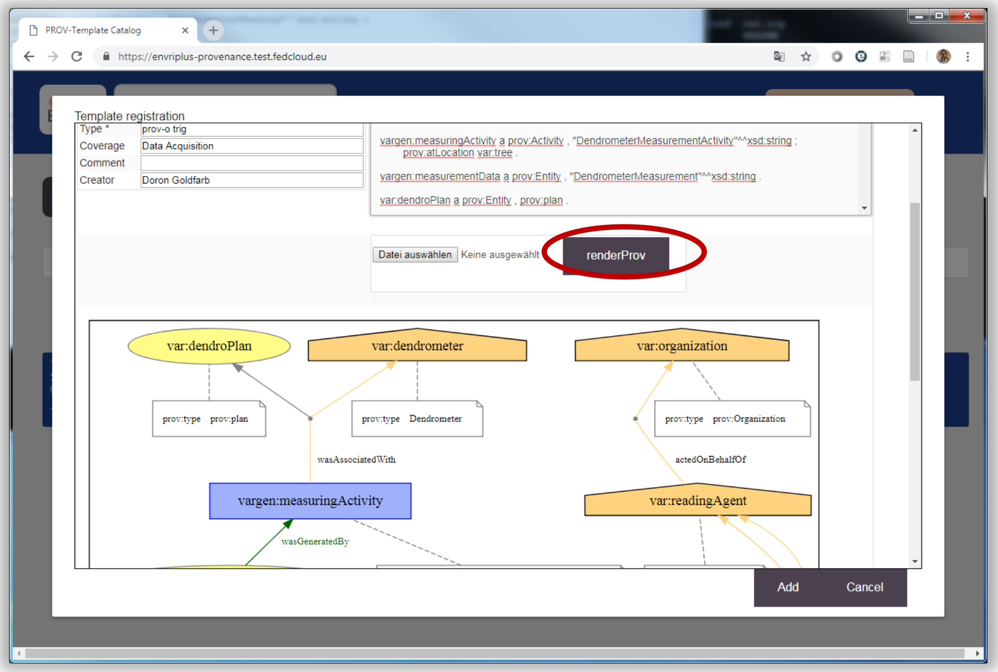
```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix vargen: <http://openprovenance.org/vargen#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix ex: <http://example.com#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .

vargen:bundle {

var:dendrometer a prov:Agent , "Dendrometer"^^xsd:string .

vargen:measuringActivity a prov:Activity , "DendrometerMeasurementActivity"^^xsd:string ;
        prov:atLocation var:tree .

vargen:measurementData a prov:Entity , "DendrometerMeasurement"^^xsd:string .

var:dendroPlan a prov:Entity , prov:plan .
```

Fields marked with a * are required.

Figure 4: Form for registering new templates

The button "renderProv" allows the creation of a visual representation of the current content of the PROV-template text area, at the same time serving as rudimentary validation tool. While Figure 5 shows the result of a successful visual rendering, Figure 6 shows an error message appearing for an invalid PROV-template. When clicking on "renderProv" without first setting the Type field with the intended PROV format, there is no individual feedback on the location of the error in the document, only a "document is empty" message is displayed. When the type field is set to one of the valid PROV formats, a more informative error message is displayed instead.

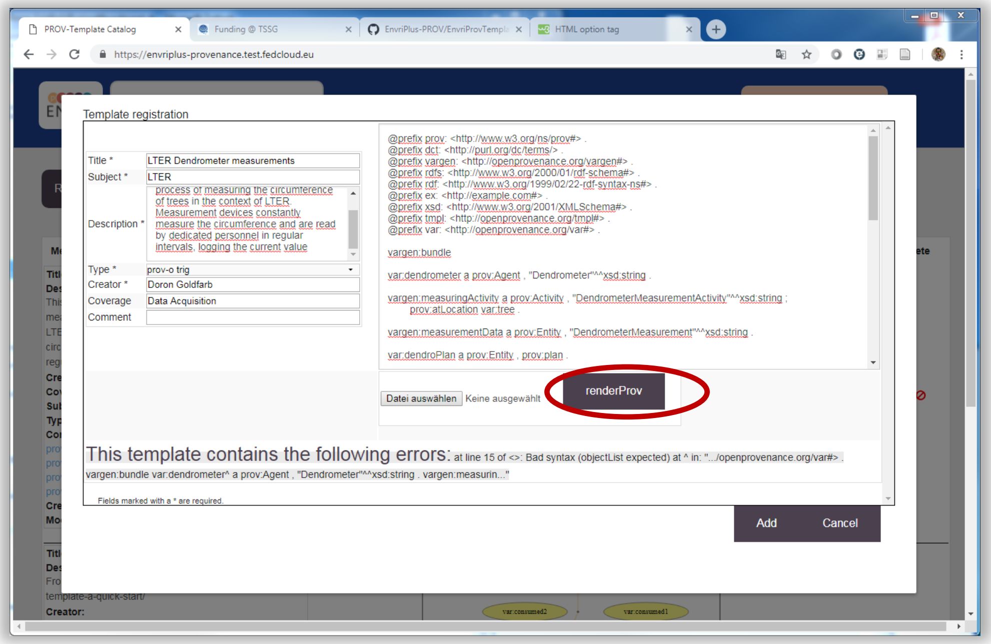
Once entered and rendered successfully, a click on "Add" stores the newly entered template into an underlying MongoDB instance. If mandatory fields are missing or the template not rendered correctly, a warning message appears instead.



Type *	prov-o trig
Coverage	Data Acquisition
Comment	
Creator	Doron Goldfarb

```
vargen:measuringActivity a prov:Activity , "DendrometerMeasurementActivity"^^xsd:string ;
    prov:atLocation var:tree .

vargen:measurementData a prov:Entity , "DendrometerMeasurement"^^xsd:string .

var:dendroPlan a prov:Entity , prov:plan .
```

Figure 5: Creating a visual representation of the new template



Title *	LTER Dendrometer measurements
Subject *	LTER
Description *	process of measuring the circumference of trees in the context of LTER. Measurement devices constantly measure the circumference and are read by dedicated personnel in regular intervals, logging the current value
Type *	prov-o trig
Creator *	Doron Goldfarb
Coverage	Data Acquisition
Comment	

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix vargen: <http://openprovenance.org/vargen#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix ex: <http://example.com#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .

vargen:bundle

var:dendrometer a prov:Agent , "Dendrometer"^^xsd:string .

vargen:measuringActivity a prov:Activity , "DendrometerMeasurementActivity"^^xsd:string ;
    prov:atLocation var:tree .

vargen:measurementData a prov:Entity , "DendrometerMeasurement"^^xsd:string .

var:dendroPlan a prov:Entity , prov:plan .
```

This template contains the following errors: at line 15 of <>: Bad syntax (objectList expected) at ^ in: ".../openprovenance.org/var#> . vargen:bundle var:dendrometer^ a prov:Agent , "Dendrometer"^^xsd:string . vargen:measurin..."

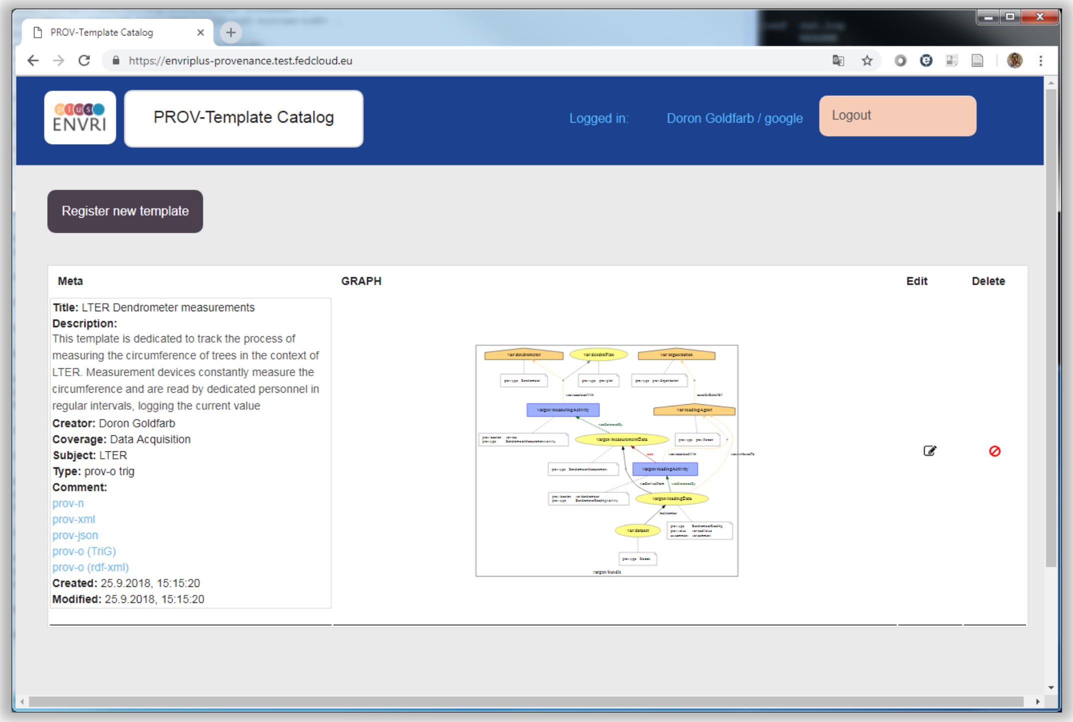
Fields marked with a * are required.

Figure 6: Error message for invalid PROV-template

List View

After successful registration, the template is shown in the list, as visible in Figure 7. All template entries belonging to the currently logged in user feature two dedicated buttons for modification or deletion. It should be noted that for one and the same user, his or her Google, GitHub or LinkedIn profiles are treated as separate accounts, i.e. templates created via a user's Google profile can only be modified by that profile, not by the same user's GitHub profile.

The metadata entered for each template is shown on the left hand side, including a creation and last modification date as well as links to download the stored template in various formats.



Meta	GRAPH	Edit	Delete
Title: LTER Dendrometer measurements **Description:** This template is dedicated to track the process of measuring the circumference of trees in the context of LTER. Measurement devices constantly measure the circumference and are read by dedicated personnel in regular intervals, logging the current value **Creator:** Doron Goldfarb **Coverage:** Data Acquisition **Subject:** LTER **Type:** prov-o trig **Comment:** prov-n prov-xml prov-json prov-o (TriG) prov-o (rdf-xml) **Created:** 25.9.2018, 15:15:20 **Modified:** 25.9.2018, 15:15:20			

Figure 7: View on template owned by currently logged in user

3 Expansion API

While the Web interface allows the registration and browsing of PROV templates, an accompanying API provides means to instantiate the templates with own bindings. Table 1 summarizes the main functionalities.

Table 1: Provided API function

URI pattern	Description
https://envriplus-provenance.test.fedcloud.eu/templates	returns json dict with template ID as key and basic metadata as value, including creator, title, description, created, modified
https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>	returns json dict containing all metadata and string representation of template and svg rendering for given template ID
https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>/[provn | provjson | provxml | trig | rdfxml]	Return template PROV doc as one of [provn | provjson |provxml | trig | rdfxml]
https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>/expand?fmt=[provn | provjson | provxml | trig | rdfxml] &writeprov=[true|false]& bindver = [v2 | v3] & bindings = <URLENCODED_BINDINGS >	HTTP GET based template expansion. Template identified via <templateID>, instantiation returned in form of <fmt>. Pass <bindings> as urlencoded string of version <bindver> via <bindings> parameter. Specify <writeprov> as true to write result into ENVRI TripleStore
https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>/e	HTTP POST based template expansion. Use URL to the left (Use all

xpand?fmt=[provn | provjson | provxml | trig | rdfxml] & bindver = [v2 | v3] &writeprov=[true|false]	parameters but <bindings> as above) and provide bindings data as payload.

The API currently provides two calls for listing/retrieving the stored content and two calls with dedicated functions. Regarding the latter, one function converts the stored template into a variety of PROV formats while the other one enables the expansion of stored templates via submitted bindings.

List all Templates

Templates are addressed via their ID, which can be looked up using the URL

https://envriplus-provenance.test.fedcloud.eu/templates

returning a JSON dictionary of all the registered templates with their ID as key.

Retrieve metadata for one stored template

Using a known template ID, calling URLs of the form

https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>

returns a JSON dict with all the data stored for the addressed template.

Convert stored template to PROV target format

Conversion can be initiated by calling URLs of the form

https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>/
[provn|provjson|provxml|trig|rdfxml]

appending one of the allowed PROV output formats accordingly.

Expand stored template

HTTP GET style

Expansion takes place by calling URLs of the form

https://envriplus-provenance.test.fedcloud.eu/templates/<templateID>/expand?
fmt=[provn|provjson|provxml|trig|rdfxml]&bindver=[v2|v3]&
bindings=<URLENCODED_BINDINGS>

using extra parameters specifying the output format of the expanded PROV document, the version of the bindings used and the bindings themselves, the latter must be provided in URLencoded form. While format and binding version default to “provn” and “v2”, the provision of a bindings file is mandatory.

Example

The URL provided below shows a working example for the template "LTER Dendrometer measurements" registered in the example above.

> https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/expand?fmt=provn&bindings=%0A%40prefix%20prov%3A%20%3Chttp%3A%2F%2Fwww.w3.org%2Fns%2Fprov%23%3E%20.%0A%40prefix%20tmpl%3A%20%3Chttp%3A%2F%2Fopenprovenance.org%2Ftmpl%23%3E%20.%0A%40prefix%20var%3A%20%3Chttp%3A%2F%2Fopenprovenance.org%2Fvar%23%3E%20.%0A%40prefix%20ex%3A%20%3Chttp%3A%2F%2Fexample.com%23%3E%20.%0Avar%3Adendrometer%20a%20prov%3AEntity%20%3B%0Atmpl%3Avalue_0%20ex%3A27%20.%0Avar%3Atree%20a%20prov%3AEntity%20%3B%0Atmpl%3A2dvalue_0_0%20ex%3A5001%20.%0Avar%3AdendroPlan%20a%20prov%3AEntity%20%3B%0Atmpl%3Avalue_0%20ex%3AdendrometerMeasurementMethodology%20.%0Avar%3Aorganization%20a%20prov%3AEntity%20%3B%0Atmpl%3Avalue_0%20ex%3ANP_Kalkalpen%20.%0Avar%3AreadingAgent%20a%20prov%3AEntity%20%3B%0Atmpl%3Avalue_0%20ex%3AKettenhummer%20.%0Avar%3Acomment%20a%20prov%3AEntity%20%3B%0Atmpl%3A2dvalue_0_0%20%22%22%20.%0Avar%3AreadValue%20a%20prov%3AEntity%20%3B%0Atmpl%3A2dvalue_0_0%20%227%22%20.%0Avar%3AendDate%20a%20prov%3AEntity%20%3B%0Atmpl%3Avalue_0%20%222018-05-09T00%3A00%3A00%22%20.%0Avar%3Adataset%20a%20prov%3AEntity%20%3B%0Atmpl%3Avalue_0%20ex%3A2018-05-09T00_00_00%20.

The bindings provided above are the URLencoded version of the following information:

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix ex: <http://example.com#> .

var:dendrometer a prov:Entity ;
        tmpl:value_0 ex:27 .
var:tree a prov:Entity ;
        tmpl:2dvalue_0_0 ex:5001 .
var:dendroPlan a prov:Entity ;
        tmpl:value_0 ex:dendrometerMeasurementMethodology .
var:organization a prov:Entity ;
        tmpl:value_0 ex:NP_Kalkalpen .
var:readingAgent a prov:Entity ;
        tmpl:value_0 ex:Kettenhummer .
var:comment a prov:Entity ;
        tmpl:2dvalue_0_0 "" .
var:readValue a prov:Entity ;
        tmpl:2dvalue_0_0 "7" .
var:endDate a prov:Entity ;
        tmpl:value_0 "2018-05-09T00:00:00" .
var:dataset a prov:Entity ;
        tmpl:value_0 ex:2018-05-09T00_00_00 .
```

HTTP POST style

The following python code provides an example for how to use the expansion API via POST requests without writing the result into the Triple Store.

```
import requests
templateID="5baa34e8d6fa333791066dcf"

bindings="""
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix ex: <http://example.com#> .
var:dendrometer a prov:Entity ;
tmpl:value_0 ex:27 .
var:tree a prov:Entity ;
tmpl:2dvalue_0_0 ex:5001 .
var:dendroPlan a prov:Entity ;
tmpl:value_0 ex:dendrometerMeasurementMethodology .
var:organization a prov:Entity ;
tmpl:value_0 ex:NP_Kalkalpen .
var:readingAgent a prov:Entity ;
tmpl:value_0 ex:Kettenhummer .
var:comment a prov:Entity ;
tmpl:2dvalue_0_0 "" .
var:readValue a prov:Entity ;
tmpl:2dvalue_0_0 "7" .
var:endDate a prov:Entity ;
tmpl:value_0 "2018-05-09T00:00:00" .
var:dataset a prov:Entity ;
tmpl:value_0 ex:2018-05-09T00_00_00 .
"""

r = requests.post(https://envriplus-provenance.test.fedcloud.eu/templates/ + templateID + \
"/expand?fmt=provjson&writeprov=false",
                    data= bindings
)

print repr(r.text)
print repr(r.headers)
```

4 Querying stored provenance

A dedicated SPARQL endpoint can be used to query Provenance documents stored during expansion when the <writeprov> Parameter has been set to true.

To query the store, a HTTP GET request needs to be sent with the desired SPARQL query embedded in the request URL, where <encoded query> is the SPARQL query encoded for insertion into a URL:

```
GET http://oil-e.vlan400.uvalight.net/prov/query?query=<encoded query>
```

The desired format for the response and the host from which the request is being sent should ideally be specified within the request header:

```
Host: <your hostname>

Accept: <your desired format>
```

A list of common formats is provided below:

```
Accept: application/sparql-results+xml (for XML, the default for SELECT queries)
Accept: application/sparql-results+json (for JSON, appropriate for SELECT queries)
Accept: text/turtle (for Turtle, the default for CONSTRUCT queries)
Accept: application/x-turtle (alternative media type for Turtle)
Accept: application/ld+json (for JSON-LD, appropriate for CONSTRUCT queries)
Accept: application/rdf+xml (for RDF/XML, appropriate for CONSTRUCT queries)
```

The provenance store is set up such that the default graph is the union of all named graphs (so one can query across all provenance bundles). To query a particular provenance bundle, the named graph to which the data was uploaded needs to be specified. This is done using a FROM or FROM NAMED statement in a SPARQL query as normal.

5 References

[Moreau, 2018] L. Moreau, B. V. Batlajery, T. D. Huynh, D. Michaelides, and H. Packer, 'A Templating System to Generate Provenance', IEEE Transactions on Software Engineering, vol. 44, no. 2, pp. 103–121, Feb. 2018.

[Goldfarb, 2018] D.Goldfarb, 'EnvriPlus PROV-Templates', 2018