D8.6
Data provenance and tracing for environmental sciences: Prototype and deployment

WORK PACKAGE 8 – Data Curation and Cataloging

LEADING BENEFICIARY: EAA (Umweltbundesamt GmbH)

Author(s):	Beneficiary/Institution
Doron Goldfarb	EAA (Umweltbundesamt GmbH/LTER)
Barbara Magagna	EAA (Umweltbundesamt GmbH/LTER)
Stephan Kindermann	DKRZ (Deutsches Klimarechenzentrum GmbH/IS-ENES)
Markus Stocker	TIB/PANGAEA
Dan Lear	MBA (Marine Biological Association of the United Kingdom/DASSH)
Kevin Paxmann	MBA (Marine Biological Association of the United Kingdom/DASSH)
Carl-Fredrik Enell	EISCAT (EISCAT Scientific Association)
Rikard Slapak	EISCAT (EISCAT Scientific Association)
Paul Martin	UvA (niversiteit van Amsterdam)
Spiros Koulouzis	UvA (Universiteit van Amsterdam)
Christian Pichot	INRA (Institut National de la Recherche Agronomique/ANAEE)

Accepted by: Keith Jeffery, NERC, WP 8 lead

Deliverable type: Demonstrator

Dissemination level: PUBLIC

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 agreement No 654182

Deliverable due date:	31.10.2018/M42
Actual Date of Submission:	31.10.2018/M42

ABSTRACT

The individual data life cycles of ENVRIplus RIs are often composed of quite heterogeneous workflows, ranging from fully automated to almost fully manual procedures. This deliverable presents a prototypical implementation of a service designed to enable the tracking of provenance at the desired granularity in a harmonized form despite of the heterogeneity of the encountered landscape. Individual use-cases contributed by different RIs discuss the applicability of the chosen approach and provide valuable feedback for further consideration.

Project internal reviewer(s):

Project internal reviewer(s):	**Beneficiary/Institution**
Malcolm Atkinson	UEDIN
Margareta Hellström	LU

Document history:

Date	**Version**
2nd October 2018	Initial Draft
8th October 2018	Google Docs version
29th October 2018	Review from Margareta Hellström
31th October 2018	Review from Malcolm Atkinson

DOCUMENT AMENDMENT PROCEDURE

Amendments, comments and suggestions should be sent to the authors (Doron Goldfarb, doron.goldfarb@umweltbundesamt.at, Barbara Magagna, barbara.magagna@umweltbundesamt.at)

TERMINOLOGY

This deliverable uses terminology based on the ENVRI Reference Model [Nieva de la Hidalga et al. 2017], which is published online as an ontology: http://www.oil-e.net/ontology/envri-rm.owl

A complete project glossary is provided online https://wiki.envri.eu/pages/viewpage.action?pageId=14452608 and in Appendix A.

PROJECT SUMMARY

ENVRIplus is a Horizon 2020 project bringing together Environmental and Earth System Research Infrastructures, projects and networks together with technical specialist partners to create a more coherent, interdisciplinary and interoperable cluster of Environmental Research Infrastructures across Europe. It is driven by three overarching goals: 1) promoting cross-fertilization between infrastructures, 2) implementing innovative concepts and devices across RIs, and 3) facilitating research and innovation in the field of environment for an increasing number of users outside the RIs.

ENVRIplus aligns its activities to a core strategic plan where sharing multi-disciplinary expertise will be most effective. The project aims to improve Earth observation monitoring systems and strategies, including actions to improve harmonization and innovation, and generate common solutions to many shared information technology and data related challenges. It also seeks to harmonize policies for access and provide strategies for knowledge transfer amongst RIs. ENVRIplus develops guidelines to enhance transdisciplinary use of data and data-products supported by applied use-cases involving RIs from different domains. The project coordinates actions to improve communication and cooperation, addressing Environmental RIs at all levels, from management to end-users, implementing RI-staff exchange programs, generating material for RI personnel, and proposing common strategic developments and actions for enhancing services to users and evaluating the socio-economic impacts.

ENVRIplus is expected to facilitate structuration and improve quality of services offered both within single RIs and at the pan-RI level. It promotes efficient and multi-disciplinary research offering new opportunities to users, new tools to RI managers and new communication strategies for environmental RI communities. The resulting solutions, services and other project outcomes are made available to all environmental RI initiatives, thus contributing to the development of a coherent European RI ecosystem.

EXECUTIVE SUMMARY

The requirements gathering activities from the involved RIs, outlined in D8.5, revealed a very heterogeneous landscape of data infrastructures in ENVRIplus, especially with respect to the level of automatization across different RIs and the various steps within their individual data life cycles. These insights made it clear that any approach to an ENVRIplus-wide provenance related service would need to cater for the encountered heterogeneity in an optimal way.

This deliverable presents a prototype for an ENVRIplus provenance service designed to cover the needs of such a wide variety of stakeholders by following an existing concept called PROV-Template. The approach is targeted at supporting interested parties in the adoption of a common data model for describing data provenance, PROV-DM, by providing a means to convert legacy and/or proprietary log data into the desired form. This is achieved via so-called templates which are a-priori representations of the final provenance data that should essentially be instantiated during the repeated execution of the underlying workflows, resulting in the desired provenance trace for each run.

The report discusses the rationale behind the chosen approach and describes its implementation, which includes a catalog to register templates, also serving as a common platform for potential reuse between different parties, and an attached service to instantiate registered templates with information from individual workflow runs. The latter is enabled by a custom implementation of a Python library which proved to be more flexible than using the only existing, Java-based, implementation.

A significant part of the deliverable covers descriptions of individual use-cases contributed by different ENVRIplus RIs, providing valuable community feedback gained from detailed individual implementation attempts and the resulting conclusions. The collected feedback suggests that the chosen approach is of interest to the community and has a number of advantages, although its use within a production setting would still require specific modifications/extensions of existing workflows. Nevertheless, this would not relieve RIs from meeting basic requirements such as having to provide structured descriptions of entities involved in the provenance traces to be tracked.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Provenance related activities

The task 8.3 "Inter RI data provenance and trace services" aims at introducing scientists and data architects to the topic of provenance. The task has to produce a report D8.5 ("Data provenance and tracing for environmental sciences: system design") and a prototype demonstrator D8.6 ("Data provenance and tracing for environmental sciences: prototype and deployment").

The activities for this task started with the requirements gathering in autumn 2015, which was organized around Task 5.1 Reference model guided RI design [ENVRIplus 2015]. A specific provenance related questionnaire was developed and distributed to the Research Infrastructures to collect provenance specific requirements. Although the feedback was rather sparse, it made clear that there was a broad interest to get more insights into this topic. An initial state of the art analysis on provenance was undertaken and incorporated in D5.1 [Atkinson et al., 2016]. However the main activities started in autumn 2017 with a virtual Kick-Off meeting in October, which had the objective to prepare the Provenance Workshop held at the 6th ENVRIweek in Malaga in November. This workshop established a work plan with actions and contributions from RIs and ICT partners. Since then a template for requirements and use case gathering was developed, distributed to several RIs and the responses were collected during the following months. Several teleconferences with provenance practices presented by EPOS/DARE and IS-ENES representatives followed. The 8.3 working team (ENVRI provenance group) participated in the regular teleconferences of RDA Provenance Patterns Working Group, the Workshop RDA-Europe Data Provenance Approaches held in Barcelona in January 2018, and the related sessions at the RDA Plenary meetings in Barcelona (April 2017) and Berlin (March 2018). At the site visits of EPOS in Rome in September 2017, of ICOS in Lund in December 2017 and of EISCAT/D4Science in Pisa in February 2018 ICT specific provenance requirements and implementations were presented and discussed. An implementation case (Use Case TC_17 Particle formation[1], see also [Chen et al. 2017]) was selected to demonstrate how provenance management can be applied. During the ENVRIweek in Zandvoort in May 2018 the outline of the demonstrator development was settled. RIs interested in testing the prototype within their individual use cases were identified and involved in the development.

As there is no one-size-fits-all system for all domains and application areas, D8.5 [Magagna et al., 2018], published in April 2018, gives basic insights into relevant aspects of provenance management needed to design provenance systems according to the explicit requirements of specific RIs. It provides a thorough analysis of the state of the art and provenance models in use. The report takes into account common requirements of RIs throughout the entire data lifecycle: from acquisition, curation, publishing, processing to use. It indicates standardised interfaces for querying, accessing and integrating provenance data and investigates emerging provenance services for e-Infrastructure projects. Furthermore, it builds upon the semantic linking framework developed in ENVRIplus Task T5.3 [Martin et al., 2018] reusing existing standards, such as W3C's PROV for possible general interoperability.

The current deliverable D8.6 describes the prototyping of a specific service that can be used in a flexible way by RIs covering mainly provenance acquisition aspects. Interested RIs were involved

[1] https://confluence.egi.eu/pages/viewpage.action?pageId=30737206

in the development phase and tested the service for its usability. Their different use cases are described demonstrating the benefit of such a service.

2 REQUIREMENTS SUMMARY

At the beginning of the development of any RI, it is important to understand the existing situation and conditions of the RI architecture related to technology of interest. To study this a particular emphasis was laid during the preparation of D8.5 [Magagna et al., 2018] on the collection of RI specific use cases and requirements regarding data provenance aspects. In the context of ENVRIplus task T8.3 use cases are described as a set of interactions between a system and and one or more actors. They specify details of a system functionality from a user perspective. For each use case more than one requirement can be listed. Requirements have a functional perspective, here the problem is approached from a solution angle which aims towards a complete developer specification. As already described this process covered two phases and resulted in a ENVRI synopsis of both use cases and requirements. Additionally, a comparison was performed with the provenance use cases provided in the Provenance Patterns Database[2] emerged from the activities of the RDA Provenance Patterns Working Group. Many of the RI use cases found related ones in the database. Table 1 lists those ENVRI use cases which are without any relation to existing ones and are thus included as new use cases in the Provenance Patterns Database:

Table 1: Use Cases introduced in the Provenance Patterns Database (a detailed description of the listed use cases can be found in chapter 3 of D8.5 [Magagna et al., 2018]

UC Nr	Name
LTER.U1.1	Non-automated data collection-observation
LTER.U1.2, ICOS.U2	Non-automated data collection-physical samples
LTER.U2	Track lineage for heterogeneous data sources via publications
LTER.U3	Track lineage for ad-hoc workflows combining scientific scripts and third party software
EPOS.U3	Monitoring
EPOS.U4	Preview and staging
EPOS.U6	Configuration of interdependent tasks (data reuse)
EPOS.U9	Selective generation of traces

[2] https://patterns.promsns.org/

The ENVRI technology landscape is characterized by a highly varying level of automation. Some RIs are built on fully automated sensor networks where human intervention is mostly limited to monitoring, interpretation and maintenance tasks. Other RIs in turn include significantly more manual steps, taking place in data acquisition, exchange, or even processing. This diversity is clearly reflected in the use cases reported by the different RIs.

In less automated settings, reported use cases appear more targeted towards the different aspects of provenance collection itself such as tracking lineage for script based workflows, provenance for automated and manual data acquisition such as human observation and physical sample based data collection, and provenance for data publishing and (re)use. In more automated settings, provenance tracking can partly be solved within the workflow management systems, which support the harmonization between distributed provenance archives into a common representation to enable provenance summaries of heterogeneous workflows. Such interoperability issues pose a challenge for large-scale provenance tracking.

Apart from provenance tracking services, a central requirement common to many RIs is the provisioning of various types of registries, since tracking provenance for processes with different agents and entities usually requires unique identifiers for each involved instance. Registries for persons, data objects, measurement devices etc. can thus be almost considered a prerequisite for meaningful provenance approaches. These registries should allow the use of PIDs (pointing to other registries) in their kernel metadata schemas, paving the way towards an overall Digital Object Network-compliant system architecture (for EOSC).

3 PROVENANCE TEMPLATES

As stated in Deliverable D8.5 [Magagna 2018], chapter 6.3 “Provenance and the ENVRI architecture“, the heterogeneous nature of environmental and Earth Science RIs stands in the way of applying one single provenance “system” across all involved communities, which has led us to instead focus on providing recommendations to foster the adoption of standard provenance data models and approaches in order to increase interoperability. As far as the shared development of dedicated technical solutions was concerned, the suggestion was to seek for approaches with broadest possible applicability across all stakeholders. In the context of task T8.3, it was thus decided to find a prototypical solution considering these constraints, balancing the intended contribution as golden mean between providing a pure standards recommendation and aiming at a rather monolithic and inflexible software solution.

As far as standards for describing provenance are concerned, PROV-DM[3] has emerged as a widely accepted basis which has been extended for more domain specific purposes. It thus stands to reason to use this model as a point of reference. Regarding a prototypical ENVRIplus service enabling the creation of PROV-DM compliant provenance traces for a broadest possible user base, the challenge was to find an approach able to take heterogeneous workflows across (and within) RIs into account, ranging from purely manual steps to highly automated computational workflows.

As it turned out, a potentially useful approach to that challenge already existed within the PROV-DM ecosystem: Mentioned in the technology review chapter 2.3 of Deliverable 8.5, the PROV-Template framework, described by Moreau at al. in [Moreau et al., 2018], suggested to be able

[3] https://www.w3.org/TR/prov-dm/

to meet the requirements by offering a way to outsource the generation of PROV-compliant provenance documents from the underlying workflows. This could be achieved by specifying *templates* describing the intended future structure of the PROV documents to be generated and instantiating them accordingly at runtime using so-called bindings. A PROV-template thus represents a valid PROV document, with the difference that its elements are declared as variables to be substituted with actual values at a later point in time. The above mentioned *bindings* are essentially a mapping between variables and their respective substitute values which is created anew for each run of the workflow for which the provenance template was created.

Given that in the most cases, the alternative would be to directly create the respective PROV compliant output within the workflow to be traced, this approach has a number of advantages:

- **"Provification" of legacy workflow output**

  In many cases, existing workflows, whether manual or automated, generate some form of protocol or log. It can safely be assumed that the basic roles of the entities mentioned in such legacy traces often resemble agents, activities or entities as specified in the PROV-DM. PROV-Templates thus enable the mapping of these legacy traces to PROV and at the same time also support their semantic annotation by making explicit the roles of the involved entities.

- **More flexible handling of changes to the created PROV documents**

  PROV-Template could be useful even in "non-legacy" settings, since in many cases, required changes to the PROV output, such as a reduction of granularity of the tracked information, could be done directly in the template without having to touch the workflow or its output itself.

Considering the timeframe of Task 8.3 in the context of the ENVRIplus project, choosing the PROV-template based approach had additional advantages:

- **Exploration of PROV as provenance data model with added value**

  With PROV being one of the de-facto standards for describing provenance, it was considered a useful exercise to explore modelling RI specific aspects with this data model. Usually, such experiments generate a number of artifacts which are helpful for establishing know-how but can not be used further. Using the PROV-template approach allows experimenting with the PROV model while at the same time the resulting PROV-templates can be potentially reused in future settings to generate actual PROV data.

- **Start working with PROV regardless of not yet implemented infrastructure elements**

  As outlined in the different RI requirements presented in Deliverable 8.5 [Magagna 2018], many infrastructure elements essential to a usable provenance architecture were not yet implemented in most of the RIs; most prominently, this was the case for various registries such as for person identifiers, datasets, etc. Using PROV-templates enabled to start working on the provenance infrastructure without having to wait for their implementation.

3.1 Basic concept

Figure 1 provides a visual rendering of a simple PROV-Template example, its PROV-O[4] based RDF representation using the TriG[5] syntax is shown in Listing 1. PROV-templates are always valid PROV-documents, where the identifiers for the involved elements are represented by variables. The latter are declared as such by using a special namespace "http://openprovenance.org/var#" with prefix "var:", a related namespace "http://openprovenance.org/vargen#" with prefix "vargen:" allows the declaration of variables which are instantiated with a randomly generated ID at runtime.

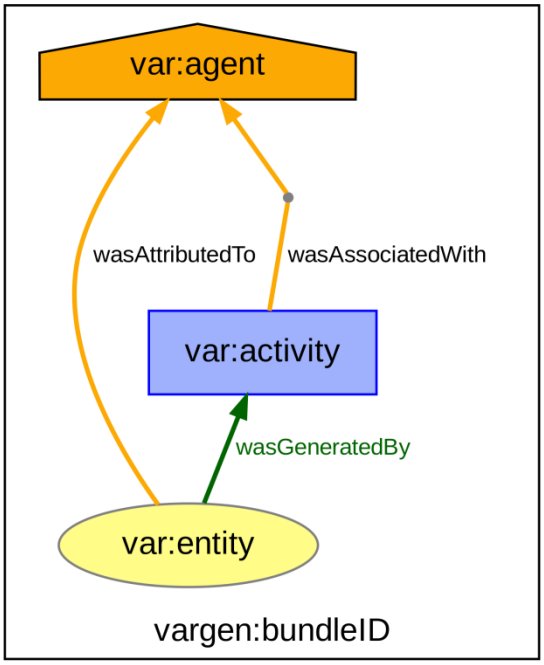

Figure 1: Basic PROV-Template example visualization

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix vargen: <http://openprovenance.org/vargen#> .

vargen:bundleID {
      var:agent a prov:Agent .
      var:activity a prov:Activity .
      var:entity a prov:Entity .
      var:activity prov:wasAssociatedWith var:agent .
      var:entity prov:wasGeneratedBy var:activity .
      var:entity prov:wasAttributedTo var:agent .
}
```

Listing 1: Basic PROV-Template example

The bindings used to expand templates can be specified using two different "flavors". The first, initially proposed variant is itself again a valid PROV document, featuring variables as entities and mapping them to actual values via a third dedicated namespace

[4] https://www.w3.org/TR/prov-o/
[5] http://www.w3.org/TR/trig/

"http://openprovenance.org/tmpl#" with prefix "tmpl:". Listing 2 shows an example for such a binding, further documentation is provided in the PROV-Template specification[6] (Section 4: Set of Bindings). Depending on the type of variable, the substituted values can be literals or URIs, the latter ideally resolving to meaningful additional information retrievable via the Web. The example below uses an explicit, non-functional namespace "http://example.com/" with prefix "ex:" for such URIs.

```
@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:agent a prov:Entity ;
  tmpl:value_0 ex:agent123 .
var:activity a prov:Entity ;
  tmpl:value_0 ex:activity123 .
var:entity a prov:Entity ;
 tmpl:value_0 ex:entity123 .
```

Listing 2: Example bindings for basic PROV-Template example expressed as PROV document

The same binding as presented in Listing 3 can also be expressed in a more recent and less complex non-PROV json format, referred to as "v3" bindings, a simple example for which is provided in Listing 3. Instead of being declared in the header, the used prefixes are specified in a dedicated "context" section. Documentation for this form of bindings specification is provided in the supplementary material[7] for [Moreau et al., 2018].

```
{
 "var": {
    "agent": [{"@id": "ex:agent123"}],
    "activity": [{"@id": "ex:activity123"}],
    "entity": [{"@id": "ex:entity123"}]
    },
 "context": {
    "ex": "http://example.com/",
    "var": "http://openprovenance.org/var#"
 }
}
```

Listing 3: Alternative representation of bindings as JSON document

Having a template and suitable bindings at hand, these two components can be processed into an instantiated PROV document. This process is called *expansion* and is illustrated below in Figure 2. Due to the separation, this setting suggests itself as a usable approach to creating PROV compliant data from within heterogeneous infrastructures and Data Life Cycle (DLC) steps.

[6] https://provenance.ecs.soton.ac.uk/prov-template/#environment

[7] https://ieeexplore.ieee.org/ielx7/32/8289825/7909036/tse-moreau-2659745-mm.zip?tp=&arnumber=7909036

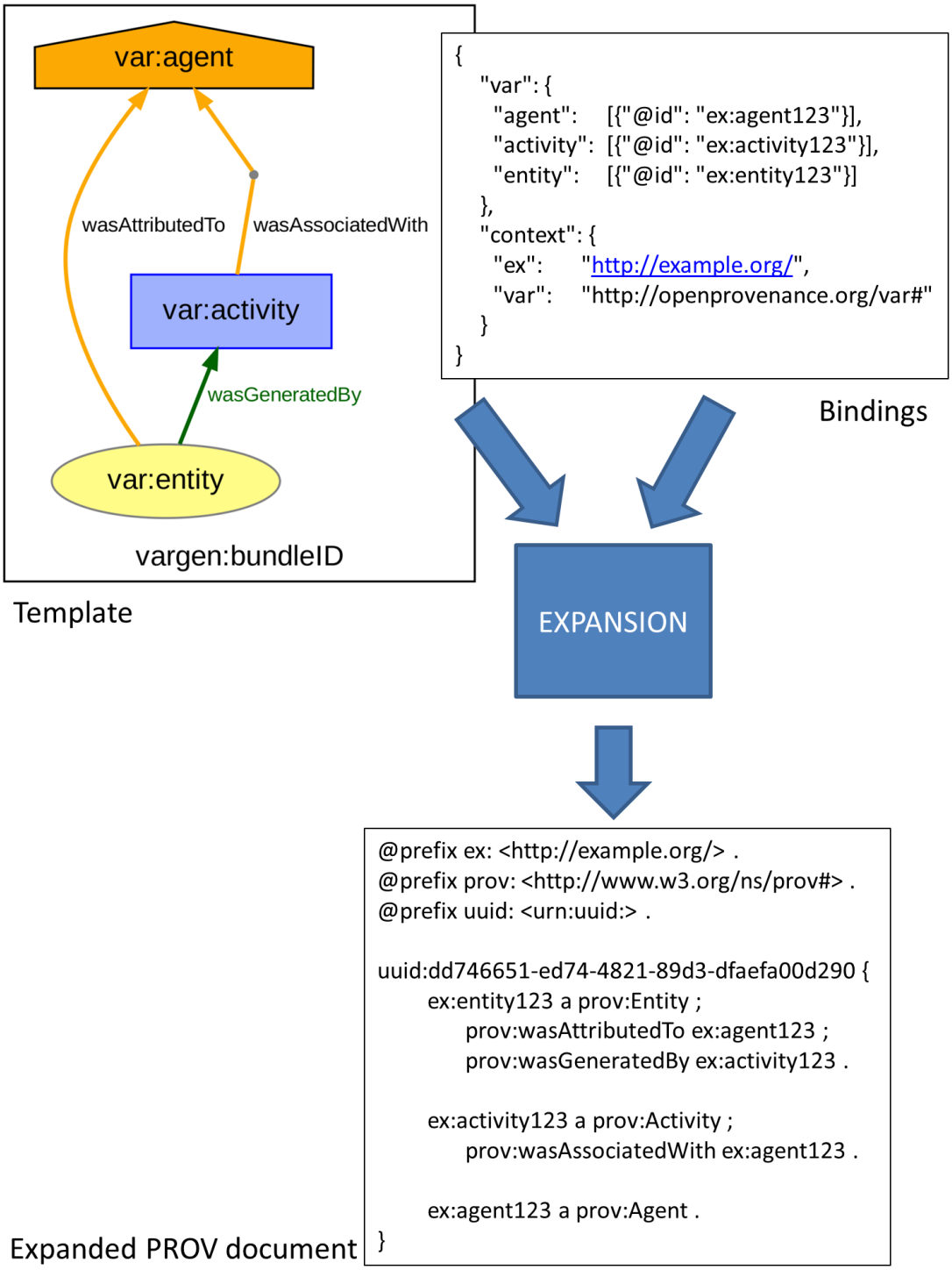

Figure 2: Illustration of PROV-Template expansion process

3.2 PROV templates in the context of ENVRIplus

For the reasons outlined above, it was decided to explore the possibilities of PROV-Templates in the context of the ENVRIplus task T8.3. This involved two aspects, the first of which was to set up a proof-of-concept service for template registration, storage and expansion. The second aspect was to ask involved communities to represent provenance specific aspects of their Data Life Cycles (DLC) using PROV-Templates. This way, they could experiment with the PROV-DM as

language to express provenance for their infrastructure and to assess what steps would be necessary to create PROV compliant data out of the respective workflows. The contributors were moreover asked to register their created templates in the prototype infrastructure so that they would become available for inspiration and reuse. Whenever possible, contributors were also encouraged to experiment with the provided expansion functionality and to assess the usability of such a service within their own RI.

3.2.1 Demarcation

Besides its application in the context of the PROV-Template framework, the notion of templates has already been used in the context of provenance, referring to related but nevertheless different things. It is thus of importance to clearly demarcate the approach followed here from the existing ones.

A general overview on the idea of PROV-Template as applied in this deliverable is given in [Moreau et al., 2018], including its positioning with respect to existing approaches in this regard. The authors especially sought to distinguish PROV-Template from the concept of “prospective provenance”, which refers to the provision of a-priori recipes for executing specific workflows, while provenance templates instead “only” refer to a-priori recipes for how the provenance data should be represented.

Provenance templates are mentioned in EUDAT2020 deliverable D8.3 “Report on Design Model and Definition of Data Directives” [Le Franc et al., 2018] as means for modeling data life cycles in the context of the EUDAT CDI. The understanding of provenance templates in this context, however, is to rather use them as means to describe prospective provenance for the declaration of data management workflows.

3.2.2 Embedding provenance within the RI architecture

Provenance information such as created via the ENVRI Provenance Service can be seen as metadata which only contains information describing the lineage of data or other assets in general (including methods, encodings, working practices, software and services).

Traditionally, this information is included in the metadata record itself. “Metadata are often classified by their purpose, including descriptive metadata, administrative metadata, and structural metadata as the most common sub-classifications. Rights management (terms and conditions), provenance, and preservation metadata are most often subcategorized under administrative metadata" (Socha et al., 2013].

Indeed, many metadata systems have legacy methods for delivering provenance such as free text fields (e.g. dc:source, in D-CAT metadata documents) or at best provide structured machine-readable, but not PROV-compliant provenance (e.g. LI-Lineage, in ISO19115-2 metadata documents).

This method is no longer considered best practice, since such provenance information cannot easily be machine-read and elements of it cannot be reused efficiently by other IT components (besides simply copying the provenance information completely[8]). This is actually the case for most metadata records available today. Even if it is stored in formats (like XML) that are in themselves machine-interpretable, there are enough not-so-well-documented and/or

[8] https://patterns.promsns.org/pattern/12

unregistered schemata in use so that a majority of datasets remain impossible to be processed even in a semi-autonomous way.

Several studies indicated that many scientists spend 80% of their time on "data wrangling", including metadata hunting, format conversions and basic quality checking [Wittenburg and Strawn 2018]. This is the main reason why we recommended the provision of information on the lineage of data in a separate record following a standard like PROV.

In the context of the PROV ecosystem, a dedicated specification called PROV-AQ[9] (Access & Query) suggests different ways of linking provenance information to resources made available via the Web. Assuming a dataset described and linked to by a metadata record in some catalog to be the most common situation relevant to provenance provision in the ENVRI context, the situation sketched by PROV-AQ would be as follows: Figure 3 represents a “legacy” situation where a catalog entry for a dataset consists of a metadata record featuring a number of fields including provenance information provided in unstructured form e.g. as free text. The metadata record contains a HTTP URI pointing to the dataset whose retrieval via that URI is handled by a dedicated service.

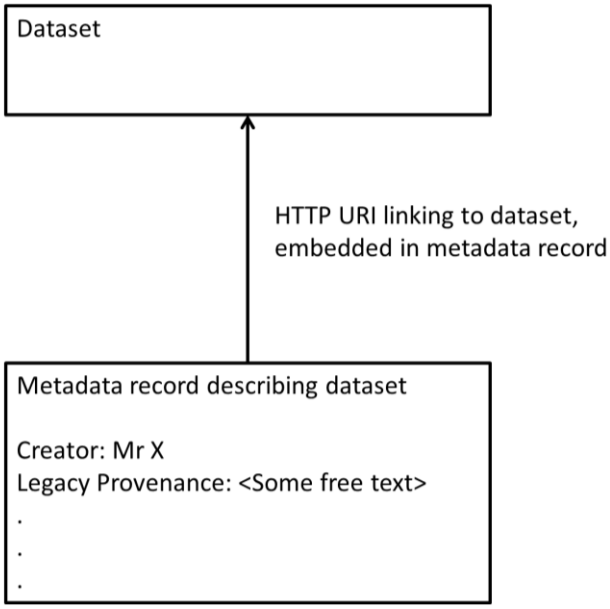

Figure 3: Metadata record describing and pointing to a dataset, “legacy” provenance information present as free text

PROV-AQ outlines a relatively non-intrusive approach to extending such configurations to be able to provide structured provenance information alongside the actual content, sketched in the PROV-AQ specification section 3.1.2. The basic idea is that when the dataset URI is requested by the client, the server in addition returns a dedicated HTTP <link> header which contains a property called “has_provenance” offering a link to the PROV document associated with the dataset. Figure 4 provides the basic process behind this approach. The link to the provenance

[9] http://www.w3.org/TR/prov-aq/

document can consist of a "simple" file link or represent a fully specified GET request to an API, e.g. a query to a HTTP SPARQL API in front of a provenance triple-store.

PROV-AQ also specifies more complex scenarios, such as extending the sketched procedure with a content negotiation approach similar to mechanisms used to serve Linked Data, or more explicitly describing provenance query services. The discussion of such advanced approaches, however, lies beyond the scope of this deliverable.

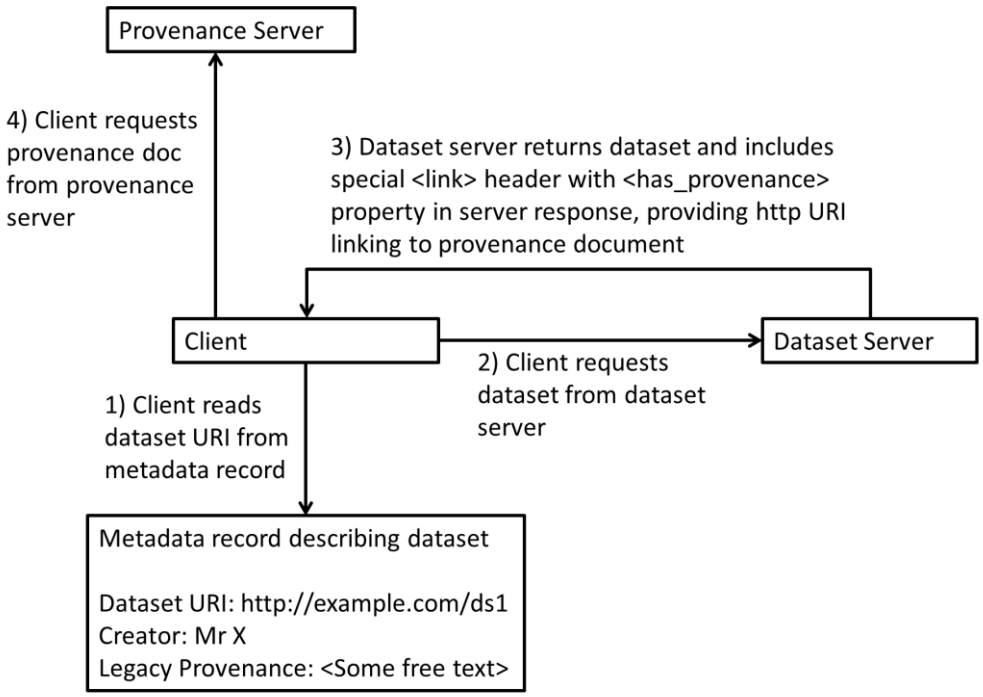

Figure 4: Retrieving provenance attached to dataset vialink provided in server response header

Apart from the approach described above as recommended by the PROV community there is currently a lot of discussion (GEDE group[10], C2CAMP initiative[11]) about building "Digital Object Networks" (DONs)[12] that will support the "cross-linking" of digital objects with their related metadata. The latter could be potentially stored in a large number of separate services (catalogs, registries or repositories) but would remain interconnected via their PIDs. In order to achieve this, the operational instances of all the necessary components — such as the provenance service described here, but also any instance or type registry, vocabulary service, trusted data catalogue and data repository — would need to support the assignment of respective Handles to all relevant items, so that they can be tied together in a both machine- and human-readable way.

ENVRI-FAIR, the follow up ENVRIplus project, will provide substantial improvements in this respect, also building on the service proposed in this deliverable so that RIs can make a concentrated effort to hook up to this machinery, exposing their data holdings (and other

[10] https://www.rd-alliance.org/groups/gede-group-european-data-experts-rda
[11] https://www.go-fair.org/implementation-networks/overview/c2camp-in/
[12] https://rd-alliance.org/group/gede-group-european-data-experts-rda/wiki/first-gede-do-workshop-september-18

research assets) and metadata in an interoperable way. These efforts must be done in close cooperation with the maintainers of the appropriate standards (DC, DCAT, CKAN, PROV, etc.) and be promoted under the EOSC umbrella so that a large community can take advantage of these common developments.

4 PROV-TEMPLATE BASED ENVRI PROVENANCE SERVICE PROTOTYPE

This chapter presents a prototype for a PROV-Template based solution which could potentially serve as (part of) a future ENVRIplus provenance service or — on the long term — even as a domain and community independent EOSC offering. The underlying rationale was that PROV-templates are a flexible solution which can be adapted to different existing infrastructures but would at the same time benefit from a set of centralized components. The latter would include a registry for the individual templates, a Web service for expanding them with bindings at runtime and storing the resulting provenance documents in a common database. Especially a common registry for templates was expected to be beneficial, serving as point of contact and potential reuse between different RIs. Directly attaching an expansion service to the registry would then make sure that the registered templates could immediately be used for instantiation, having the additional advantage that providing a single implementation of an expansion algorithm would remove a potential source for inconsistencies in the output. A common storage for generated PROV documents would in turn enable RIs to start tracking and using PROV compliant provenance data even without having to set up and maintain the required infrastructure themselves. Last but not least, such an integrated approach to PROV-template (catalog, expansion service, provenance storage) would also generally represent a novel contribution beyond the immediate context of ENVRIplus.

Following the rationale outlined above, the basic architecture of the implemented prototype is shown in Figure 5. A dedicated MongoDB[13] instance, shown at the left center bottom of the Figure, serves as storage for registered PROV-Templates and is accessible via a Web Service shown above in the Figure, implemented using Python and the Flask[14] framework. The Web service offers an API with basic (CRUD) functions for **c**reating, **r**eading, **u**pdating and **d**eleting stored templates, the latter two requiring authentication. This is provided by the inclusion of the Authomatic[15] library which allows using existing Social Media profiles supporting the Oauth protocol, such as Google, GitHub or LinkedIn, for authentication without having to implement this functionality by oneself.

A dedicated Web interface, implemented using vue.js[16] and shown at the left top of Figure 5, uses the CRUD API for managing and browsing templates. It allows users to authenticate using one of the above mentioned Social Media profiles and to subsequently register and update their PROV-Templates. Registration currently uses Dublin Core Metadata elements for describing new templates and provides a basic validation mechanism, mainly checking for a syntactically correct PROV style, which is required to be passed before new templates can be added. Not requiring authentication, the main view provides a list of all currently registered templates including their metadata and SVG[17] renderings of their content. If a user is currently authenticated with his or

[13] https://www.mongodb.com/
[14] http://flask.pocoo.org/
[15] http://authomatic.github.io/authomatic/
[16] https://vuejs.org/
[17] https://www.w3schools.com/graphics/svg_intro.asp

her Social Media profile, entries for those templates created by that user additionally feature buttons for their update or deletion. The service is available online, hosted by EGI, using the temporary URL https://envriplus-provenance.test.fedcloud.eu/ and the source code for back- and frontend available on Github at https://github.com/EnvriPlus-PROV/ProvTemplateCatalog.

A manual for using the service is provided via a dedicated link on the front page of the service and also via the URL https://www.envri.eu/provenancetemplates. The latter URL will also serve as access point to the service if the temporary URL will possibly become obsolete.

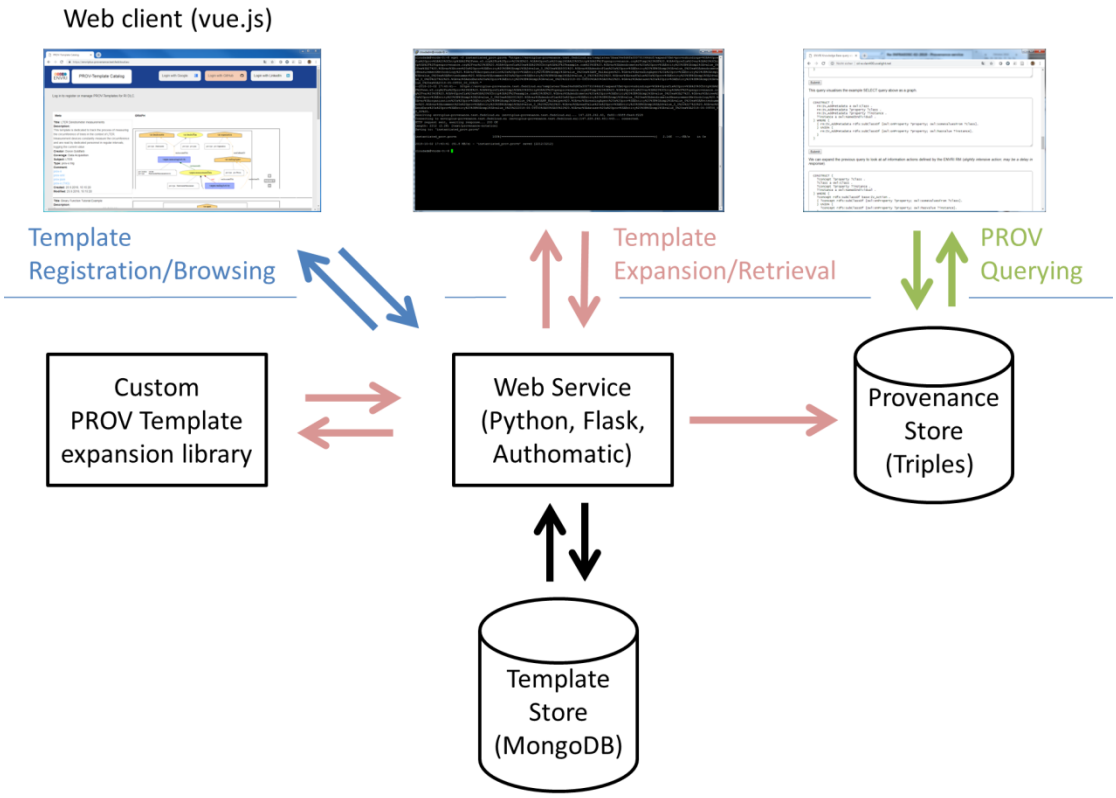

Figure 5: Prototype architecture

Listing 4 shows an example entry for the information stored for each template in the MongoDB template store. The automatically generated ID also serves as external reference to the stored template, used in API calls for example. As mentioned before, the user-editable metadata fields for describing templates follow the Dublin Core standard[18]. While they, besides the "type" field, currently allow free text entries, the field "coverage", and probably also the fields "subject" and "creator", should eventually reference controlled vocabularies / person registries in future. The fields "prov" and "provsvg" are in turn dedicated to storing the originally registered PROV content and the converted SVG representation[19], the latter mainly for performance reasons in order to avoid the necessity of repeated conversion. The "retr_url_<XYZ>" fields serve as shortcuts to API calls for retrieving the stored template in various formats. Last but not least, the "owner" dictionary stores information about the third party social media profile used for

[18] http://dublincore.org/specifications

[19] SVG stands for Scalable Vector Graphics, an XML based vector image format which can be displayed in modern browsers, https://www.w3.org/TR/SVG11

authentication during the registration of the template at hand, its “userid” key stores the ID returned by the used profile, while the “siteid” key refers to an enumerated list of available third party profiles for authentication, read from a service configuration file[20].

```
_id:              ObjectId("5baa34e8d6fa333791066dcf")
comment:          ""
description:      This template is dedicated to track the process of measuring …
creator:          Doron Goldfarb
coverage:         Data Acquisition
owner:             {
                        siteid: 1
                        userid: <USER-ID-HERE>
                   }
provsvg:          "<?xml version="1.0" encoding="UTF-8" standalone="no"?>\n<!DOCTYPE svg…
subject:           "LTER"
created:           "25.9.2018, 15:15:20"
prov:             @prefix prov: <http://www.w3.org/ns/prov#> .\n@prefix dct: <http://...
title:            LTER Dendrometer measurements
modified:         25.9.2018, 15:15:20
type:            rov-o trig
retr_url_trig:    https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/trig
retr_url_json:    https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/provjson
retr_url_provn:   https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/provn
retr_url_rdfxml:  https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/rdfxml
retr_url_xml:     https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/provxml
```

Listing 4: Dublin Core representation of PROV-Template catalog records

4.1 Template expansion via custom Python implementation

Besides offering basic CRUD operations, the Web service API also supports the expansion of registered templates via dedicated REST calls using the respective bindings passed as parameters “on the fly”. Template expansion is performed using functions provided by a dedicated custom Python library created throughout task 8.3 by DKRZ and EAA, available on Github at https://github.com/EnvriPlus-PROV/EnvriProvTemplates. This library basically mimics the functionality of the only existing PROV-template expansion implementation to date, which is provided as part of the Java based ProvToolbox[21] suite.

The main motivation for creating a custom Python implementation was based on DKRZ requirements to include expansion functionality in existing Python workflows, which could be achieved in a much more elegant way via a native Python library. Another incentive arose from

[20] For additional information, consult the code documentation at https://github.com/EnvriPlus-PROV/ProvTemplateCatalog regarding the “config.py” file

[21] https://github.com/lucmoreau/ProvToolbox

experiments conducted by EAA, which revealed some cases of unclear behaviour of the existing expansion implementation with respect to the PROV-Template specification[22]. Although the maintainers of the ProvToolbox code suite were very responsive to requests in this regard, creating a custom implementation that allowed much greater flexibility.

Interpreting unclear specification issues

One example for an unclear specification issue was found to be the behaviour during expansion of regular variables linked to variables specified using the *vargen* namespace:

According to the PROV-Template specification, vargen variables at element identifier position do not need to be bound to mandatory identifiers in the bindings such as it would be the case for regular variables at the same position - If unbound, vargen variables are assigned a randomly created ID instead. Another PROV-Template feature in turn allows using a special statement "tmpl:linked" to link two or more variables together into so-called *link groups*, with the effect that no cartesian expansion (see [Moreau et al., 2018] for the concept of cartesian expansion) takes place between them, provided that all the variables within a link group are bound to the same number of instances in the bindings, otherwise resulting in an error.

An intuitive interpretation of the behavior during the expansion of templates featuring link-groups consisting of a mixture of vargen variables and regular variables would be that while the regular variables of course all need to be bound to the same number of instances in the bindings, it could be assumed that vargen variables within the same link-group would instead get instantiated with the same number of automatically generated random IDs. This would be beneficial to a number of use-cases, such as linking one by one a set of distinct entities with known identifiers to a similar set of generating activities for which exact start times but no identifiers are available.

While the PROV-Template specification is not clear regarding the above interpretation, the original implementation currently does not support this behavior and returns an error. It was thus decided to implement the desired behavior in the custom expansion solution.

4.2 Storage for expanded documents

Expanded PROV documents can optionally be written into a dedicated section of the ENVRIplus Triple Store[23] currently maintained by UvA, where they can potentially be queried together across individual template expansions, templates and even RIs. One issue in this regard was how to separate individual PROV documents from each other in the store, which is usually done via named graphs. It was thus necessary to decide on a related naming scheme, which was specified as follows: If an expanded PROV document has a bundle ID, this identifier is appended with a timestamp and the combined string used as ID for the named graph. If there is no Bundle ID, the ID of the expanded template is used instead, similarly appended with a timestamp. In a long term, the assignment of identifiers for individual template expansions should be reconsidered in a larger, e.g. EOSC, context.

[22] https://provenance.ecs.soton.ac.uk/prov-template/
[23] http://oil-e.vlan400.uvalight.net/

5 PROVENANCE TEMPLATE CONTRIBUTIONS

The individual sections in this chapter represent the community experiments performed by various ENVRIplus partners. The experiments covered individual parts of the different Data Life Cycles (DLC), ranging from data acquisition to data use. Table 2 provides an overview on the different DLC steps covered, following the ENVRI Reference Model approach on the information objects lifecycle[24] [Atkinson et al., 2016, page 26].

Table 2: Overview on different DLC steps covered in provenance record by RIs

	LTER	AnaEE	EISCAT	DASSH	IS-ENES	VRE4EIC	Particle
Data Acquisition	X	X	X				
Data Curation				X	X		
Data Publishing			X	X	X		
Data Processing					X	X	
Data Use			X		X		X

Besides covering different lifecycle steps, the provided examples also represent a wide spectrum regarding automation, ranging from attempts to provide templates for the results of purely manual acquisition protocols to those for representing more automated workflows. This variety of settings, also with respect to the underlying technologies, is reflected in the different approaches to creating and instantiating PROV-templates followed by the individual contributions, briefly summarized below.

DASSH provide a prototypical use-case for PROV-Template in the context of quality assurance processes performed in data ingestion workflows. A dedicated template is instantiated with bindings extracted from logs of human Q&A activity recorded in the GitLab[25] versioning system.

The contribution by *LTER* focuses on representing a data acquisition process whose results are stored in an Excel sheet. The structure of the information present there is used to create a PROV-template capturing the underlying acquisition workflow, which is later instantiated by extracting the actual values from the spreadsheet.

EISCAT provides two different PROV-templates modeled to capture a chain of events mainly covering the acquisition of data via Radar sites, subsequent dataset publishing and its use. This use-case shows how different steps in the genesis of scientific results, each step usually resulting in data of a certain „level“, can be represented by individual PROV „sub-chains“ which can be concatenated to represent the full chain of events leading to a certain result.

[24] https://confluence.egi.eu/display/EC/Model+Overview
[25] https://about.gitlab.com/

The PROV experiment performed by *TIB/Pangaea* in the context of the ENVRIplus Science Demonstrator 6, referred to as 'Particle Formation' throughout this deliverable, involves the creation and instantiation of two different PROV-templates covering two subsequent data analysis steps each resulting in data of different levels, as executed in workflows running on a D4Science VRE.

Two different use cases each involving PROV-templates are discussed by *IS-ENES*. One use-case deals with the representation of structurally comparable but heterogeneous data ingestion workflows by different projects via PROV-Templates, which requires the automatic creation of project specific PROV-templates from dedicated configuration files. The second use-case considers PROV-template for representing the activities around the evaluation of earth system models.

The log file conversion use-case explored by UvA in the context of *VRE4EIC* describes the aggregation of provenance data from different sources by the example of a metadata conversion workflow. A common provenance model is represented as PROV-template which is instantiated using the ENVRI Provenance service prototype. This illustrates the effectiveness of PROV as Lingua Franca for collecting provenance information.

5.1 IS-ENES

Stephan Kindermann - DKRZ, German Climate Computing Center, Hamburg, Germany

Background and Use Case selection

As familiarity with W3C PROV standard conforming provenance descriptions in general and knowledge of the PROV templating approach is not yet widespread in the IS-ENES community two concrete use cases were chosen which are related to two concrete application areas with a clear short to mid term vision of PROV (templating) adoption. The two use cases are also related to two different stages in the IS-ENES data life cycle:

The first use case illustrates the usage of prov templates to describe the data ingest process at an IS-ENES data center (DKRZ). The process is structured based on a generic data ingest workflow, consisting of generic workflow steps (e.g. data ingest request handling, data transfer and ingest, data quality control, data publication, data archiving). Each step has well defined associated actors, input/output entities and activities as well as associated generic attribute information. Yet detailed workflow steps related information (e.g. entity attributes) as well as workflow step composition are different for different data projects. Thus the specific challenge in this case is to enable the automatic generation of project specific PROV templates based on specific project related configuration information.

The second use case is related to a community effort developing a diagnostics and performance metrics tool for the evaluation of Earth System Models, the ESMValTool[26]. ESMValTool will be run routinely at some large ESGF sites (e.g.. DKRZ) to provide timely diagnostic results for the ongoing CMIP6 experiments. Also in the context of COPERNICUS the ESMValTool is applied to generate e.g. climate indicators for CMIP5. Results produced by the ESMValTool need to be clearly characterized with respect to various provenance aspects: used input data and applied preprocessing steps as well as climate diagnostics, actors involved, software versions used etc.

[26] https://www.esmvaltool.org

A first sketch of how a PROV based representation of this information could look like was developed by the ESMValTool community[27]. This sketch was taken as the starting point to define a prov template to represent provenance information for ESMValTool generated data products.

Automatic PROV template generation

The data ingest pipeline at DKRZ is supported by a set of tools to manage the related data management tasks and to store management related information e.g. who requested data ingest as part of which project, generic metadata describing the data collection, which data quality control procedures were applied, who published the data e.g. to be accessible as part of the Earth System Grid Federation (ESGF) etc. An integrated environment is in development supporting the collection and storage of all data management related information of the data ingest process[28]. The workflow and workflow steps associated to specific data projects can be configured using a json format[29].

A simple tool was developed taking this json file as input and generating a high level PROV template description as output. The purpose of this template is on one hand side to be instantiated to generate concrete W3C PROV conforming documents characterizing a specific data ingest process. On the other side, and currently more important, this template is used as a means to discuss the best generic way to represent the data ingest workflow based on PROV. Two alternative representations which are automatically generated based on the workflow json description are shown in Figure 6 and Figure 7 (both omitting entity attributes). The second version is explicitly including a final data collection phase, to merge all workflow steps related data management information in a final combined result (as the final PROV document itself is implicitly linking and including all relevant information it is a point of discussion whether an explicit representation of this is necessary).

[27] see github issue https://github.com/ESMValGroup/ESMValTool/issues/240#issuecomment-399019254

[28] https://github.com/IS-ENES-Data/submission_forms

[29] https://github.com/IS-ENES-Data/submission_forms/blob/master/dkrz_forms/config/workflow_steps.py

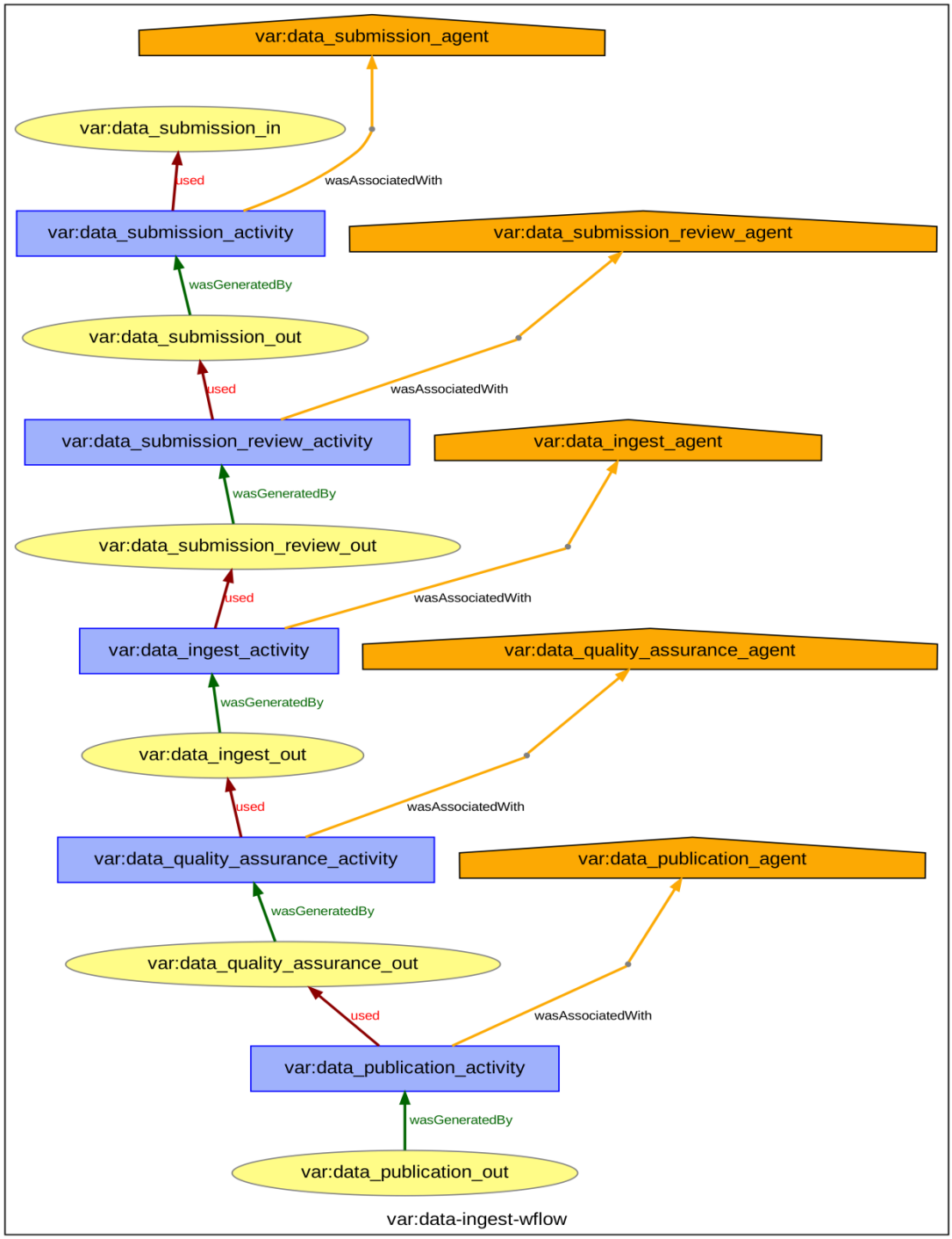

Figure 6: PROV-Template for the data ingest workflow at a IS-ENES data center. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bd71d13d6fa3360dd4f41cd/svg

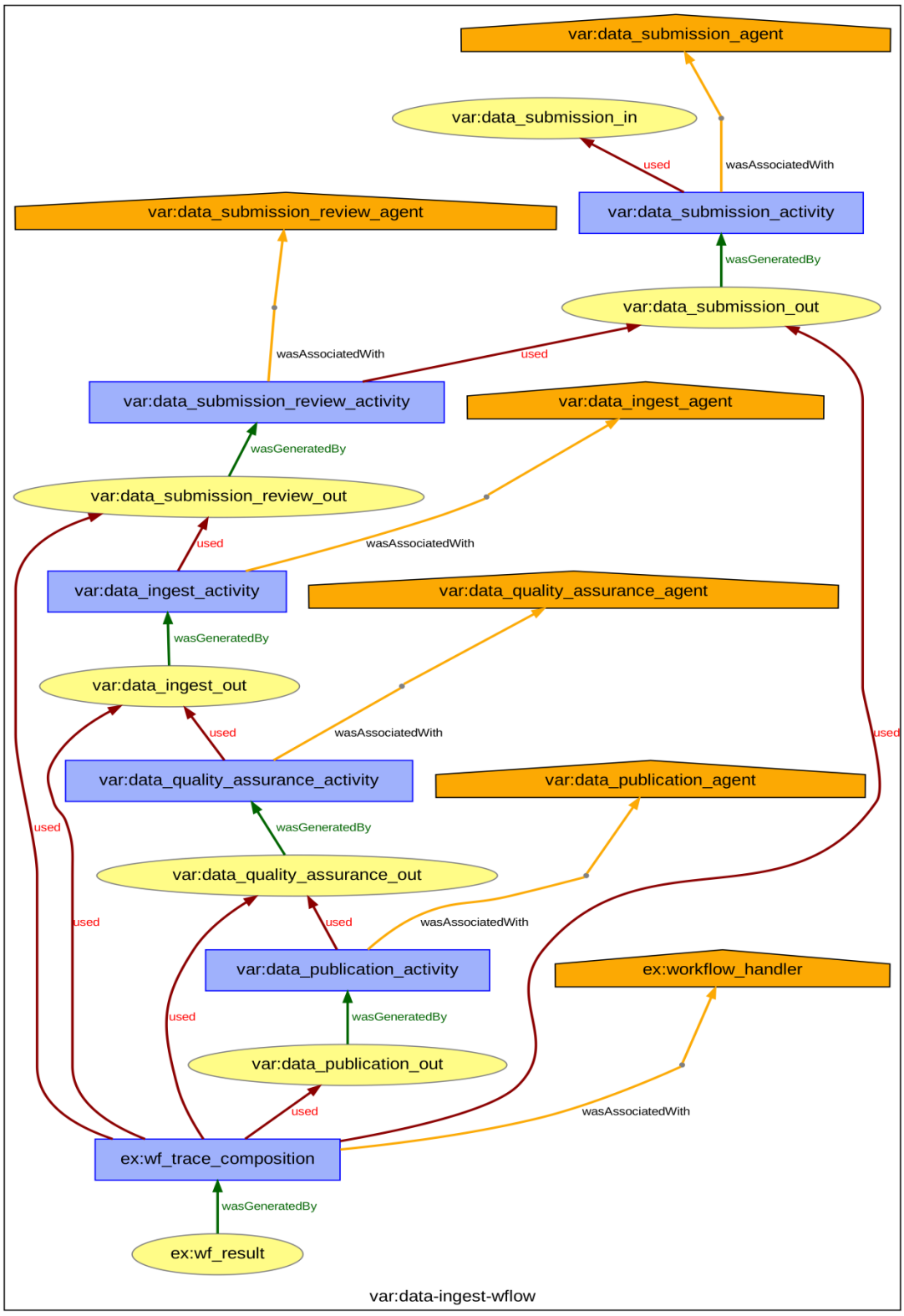

Figure 7: PROV-Template for the data ingest workflow at a IS-ENES data center with explicitly modelled trace composition activity. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bbc991ad6fa3376fe116ab2/svg

PROV templates for climate data diagnostics and performance metrics:

The Earth System Model eValuation Tool (ESMValTool) is a community diagnostics and performance metrics tool for the evaluation of Earth System Models (ESMs) that allows for routine comparison of single or multiple models, either against predecessor versions or against observations(ESMValTool, http://esmvaltool.org). The tool is composed of a large collection of individual diagnostics, which are organized in a common framework with respect to configuration, pre-processing and data input as well as data output. The tool development is organized as a community effort on github (https://github.com/ESMValGroup/ESMValTool).

All data products generated by the ESMValTool are characterized by:

- the input data sets used (organized in collections)
- an internal preprocessing step generating an internal intermediate result
- a climate diagnostics application (associated with a set of authors) associated with a specific configuration
- an overall description and configuration (associated with a set of authors)

The PROV-trig template representation for data products generated by the ESMValTool can be derived as shown in Listing , whose graphical representation is shown in Figure 8.

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix evt: <http://www.esmvaltool.org/scheme> .
@prefix ex: <http://example.org/> .
@prefix ex1: <http://example.org/1/> .
@prefix ex2: <http://example.org/2/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix gen: <http://enes.org/provgen> .
@prefix ns1: <http://www.esmvaltool.org/> .
@prefix ns2: <http://enes.org/> .
@prefix orcid: <http://orcid.org/> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix vargen: <http://openprovenance.org/vargen#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

{
        var:outfile a prov:Entity ;
        ns1:schemecaption "var:caption"^^xsd:string ;
        ns1:schemedomain "var:domain"^^xsd:string ;
        ns1:schemeplottype "var:plottype"^^xsd:string ;
        ns1:schemerealm "var:realm"^^xsd:string ;
        ns1:schemetheme "var:theme"^^xsd:string ;
        ns1:schemevariable "var:variable"^^xsd:string ;
        prov:qualifiedGeneration [ a prov:Generation ;
        prov:activity ns1:schemediagrun ;
        prov:atTime "var:diagrun"^^xsd:dateTime ] ;
        prov:wasDerivedFrom ns1:schemerecipe .

        ns2:provgendata_collection a prov:Entity,
        "prov:Collection"^^xsd:string ;
        ns2:provgenscript "gen_data_collection(provdoc,provcollection,ds_dict)"^^xsd:string .

        var:Author_diag a prov:Agent,
        "prov:Person"^^xsd:string ;
        foaf:name "var:name"^^xsd:string .
```

```
        var:Author_nml a prov:Agent,
        "prov:Person"^^xsd:string ;
        foaf:name "var:name"^^xsd:string .

        ns1:schemediag_setting a prov:Entity .

        ns1:schemediagnostic a prov:Entity ;
        ns1:schemedescription "var:diag_description"^^xsd:string ;
        ns1:schemereferences "var:references"^^xsd:string ;
        ns1:schemestatistics "var:statistics"^^xsd:string ;
        prov:wasAttributedTo var:Author_diag .

        ns1:schemediagrun a prov:Activity ;
        prov:used ns1:schemediag_setting,
        ns1:schemediagnostic,
        ns1:schemepreproc_file,
        ns1:schemesoftware .

        ns1:schemepreproc_file a prov:Entity ;
        prov:qualifiedGeneration [ a prov:Generation ;
        prov:activity ns1:schemepreprocrun ;
        prov:atTime "var:preprocrun"^^xsd:dateTime ] .

        ns1:schemepreproc_setting a prov:Entity ;
        ns1:schemecmor "var:cmor_fixes"^^xsd:string ;
        ns1:schemederivation "var:derivation"^^xsd:string ;
        ns1:schemelevelint "var:levelint"^^xsd:string ;
        ns1:schememasking "var:masking"^^xsd:string ;
        ns1:schememultimeanstat "var:multimeanstat"^^xsd:string ;
        ns1:schemeregridding "var:regridding"^^xsd:string ;
        ns1:schemetimesel "var:timesel"^^xsd:string .

        ns1:schemepreprocrun a prov:Activity ;
        prov:used ns2:provgendata_collection,
        ns1:schemepreproc_setting,
        ns1:schemesoftware .

        ns1:schemerecipe a prov:Entity ;
        ns1:schemedescription "var:nml_description"^^xsd:string ;
        ns1:schemeproject "var:project"^^xsd:string ;
        ns1:schemereferences "var:nml_references"^^xsd:string ;
        prov:wasAttributedTo var:Author_nml .

        ns1:schemesoftware a prov:Entity ;
        ns1:schemeESMValTool "var:ESMValTool"^^xsd:string ;
        ns1:schemeNCL "var:NCL"^^xsd:string ;
        ns1:schemePython "var:Python"^^xsd:string .
}
```

Listing 5: PROV-Template for representing data products generated by the ESMValTool

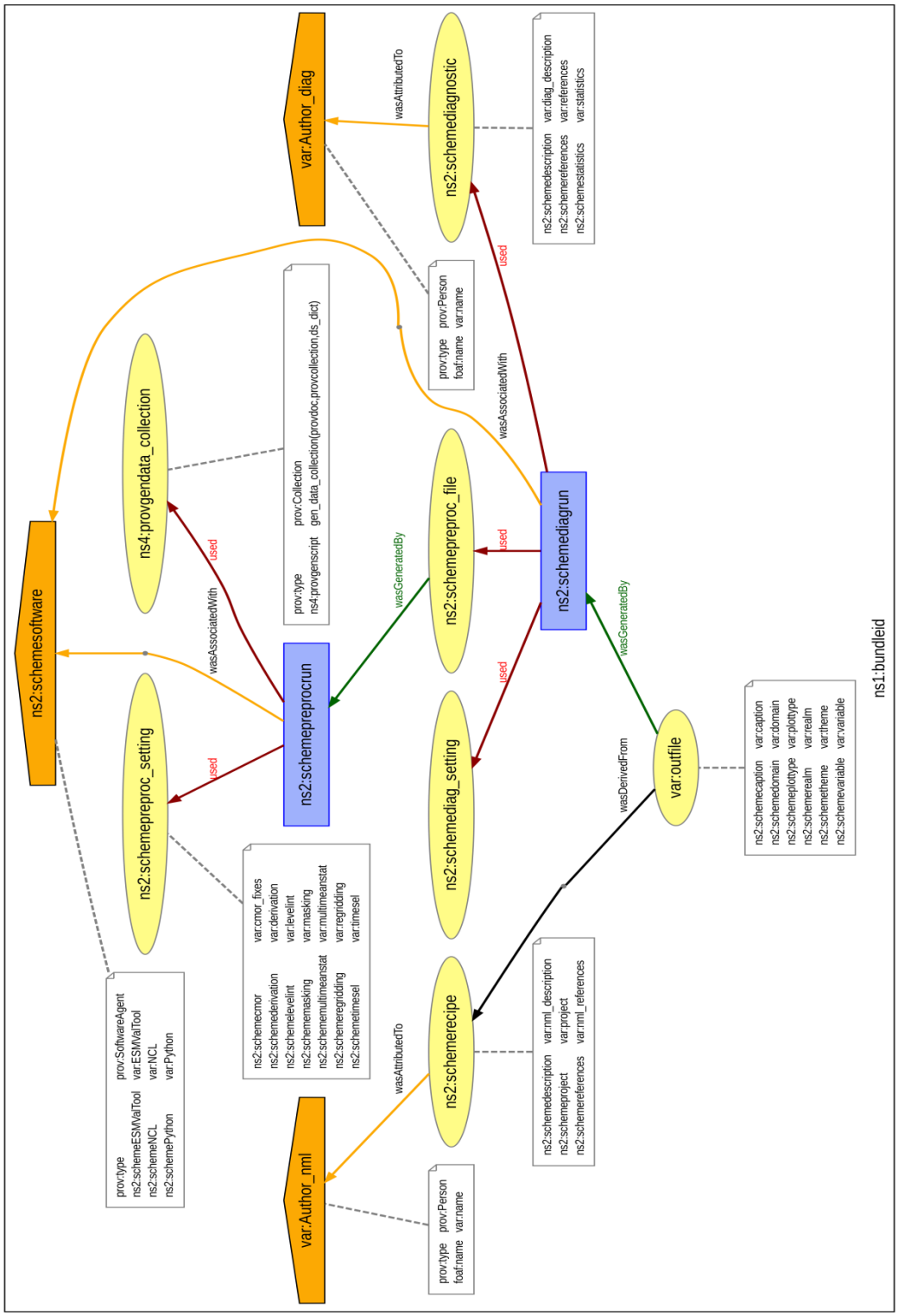

Figure 8: Visualization of PROV-Template for climate data diagnostic results produced by the ESMValTool. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bbc9493d6fa3376fe116ab1/svg

The internal structure of the data collection used in the preprocessing step is not explicitly represented in this template (normally dozens, to hundreds of files are involved organized in datasets, which are summarized in collections). The instantiation of these data collections as part of the PROV templating approach has several drawbacks: The expression of the internal structure of data collections at the template level would be difficult to achieve (e.g. several different templates would be necessary to describe different types of data collections) and the instantiation would be problematic as well (clear thinking about the necessary tmp:linked attributes in the template, to achieve the desired expansion structure).

Thus an explicit (community specific) expansion rule for data collections was implemented to be applied in a second instantiation step of the template. The approach followed can be generalized to enrich template instantiation with community specific expansion mechanisms:

The expansion function to be applied is provided as a parameter to the PROV collection entity (the parameter is given as gen:script associated value - "gen_data_collection" in the above description). This function needs to be defined in the context the template expansion is done. Listing 6 shows an example for such a function definition.

```
def gen_data_collection(doc,collection,pars):

    id = 0
    for ds_id in pars['ds_ids']:
        ds_info = pars[ds_id]
        dataset =  doc.entity('evt:'+ds_id,
            {'ex:source': ds_info['source'],
             'ex:name':ds_id,
             'dc:identifier':ds_info['tracking_id']
        })

        for sfile in ds_info['files']:
            file_info = ds_info[sfile]
            file_ent = doc.entity('evt:'+sfile,
                  {'dc:identifier': file_info['tracking_id'],
                   'ex:name': sfile
                  })
            doc.used(dataset,file_ent)

        doc.hadMember(collection, dataset)
    return doc


def get_expscript(collection):
    ''' retrieve property describing expansion script
    '''
    attrs = collection.attributes
    for (aname,avalue) in attrs:
        print(aname.localpart)
        if aname.localpart == 'script':
            return avalue
    return 'no script'

def evaluate_expression(expression, doc, collection, pars):
    provdoc = doc
    provcollection = collection
    ds_dict = pars
    result = eval(expression)
    return result


### instantiation information
mypars = {'ds_ids':['ds1','ds2','ds3'],
          'ds1':  {
              'files':['ds1file1','ds1file2'],
              'source':'ESGF DKRZ',
              'tracking_id':'hdl.net:222/1111.1111.1111',
```

```
        'ds1file1': {'tracking_id':'hdl.net:111/000.000.0001'},
        'ds1file2': {'tracking_id':'hdl.net:111/000.000.0002'}
    },
    'ds2': {
        'files' : ['ds2file1'],
        'source':'ESGF DKRZ',
        'ds2file1':{'tracking_id':'hdl.net:111/000.000.0003'},
        'tracking_id':'hdl.net:222/444.444.444',
    },
    'ds3':{
        'files' : [],
         'source':'Obs DKRZ',
        'tracking_id':'hdl.net:222/000.000.666'
    }
    }


################################################################
# Apply expansion rule which is defined as entity attribute

result = evaluate_expression(script,new1,collection,mypars)
print(result.get_provn())
```

Listing 6: Applying community specific expansion rules in a generic setting

An example result of the application of this additional community specific expansion rule is accessible at https://github.com/stephank16/enes_graph_use_case/blob/master/prov_templates/esmval-prov.ipynb

Community adoption aspects

The work on adopting the PROV templating approach in the above described concrete use case scenarios revealed several aspects summarized here as a conclusion:

- There are many alternatives to provide PROV template descriptions characterizing specific use cases. A (curated) catalog of PROV template descriptions can help to exchange experiences between domain experts on best practices.
- Normally first versions of template definitions are using “local” namespaces. Domain experts need to iterate with linked data experts on which generally adopted vocabularies can be best applied to characterize the attributes of the PROV entities. Also here cross domain collection of best practices would be very helpful.
    - some cross domain attributes of importance are e.g. persistent identifiers (currently associated to the Dublin Core element ‘dc:identifier’ in the collection expansion implemented for the ESMValTool template). Other examples include DOI attributes, and citation information in general associated to PROV actors characterized as “authors”.
- Expression of generic templates is restricted by the mechanisms the PROV template expansion algorithm provides (cartesian product and linked attribute expansion). These mechanisms can be too restrictive to generalize specific community use case scenarios. This was resolved in the above example by applying a community specific expansion step to generate the final fine granular PROV representation.
- PROV templates (especially graphical representations of them) provide a good vehicle to discuss prevenance representation aspects within a community. Especially the separation on the generic structure (PROV entities and associated variables as well as related vocabularies) and the “what” (the concrete variable values) is important. The

"what" is normally very clear from the concrete use case - the overall discussion for structuring in a PROV standard compliant way can be guided by the design of PROV templates. This is also currently the main purpose of the prov templates described above within the IS-ENES community.

5.2 EISCAT

Carl-Fredrik Enell, EISCAT Scientific Association, Kiruna, Sweden
Rikard Slapak, EISCAT Scientific Association, Kiruna, Sweden

Background

EISCAT[30] is an international research association which operates several high power large aperture radar systems: one UHF radar and one tristatic VHF radar in Tromsø, Norway, with remote receivers in Kiruna, Sweden, and Sodankylä, Finland, and one dual-antenna VHF radar in Longyearbyen on Svalbard, Norway. The radars have multiple purposes, and the standard mode of observation is to retrieve ionospheric parameters from incoherent scattering of radio waves by electrons in the ionosphere.

EISCAT has member institutes in several countries around the world. Observation time is granted to the member countries in proportion to their contribution to the association. Scientists apply for observation time on a campaign experiment basis.

EISCAT is currently constructing a new radar system, EISCAT_3D[31], which will be able to operate more continuously and observe volumes of the atmosphere. It will generate significantly more data than the existing radars. EISCAT_3D is an ESFRI Landmark project and a participating RI in ENVRIplus. This study is part of ongoing development of data models and the necessary data identification and provenance standards for EISCAT_3D.

The following is a description of the observational workflow and the required provenance items of the existing EISCAT radars. The templates will be extended as the EISCAT_3D data management projects progress (these include EOSC[32] and NeIC[33] projects). Python software will be a major part of the data handling system, building on e.g. DIRAC[34], Rucio[35] and similar services from the particle physics community as well as metadata harvesting to B2FIND[36], B2SHARE[37] and similar search portals, and thus the PROV template libraries will be a natural extension.

[30] https://www.eiscat.se/
[31] https://www.eiscat.se/about/eiscat3d7/
[32] https://www.eosc-hub.eu/research-communities/eiscat3d-agile-data
[33] https://neic.no/e3dds/
[34] http://diracgrid.org/
[35] https://rucio.cern.ch/
[36] http://b2find.eudat.eu/
[37] http://b2share.eudat.eu/

Workflow: Producing the data - running an EISCAT experiment

The existing EISCAT radars run on campaign basis and each observation usually has a dedicated purpose. An EISCAT experiment starts by requesting observation time, and allocated experiments are inserted into the EISCAT Schedule, which is accessible online (https://www.eiscat.se/schedule/schedule.cgi). Figure 9 shows an example of a requested experiment: in this case a Common Programme (CP) experiment, providing data to all EISCAT members. Experiment time is also allocated to individual EISCAT members, which is called Special Programme (SP) experiments. For SP experiments the Resources line would specify the member and used time in hours, such as FI(5) for 5 hours of time allocated to Finland.

The experiment is defined by two main items: the pulse code programme, which defines the resolution in range and time, and the antenna scan pattern. In the example below these are called “folke” and “ip4”, and “bella” and “lowel”, respectively for the two radar systems in question.

The experiment is started at the specified time by entering a command as shown in the Notes in the command line console of the EISCAT Realtime Operating System (EROS). The data are first stored locally at the radar sites and then regularly transferred to the archive at EISCAT Headquarters. The location of the archived data is inserted into a MySQL database called **disk_archive**. This catalogue contains pointers to hourly directories and the Resource information as described above.

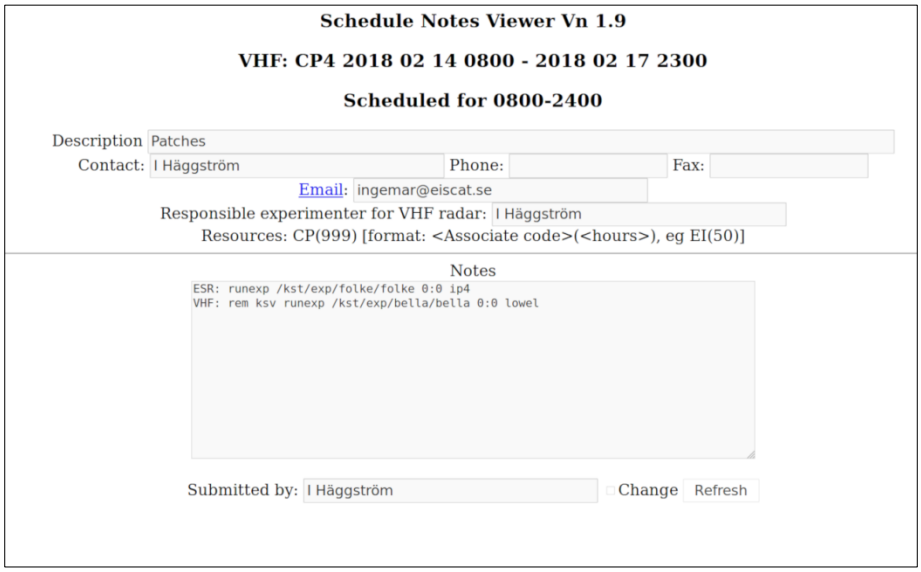

Figure 9: An example of a schedule entry for a requested experiment using the EISCAT VHF radar in Tromsø, Norway.

Summary of workflow

1. User requests experiment
2. User and EISCAT staff run the experiment
3. Data files are transferred to main archive at EISCAT Headquarters
4. Data catalogue items are inserted in disk_archive database

Provenance items describing the data production workflow

- Contact address of responsible experimenter
- Scheduled start and stop time
- Actual start and stop time
- Resources (used experiment hours per EISCAT member)
- Experiment details: pulse code and antenna scan programmes (name and version, e.g. bella v 2, ip4 v 1)

Workflow: Analysing and visualising the experiment data

The data from an EISCAT experiment are usually stored in the autocorrelation function domain (which can be thought of as power spectra) in Matlab files. These files are referred to as EISCAT Level 2 data, Level 1 being the raw signals that the receivers decode. The typical use is to retrieve and plot ionospheric parameters by fitting modelled spectra to these data. EISCAT provides a Matlab package called GUISDAP to perform this analysis. The GUISDAP results constitute EISCAT Level 3 data.

The users can browse and download Level 2 data through the EISCAT Schedule . The system checks that the IP address of the user is in a member country and if so, download of data files is granted, given that the data embargo rules are fulfilled. The user can select to download files for further analysis or optionally run the GUISDAP analysis online at EISCAT. Figure 10 and 11 below show an example of data browsing and download.

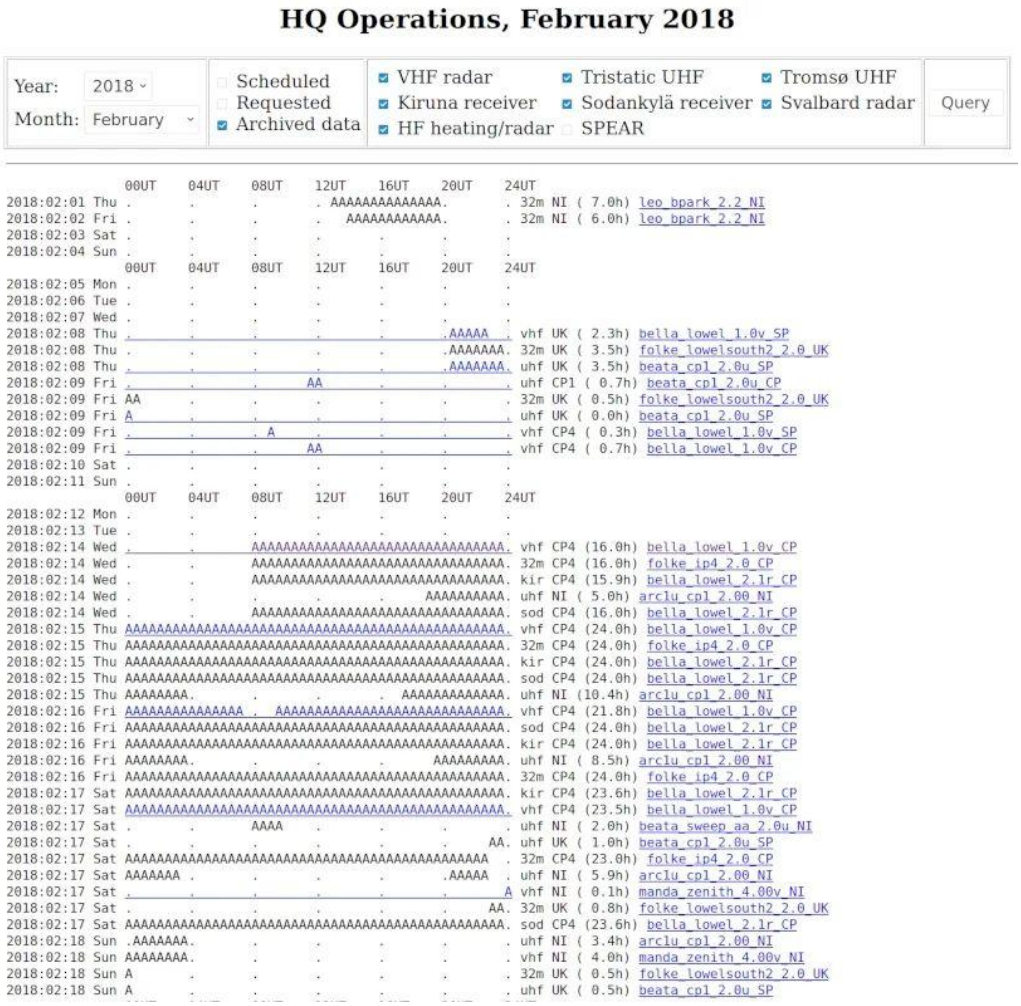

Figure 10: Searching for data through the schedule interface, which queries the MySQL file catalogue database. Clicking the experiment names on the right opens a Level 2 data download page, whereas the Archived (A) links on the left take the user to Level 3 data in the Madrigal system, produced by routine analysis with standard settings.

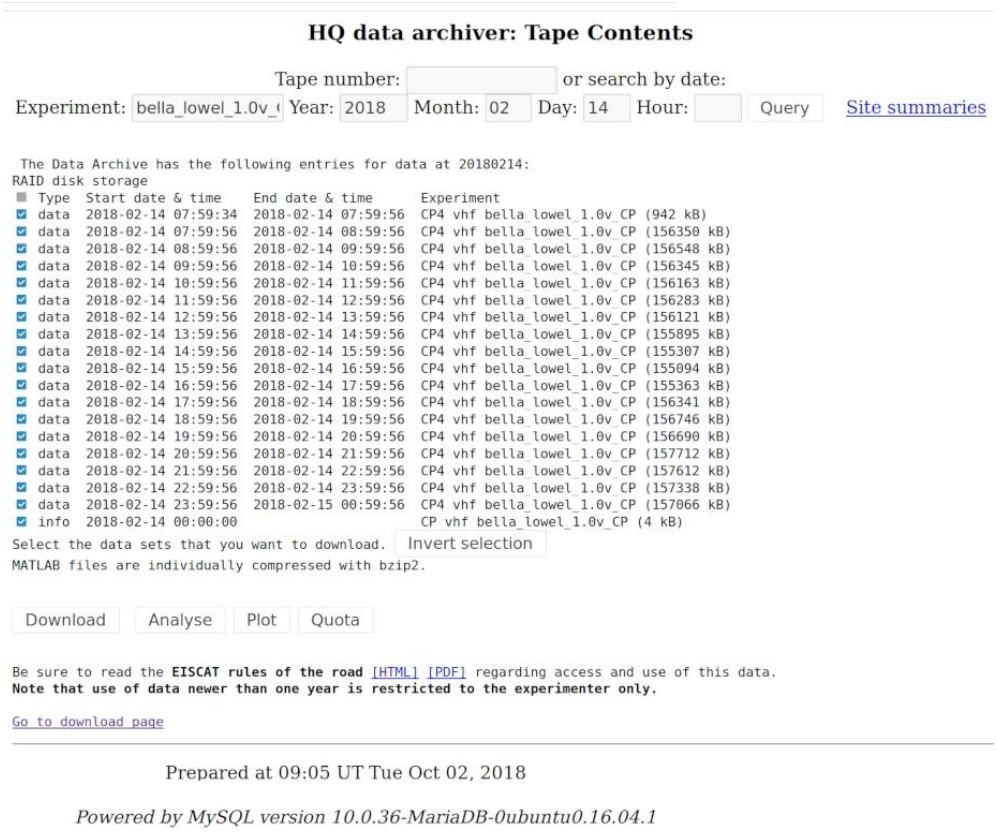

Figure 11: The Level 2 data download page. Here the user can select an interval of data and can then select to reanalyse, plot or download the files. In the latter case a tar archive will be downloaded. The user can then use GUISDAP to reproduce Level 3 physical parameter data with different settings, such as integration time.

In either case, the user has to set parameters for GUISDAP as shown in Figure 12. Many of the settings are fixed for a given experiment but importantly, the integration time has to be selected, i.e. how long intervals of Level 2 data should be averaged before analysis. This is a tradeoff between time resolution and signal to noise ratio. Also, a calibration constant can be given in cases when the measured power is wrong, typically when there is snow in the antenna. Parameter analysis is made by GUISDAP, by fitting theoretical spectra to the data (Figure 13), and the results (Level 3 data) are saved to Matlab-compatible files, which then can be visualised with a script called *vizu.m*. This produces the standard EISCAT plots as shown in Figure 14.

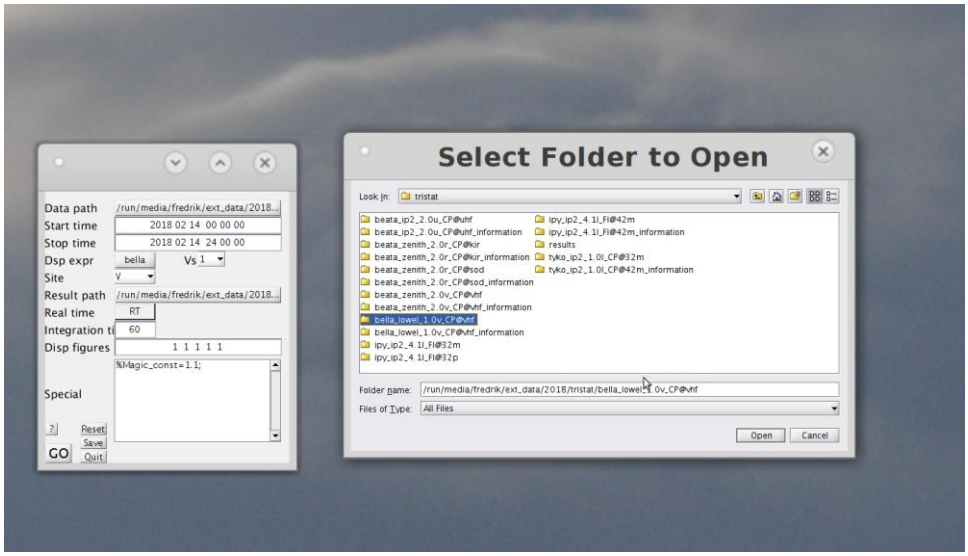

Figure 12: The GUISDAP start window and input file selection window. The user selects the location of the Level 2 files and the time interval to analyse, the correct analysis configuration for the used radar pulse code, integration time, special analysis parameters such as a calibration constant ("Magic_const" - included but commented out in this figure) , where to store the resulting Level 3 files, and what figures to display (including whether to run vizu.m automatically on the analysed data).

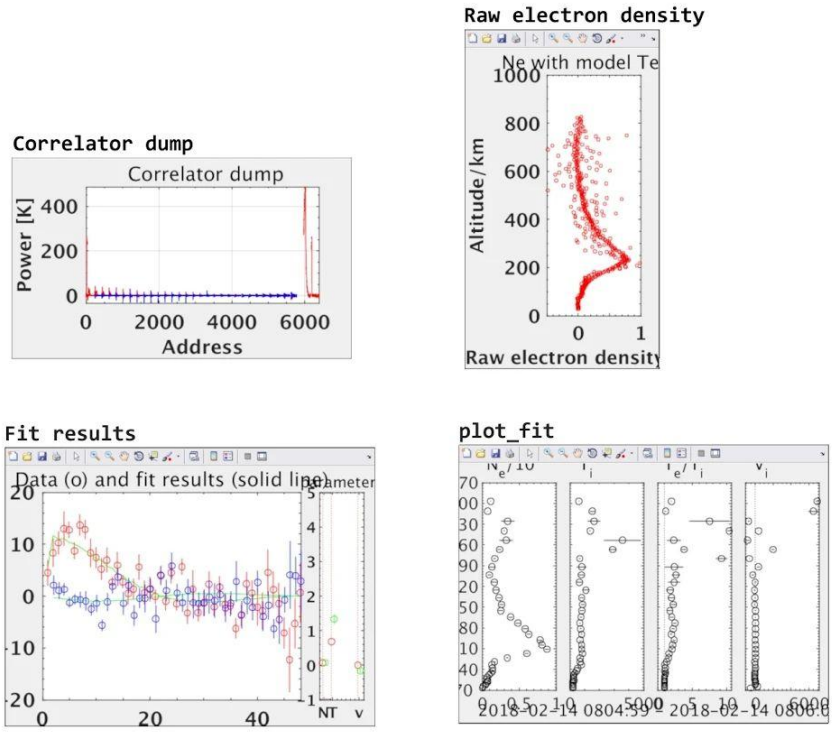

Figure 13: Running GUISDAP. This figure shows the raw data dump and how parameters are analysed by fitting theoretical spectra to the data, as function of the four parameters 1. electron density, 2. ion temperature, 3. electron/ion temperature ratio and 4. line of sight Doppler velocity.

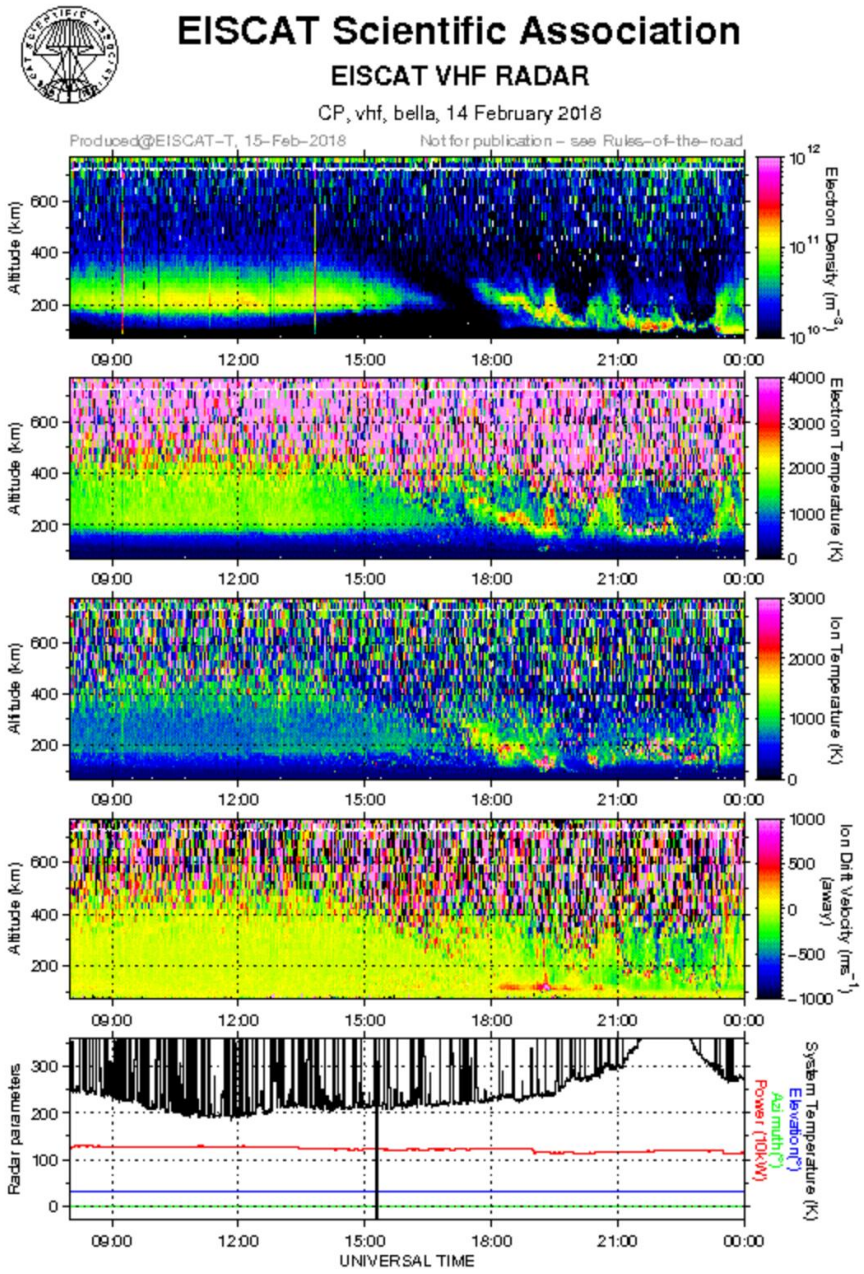

Figure 14: Visualisation of data from the above experiment, analysed by GUISDAP and plotted with the "vizu" script.

Summary of workflow

1. User searches for data.
2. User requests Level 2 data download. Authorisation is checked (in the existing system we check that the IP address is in a member country) and if access is granted a tar archive is produced.
3. User runs GUISDAP analysis on downloaded Level 2 files. Parameters like integration time, calibration constant and many more can be selected. GUISDAP writes the Level 3 results to Matlab files. See Figure 12 and Figure 13 for an example.
4. The results can subsequently be plotted with the "vizu" script included in GUISDAP. This will be done automatically after the GUISDAP run if the last of the "Disp figures" parameters is set to 1. See Figure 14 for an example.

Provenance items of analysis workflow

- User end experiment information
- GUISDAP version and configuration

PROV templates and diagrams

The following is a PROV modelling of the relevant provenance information from the two workflows described above: running an experiment and producing Level 2 data (henceforth the 'Produce' workflow), and analysing the data to produce Level 3 data and visualisations (the 'Consume' workflow). The provenances are shown in two separate charts (Figure 15 and Figure 16, respectively), and involve agents (pentagons), activities (rectangles), and entities (ellipses) and how they are connected to each other.

'Produce' provenance

As described above the 'produce' chain goes from a scheduled experiment to the actual generation and storage of Level 2 data retrieved during the experiment. The PROV template chart for this chain is shown in Figure 15. The end products are stored files and entries into the data catalogue. The provenance information connected to them are creation dates (*var:catalogueEntryCreationDate* and *var:storedFileCreationDate*, respectively) as well as a URL specifying the location of the stored files (var:fileURL).

Connected to the running of the experiment are the start and stop time of the experiment (*var:actualStartTime* and *var:actualEndTime*, respectively), which can differ from the scheduled start and stop times (*var:scheduledStart* and *var:scheduledStop*) as given in the information for the planned experiment (information that needs to be extracted from data files or system logs). Other important provenance items in the experiment definition are the antenna scan pattern (*var:antennaScanPattern*), experiment resources (*var:experimentResources*), the radar pulse code, which sets the resolution in time and range (*var:radarPulseCode*) and a URL to the schedule (*var:scheduleURL*). The scheduled experiment is attributed to a principal investigator (prov:Person), to which contact information such as e-mail address and affiliation is connected.

The experiment is run using the EISCAT Radar Operating System (EROS), to which provenance items such as configuration, controller, settings and version are related (*var:radarConfiguration*, *var:radarController*, *var:radarSettings* and *var:radarVersion*, respectively).

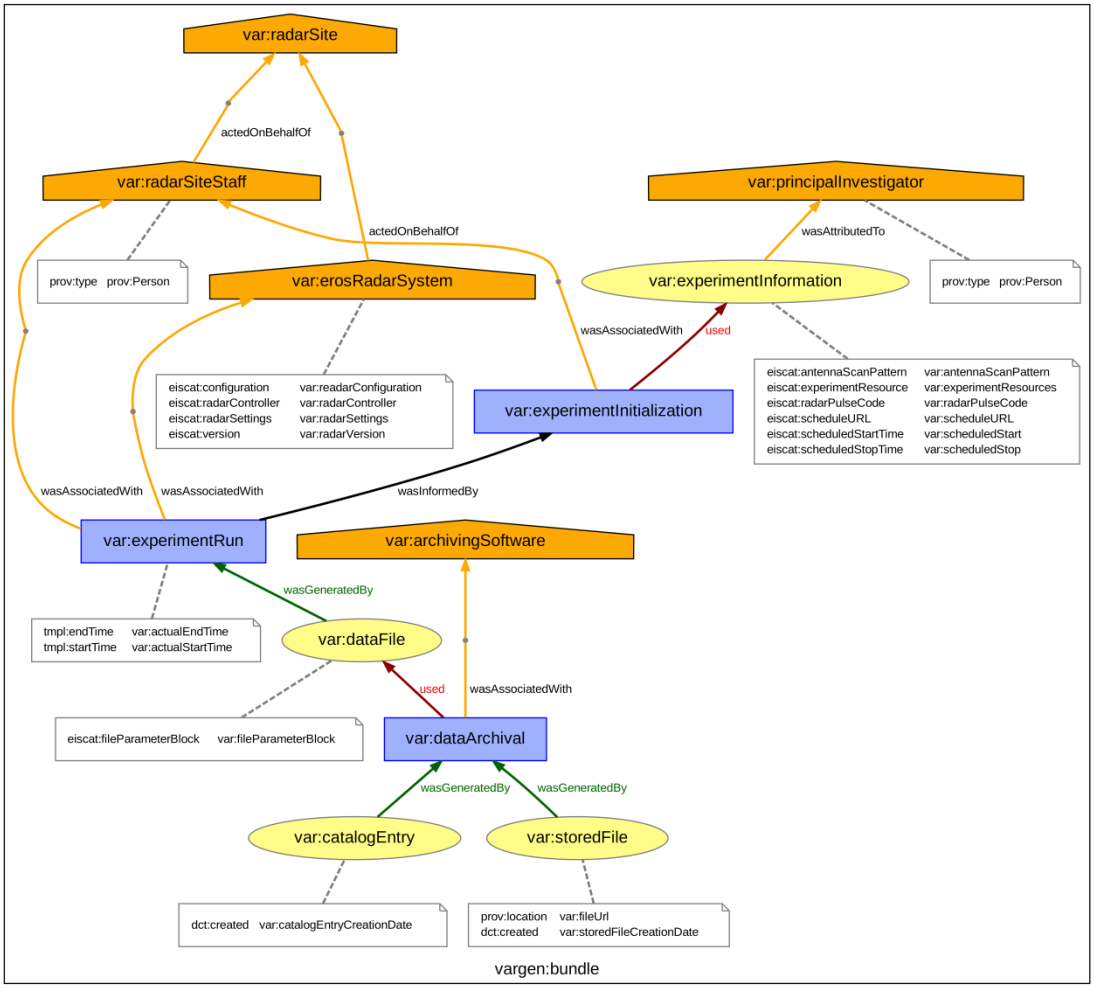

Figure 15: The PROV template describing the 'produce' provenance. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bc08800d6fa331dc528b522/svg

'Consume' provenance

The 'consume' chain describes the steps from a user querying specific data to the generation of files containing Level 3 plasma parameter data and visualisation of the data. The PROV template chart for this chain is shown in Figure 16. The experiment information (type, version, site) of the analysed data is saved as well as information regarding the user (analyst) and analysis (start and stop time, integration time, time of the analysis). The parameter fitting software (GUISDAP) used for obtaining the plasma parameters (from analysing the Level 2 data obtained as a result from the query) and the visualisation software ('*Vizu.m*') for plotting are continuously updated/changed/improved and therefore the configuration and version information about them are part of the provenance. Any other important information about the GUISDAP configuration needs to be stored. A pointer to the calibration information (if there exists any for the specific experiment) will be in the math-file containing the power spectral data.

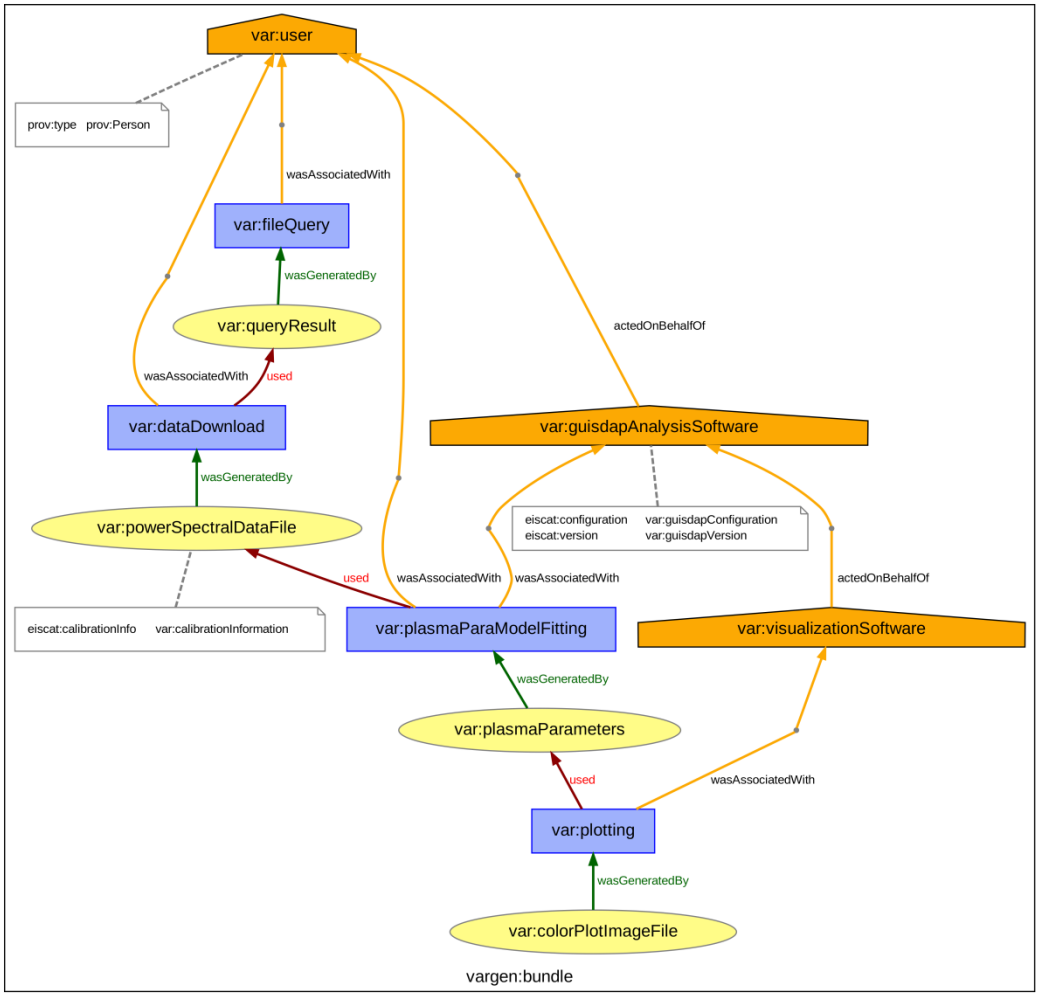

Figure 16: The PROV template describing the 'consume' provenance. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bc0a532d6fa331dc528b523/svg

Conclusions

The provenance information which needs to be attached to the data acquisition and analysis workflows of the legacy EISCAT radars was successfully modelled and PROV-templates were generated (Figure 15 and Figure 16). The study considered the present EISCAT radar system and can be considered to be a pre-study, that will be further developed for the future EISCAT radar system: EISCAT_3D. Even though there will be necessary modifications, changes and additions to the provenances of the new radar system, the PROV-templates presented in this document will serve as starting points. In summation, we find the approach of the PROV-template promising and intuitive, and that it therefore will be helpful and applicable in the context of EISCAT_3D development.

5.3 Data flow and provenance management in the AnaEE Information System (ANAEE)

Christian Pichot - INRA Provence-Alpes-Côte d'Azur, Fr

"Data" has to be understood as 'information'. Within the AnaEE IS context, it mainly covers data produced by the experimental platforms, and metadata about data, datasets and projects of experimentation. It also covers various types of documents such as protocols, sensor description.

AnaEE[38] is a Research Infrastructure devoted to the study of continental ecosystems (grassland, forest, freshwater) and their biodiversity. AnaEE offers to the research community experimental facilities for manipulations of managed and unmanaged terrestrial and aquatic ecosystems. These facilities are provided by distributed national experimental (open-air, mesocosms or enclosed), analytical and modelling platforms.

The Information System is (or will be) based on **a distributed architecture** gathering information from platform databases (located in the different sites) and modelling platforms. These metadata will feed a AnaEE portal for 'metadata and access to the resources' (Figure 17 from [Clobert et al., 2018] for IS implementation in AnaEE-France). In addition to this AnaEE 'access to the resources' portal, a discovery portal provides standardised (ISO 19115/19139) metadata about the experimental platforms, the programmes of experimentation and the datasets.

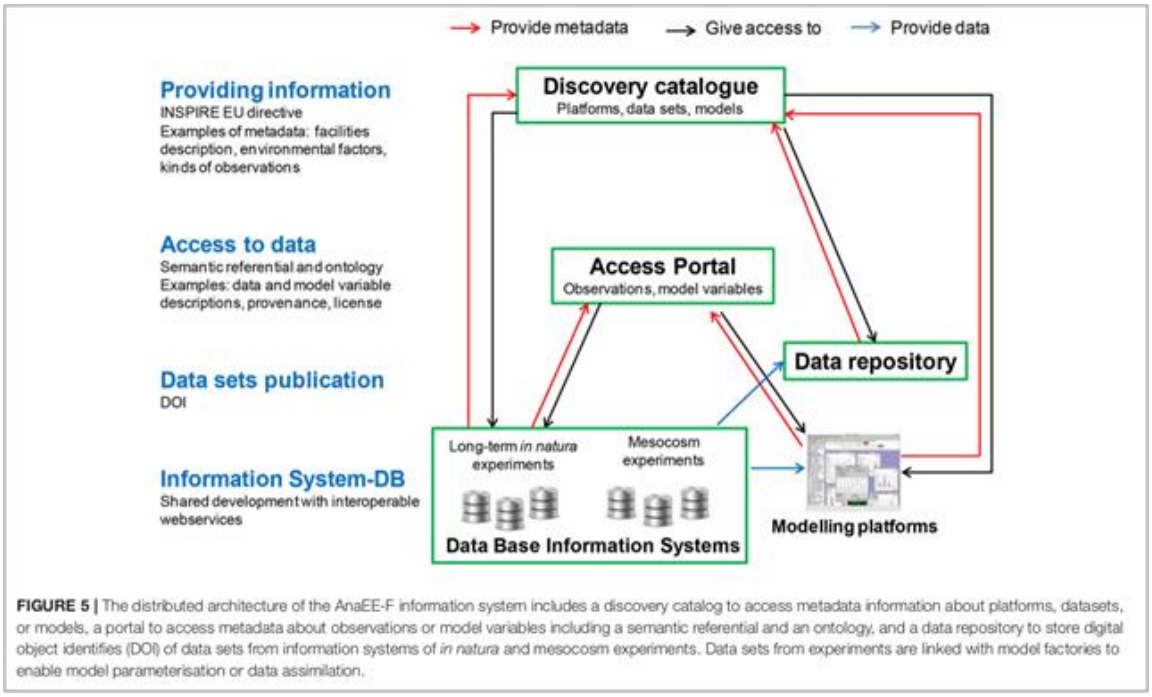

FIGURE 5 | The distributed architecture of the AnaEE-F information system includes a discovery catalog to access metadata information about platforms, datasets, or models, a portal to access metadata about observations or model variables including a semantic referential and an ontology, and a data repository to store digital object identifies (DOI) of data sets from information systems of *in natura* and mesocosm experiments. Data sets from experiments are linked with model factories to enable model parameterisation or data assimilation.

Figure 17: AnaEE-F Information System architecture [Clobert et al., 2018]

In order to harmonize and valorize the heterogeneous data generated by the experimental platforms, a **semantic interoperability** is developed based on the sharing of vocabularies (AnaeeThes thesaurus and an OBOE-based ontology) and the generic modelling of the experiments and of the studied variables (Figure 18). The discovery and access portals are fed by

[38] https://www.anaee.com/

information (rdf triples) produced by the semantic annotation of AnaEE distributed resources: relational databases and modelling platforms. A first pipeline is developed for the automation of the annotation process and the production of the semantic data. A second pipeline is devoted to the exploitation of these semantic data through the generation i) of standardised ISO and GeoDCAT metadata records and ii) of data files (NetCDF format) from selected parameters (experimental sites, years, experimental factors, measured variables…). These tools are part of the ENVRIplus service portfolio and will be usable in different contexts of ontologies and databases.

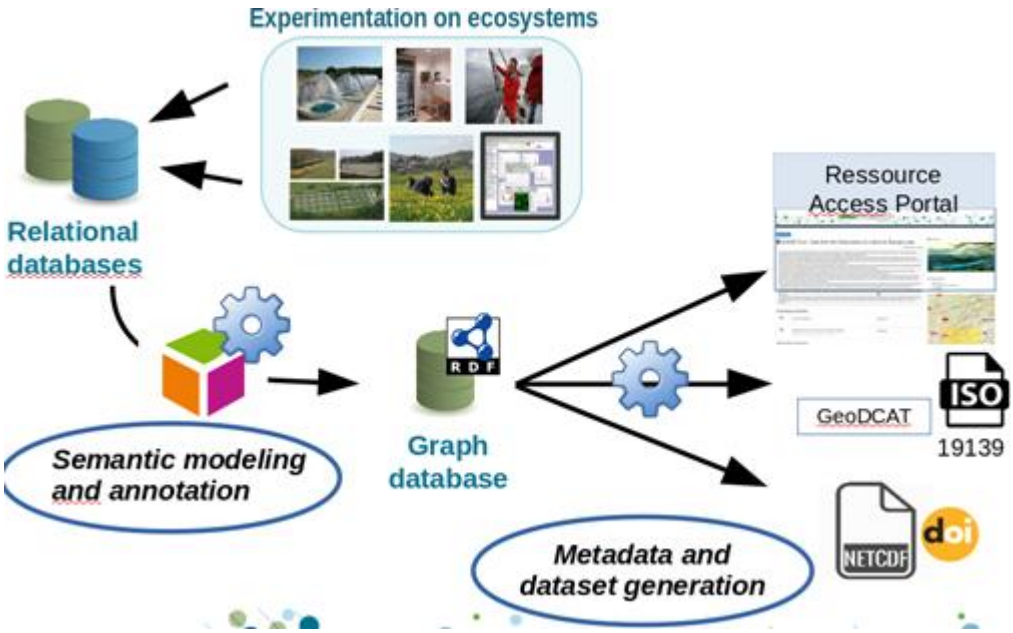

Figure 18: Developing semantic interoperability in ecology and ecosystem studies: the AnaEE infrastructure framework (adapted from Pichot et al. ICEI, Jena September 2018)

The on-going analysis of AnaEE **information flow** allows for the identification of i) the nature of each flow, ii) the data formats and iii) the used medium (Figure 19).

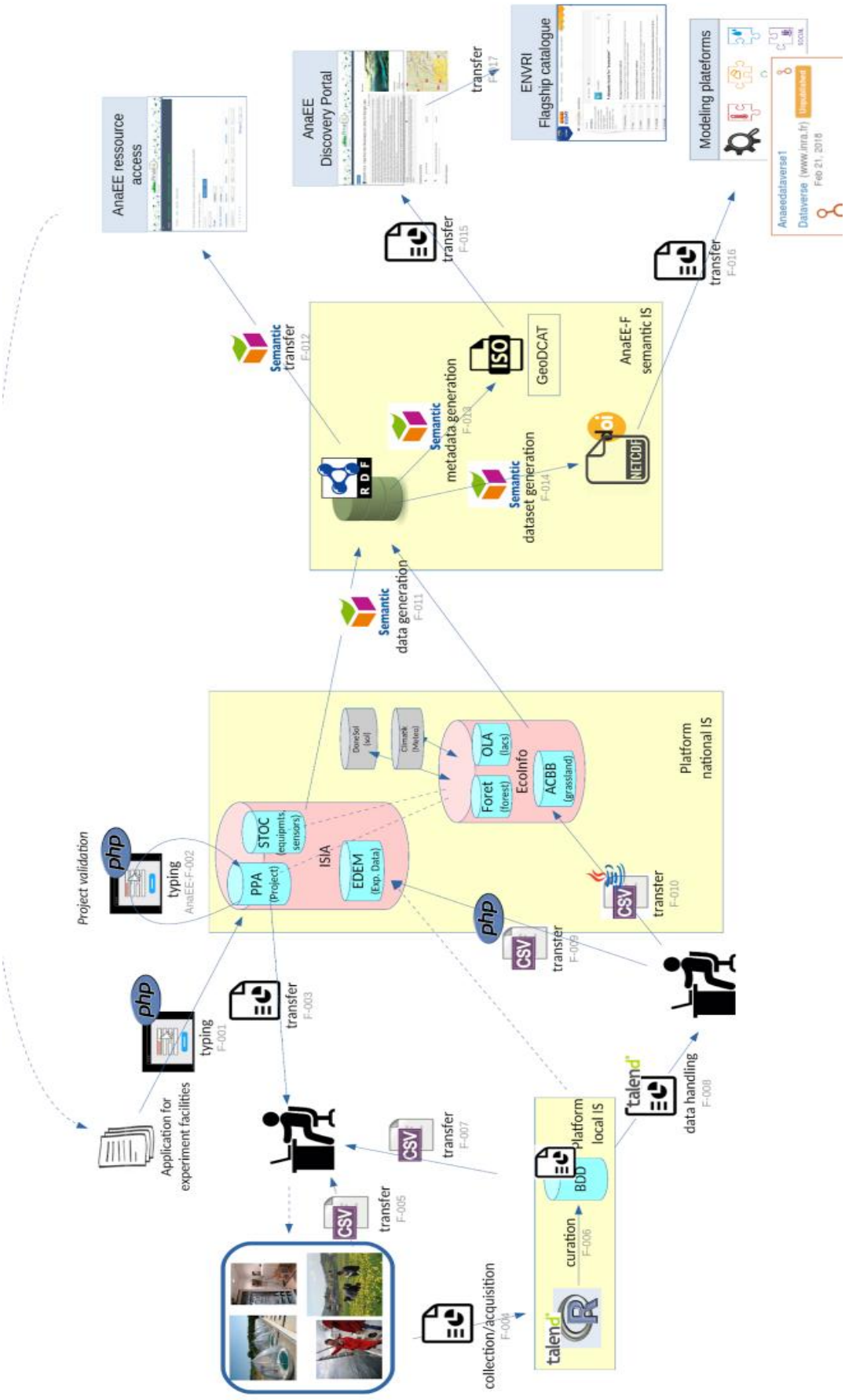

Figure 19: Information flow diagram for the AnaEE-F Information System

The information flow diagram is used to identify how Provenance components will be managed.

Within the recent Provenance survey, 10 requirements relevant for the different phases of the AnaEE Data Life Cycle were listed:

AnaEE-R1 - Provenance for the experimental facilities
AnaEE-R2 - Provenance for the variable/trait observed/measured
AnaEE-R3 - Provenance for the experimental design
AnaEE-R4 - Provenance for the spatial and temporal context
AnaEE-R5 - Provenance for the data acquisition tool
AnaEE-R6 - Provenance for actors
AnaEE-R7 - Provenance for the data curation process
AnaEE-R8 - Provenance for the data annotation process
AnaEE-R9 - Provenance for the metadata generation process
AnaEE-R10 - Provenance for the dataset generation/ identification/publication

These requirements will be met either by the information managed by the AnaEE-IS or by the use of additional tool(s). Preliminary results from this analysis are summarized in Table 3.

Whatever IS is used for their management (AnaEE IS or complementary tools), the provenance elements would be produced following the PROV ontology. As a first example for the use of 'complementary tools', the PROV template shown in Figure 20 was used to model the role of agents in the data acquisition process (https://envriplus-provenance.test.fedcloud.eu/ for "MeasuringAgent"). Within the frame of OBOE[39], this PROV information describes people who realize a measurement of one characteristic for one entity (e.g diameter at breast height of a tree). The measurement tools and the related protocol can also be provided.

Most of the provenance information managed in the AnaEE IS will be first produced as part of the semantic data/metadata characterisation using the OBOE based ontology and additional ones (FOAF, PROV, SSN…). The full PROV format generation of this information would be automated through devoted pipelines.

[39] https://semtools.ecoinformatics.org/oboe

Table 3: Identification of the tools to be used for the management of Provenance in AnaEE.

requirement ID	Meet by AnaEE IS (Yes, Not Yet, No)	comments	complementary tools
AnaEE-R1	Y	platforms DOI and specific facilities, ISIA IS and AnaEE service catalog.	
AnaEE-R2	Y	AnaEE reference naming vocabularies, aligned with external resources	
AnaEE-R3	Y	AnaEE reference naming vocabularies, aligned with external resources	
AnaEE-R4	Y	Spatial and temporal features from the OBOE basedontology	
AnaEE-R5	Y/N	Sensors features from the STOC IS. Only partially managed in the EcoInfo SI (long term experiments)	PROV Template
AnaEE-R6	NY	Not yet implemented. May not fully meet this requirements, at all steps of the DataLife Cycle	PROV Template. Initiated at https://envriplus-provenance.test.fedcloud.eu/ for "MeasuringAgent"
AnaEE-R7	NY	Should fully meet this requirements.	
AnaEE-R8	Y, partially	Ongoing. Linked to the 'semantic annotation pipeline'	
AnaEE-R9	NY	Linked to the 'semantic exploitation pipeline'	
AnaEE-R10	NY	Linked to the 'semantic exploitation pipeline'	

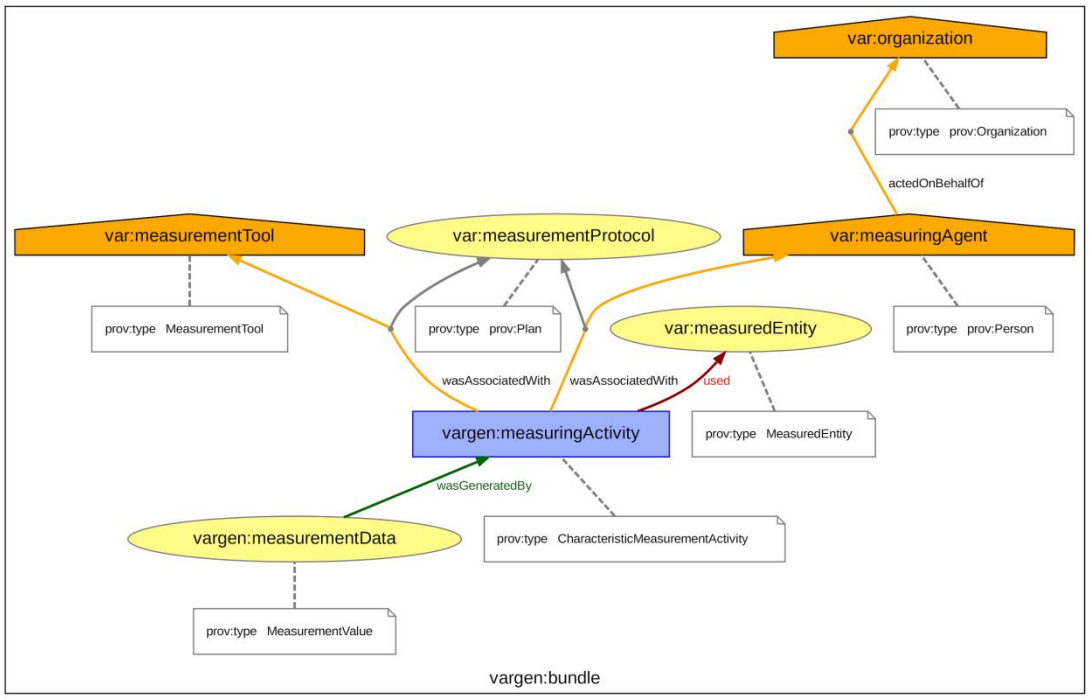

Figure 20: Role of agents in the AnaEE data acquisition process. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bc35c13d6fa3327093d3067/svg

5.4 An investigation into the usefulness and feasibility of utilizing the PROV-TEMPLATE system within a data archive centre (DASSH)

Kevin Paxman – DASSH, The Marine Biological Association, Plymouth, UK
Dan Lear - DASSH, The Marine Biological Association, Plymouth, UK

Background

An investigation was conducted into whether it was both feasible and worthwhile to implement the PROV-TEMPLATE system[40] into the procedures of DASSH (The Archive for Marine Species and Habitats Data). Although we publish metadata from the data provider concerning dataset collection, DASSH does not currently expose any provenance information relating to the processes to which we've subjected the data. Indeed we have only just begun standardising the logging of these processes. PROV is a prime candidate for a standard in which to preserve this information, and the PROV-TEMPLATE system potentially provides us a useful mechanism for the automatic generation of PROV documents.

Selected Data Life Cycle (DLC) segment

For this pilot project, it was decided to focus on the portion of the DLC that DASSH primarily concerns itself with – data transformation, validation, correction and archival. It is hoped that by focusing on this as opposed to data collection and initial processing, we will be making novel contributions to the overall project.

DASSH's primary role is to collect marine biological datasets from various UK bodies, which we standardise and publish. If the dataset received is not in the MEDIN format (Marine

[40] https://provenance.ecs.soton.ac.uk/prov-template/

Environmental Data and Information Network) then the data is converted into MEDIN manually by one of out staff. Datasets are then subjected to manual and automated validation steps, with each iteration going back to the original staff member for corrections. This process is described in Figure 21.

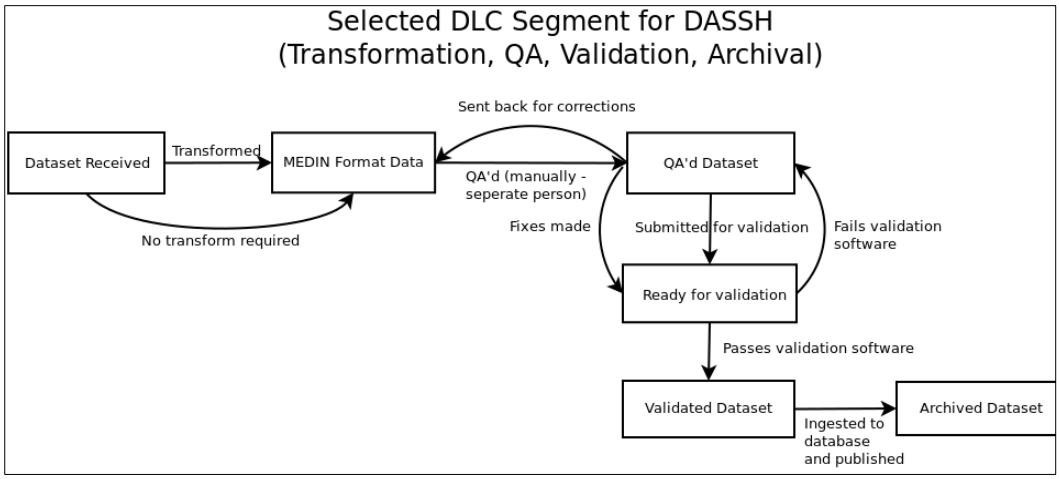

Figure 21: DASSH data transformation, QA, validation and archival process

Potential benefit of PROV to DASSH

There are number of aspects of our procedures that it would be useful for our team to record using PROV. These are both items that we already log but are not yet in a standardized interchangeable format and things not currently recorded at all.

Examples:
- Was the dataset transformed? From which format?
- Who transformed it? How long did it take?
- Who QA'd it? How many changes did it need?
- How many passes of the validation software were needed?
- Which version of the validation software was used?
- Which intermediate files were used for which stages?
- How long was spent on each stage?

These information points would be useful internally for tracking problems and externally for assessing data quality.

Example PROV document for DASSH DLC

An example PROV document in PROV-N notation was manually constructed to demonstrate the idealized form of a DASSH PROV document. It shows DASSH staff and DASSH software interacting with each other to transform, validate, correct and ingest (publish) an example dataset. It also tracks the various intermediate files and notes the end times for the various steps. This is shown in full in Listing 7.

```
document
  default <http://example.org/default>
  prefix dassh <http://dassh.ac.uk/provenance>
  bundle dassh:DASSHDT00000451
    agent(dassh:kevpax,[prov:type='prov:Person'])
    agent(dassh:matarn,[prov:type='prov:Person'])
    agent(dassh:annluf,[prov:type='prov:Person'])
    agent(dassh:validator_v2.4,[prov:type='prov:SoftwareAgent'])
    agent(dassh:ingestor_v3.7,[prov:type='prov:SoftwareAgent'])
    entity(dassh:DASSHDT00000451_v1,[dassh:fileName="DASSHDT00000451_AS01.csv"])
    activity(a1,2018-10-02T01:44:36,2018-11-03T01:45:36,[dassh:act_type="transform",
                                                         dassh:source_format="cefas"])
    wasAssociatedWith(a1,dassh:annluf,-)
    used(a1,dassh:DASSHDT00000451_v1,-)
    entity(dassh:DASSHDT00000451_v2,[dassh:fileName="DASSHDT00000451_AS01.csv"])
    wasGeneratedBy(dassh:DASSHDT00000451_v2,a1,-)
    activity(a2,2018-10-04T01:44:36,2018-10-
                    04T01:45:36,[dassh:act_type="qa",dassh:result="pass"])
    wasAssociatedWith(a2,dassh:matarn,-)
    used(a2,dassh:DASSHDT00000451_v2,-)
    activity(a3,2018-10-04T01:44:36,2018-10-
                    04T01:45:36,[dassh:act_type="validation",dassh:result="fail"])
    wasAssociatedWith(a3,dassh:kevpax,-)
    wasAssociatedWith(a3,dassh:validator_v2.4,-)
    used(a3,dassh:DASSHDT00000451_v2,-)
    activity(a4,2018-10-04T01:44:36,2018-10-04T01:45:36,[dassh:act_type="edit"])
    wasAssociatedWith(a4,dassh:annluf,-)
    used(a4,DASSHDT00000451_v2,-)
    entity(dassh:DASSHDT00000451_v3,[dassh:fileName="DASSHDT00000451_AS01.csv"])
    wasGeneratedBy(dassh:DASSHDT00000451_v3,a4,-)
    activity(a5,2018-10-05T01:44:36,2018-10-
                    05T01:45:36,[dassh:act_type="validation",dassh:result="pass"])
    wasAssociatedWith(a5,dassh:kevpax,-)
    activity(a6,2018-10-05T01:44:36,2018-10-05T01:45:36,[dassh:act_type="ingestion"])
    wasAssociatedWith(a5,dassh:kevpax,-)
    wasAssociatedWith(a3,dassh:ingestor_v3.7,-)
    used(a6,dassh:DASSHDT00000451_v3,-)
  endBundle
endDocument
```

Listing 7: PROV-Tempalte describing the DASSH data transformation process

This PROV document has been visualised in **Fehler! Verweisquelle konnte nicht gefunden werden.** using the Python library PROV[41].

In preparation, the above document had to be converted from PROV-N to PROV-JSON using provconvert from the java software ProvToolbox[42].

[41] https://github.com/trungdong/prov
[42] https://lucmoreau.github.io/ProvToolbox

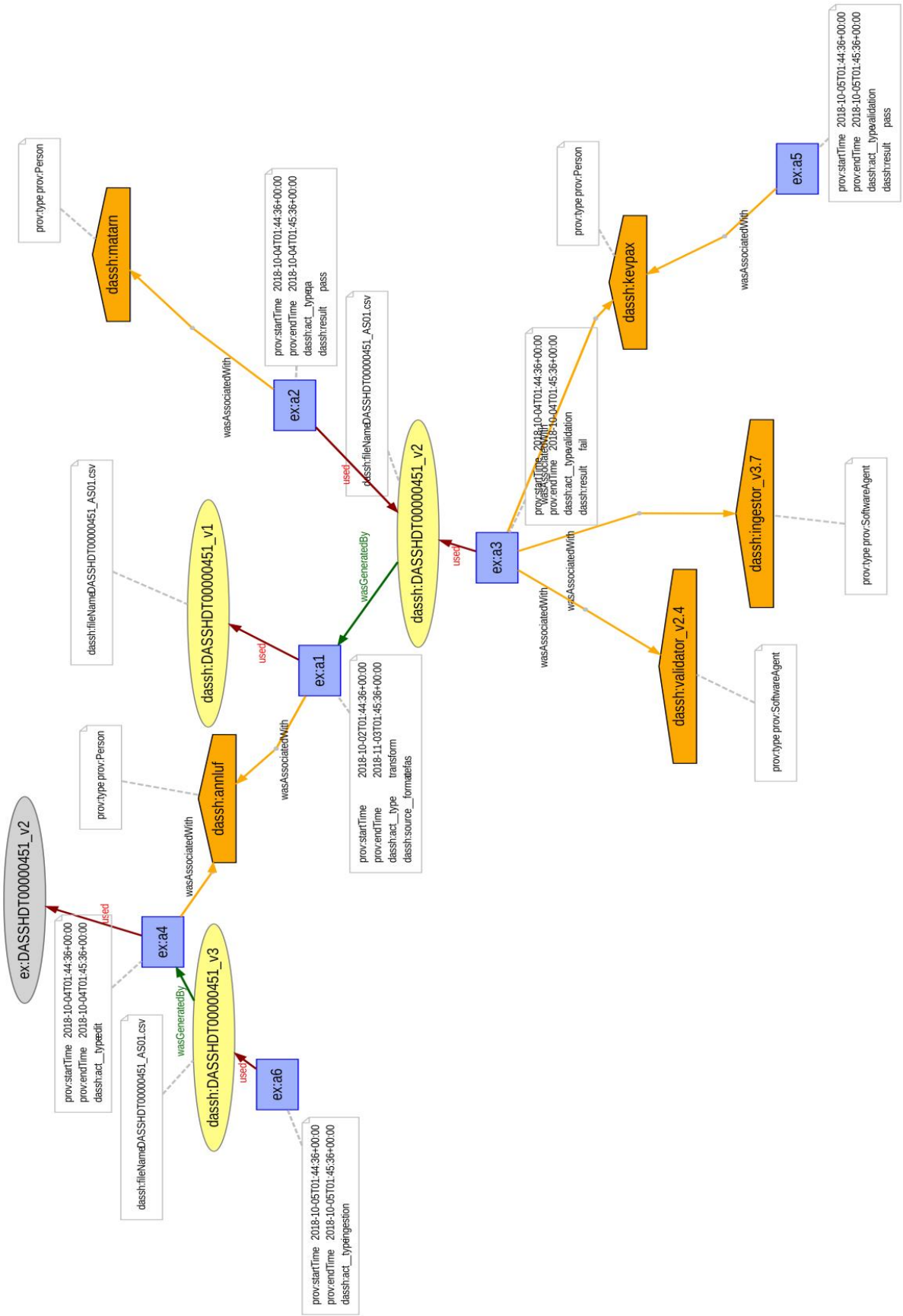

Figure 22: Visualization of the PROV document describing the DASSH data transformation process

Generating PROV documents automatically

Obviously it is not feasible to create a valid PROV document for every dataset by hand. The PROV-TEMPLATE system describes a process in which a bindings file and template are used together with provconvert in order to generate a PROV document.

The system is designed to generate these documents from CSV/TSV style log files. The main problem for DASSH is that we do not store log files of this nature. It is only recently that we began logging these steps at all, and the place we use for this is GitLab. Gitlab is a web-based manager of git repositories (software versioning system) and issue tracker. We use it to track overall projects (such as data ingestion) and individual datasets (as an Issue). It quickly became clear that to generate the log files needed by the templating system we would have to use the GitLab API to generate them.

GitLab as a standardised logging system

There is an excellent API for GitLab[43]. The main challenge is to log events in GitLab in a systematic way so that they can be harvested programmatically. DASSH's standard method is to write a comment – parsing these for keywords was discarded as too unreliable. Another feature that GitLab provides and which we were already using to some extent is Labels. These are custom created markers which can be added and taken away from an Issue at any point. As well as the Label itself, GitLab records the time the Label was added and the user who added it. By creating a Label for every activity we wish to log, they could provide an excellent way to standardise the logging of DASSH DLC steps between staff members and datasets.

Example labels (already in use):
- Started
- In MEDIN format
- Ready for QA
- Ready for ingestion (QA finished)
- Blocked

Example potential Labels to be added:
- Transform required
- Submitted for validation
- Passed Validation
- Corrections Required
- Published

By expanding the number of these Labels and applying them in a more rigorous way, we would provide a standardized logging system from which CSV logfiles could be generated, via the GitLab API and a parent script.

Generating log files from GitLab

GitLab uses a RESTful API. The two utilized commands are:

- Get an Issue description (to retrieve general information about the dataset)[44]

[43] https://docs.gitlab.com/ee/api/
[44] https://gitserver.mba.ac.uk/api/v4/projects/19/issues/153

- Get notes for an Issue (includes the Labels)[45]

A script was written to extract the desired information from Gitlab and to build the CSV file (gitlab_to_log.py) and is given below in Listing 8.

We have attempted to make the script as general as possible but it will require modification before being used by others, especially the part that parses the description for dataset information. The Labels section should be more portable providing that the Label dictionary is updated with the custom Labels used by the RI in question.

Example command:
gitlab_to_log.py –gitlab 'https://gitserver.mba.ac.uk' --issue_num 102 --token 1K34JK234H22H34 --logfile issue_102_logfile.csv

(note – the token is just an example)

```
#!/usr/bin/python3
# A script to create a log file from gitlab for a given dataset
# This file could feed into PROV-TEMPLATE system

import argparse
import requests
import copy

# get arguments
parser = argparse.ArgumentParser(description=
                                 "Script to to create a CSV log file from information
                                 held in GitLab"
                                 "It assumes one Issue = one dataset and uses Labels to
                                 extract "
                                 "standardised information")

parser.add_argument('-i','--issue_num',
            required=True,
            dest='issueNum',
            help="The issue number")
parser.add_argument('-l','--logfile',
            required=True,
            dest='logfile',
            help="The log file to be written out")
parser.add_argument('-t','--token',
            required=True,
            dest='privateToken',
            help="The private token to be used to access gitlab")
parser.add_argument('-n','--namespace',
            default='dassh',
            dest='namespace',
            help="The namespace to be prepended to values")
parser.add_argument('-g','--gitlab',
            default='https://gitserver.mba.ac.uk',
            dest='gitlab',
            help="The location of the GitLab instance")
parser.add_argument('-p','--project_num',
            default='100',
            dest='projectNum',
            help="The GitLab project number")
args = parser.parse_args()

# The gitlab label numbers and what they mean
```

[45] https://gitserver.mba.ac.uk/api/v4/projects/19/issues/133/notes

```
labels = { 'transform':'28',
           'QA':'29',
           'metaDataRecorded':None,
           'readyIngest':None,
           'validated':None
}

# request issue
r=requests.get(args.gitlab+'/api/v4/projects/'+args.projectNum+'/issues/'+args.issueNu
m,
               headers={'PRIVATE-TOKEN':args.privateToken})
issue = r.json() # convert response to json (actually dictionary)

# request issue notes
r=requests.get(args.gitlab+'/api/v4/projects/'+args.projectNum+'/issues/'+args.issueNu
m+'/notes',
               headers={'PRIVATE-TOKEN':args.privateToken})
notes = r.json()

# Get guid and others from description
description = issue['description']
description = description.split('\n')
for pos,section in enumerate(description):
   if 'resource id' in section.lower():
      resourceID=description[pos+1].lstrip('>')
   if 'data location' in section.lower():
      dataLocation=description[pos+1].lstrip('>')
      dataLocation=dataLocation.split('\\')[-1]
   if 'guid' in section.lower():
      metadataID=description[pos+1].lstrip('>')

# determines order of fields in output log
logFields=['metadataid','file','user','department','activity','software','time']

# set up template for log line
# some of these values will be overwritten if necessary
logLineTemplate={'metadataid':'uuid:'+metadataID,
                 'file' : args.namespace+':'+dataLocation,
                 'user': args.namespace+':NULL',
                 'department':args.namespace+':dassh',
                 'activity':args.namespace+':NULL',
                 'software':args.namespace+':NULL',
                 'time': args.namespace+':NULL'}

# function to turn a log dict into a csv line
def createLogLine(logDict):
   outputLine=''
   for field in logFields:
      outputLine+=logDict[field]
      outputLine+=','
   outputLine=outputLine.rstrip(',')
   outputLine+='\n'
   return(outputLine)

# open output log file
logFile=open(args.logfile,'w')

# write header
logFile.write(','.join(logFields)+'\n')

# go through notes backwards (as they are stored most recent first)
for note in reversed(notes):
   body=note['body']
   # skip if no labels in this note/comment
   if 'label' not in body:
      continue
   log = copy.copy(logLineTemplate)
   log['user'] = args.namespace+':'+note['author']['username']
```

```
   log['time'] = '"'+note['updated_at']+'"'
   # If any of our defined labels are found
   # then write out a line of the log file
   for activity,labelNum in labels.items():
     if labelNum and 'added ~'+labelNum in body:
        log['activity'] = args.namespace+':'+activity
        logLine=createLogLine(log)
        logFile.write(logLine)
   logFile.close()
```

Listing 8: Python script extracting information from Gitlab and converting it to CSV

Output log file

An example CSV log file generated by the above script is shown in Table 4.

Table 4: CSV output extracted from Gitlab

```
metadataid, file, user, department, activity, software, time

uuid:6abc6408ad88364495e4a0451913a286, dassh:DASSHDT00000033-AS02.xlsx,
dassh:tholan,dass:dassh, dassh:transform, dassh:NULL, "2018-08-
28T16:16:43.966+01:00"

uuid:6abc6408ad88364495e4a0451913a286, dassh:DASSHDT00000033-AS02.xlsx,
dassh:matarn, dassh:dassh, dassh:QA, dassh:NULL, "2018-08-
31T11:06:03.608+01:00"
```

Which has the fields:
metadataid – our unique identifier for this dataset
file – the file associated with this activity
user – the staff member associated with this activity
department - 'dassh'
activity – e.g. QA, Transformation, Validation, Correction
software – which software (including version) was used, if applicable
time – the end time of the activity

These match the entries in our bindings file.

The template and bindings files

To convert the CSV log file into a PROV document we require two things:

- A PROV Template – describes the entities and the links between them
- A PROV bindings file with placeholders – binds the entities to their values

Our versions used are both shown in Listing 9 and Figure 23**Fehler! Verweisquelle konnte nicht gefunden werden.**.

The DASSH template (dassh_template.provn) is shown here. It utilizes:
- 'var:' to assign variables from the bindings file
- 'vargen:' to generate unique uuid on the fly for associations
- 'tmpl:endTime' to assign the time to the activity (due to limitations this is not able to be mapped as a 'var:')

```
document
   prefix prov <http://www.w3.org/ns/prov#>
   prefix tmpl <http://openprovenance.org/tmpl#>
   prefix var <http://openprovenance.org/var#>
   prefix vargen <http://openprovenance.org/vargen#>
   prefix ex <http://example.com#>
   bundle ex:a
      entity(var:metadataid)
      entity(var:file)
      agent(var:department, [prov:type = 'prov:Organization'])
      agent(var:user, [prov:type = 'prov:Person'])
      agent(var:software, [prov:type = 'prov:SoftwareAgent'])
      activity(var:activity, [tmpl:endTime = 'var:time'])
      wasAssociatedWith(vargen:id; var:activity, var:user, -,
                                        [prov:role='var:personRole'])
      actedOnBehalfOf(var:user, var:department,-)
      wasGeneratedBy(var:file, var:activity,-)
      wasAttributedTo(var:file, var:user)
      hadMember(var:metadataid, var:file)
   endBundle
endDocument
```

Listing 9: PROV-Template to be substituted with CSV values

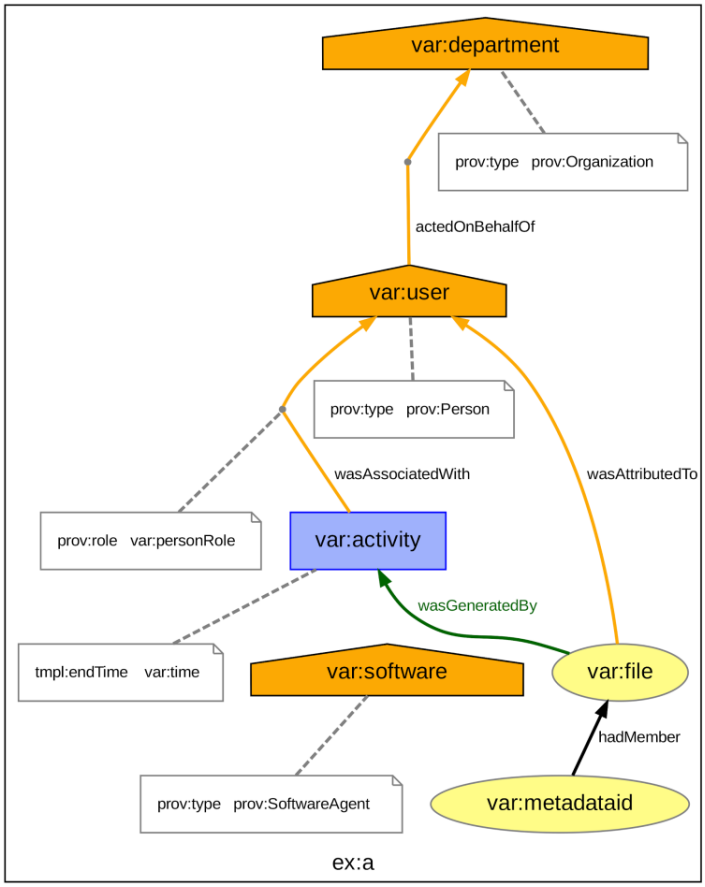

Figure 23: Visualization of PROV-Template to be substituted with CSV values. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bb71691d6fa335484d16d6e/svg

The DASSH bindings file with placeholders (dassh_bindings.ttl) is shown in Listing 10. The bindings file is given in Turtle notation (.ttl) as suggested by the PROV-TEMPLATE document. It has placeholders as described in the same, to be filled from a line of a log file.

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix uuid: <urn:uuid:> .
@prefix dassh: <http://dassh.ac.uk/prov#> .

var:metadataid a prov:Entity ;
    tmpl:value_0 $colX1 .
var:file a prov:Entity ;
    tmpl:value_0 $colX2 .
var:user a prov:Entity ;
    tmpl:value_0 $colX3 .
var:department a prov:Entity ;
    tmpl:value_0 $colX4 .
var:activity a prov:Entity ;
    tmpl:value_0 $colX5 .
var:software a prov:Entity ;
    tmpl:value_0 $colX6 .
var:time a prov:Entity ;
    tmpl:value_0 $colX7 .
```

Listing 10: Bindings structure to be populated with CSV values

Populating the bindings template

In the PROV-TEMPLATE example a bash script was used to place the log file values into a bindings template. We have decided to separate the bindings template from the script and also to re-write it in Python to try to improve operability between operating systems. The script also allows the specification of different separators (TSV/CSV/etc) and placeholder styles. The script to populate a binding template to create full bindings files is called fill_bindings_template.py and it is shown in Listing 11.

It takes a bindings template and logfile and outputs one binding file per row. It also creates a list of bindings files created (by default called bindingsFileList.txt).

An example command is:
fill_bindings_template.py --header –bindings_template dassh_bindings.ttl --logfile issue_logfile.csv –seperator ‘,’ –placeholder ‘$colX’

```
#!/usr/bin/python3
# A script to take a bindings template and a log file and generates a bindings file
for each row
# to be used in the PROV-TEMPLATE system
# output files will take form bindings_file_N.ttl
# also creates a list of files created (default bindingFilesList.txt)

import argparse

# get arguments
parser = argparse.ArgumentParser(description="Takes a bindings template and a log file
"
                                             "and generates a bindings file for each
                                             row")
parser.add_argument('-t','--bindings_template',
            required=True,
            dest='bindingsTemplate',
            help="The bindings template to be used")
parser.add_argument('-l','--logfile',
            dest='logfile',
            required=True,
            help="The CSV log file to be used")
parser.add_argument('-H','--header',
```

```
            dest='header',
            action='store_true',
            help="Specifies if the log file contains a header (which will be
            skipped)")
parser.add_argument('-s','--seperator',
            dest='seperator',
            default=',',
            help="The seperator in the log file - for CSV this will be ','")
parser.add_argument('-p','--placeholder',
            dest='placeholder',
            default='$colX',
            help="The style of placeholder used. A number starting from one will "
            "be appended and incremented for each column")
parser.add_argument('-b','--bindings_list',
            dest='bindingsList',
            default='bindingsFileList.txt',
            help="The file to hold the list of bindings files created")
args = parser.parse_args()

# read log file
logFile=open(args.logfile,'r')
logEntries=[]
while True:
   line=logFile.readline()
   if not line:
      break
   values=line.split(args.seperator)
   logEntries.append(values)

# read bindings template file
bindingsTemplateFile = open(args.bindingsTemplate,'r')
bindingsTemplateBlank = bindingsTemplateFile.read()
bindingsTemplateFile.close()

bindingsFileList=open(args.bindingsList,'w')
for rowNum,row in enumerate(logEntries):
   # skip header if present
   if rowNum == 0 and args.header:
      continue
   # replace the placeholders in the template
   bindingsTemplate=bindingsTemplateBlank
   for pos,value in enumerate(row,1):
      placeholder=args.placeholder+str(pos)
      # get rid of .ttl protected values in non-literals (haven't found a way of
      escaping these)
      if value[0] != '"' or value[-1] != '"':
         value = value.replace('.','').replace(';','').replace('@','')
      bindingsTemplate = bindingsTemplate.replace(placeholder,value)
   # write out a bindings file
   bindingsFileName = 'bindingsFile_'+str(rowNum)+'.ttl'
   bindingsFile = open(bindingsFileName,'w')
   bindingsFile.write(bindingsTemplate)
   # add bindings file name to list
   bindingsFileList.write(bindingsFileName+'\n')
bindingsFileList.close()
```

Listing 11: Python script to create bindings from CSV values

Note the section which removes protected ttl characters (‘.’, ’;’ and ’@’) inside non-literal values. For example the filename value dassh:’DASSHDT00000033-AS02.xls’ was causing a problem because of the period in the name. Although escaping these with ‘\’ should be valid syntax, provconvert throws an error when these are present. The only solution found was to remove them entirely.

The overall script

A wrapper script was developed to bring all the elements together (create_prov.py). Python was used in an attempt to introduce OS agnosticism as opposed to the bash script utilized in the PROV-TEMPLATE document example.
The script takes as input:
- A PROV-TEMPLATE template
- A 'bindings template' e.g. a bindings file with placeholders
- The issue number from GitLab
- The password (private token) used to access GitLab

Other optional parameters are possible to add further customization.

The script performs the following steps:
- Uses gitlab_to_log.py to create a CSV from a given issue on GitLab
- Uses fill_bindings_template.py to create a bindings file for each row of the CSV
- Uses provconvert to create a PROV document from each bindings file and the template
- Uses provconvert to merge the PROV documents into a single output document

An example command is:
create_prov.py -t dassh_template.provn -b dassh_bindings.ttl -i 22 -p 1K34JK234H22H34

The script is given in full in Listing 12.

```
#!/usr/bin/python3

# A script to create a PROV document (provn) from a dataset's GitLab logs
# Uses the PROV-TEMPLATE system

import os
import argparse

# get arguments
parser = argparse.ArgumentParser(description="Script to create a PROV document from "
                                             "a GitLab issue. Uses the PROV-TEMPLATE
                                             system.")
parser.add_argument('-t','--template',
            required=True,
            help="A PROV-TEMPLATE template file")
parser.add_argument('-b','--bindings_template',
            required=True,
            dest='bindingsTemplate',
            help="A bindings file with placeholders")
parser.add_argument('-l','--logfile',
            dest='logFile',
            default='issue_logfile.csv',
            help="The log file to be written out")
parser.add_argument('-o','--output_file',
            dest='outputFile',
            default='provDoc_final.provn',
            help="The final output PROV document")
parser.add_argument('-i','--issue_num',
            required=True,
            dest='issueNum',
            help="The issue number of the dataset in GitLab")
parser.add_argument('-p','--private_token',
            dest='privateToken',
            required=True,
            help="The private token to be used to access GitLab")
args = parser.parse_args()

# run gitlab_to_log to extract a CSV log file for a gitlab issue
os.system('gitlab_to_log.py --issue_num '+args.issueNum+' --logfile '+args.logFile+' -
```

```
-token '+args.privateToken)

# Create a number of bindings files using the bindings template and a logfile
os.system('fill_bindings_template.py -H --bindings_template '+args.bindingsTemplate+'
--logfile '+args.logFile)

# open bindings file list - the default from fill_bindings_template.py is
bindingFilesList.txt
bindingsFileList = open('bindingsFileList.txt','r')

# open a mergelist to record output prov documents to be merged
mergeList = open('prov_mergelist.txt','w')

# For each bindings file, combine with template to create a prov document
while True:
   bindingsFileName = bindingsFileList.readline()
   if not bindingsFileName:
      break
   bindingsFileName=bindingsFileName.strip()
   provDocFileName =
   bindingsFileName.replace('bindingsFile','provDoc').replace('.ttl','.provn')
   os.system('provconvert -infile '+args.template+' -bindings '+bindingsFileName+' -
   outfile '+provDocFileName)
   mergeList.write(provDocFileName+'\n')

# close files
bindingsFileList.close()
mergeList.close()

# Merge into single document
os.system('provconvert -flatten -merge prov_mergelist.txt -outfile '+args.outputFile)
```

Listing 12: Wrapper script performing value extraction from Gitlab, binding creation and tempate expansion

Output

Output from the above script is shown in Listing 13. It is from a real dataset and describes two activities – a transformation by one staff member and a QA check by another.

It does not yet reach the desired detail as shown in our initial PROV example – the main hurdle to this is the introduction of appropriate Labels and the training of staff to add these systematically.

Other noted issues with the output are that it does not preserve whitespace formatting from the template, which makes it harder for humans to read. It has also erroneously removed the 'prov:' prefix definition from the template. Both problems are attributed to current issues with provconvert.

```
document
prefix uuid <urn:uuid:>
prefix dassh <http://dassh.ac.uk/prov#>
entity(uuid:6abc6408ad88364495e4a0451913a286)
entity(dassh:DASSHDT00000033-AS02xlsx)
activity(dassh:QA,-,-)
activity(dassh:transform,-,-)
agent(dassh:dassh,[prov:type = 'prov:Organization'])
agent(dassh:tholan,[prov:type = 'prov:Person'])
agent(dassh:matarn,[prov:type = 'prov:Person'])
agent(dassh:NULL,[prov:type = 'prov:SoftwareAgent'])
wasGeneratedBy(dassh:DASSHDT00000033-AS02xlsx,dassh:transform,-)
wasGeneratedBy(dassh:DASSHDT00000033-AS02xlsx,dassh:QA,-)
```

```
wasAssociatedWith(uuid:bc07b090-8756-40cd-8b68-10b4facb670c;dassh:QA,dassh:matarn,-)
wasAssociatedWith(uuid:463e5fd6-57bf-4013-bd52-
e1c320088528;dassh:transform,dassh:tholan,-)
wasAttributedTo(dassh:DASSHDT00000033-AS02xlsx, dassh:tholan)
wasAttributedTo(dassh:DASSHDT00000033-AS02xlsx, dassh:matarn)
hadMember(uuid:6abc6408ad88364495e4a0451913a286,dassh:DASSHDT00000033-AS02xlsx)
actedOnBehalfOf(dassh:tholan,dassh:dassh,-)
actedOnBehalfOf(dassh:matarn,dassh:dassh,-)
endDocument
```

Listing 13: Resulting PROV output for DASSH process

Conclusion

We have demonstrated that it is possible to automatically generate PROV documents for DASSH using the PROV-TEMPLATE system. We have also laid the groundwork for extracting CSV log files from GitLab using Labels as a standardised logging system. We hope our three scripts (gitlab_to_log.py, fill_bindings_template.py and create_prov.py) will be useful to others attempting to implement the PROV-TEMPLATE system, as well as our published template and bindings files.

Although the technology has been tested successfully, the main hurdle for implementation in this Data Archive Centre remains the inconsistency of our logging practices. For the outlined GitLab extraction system to work, we must require staff members to record the appropriate Label at the appropriate time and enforce this in some way. This will necessitate some retraining and the development of new procedures. We also need to expand the types of Labels used to cover all the activities we wish to track. It may be beneficial to standardise Label names and uses between RIs.

Finally, we would also need to create the dassh.ac.uk/prov namespace referenced in our template and create the required definitions.

5.5 LTER

Doron Goldfarb – EAA (Umweltbundesamt GmbH), Austria

Background

Long-Term Ecosystem Research (LTER) is an essential component of worldwide efforts to better understand ecosystems. Through research and monitoring, LTER seeks to improve our knowledge of the structure and functions of ecosystems and their long-term response to environmental, societal and economic drivers. LTER contributes to the knowledge base informing policy and to the development of management options in response to the Grand Challenges under Global Change.

LTER Europe[46], established in 2003, aims to improve harmonisation and standardisation of long term observation in the ecosytem domain. With its network of sites and researcher is one of the European scale infrastructures in place providing data and expertise in order to bridge the gap between science and decision making. Getting an overview and access to information on available data and scientific observation infrastructures is therefore one of the central points to the success for the implementation of data intensive science. In 2018

[46] https://www.lter-europe.net/

eLTER was listed on the ESFRI roadmap for ecosystem based research infrastructures with the plan for the formalisation in the coming years.

The European LTER network is comprising about 420 formally acknowledged ecosystem research sites (65% terrestrial, 26% aquatic and 9% transitional waters LTER Sites) and 35 LTSER Platforms for socio ecological research at the regional scale in 26 member countries. The infrastructures are operated by around 100 institutions being responsible for the collection and quality assessment of the data. The site infrastructure as well as data collected at the sites are documented with DEIMS-SDR[47], providing also a unique registration of the observation facilities.

The curation of data and the provision of metadata are key processes in each of the research sites. Getting an added value on the comparison of the ecological data, also between different projects, requires that the data is available and discoverable for usage by researchers. A distributed infrastructure allows each site to maintain data that has been collected. The eLTER Information System[48] provides a single access point to data from LTER sites.

Workflow and use case selection

LTER focuses on the simultaneous observation of different ecosystem compartments and processes within a defined area. Observations range from continuous automated sensor based measurements (e.g. meteorology, soil temperature) to periodic human based observations (e.g. vegetation surveys or soil surveys). As of today, it is still often the case that measurements are either directly taken by hand or read from “analog” measurement devices by humans, noted by hand in dedicated forms which are later transferred to digital representations such as Excel sheets. Since data generated that way are integrated into larger scale datasets made available for later re-use, it is of strong interest to describe their provenance using similar approaches as applied to data generated and processed in more automated settings.

The use-case presented in this section addresses these issues by describing an example workflow for extracting provenance information from Excel sheets in the context of LTER data acquisition. It shows how information about the data acquisition process which is only present in implicit form in the Excel sheet can be made more explicit in form of a PROV-template, which is subsequently populated with the extracted data. This approach is of special interest since it can potentially be applied to similar processes such as sample based observations, e.g. soil water sampling or deposition, where additional information related to the sampling and sample treatments is recorded alongside the observed values.

Workflow - Regular recording of stem increment using dendrometers

The worksheet shown in Figure 24 is used for collecting Integrated Monitoring results from various measuring devices at the Austrian LTER site Zöbelboden[49] and the example PROV conversion will focus on some aspects thereof, including the following information: the upper left side of the worksheet features date and time information and the name(s) of the personnel responsible for the measurement readings. On the right side, the section “Stammzuwachs” (stem

[47] https://deims.org
[48] https://data.lter-europe.net/
[49] http://www.lter-austria.at/zoebelboden/

growth or stem increment) represents tree growth measurement information collected from dendrometer devices attached to a selection of trees, measuring relative changes of their circumference. The leftmost column of this section represents the ID for each tree (MONr.), the next column (Dendrometer) the ID of the attached Dendrometer, followed by the reading of its currently displayed value performed by the responsible personnel ([cm]). The fourth column (neu) is not used for this example, while the fifth and last column (Anmerkung) indicates if there is a specific comment for one of the readings. The remainder of this section describes the process of turning this information into PROV via PROV-template.

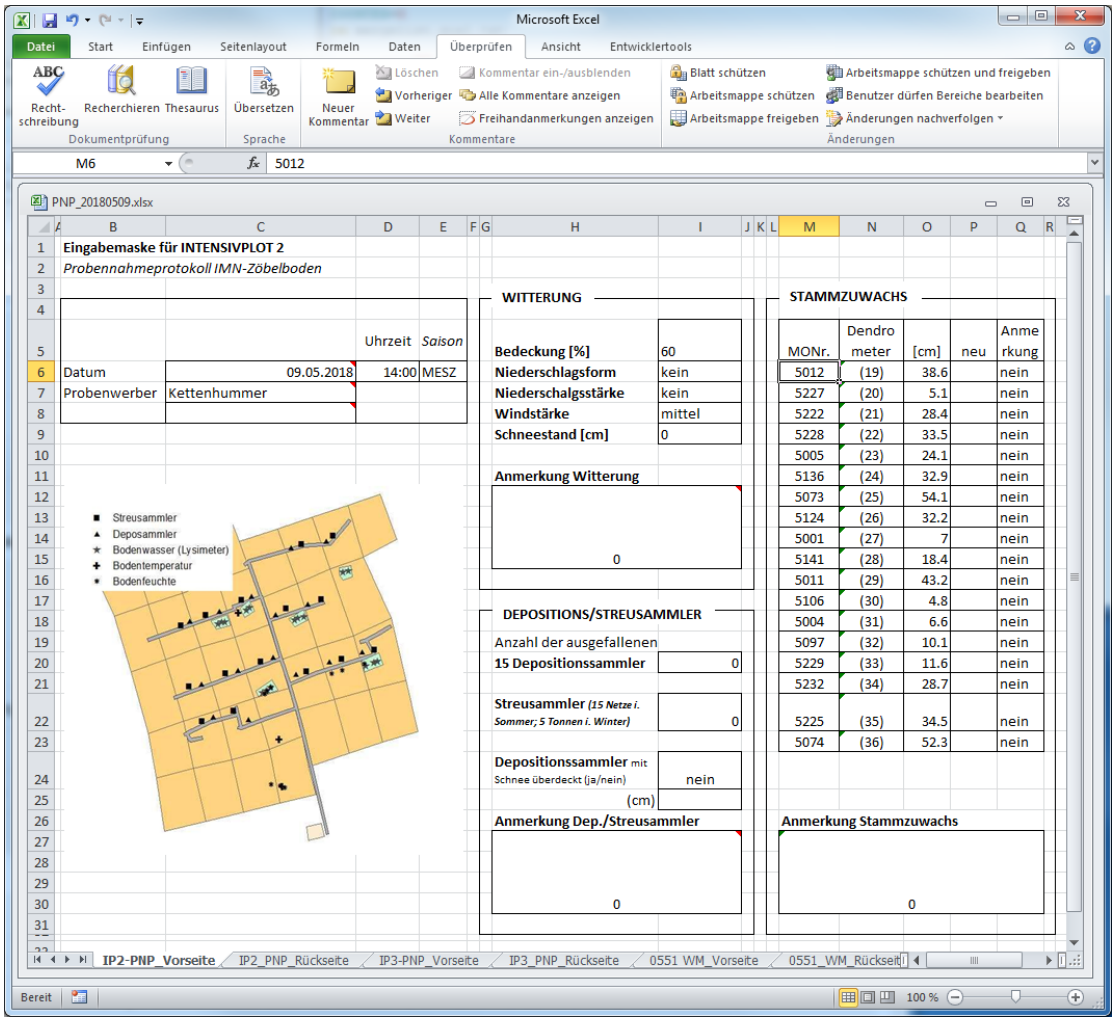

Figure 24: Spreadsheet for collecting Integrated Monitoring results at the LTER site Zöbelboden

5.5.1 Extracting information from Excel to columns

The information from the Excel worksheet is extracted and converted to column form using the Python openpyxl library. The script shown in Listing 14 uses this library to extract and rearrange the information from the sheet from Figure 24 into columns, resulting in a row wise representation as shown in Table 5.

```
import openpyxl
import sys
import dateutil.parser as parser
import dateutil.tz as tz

def checkNone(data):
        if data==None:
                return ""
        else:
                return str(data)

wb=openpyxl.load_workbook(sys.argv[1])
data=wb["IP2-PNP_Vorseite"]

date=data["c6"].value

counter=7
persons=list()
run=True
while run:
        index="c"+str(counter)
        run=True
        try:
                if data[index].value!=None and data[index].value!="":
                        persons.append(data[index].value)
                else:
                        run=False
        except:
                run=False
                pass
        counter+=1

counter=6
trees=dict()
run=True
fnames=[data["m5"].value,
data["n5"].value,data["o5"].value,data["p5"].value,data["q5"].value]
while run:
        index="m"+str(counter)
        run=True
        try:
                if data[index].value!=None and data[index].value!="":
                        idx=checkNone(str(data[index].value))
                        trees[idx]=dict()
                        tree=trees[idx]
                        tree[fnames[1]]=checkNone(data["n"+str(counter)].value)
                        tree[fnames[2]]=checkNone(data["o"+str(counter)].value)
                        tree[fnames[3]]=checkNone(data["p"+str(counter)].value)
                        tree[fnames[4]]=checkNone(data["q"+str(counter)].value)
                        trees[idx]=tree
                else:
                        run=False
        except:
                run=False
                pass
        counter+=1

for p in persons:
        for t in trees:
```

```
                print str(date.isoformat()) + "\t" + p + "\t" + t + "\t" + \
                        trees[t][fnames[1]] + "\t" + \
                        trees[t][fnames[2]] + "\t" + \
                        trees[t][fnames[3]] + "\t" + \
                        trees[t][fnames[4]]
```

Listing 14: Python script for extracting column-wise data from Excel

Table 5: Information from Excel sheet as columns

2018-05-09T14:00:00+02:00	Kettenhummer	5011	(29)	43.2	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5012	(19)	38.6	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5228	(22)	33.5	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5005	(23)	24.1	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5004	(31)	6.6	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5229	(33)	11.6	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5222	(21)	28.4	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5232	(34)	28.7	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5227	(20)	5.1	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5001	(27)	7	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5225	(35)	34.5	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5141	(28)	18.4	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5073	(25)	54.1	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5124	(26)	32.2	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5136	(24)	32.9	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5097	(32)	10.1	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5106	(30)	4.8	nein
2018-05-09T14:00:00+02:00	Kettenhummer	5074	(36)	52.3	nein

5.5.2 Converting columns to PROV

In order to convert the column information to PROV, a PROV template first needs to be created, reflecting the intended provenance chain for the present information. One approach (out of many possible ones) is to represent the chain as two separate processes. The first process consists of the dendrometers attached to the individual trees, performing continuous measurements of their circumference, resulting in continuous measurements displayed on each dendrometer. The second process consists of dedicated personnel visiting the different trees, taking readings of the current measurements displayed on the dendrometers, resulting in individual fixed data values, potentially with additional comments such as the state of the measuring device or other contextual information. These data values then become members of the resulting dataset, to which the collected provenance information could be attached.

An example PROV-template document for this view is provided in Listing 15 in RDF-TriG form, its visualization is shown in Figure 25. For pure demonstration purposes, it features two currently "hardcoded" elements, "var:organization" and "var:dendroPlan" for which no information is present in the data extracted from the example Excel spreadsheet. The former could be used to encode the affiliation of the personnel performing the readings, the latter is used as an example how explicit information about the methodology behind the dendrometer measurements could be attached via a prov: plan property assigned to the associations between dendrometer and

their related measurement activities. These two special cases aside, the “vargen” namespace is extensively used for those elements for which no information is explicitly provided via the data extracted from the spreadsheet: This way, each measuring/reading activity instance and the resulting measurement/reading entity instances get assigned with randomly generated IDs.

```
@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#> .


ex:b {
        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 .

        vargen:dendroAssoc a prov:Association ;
                prov:agent var:dendrometer .

        vargen:measuringActivity prov:qualifiedAssociation vargen:dendroAssoc .

        vargen:dendroAssoc prov:hadPlan var:dendroPlan .

        vargen:measurementData prov:wasGeneratedBy vargen:measuringActivity .

        var:dataset a prov:Entity , "Dataset"^^xsd:string .

        vargen:readingData a prov:Entity , "DendrometerReading"^^xsd:string ;
                prov:value var:readValue ;
                ex:comment var:comment .

        var:department a prov:Agent , prov:Organization .

        var:readingAgent a prov:Agent , prov:Person .

        vargen:readingActivity a prov:Activity , "DendrometerReadingActivity"^^xsd:string ;
                prov:atLocation var:dendrometer ;
                prov:wasAssociatedWith var:readingAgent .

        var:readingAgent prov:actedOnBehalfOf var:department .

        vargen:readingData prov:wasGeneratedBy vargen:readingActivity ;
                prov:wasAttributedTo var:readingAgent ;
                prov:wasDerivedFrom vargen:measurementData .

        vargen:readingActivity prov:used vargen:measurementData .

        var:dataset prov:hadMember vargen:readingData .
}
```

Listing 15: PROV-template for dendrometer readings

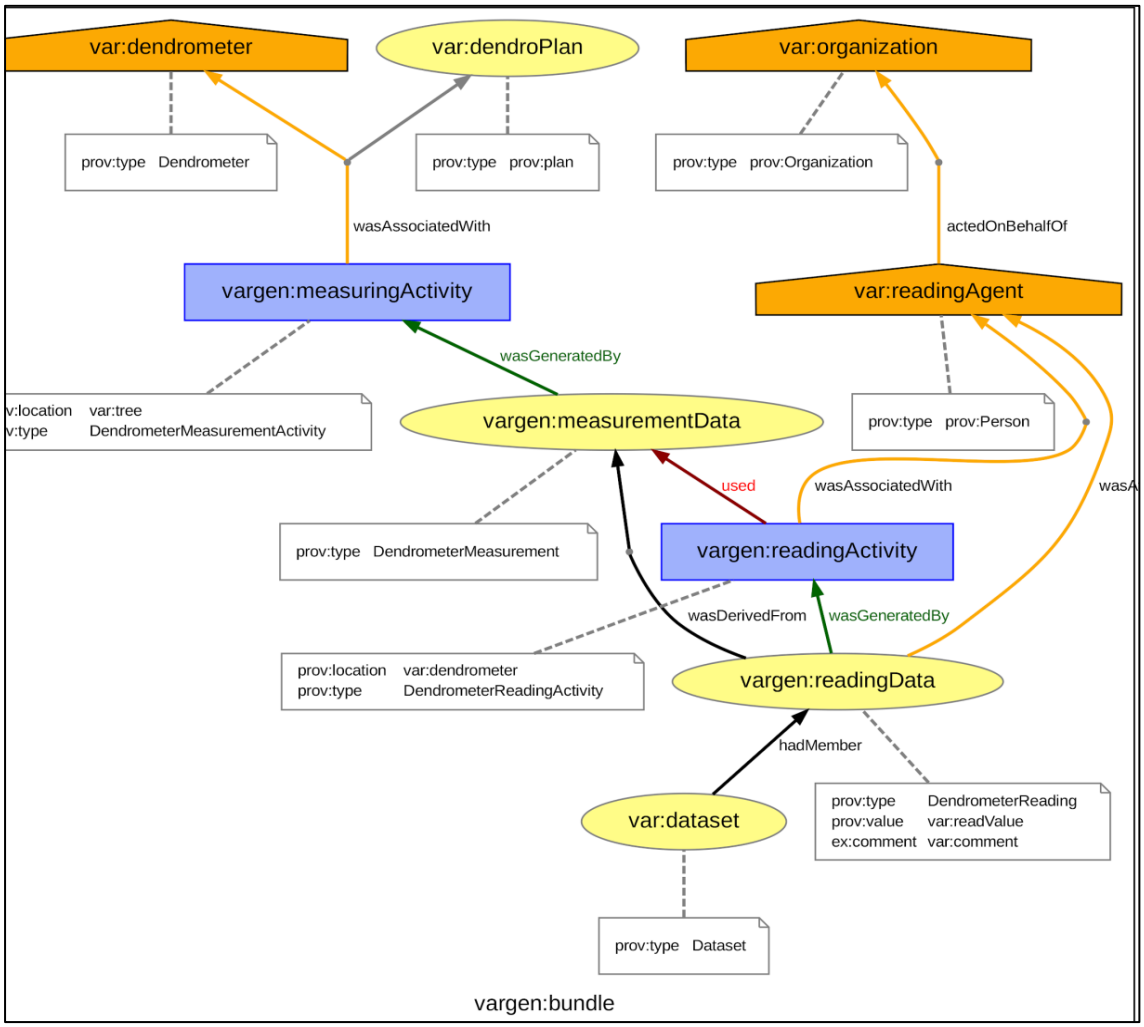

Figure 25: Visualization of PROV-template for dendrometer readings. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5baa34e8d6fa333791066dcf/svg

The template from Listing 15 is populated via row-wise bindings generated from iterating Table 5 Listing 16 shows a script for creating the bindings. In this example, "`ex:dendrometerMeasurementMethodology`" and "`ex:NP_Kalkalpen`" represent hardcoded values for the description of the measurement methodology in "`var:plan`" and the affiliation of the reading personnel "`var:organization`". The remaining variables are instantiated using values extracted from the table.

```
COUNTER=0
rm mergelist.qual.txt

while IFS=$'\t'; read col1 col2 col3 col4 col5 col6 col7
do
        cat <<EndOfMessage > tmpfile_${COUNTER}.qual.ttl
                        @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:$col4 .
                        var:tree a prov:Entity ;
                                   tmpl:2dvalue_0_0 ex:$col3 .
                        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:$col2 .
                        var:comment a prov:Entity ;
                                   tmpl:2dvalue_0_0 "$col7" .
                        var:readValue a prov:Entity ;
                            tmpl:2dvalue_0_0 "$col5" .
                        var:dataset a prov:Entity ;
                               tmpl:value_0 ex:$(echo $col1 | tr ":" "_") .
EndOfMessage

        echo "making expanded prov"
        provconvert -infile ${1} -bindings tmpfile_${COUNTER}.qual.ttl -outfile
${1}.out.${COUNTER}.qual.provn

        echo "file, ${1}.out.${COUNTER}.qual.provn, provn" >> mergelist.qual.txt

        COUNTER=$((COUNTER + 1))
done < ${2:-/dev/stdin}

echo "merging provn"
provconvert -flatten -merge mergelist.qual.txt  -outfile merged.qual.provn
```

Listing 16: Bash script for row-wise template expansion for Excel example

The PROV result for the expansion and merging of the first two rows of Table 5 is shown in Figure 26. The hardcoded instance “ex:dendroMeterMeasurementMethodology” is now referenced by the associations between each of the two dendrometers and their respective measurement activities, showing how global or individual information about the steps behind specific associations between agents and their activities can be referenced in PROV.

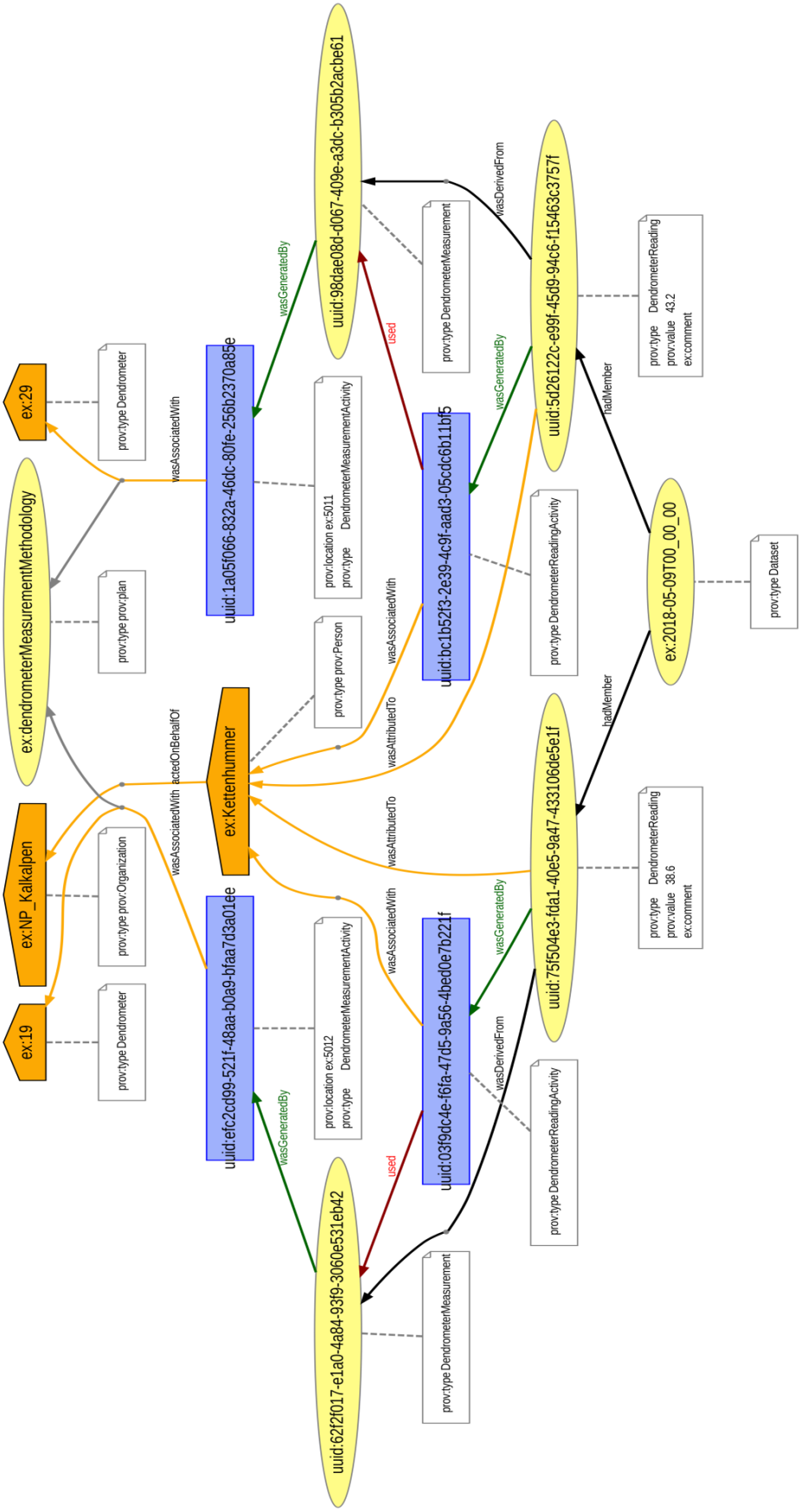

Figure 26: PROV document for the first two rows of the Excel extraction example

5.5.3 Conclusions

This experiment served as proof-of-concept that it is possible to extract legacy data from Excel spreadsheets and convert them to a PROV representation using PROV-template. It showed that the approach is generally feasible and could be used in production settings, although in some cases requiring the modification of the forms used for data collection. The latter insight underscored the general usefulness of trying to make the structure of the present information more explicit.

Experimenting with the PROV representation of the data acquisition workflow thus supported the formal analysis of the latter, helping to identify relevant elements to be captured in the provenance trace. The separation between the dendrometer measurements and their respective readings acquired via human personnel highlighted the general issue of choosing the suitable granularity when modeling and capturing provenance chains. In this regard, PROV-Template proved to be a useful way to modify the instantiated PROV-output accordingly.

Using the tested approach in a production setting would ideally go hand in hand with the establishment of dedicated registries for individual entity instances such as for persons, organizations and measurement devices, which should be represented by IDs dereferenceable to additional information about the respective agents and/or entities.

5.6 Particle Formation (TIB/PANGAEA)

Markus Stocker - TIB/PANGAEA, Germany

The ENVRIplus Science Demonstrator 6 (see ENVRIplus Deliverable D9.2) describes a service prototype that supports aerosol scientists in studying atmospheric new particle formation events by moving data analysis from local computing environments to interoperable infrastructures, thus harmonizing data analysis and the syntax and semantics of data derived from analysis.

As researchers analyse data, the infrastructure generates provenance. The workflows involves two activities that transform primary data into secondary and then tertiary data. Primary data are processed observational data, namely particle size distribution, originally acquired from a sensor network. Observational data are interpreted visually to determine whether a new particle formation event occurred (on a particular day and location). The result of such interpretation is secondary data, namely data about the event, such as the day and location at which the event occurred, the duration and classification of the event, and possibly other event attributes. In a second activity, data about events are analysed statistically e.g., mean event duration. Statistical data are tertiary data in this context.

We have used provenance templates [Moreau et al., 2018] to prototype how a system can specify templates that encode the provenance of tertiary data from secondary and originally primary data, as well as the activities and agents involved, and the time at which the activity was executed.

Listing 17 shows the Template I used to encode the provenance of secondary data, its visualization is shown in Figure 27. Secondary data are a PROV Entity and are derived from

interpreting the visualization of primary data. Secondary data are generated by an activity of 'data visualization' (OBI_0200111). The activity used the primary data and is associated with an agent, here an individual researcher.

```
@prefix obo: <http://purl.obolibrary.org/obo/> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix vargen: <http://openprovenance.org/vargen#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

vargen:bundleId {
    var:data a prov:Entity ;
        prov:wasDerivedFrom var:image ;
        prov:wasGeneratedBy vargen:visualization .

    vargen:visualization a prov:Activity, obo:OBI_0200111;
        tmpl:endTime var:t2 ;
        tmpl:startTime var:t1 ;
        prov:used var:image ;
        prov:wasAssociatedWith var:researcher .

    var:researcher a prov:Agent,
            prov:Person .

    var:image a prov:Entity .
}
```

Listing 17: Template I describing the provenance of secondary data about events as they are derived from visual products of primary data.

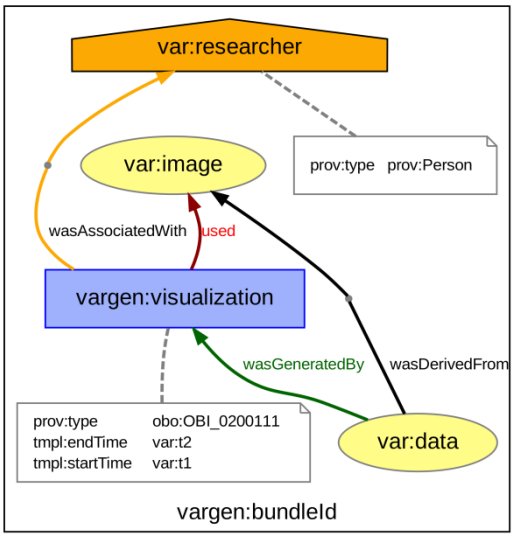

Figure 27: Visualization of Template I, available at https://envriplus-provenance.test.fedcloud.eu/templates/5bb24bfad6fa333a440a6613/svg

This general template can now be instantiated by providing bindings i.e., concrete substitutions of the variables with values. A possible binding is shown inListing 18. PROV Entities are identifiers of digital objects created, consumed, and curated on a D4Science VRE. We use ORCID to identify the researcher (PROV Agent) in provenance. Finally, we bind two timestamps for the start and end times of the activity, which is here instantaneous.

```
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix orcid: <http://orcid.org/> .
@prefix d4science: <https://data.d4science.org/> .

var:image a prov:Entity ;
     tmpl:value_0 d4science:MzhkMUdQZmRrSkxOc2kzWHA0amdlNlZTbW5yRWdGdUZHbWJQNStIS0N6Yz .

var:data a prov:Entity ;
     tmpl:value_0 d4science:K0JMcUorTjJib1Bka0hVdDI4SmU5N21wQmpubHBJOXdHbWJQNStIS0N6Yz0 .

var:researcher a prov:Entity ;
     tmpl:value_0   orcid:0000-0001-5492-3212 .

var:t1 a prov:Entity ;
     tmpl:2dvalue_0_0  "2018-09-28T14:59:27.177710+02:00" .

var:t2 a prov:Entity ;
     tmpl:2dvalue_0_0  "2018-09-28T14:59:27.177710+02:00" .
```

Listing 18: Example bindings for Template I with values for variables.

A similar discussion can be made for the second activity i.e. the computation of descriptive statistics (specifically mean event duration) from a dataset describing events, in particular their individual durations. Listing 19displays the Template II used to encode the provenance of such tertiary data, its visualization is shown in Figure 28. Tertiary data are a PROV Entity and 'scalar measurement datum' (IAO_0000032) and are derived in the activity of 'arithmetic mean calculation' (OBI_0200079) from datasets (IAO_0000100) about events.

```
@prefix obo: <http://purl.obolibrary.org/obo/> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tmpl: <http://openprovenance.org/tmpl#> .
@prefix var: <http://openprovenance.org/var#> .
@prefix vargen: <http://openprovenance.org/vargen#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

vargen:bundleId {
    var:dataset a prov:Entity, obo:IAO_0000100 .

    var:datum a prov:Entity, obo:IAO_0000032, obo:OBI_0000679 ;
        prov:wasDerivedFrom var:dataset ;
        prov:wasGeneratedBy vargen:calculation .

    vargen:calculation a prov:Activity, obo:OBI_0200079 ;
        tmpl:endTime var:t2 ;
        tmpl:startTime var:t1 ;
        prov:used var:dataset ;
        prov:wasAssociatedWith var:researcher .
    var:researcher a prov:Agent,
            prov:Person .
}
```

Listing 19: Template II describing the provenance of tertiary data about mean event durations as they are derived from datasets about events.

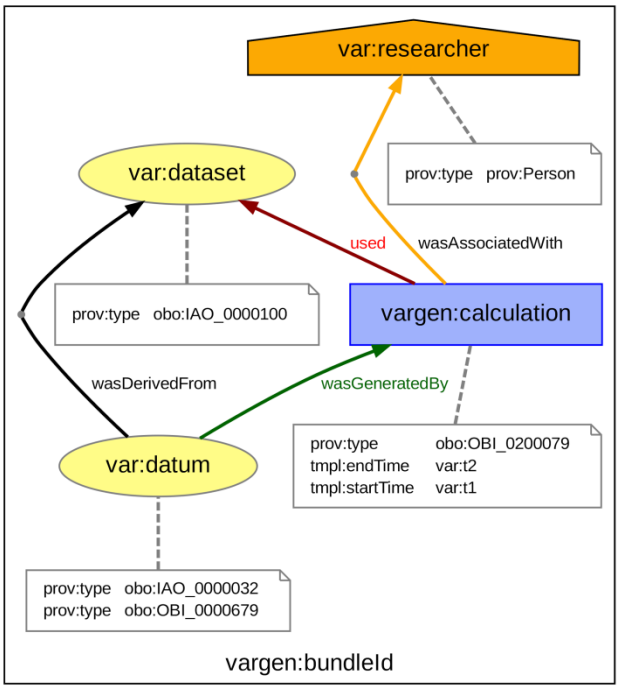

Figure 28: Visualization of Template II, available at https://envriplus-provenance.test.fedcloud.eu/templates/5bb64331d6fa334cdfe68062/svg

The bindings for Template II are similar to the ones shown in Listing 18 for Template I and are thus omitted here. Instead, Listing 20 shows the result of applying bindings to Template II. This PROV-N document represents provenance instance data relating a concrete descriptive statistic to the dataset from which it was computed, and the involved agent. This document can easily be converted e.g., to RDF/XML and thus be stored to a conventional triple store using existing provenance tools.

```
document
  prefix xml <http://www.w3.org/XML/1998/namespace>
  prefix tmpl <http://openprovenance.org/tmpl#>
  prefix uuid <urn:uuid:>
  prefix obo <http://purl.obolibrary.org/obo/>
  prefix tdata <http://avaa.tdata.fi/web/smart/smear/>
  prefix rdfs <http://www.w3.org/2000/01/rdf-schema#>
  prefix var <http://openprovenance.org/var#>
  prefix rdf <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
  prefix orcid <http://orcid.org/>
  prefix vargen <http://openprovenance.org/vargen#>
  prefix d4science <https://data.d4science.org/>

  bundle uuid:55bfe9fa-b5ca-4b7c-b898-3e45724818c4
    prefix orcid <http://orcid.org/>
    prefix obo <http://purl.obolibrary.org/obo/>
    prefix tdata <http://avaa.tdata.fi/web/smart/smear/>
    prefix uuid <urn:uuid:>
    prefix d4science <https://data.d4science.org/>

    agent(orcid:0000-0001-5492-3212, [prov:type='prov:Person'])
    entity(d4science:ZHBQWDMrQzJaT2JOc2kzWHA0amdlNVVVN0s5VlhtdkFHbWJQNStIS0N6Yz0,
[prov:type='obo:OBI_0000679'])
    activity(uuid:209dda6f-a2e2-421f-82f6-38e11e6ec632, 2018-09-28T13:12:52.976691+02:00,
2018-09-28T13:12:52.976691+02:00, [prov:type='obo:OBI_0200079'])
    entity(tdata:c1c3d1e86b786a36a16d9ee4312cd292, [prov:type='obo:IAO_0000100'])
    used(uuid:209dda6f-a2e2-421f-82f6-38e11e6ec632, tdata:c1c3d1e86b786a36a16d9ee4312cd292,
-)
```

```
    wasDerivedFrom(d4science:ZHBQWDMrQzJaT2JOc2kzWHA0amdlNVVVN0s5VlhtdkFHbWJQNStIS0N6Yz0,
tdata:c1c3d1e86b786a36a16d9ee4312cd292, -, -, -)
    wasGeneratedBy(d4science:ZHBQWDMrQzJaT2JOc2kzWHA0amdlNVVVN0s5VlhtdkFHbWJQNStIS0N6Yz0,
uuid:209dda6f-a2e2-421f-82f6-38e11e6ec632, -)
    wasAssociatedWith(uuid:209dda6f-a2e2-421f-82f6-38e11e6ec632, orcid:0000-0001-5492-3212,
-)
  endBundle
endDocument
```

Listing 20: Result of applying bindings to Template II.

The experiments described here suggest that the key advantage of using provenance templates lays in the possibility of decoupling the creation of provenance instance data from the program code that performs the actual data analysis. This enables a cleaner separation of concerns and system modularization. It also provides stability when that software changes.

On the other side, implementing the application of bindings to templates as a service means that systems rely on the availability of the service to generate provenance data. Since provenance occurs in activities, and activities are individually unique in time, systems need to guarantee relevant data are captured or logged in some form even if required services are temporarily unavailable. Furthermore, systems that generate provenance data fast (e.g., high frequency fully automated processes) may not be able to rely on a service for performance reasons.

In summary, especially for complex systems with intricate provenance data, we suggest that provenance templates is an interesting approach worth considering.

5.7 Log file conversion and harmonization (UvA/VRE4EIC)

Paul Marin, UvA, Netherlands

A major concern inherent in the effective use of PROV (or indeed any other provenance model) is the integration of logs, metadata, version histories and other provenance sources into a generally queryable substrate (e.g. a single virtual provenance graph). One use-case being currently investigated in the context of projects such as ENVRIplus and VRE4EIC[50] (concerning the general integration of virtual research environments with research infrastructure at multiple levels) is that of automatically integrating provenance information generated at the infrastructural, workflow and individual service levels. Specifically, how records generated during the execution of a Taverna workflow at multiple levels—by the workflow server, the underlying infrastructure and the actual services being invoked—can be automatically gathered together within a single provenance model, perhaps described (partially or wholly) using a PROV template and instantiated with the appropriate bindings on demand. Figure 29 illustrates the basic concept: a workflow collects, transforms and visualises data from or via multiple services and e-infrastructures, all of which produce some form of provenance. The challenge is thus how to gather and integrate that data.

[50] https://www.vre4eic.eu/

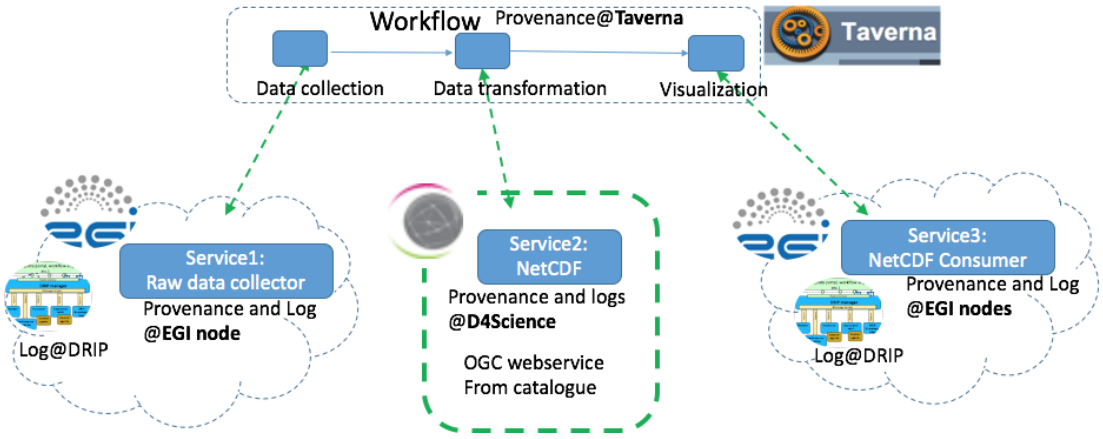

Figure 29: Vision of the provenance integration workflow as might be implemented using ENVRI services.

Figure 30 provides an example of a Web service as a workflow element in Taverna, as seen within the Taverna workbench GUI. Taverna is able to record metadata about a workflow, in particular the type, inputs and outputs of every element, in RDF and also generate some accompanying PROV data. Thus there is already some existing provenance data in suitable form available from parts of the system-to-be-integrated; the next step therefore is to bring those parts of the system not producing PROV-compliant data into a format conductive to integration.

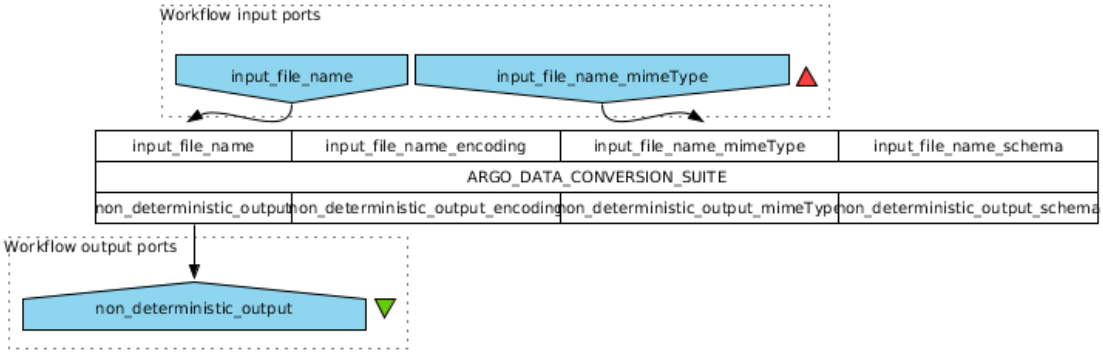

Figure 30: Example of a provenance-generating Taverna workflow element (Argo data conversion suite).

Thus a key feature of this multi-level provenance collection must be the conversion of infrastructural logs detailing various events that may happen during the execution of an scientific application on e-infrastructure, e.g. the running of a data subscription service on a scientific Cloud. Such events might be generated by exceptions thrown by the application code, or by the reception of requests via HTTP. In order to integrate these requests with the provenance information generated by the overarching workflow system, it is necessary to convert from textual entries to valid PROV triples, and it is necessary to be able to do this for batches of entries at once, so as to reduce the load on the conversion service.

The PROV-templating system developed within ENVRIplus was used to do this conversion; a simple template that supported the creation of multiple parallel PROV data structures was created along with a simple service for converting the textual logs into sets of bindings to send to the template service. The template is shown Figure 31.

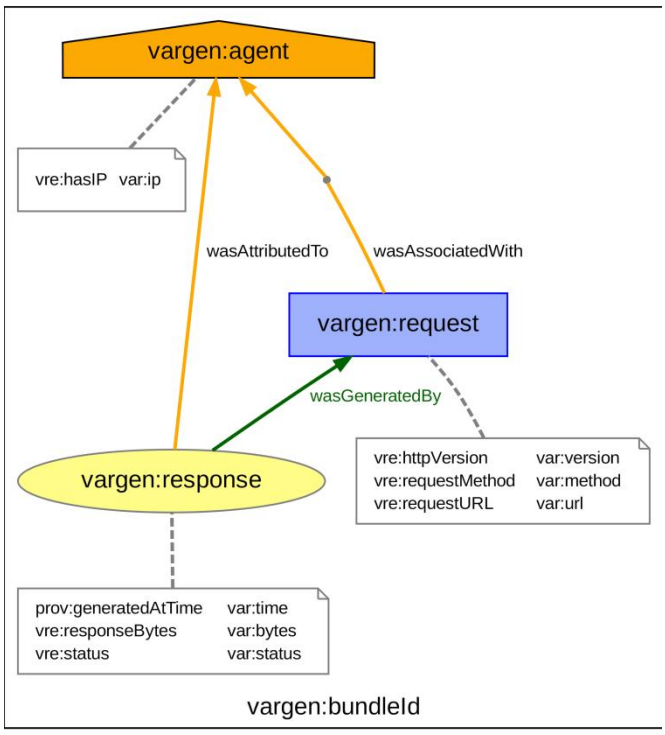

Figure 31: Template for converting logged HTTP requests into PROV data. Visualization available at https://envriplus-provenance.test.fedcloud.eu/templates/5bb768bad6fa335484d16d6f/svg

A typical log entry looks as so:

```
10.255.0.2 - - [27/Sep/2018:07:31:43 +0000] "GET /cue/rest/argo/get
HTTP/1.1" 200 1103475
```

Each log entry is composed of a number of parts: an IP address, a time stamp, a HTTP request (consisting internally of method, URI and protocol version), the resulting HTTP response code, and the response body. We model this log entry in PROV as a standard triplet of an Agent, an Activity and an Entity; in this model, the agent is the resource making the HTTP request (which is only discernible directly from the log by its IP address), the activity is the HTTP request itself (consisting of method, URI and protocol version) and the entity is the response (consisting of response code, time of generation and response body). Furthermore, we rely on the template service itself to assign UUIDs to individual requests and responses, but assign agent identifiers ourselves in order to allow us to group requests and responses made by the same agent where we are able to confirm from other parts of the system that the originator of a number of requests is indeed the same resource. We also take advantage of the linking grammar provided by the PROV-template specification to link each response to each request to each agent such that if we provide multiple log entries to the template service at once (as we typically do), we generate parallel PROV data for each entry rather than the cartesian product of all resources. An example of a binding produced for input into the template service is shown inListing 21; note that the first set of bindings corresponds to the log example above.

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

```
@prefix vre: <https://www.vre4eic.eu/log#> .

var:agent a prov:Entity .
var:ip a prov:Entity .
var:status a prov:Entity .
var:bytes a prov:Entity .
var:url a prov:Entity .
var:method a prov:Entity .
var:version a prov:Entity .
var:time a prov:Entity .

var:agent tmpl:value_0 vre:ag1 .
var:ip tmpl:2dvalue_0_0 "10.255.0.2" .
var:status tmpl:2dvalue_0_0 "200" .
var:bytes tmpl:2dvalue_0_0 "1103475" .
var:url tmpl:2dvalue_0_0 "/cue/rest/argo/get" .
var:method tmpl:2dvalue_0_0 "GET" .
var:version tmpl:2dvalue_0_0 "1.1" .
var:time tmpl:value_0 "2018-09-13T12:51:13+00:00" .

var:agent tmpl:value_1 vre:ag2 .
var:ip tmpl:2dvalue_1_0 "10.255.0.2" .
var:status tmpl:2dvalue_1_0 "200" .
var:bytes tmpl:2dvalue_1_0 "28172" .
var:url tmpl:2dvalue_1_0
"/cue/rest/argo/get?geospatial_lat_min=31.000&geospatial_lat_max=38.200&geospatial_lon_min=1
47.000&geospatial_lon_max=147.100&flowlabel=" .
var:method tmpl:2dvalue_1_0 "GET" .
var:version tmpl:2dvalue_1_0 "1.1" .
var:time tmpl:value_1 "2018-09-13T12:51:13+00:00" .

var:agent tmpl:value_2 vre:ag3 .
var:ip tmpl:2dvalue_2_0 "10.255.0.2" .
var:status tmpl:2dvalue_2_0 "200" .
var:bytes tmpl:2dvalue_2_0 "1103475" .
var:url tmpl:2dvalue_2_0 "/cue/rest/argo/get?geospatial_lat_min=31.000" .
var:method tmpl:2dvalue_2_0 "GET" .
var:version tmpl:2dvalue_2_0 "1.1" .
var:time tmpl:value_2 "2018-09-13T13:12:56+00:00" .
```

Listing 21: Example of binding data to be sent to the template service.

Posting the above bindings to the template service produces the PROV document in Listing 22.

```
document
  prefix xml <http://www.w3.org/XML/1998/namespace>
  prefix tmpl <http://openprovenance.org/tmpl#>
  prefix uuid <urn:uuid:>
  prefix rdfs <http://www.w3.org/2000/01/rdf-schema#>
  prefix rdf <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
  prefix vre <https://www.vre4eic.eu/log#>
  prefix var <http://openprovenance.org/var#>
  prefix vargen <http://openprovenance.org/vargen#>

  bundle uuid:0f38304a-7048-4259-8dd4-a1053e7b16df
    prefix uuid <urn:uuid:>
    prefix vre <https://www.vre4eic.eu/log#>

    agent(vre:ag1, [vre:hasIP="10.255.0.2"])
    agent(vre:ag2, [vre:hasIP="10.255.0.2"])
    agent(vre:ag3, [vre:hasIP="10.255.0.2"])
    activity(uuid:e519688a-515e-49b8-8a1d-4464ac20701f, -, -, [vre:requestMethod="GET",
vre:httpVersion="1.1", vre:requestURL="/cue/rest/argo/get"])
    activity(uuid:ee98c95b-e565-4698-b438-592a9f8df96a, -, -, [vre:requestMethod="GET",
vre:httpVersion="1.1",
vre:requestURL="/cue/rest/argo/get?geospatial_lat_min=31.000&geospatial_lat_max=38.200&geosp
atial_lon_min=147.000&geospatial_lon_max=147.100&flowlabel="])
    activity(uuid:9071a091-d810-4a27-afa3-82275dc2dbab, -, -, [vre:requestMethod="GET",
vre:httpVersion="1.1", vre:requestURL="/cue/rest/argo/get?geospatial_lat_min=31.000"])
    entity(uuid:7becc71a-a038-4604-9657-b6eb6025b618, [prov:generatedAtTime="2018-09-
```

```
13T12:51:13+00:00", vre:responseBytes="1103475", vre:status="200"])
    entity(uuid:98dfc820-0ec6-4db7-b3e4-96aee2ed56c3, [prov:generatedAtTime="2018-09-
13T12:51:13+00:00", vre:responseBytes="28172", vre:status="200"])
    entity(uuid:69f3ce5c-3923-43c8-a711-1c5e5fe9d92e, [prov:generatedAtTime="2018-09-
13T13:12:56+00:00", vre:responseBytes="1103475", vre:status="200"])
    wasAssociatedWith(uuid:e519688a-515e-49b8-8a1d-4464ac20701f, vre:ag1, -)
    wasAssociatedWith(uuid:ee98c95b-e565-4698-b438-592a9f8df96a, vre:ag2, -)
    wasAssociatedWith(uuid:9071a091-d810-4a27-afa3-82275dc2dbab, vre:ag3, -)
    wasGeneratedBy(uuid:7becc71a-a038-4604-9657-b6eb6025b618, uuid:e519688a-515e-49b8-8a1d-
4464ac20701f, -)
    wasGeneratedBy(uuid:98dfc820-0ec6-4db7-b3e4-96aee2ed56c3, uuid:ee98c95b-e565-4698-b438-
592a9f8df96a, -)
    wasGeneratedBy(uuid:69f3ce5c-3923-43c8-a711-1c5e5fe9d92e, uuid:9071a091-d810-4a27-afa3-
82275dc2dbab, -)
    wasAttributedTo(uuid:7becc71a-a038-4604-9657-b6eb6025b618, vre:ag1)
    wasAttributedTo(uuid:98dfc820-0ec6-4db7-b3e4-96aee2ed56c3, vre:ag2)
    wasAttributedTo(uuid:69f3ce5c-3923-43c8-a711-1c5e5fe9d92e, vre:ag3)
  endBundle
endDocument
```

Listing 22: Example of PROV data generated for a log file with three entries.

The ability on the part of the PROV template system is the ability to generate multiple sets of provenance entities in a single ronvert logging data in batches, reducing the intensity of HTTP requests sent to the templating service. With this, it becomes more feasible to use microservices such as the ENVRI provenance service as part of a servequest. This is particularly useful for high frequency provenance generation (for example the generation of PROV triples based on individual entries in a service log) as it allows us to cice-oriented workflow, rather than needing to integrate the provenance conversion functionality directly into the logging framework for performance reasons.

Once all constituent elements have been converted into PROV using the methodology described above, all provenance data can be ingested into a triple store (such as the ENVRI Knowledge Base) for query via a SPARQL endpoint, or the RDF documents uploaded into a suitable document store, another functionality currently offered by the ENVRI provenance service. The next step to be carried out in future is to adjust the precise formalisms used for the converted PROV to bring them into better alignment (i.e. to provide a standard integrated model that semantically links all the different sources of provenance information), and thus allow full integration, allowing querying across the constituent subsystems, for example to link infrastructural events to application-level events based on timestamp information.

To summarise, the principal advantage of the PROV template system is to permit the separation of the generation of bindings (based on a set of known parameters) extracted from log files of various different formats from the actual modelling of the integrated provenance. This affords us the freedom to adjust the provenance model later without necessitating changes to the actual provenance acquisition pipeline, which also expedites further integration efforts by giving us a robust baseline to work with. The ENVRI provenance service makes this a simple matter, and allows us to share and invoke the templates created from a variety of different contexts, again without requiring complex integration into our existing workflows.

6 CONCLUSION AND OUTLOOK

As far as the general approach using PROV-template was concerned, the majority of the representatives behind the different use case contributions stated that they considered its main intention to separate PROV structure and actual instance values as useful. This on the one hand allowed them to focus on the intended structure of the PROV-output in a more generic and formal manner, while on the other hand it suggested practical advantages due to the separation of log output and its PROV-conforming representation.

Focusing on the structure of the intended PROV output at the same time raised a number of issues that required attention when considering the application in production settings. Although suggesting a nonintrusive approach to introducing PROV into existing workflows, it quickly became clear that many existing solutions nevertheless would have to be modified/extended in one or the other way. Such cases were for example that existing output could not be reformatted into the required explicit structured form in a straightforward manner — e.g. because important information was provided as free text or underlying processes were not consistent enough yet — or that the vocabularies required to describe attributes of the modeled entities beyond what was provided by the PROV-DM itself had yet to be found. Moreover, decoupling PROV output from the underlying processes of course did not remove basic requirements for unleashing the full potential of PROV, such as using dereferenceable IDs for involved agents and entities, resolving to useful additional information when required. Another rather general issue that arose during the consideration of the structure of the PROV output concerned its granularity, suggesting that it is not trivial to find the right balance between “very detailed” and “rather superficial”. Especially in this regard, however, PROV-template could be helpful since it allows filtering very detailed “raw” logs via templates of adaptable complexity at any later point in time, adapting to changing requirements.

As far as the provided prototype service was concerned, users highlighted the potential usefulness of a curated catalog of PROV-templates for exchanging best practices amongst experts. However, they raised concerns regarding the reliability and realistically achievable up-time of any centralized expansion web service for applications in real-time or high frequency settings, where missing provenance due to unavailable or overloaded service infrastructure would be a serious issue. As outlined by others, however, some of the features of the expansion service were explicitly targeted at allowing batch-wise handling of PROV-template expansions which was in turn stated to especially enable its use in high frequency settings. In any case, in the event of a downtime of the expansion service, the separation of log output and PROV-template expansion would allow uninterrupted local logging of “raw” data which could be expanded via the central service at a later point in time. Moreover, the public availability of the underlying expansion library enables its integration into time critical offline configurations.

The proposed service is obviously only one out of possible alternative approaches to deal with the acquisition of provenance information. Within the limited resources available for this task, we still wanted to develop a practical solution for RIs that do not yet have any provenance management in place. Others, such as the EPOS-integrated infrastructures which have CERIF[51]-based data management in place, could think of optimizing their provenance related information using the temporally defined role-based relationships between instances of entities to produce

[51] the Common European Research Information Format, see https://www.eurocris.org/cerif/main-features-cerif

enriched provenance metadata [Compton et al., 2014]. However, this would require a profound mapping effort to PROV-O as well as the integration of causal-effect relationships among the entities and activities involved and re-used across processing tasks in the CERIF model [Bailo et al., 2016].

There is certainly potential to improve the functionality of the proposed ENVRI Provenance Service in many ways. The user-friendliness of the service itself could be improved, e.g. by providing a visual editor for constructing PROV-templates or dashboard like features to explore the collected provenance data. Moreover, the provenance of the template creation and expansion process could be traced as well, i.e. by tracing the version histories of updated templates as well as the components involved in each expansion. Such upgrades would, however, require significant resources for design and implementation. Best practice patterns for the appropriate provenance modelling could be provided to help in the creation of provenance templates. In this regard the Provenance Patterns Database[52] could be useful if accordingly filtered for the purpose.

Another important issue regards sustainability, both with respect to hosting the service and to its further development. This question mainly depends on how actual service uptake will develop and if it would rather remain in the environmental sciences context or also happen in a larger, cross-domain setting. In this regard, the existence of the service should be communicated to an as wide as possible audience. As far as further development is concerned, the availability of the source code via GitHub invites potentially interested parties to participate in the evolution of the service, the flexible approach behind PROV-Template also invites to seek for dedicated project-based funds for a variety of application scenarios. As far as hosting is concerned, the service is currently located by EGI and as long as future funding via one or more members of an EGI Virtual Organization[53] can be secured, it should be considered to continue maintaining it there and to register it e.g. in the EOSC-hub Service catalogue[54]. To make the service fit into the EOSC framework, it would moreover be necessary to redesign the approach to user authentication by adding compatibility with EduGain[55], similar (SAML-based) AAI[56] services or B2ACCESS[57].

Additionally it could be discussed which metadata standard would be appropriate to better describe the PROV-templates stored in the catalogue. The intention is to provide to the RI curator a selection of potential templates to be reused for the recording of the procedure used to generate the RI's output. Thus it makes sense to exploit the vocabulary already defined in the ENVRI Reference Model which addresses all relevant actors, components, data objects and actions involved in the data life cycle for the respective fields of the metadata standard chosen. In case of the proposed Dublin Core standard[58] the "type" and the "coverage" fields would benefit from reusing controlled vocabularies, whereas "creator" should be linked to person registries like ORCID[59]. As far as subject specific keywords are concerned, the "subject" field could also point to dedicated controlled vocabularies but users should also be able to enter free keywords as well.

[52] https://patterns.promsns.org
[53] https://wiki.egi.eu/wiki/Federated_Cloud_user_support
[54] https://eosc-hub.eu/catalogue
[55] https://edugain.org/
[56] Authentication and Authorization Infrastructure
[57] https://www.eudat.eu/services/b2access
[58] http://dublincore.org/specifications/
[59] https://orcid.org/signin

The correct and practical linking between data, metadata and provenance should be addressed as well in further developments of provenance-related approaches. PROV-AQ[60] has laid down the basis for this but it does not provide the answer to how this interrelates to metadata standards, where more summary provenance statements are required to be included in specific metadata fields. In this context it is also worth to elaborate which contextual provenance information is needed at which granularity level of the data.

Along with the implementation of the ENVRI Provenance Service a proper incorporation of provenance related issues in the ENVRI RM viewpoints [Nieva de la Hidalga et al. 2017] would be necessary. In the Science Viewpoint additional Community Roles (such as the provenance curator) and Behaviors have to provided. Specific activity diagrams could be modelled to describe better the user-service interactions for producing provenance records with the proposed approach. The Information Viewpoint should include the concepts of provenance and provenance templates as Information Objects and all related Information Action Types for the provenance management. In the computational viewpoint new Service Objects such as the expansion service, to mention an example, have to be introduced. Accordingly also the Engineering and the Technology Viewpoints have to be adapted. These additional elements have to be considered in future releases of the RM.

[60] https://www.w3.org/TR/prov-aq/

7 REFERENCES

[Atkinson et al., 2016] M. Atkinson, A. Hardisty, R. Filgueira, C. Alexandru, A. Vermeulen, K. Jeffery, T. Loubrieu, L. Candela, B. Magagna, P. Martin, Y. Chen and M. Hellström: A consistent characterisation of existing and planned RIs. ENVRIplus Deliverable 5.1, submitted on April 30, 2016. Available at http://www.envriplus.eu/wp-content/uploads/2016/06/A-consistent-characterisation-of-RIs.pdf.

[Bailo et al., 2017] D. Bailo, , D. Ulbricht, M. L. Nayembil, L. Trani, A. Spinuso, und K. G. Jeffery, 'Mapping solid earth Data and Research Infrastructures to CERIF'. Procedia Computer Science 106 (2017): 112–121.

[Chen et al., 2017] Y. Chen, B. Grenier, M. Hellström, A. Vermeulen, M. Stocker, R. Huber, B. Magagna, I. Häggström, M. Fiebig, P. Martin, D. Vitale, G. Judeau, T. Carval, T. Loubrieu, A. Nieva, K. Jeffery, L. Candela and J. Heikkinen: Service deployment in computing and internal e-Infrastructures. ENVRIplus Deliverable 9.1, submitted on August 31, 2017. Available at http://www.envriplus.eu/wp-content/uploads/2015/08/D9.1-Service-deployment-in-computing-and-internal-e-Infrastructures.pdf).

[Clobert et all., 2018] J. Clobert, A. Chanzy, J.-F Le Galliard, A. Chabbi, L. Greiveldinger, T. Caquet, M. Loreau, C. Mougin, C. Pichot, J. Roy, L. Saint-André, (2018). How to integrate experimental research approaches in ecological and environmental studies: AnaEE France as an example. Frontiers in Ecology and Evolution, 6 (43). , DOI : 10.3389/fevo.2018.00043

[Compton et al., 2014] M. Compton, D. Corsar, and K. Taylor, 'Sensor Data Provenance: SSNO and PROV-O Together At Last.' In TC/SSN@ ISWC, 67–82, 2014.

[ENVRIplus 2015] ENVRIplus Description of Work (DoW), public part. ENVRIplus Grant Agreement, Annex 1, part A. Horizon 2020 project no. 654182. Associated with document Ref. Ares(2015)1488547. Available at http://www.envriplus.eu/wp-content/uploads/2015/08/ENVRIplus_DoW_public.pdf.

[Le Franc et al., 2018] Y. Le Franc, T. Cortes, A. Rajapakse, A. Chernov, A. Queralt, E. Dima, X. Pivan, C. Pagé and J. Ezelin: "Report on Design Model and Definition of Data Directives". EUDAT-2020 deliverable D8.3, submitted April 19, 2018. Available at http://doi.org/10.23728/b2share.485149318e464ba9b348f9a75bb519da.

[Magagna et al., 2018] M. Magagna, D. Goldfarb, P. Martin, F. Toussaint, S. Kindermann, M. Atkinson, K. Jeffery, M. Hellström, M. Fiebig, A. Nieva de la Hidalga and A. Spinuso: Data provenance and tracing for environmental sciences: system design. ENVRIplus Deliverable D8.5, submitted on April 30, 2018. Available at http://www.envriplus.eu/wp-content/uploads/2015/08/D8.5-Data-provenance-and-tracing-for-environmental-sciences-system-design.pdf.

[Martin et al., 2018] P. Martin, Z. Zhao, B. Magagna: A definition of the ENVRIplus semantic linking framework at conceptual and formal levels. ENVRIplus Deliverable D5.3, submitted on April 30, 2018. Available at http://www.envriplus.eu/wp-content/uploads/2015/08/D5.3-A-definition-of-the-ENVRIPLUS-Semantic-linking-framework-at-conceptual-and-formal-levels.pdf.

[Moreau et al., 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, 2018.

[Nieva de la Hidalga et al. 2017] Abraham Nieva de la Hidalga, Barbara Magagna, Markus Stocker, Alex Hardisty, Paul Martin, Zhiming Zhao, Malcolm Atkinson, and Keith Jeffery. The ENVRI Reference Model (ENVRI RM) version 2.2, November 2017.

[Socha et al., 2013] Y.M. Socha, ed., “Out of cite, out of mind: The current state of practice, policy, and technology for the citation of data”. Data Science Journal vol. 12, 13 Sept 2013.https://doi.org/10.2481/dsj.OSOM13-043

[Wittenburg and Strawn 2018], P. Wittenburg and G. Strawn: "Common Patterns in Revolutionary Infrastructures and Data", February 2018. http://doi.org/10.23728/b2share.4e8ac36c0dd343da81fd9e83e72805a0.

APPENDIX

A. GLOSSARY

A1. Acronyms and abbreviations specific to this deliverable

IRI	Internationalized Resource Identifiers
PROV	W3C PROV family of documents (https://www.w3.org/TR/prov-overview/)
PROV-AQ	mechanisms for accessing and querying provenance
PROV-DM	the PROV data model for provenance
PROV-N	a notation for provenance aimed at human consumption
PROV-Template	describes the intended future structure of PROV documents
PROV-XML	an XML schema for the PROV data model
SVG	Scalable Vector Graphics
URI	Uniform Resource Identifier

A2. Other acronyms and abbrevations used in the ENVRIplus context

CCSDS	Consultative Committee for Space Data Systems
CMIS	Content Management Interoperability Services
CERIF	Common European Research Information Format
DDS	Data Distribution Service for Real-Time Systems
ENVRI	Environmental Research Infrastructure
ENVRI_RM	ENVRI Reference Model
EOSC	European Open Science Cloud
ESFRI	European Strategy Forum on Research Infrastructures
ESFRI-ENV RI	ESFRI Environmental Research Infrastructure
GIS	Geographic Information System
IEC	International Electrotechnical Commission
ISO	International Organisation for Standardization
OAIS	Open Archival Information System
OASIS	Organization for the Advancement of Structured Information Standards

OBOE	Semantic model for observational data
ODP	Open Distributed Processing
OGC	Open Geospatial Consortium
OMG	Object Management Group
ORCHESTRA	Open Architecture and Spatial Data Infrastructure for Risk Management
ORM	OGC Reference Model
OSI	Open Systems Interconnection
OWL	Web Ontology language
SOA	Service Oriented Architecture
SOA-RM	Reference Model for Service Oriented Architecture
RDF	Resource Description Framework
RM-OA	Reference Model for the ORCHESTRA Architecture
RM-ODP	Reference Model of Open Distributed Processing
UML	Unified Modelling Language
W3C	World Wide Web Consortium
UML4ODP	Unified Modelling Language For Open Distributed Processing

A3. ENVRI RM terminology

Access Control: A functionality that approves or disapproves of access requests based on specified access policies.

Acquisition Service: Oversight service for integrated data acquisition.

Active role: A active role is typically associated with a human actor.

Add Metadata: Add additional information according to a predefined schema (metadata schema). This partially overlaps with data annotations.

Annotate Data: Annotate data with meaning (concepts of predefined local or global conceptual models).

Annotate Metadata: Link metadata with meaning (concepts of predefined local or global conceptual models). This can be done by adding a pointer to concepts within a conceptual model to the data. If e.g. concepts are terms in and SKOS/RDF thesaurus, published as linked data then this would mean entering the URL of the term describing the meaning of the data.

Annotation: (verb) The action of annotating or making notes. (noun) A note added to anything written, by way of explanation or comment.

Annotation Service: Oversight service for adding and updating records attached to curated datasets.

Assign Unique Identifier: Obtain a unique identifier and associate it to the data.

Authentication: A functionality that verifies a credential of a user.

Authentication Service: Security service responsible for the authentication of external agents making requests of infrastructure services.

Authorisation: A functionality that specifies access rights to resources.

Authorisation Service: Security service responsible for the authorisation of all requests made of infrastructure services by external agents.

Backup: A copy of (persistent) data so it may be used to restore the original after a data loss event.

Behaviour : A behaviour of a community is a composition of actions performed by roles normally addressing separate business requirements.

Build Conceptual Models: Establish a local or global model of interrelated concepts.

Capacity Manager: An active role, which is a person who manage and ensure that the IT capacity meets current and future business requirements in a cost-effective manner.

Carry out Backup: Replicate data to an additional data storage so it may be used to restore the original after a data loss event. A special type of backup is a long term preservation.

Catalogue service: Oversight service for cataloguing curated datasets.

Check Quality: Actions to verify the quality of data.

Citation: from the ENVRI RM perspective, citation is defined as a pointer from a publication to:

- data source(s)
- and/or the owner(s) of the data source(s)
- a description of the evaluation process, if available
- a timestamp marking the access time to the data sources, thus reflecting a certain version

Citizen (synonyms: General Public, Media): An active role, a person, who is interested in understanding the knowledge delivered by an environmental science research infrastructure, or discovering and exploring the Knowledge_Base_Glossary enabled by the research infrastructure.

Citizen Scientist: An active role, member of the general public who engages in scientific work, often in collaboration with or under the direction of professional scientists and scientific institutions (also known as amateur scientist).

Community: A collaboration which consists of a set of roles agreeing their objective to achieve a stated business purpose.

Concept: Name and definition of the meaning of a thing (abstract or real thing). Human readable definition by sentences, machine readable definition by relations to other concepts (machine readable sentences). It can also be meant for the smallest entity of a conceptual model. It can be part of a flat list of concepts, a hierarchical list of concepts, a hierarchical thesaurus or an ontology.

Conceptual Model: A collection of concepts, their attributes and their relations. It can be unstructured or structured (e.g. glossary, thesaurus, ontology). Usually the description of a concept and/or a relation defines the concept in a human readable form. Concepts within ontologies and their relations can be seen as machine readable sentences. Those sentences can be used to establish a self-description. It is, however, practice today, to have both, the human readable description and the machine readable description. In this sense a conceptual model can also be seen as a collection of human and machine readable sentences. Conceptual models can reside within the persistence layer of a data provider or a community or outside. Conceptual models can be fused with the data (e.g. within a network of triple stores) or kept separately.

Coordination Service: An oversight service for data processing tasks deployed on infrastructure execution resources.

Data Acquisition Community. A community, which collects raw data and bring (streams of) measures into a system.

Data Acquisition Subsystem: A subsystem that collects raw data and brings the measures or data streams into a computational system.

Data Analysis: A functionality that inspects, cleans, transforms data, and provides data models with the goal of highlighting useful information, suggesting conclusions, and supporting decision making.

Data Assimilation: A functionality that combines observational data with output from a numerical model to produce an optimal estimate of the evolving state of the system.

Data Broker: Broker for facilitating data access/upload requests**.**

Data Cataloguing: A functionality that associates a data object with one or more metadata objects which contain data descriptions.

Data Citation: A functionality that assigns an accurate, consistent and standardised reference to a data object, which can be cited in scientific publications.

Data Collection: A behaviour performed by a *Data Collector* that control and monitor the collection of the digital values from a *sensor* instrument or a human sensor, such as a *Measurer* or a *Observer*, associating consistent time-stamps and necessary metadata.

Data Collector: Active or passive role, adopted by a person or an instrument collecting data.

Data Consumer: Either an active or passive role, which is an entity who receives and use the data.

Data Curation Community: A community, which curates the scientific data, maintains and archives them, and produces various data products with metadata.

Data Curation Subsystem: A subsystem that facilitates quality control and preservation of scientific data.

Data Curator: An active role, which is a person who verifies the quality of the data, preserve and maintain the data as a resource, and prepares various required data products.

Data Discovery & Access: A functionality that retrieves requested data from a data resource by using suitable search technology.

Data Exporter: Binding object for exporting curated datasets.

Data Extraction: A functionality that retrieves data out of (unstructured) data sources, including web pages, emails, documents, PDFs, scanned text, mainframe reports, and spool files.

Data Identification: A functionality that assigns (global) unique identifiers to data contents.

Data Importer: An Oversight service for the import of new data into the data curation subsystem.

Data infrastructure: a collection of data assets, organisations that operate and maintain them and guides describing how to use and manage the data. A data infrastructure is sustainably funded and has oversight that provides direction to maximise data use and value by meeting the needs of society. Data infrastructure includes technology, processes and organisation.

Data management: a process development and execution of architectures, policies, practices and procedures in order to manage the data lifecycle needs of a specific research community.

Data management plan (DMP): a formal document that outlines how data are to be handled both during a research project and after the project is completed.

Data Mining: A functionality that supports the discovery of patterns in large data sets.

Data Originator: Either an active or a passive role, which provide the digital material to be made available for public access.

Data Processing Control: A functionality that initiates the calculation and manages the outputs to be returned to the client.

Data Processing Subsystem: A subsystem that aggregates the data from various resources and provides computational capabilities and capacities for conducting data analysis and scientific experiments.

Data Product Generation: A functionality that processes data against requirement specifications and standardised formats and descriptions.

Data Provenance: Information that traces the origins of data and records all state changes of data during their lifecycle and their movements between storages.

Data Provider: Either an active or a passive role, which is an entity providing the data to be used.

Data Publication: A functionality that provides clean, well-annotated, anonymity-preserving datasets in a suitable format, and by following specified data-publication and sharing policies to make the datasets publically accessible or to those who agree to certain conditions of use, and to individuals who meet certain professional criteria.

Data Publication Community: A community that assists the data publication, discovery and access.

(Data Publication) Repository: A passive role, which is a facility for the deposition of published data.

Data Publishing Subsystem: A subsystem that enables discovery and retrieval of data housed in data resources.

Data Quality Checking: A functionality that detects and corrects (or removes) corrupt, inconsistent or inaccurate records from data sets.

Data Service Provision Community: A community that provides various services, applications and software/tools to link, and recombine data and information in order to derive knowledge.

Data State: Term used as defined in ISO/IEC 10746-2. At a given instant in time, data state is the condition of an object that determines the set of all sequences of actions (or traces) in which the object can participate.

Data Storage & Preservation: A functionality that deposits (over long-term) the data and metadata or other supplementary data and methods according to specified policies, and makes them accessible on request.

Data Store Controller: A data store within the data curation subsystem.

Data Transfer Service: Oversight service for the transfer of data into and out of the data curation subsystem.

Data Transmission: A functionality that transfers data over communication channel using specified network protocols.

Data Transporter: Generic binding object for data transfer interactions.

Data Use Community: A community who makes use of the data and service products, and transfers the knowledge into understanding.

Data Use Subsystem: A subsystem that provides functionalities to manage, control, and track users' activities and supports users to conduct their roles in the community.

Describe Service: Describe the accessibility of a service or process, which is available for reuse, the interfaces, the description of behaviour and/or implemented algorithms.

Design of Measurement Model: A behaviour that designs the measurement or monitoring model based on scientific requirements.

Do Data Mining: Execute a sequence of metadata / data request --> interpret result --> do a new request

e-Infrastructure: a combination and interworking of digitally-based technology (hardware and software), resources (data, services, digital libraries), communications (protocols, access rights and networks), and the people and organisational structures needed to support modern, internationally leading collaborative research be it in the arts and humanities or the sciences.

Educator (synonym: Trainer): An active role, which is a person who makes use of the data and application services for education and training purposes.

Engineer (synonym: Technologist): An active role, which is a person who develops and maintains the research infrastructure.

Environmental Scientist: An active role, which is a person who conduct research or perform investigation for the purpose of identifying, abating, or eliminating sources of pollutants or hazards that affect either the environment or the health of the population. Using knowledge of various scientific disciplines, may collect, synthesize, study, report, and recommend action based on data derived from measurements or observations of air, food, soil, water, and other sources.

ENVRI Reference Model: A common ontological framework and standards for the description and characterisation of computational and storage systems of ESFRI environmental research infrastructures.

Experiment Laboratory: Community proxy for conducting experiments within a research infrastructure.

Field Laboratory: Community proxy for interacting with data acquisition instruments.

Final review: Review the data to be published, which will not likely be changed again.

Free text annotation: to add a short explanation or opinion to a text or drawing (equivalent to the dictionary definition of annotation).

Instrument Controller: An integrated raw data source.

Knowledge Base: (1) A store of information or data that is available to draw on. (2) The underlying set of facts, assumptions, and rules which a computer system has available to solve a problem.

Knowledge infrastructure: robust networks of people, artifacts, and institutions that generate, share, and maintain specific knowledge about the human and natural worlds.

Mapping Rule: Configuration directives used for model-to-model transformation.

(Measurement Model) Designer: An active role, which is a person who design the measurements and monitoring models based on the requirements of environmental scientists.

Measurement result: Quantitative determinations of magnitude, dimension and uncertainty to the outputs of observation instruments, sensors (including human observers) and sensor networks.

Measurer: An active role, which is a person who determines the ratio of a physical quantity, such as a length, time, temperature etc., to a unit of measurement, such as the meter, second or degree Celsius.

Metadata: Data about data, in scientific applications is used to describe, explain, locate, or make it easier to retrieve, use, or manage an information resource.

Metadata Catalogue: A collection of metadata, usually established to make the metadata available to a community. A metadata catalogue has an access service.

Metadata Harvesting (Publishing Community Role): A behaviour performed by a metadata harvester to gather metadata from data objects in order to construct catalogues of the available information. A functionality that (regularly) collects metadata (in agreed formats) from different sources.

Metadata Harvester (Publishing Community Role): A passive role performed by a system or service collecting metadata to support the construction/selection of a global conceptual model and the production of mapping rules.

Metadata State:

- raw: are established metadata, which are not yet registered. In general, they are not shareable in this status
- registered: are metadata which are inserted into a metadata catalogue.

- published: are metadata made available to the public, the outside world. Within some metadata catalogues registered.

Passive Role: A passive role is typically associated with a non-human actor.

Perform Mapping: Execute transformation rules for values (mapping from one unit to another unit) or translation rules for concepts (translating the meaning from one conceptual model to another conceptual model, e.g. translating code lists).

Persistent Data: Term (data) used as defined in ISO/IEC 10746-2. Data is the representations of information dealt by information systems and users thereof. Data which are persistent (stored).

Perform Measurement or Observation: Measure parameter(s) or observe an event. The performance of a measurement or observation produces measurement results.

PID Generator: A passive role, a system which assigns persist global unique identifiers to a (set of) digital object.

PID Registry: A passive role, which is an information system for registering PIDs.

PID Service: External service for persistent identifier assignment and resolution.

Policy Maker (synonym: Decision Maker): An active role, a person, who makes decisions based on the data evidences.

Process Control: A functionality that receives input status, applies a set of logic statements or control algorithms, and generates a set of analogue / digital outputs to change the logic states of devices.

Process Controller: Part of the execution platform provided by the data processing subsystem.

Process Data: Process data for the purposes of:

- converting and generating data products
- calculations: e.g., statistical processes, simulation models
- visualisation: e.g., alpha-numerically, graphically, geographically

Data processes should be recorded as provenance.

Provenance: The pathway of data generation from raw data to the actual state of data.

Publish Data: Make data public accessible.

Publish Metadata: Make the registered metadata available to the public.

QA Notation: Notation of the result of a Quality Assessment. This notation can be a nominal value out of a classification system up to a comprehensive (machine readable) description of the whole QA process.

Quality Assessment (QA): Assessment of details of the data generation, including the check of the plausibility of the data. Usually the quality assessment is done by predefined checks on data and their generation process.

Query Data: Send a request to a data store to retrieve required data.

Query Metadata: Send a request to metadata resources to retrieve metadata of interests.

Observer: An active role, which is a person who receives knowledge of the outside world through the senses, or records data using scientific instruments.

Raw Data Collector: Binding object for raw data collection.

Reference Mode: A reference mode is an abstract framework for understanding significant relationships among the entities of some environment.

Register Metadata: Enter the metadata into a metadata catalogue.

Research Infrastructure: means facilities, resources and related services that are used by the scientific community to conduct top-level research in their respective fields and covers major scientific equipment or sets of instruments; knowledge-based resources such as collections, archives or structures for scientific information; enabling Information and Communications Technology-based infrastructures such as Grid, computing, software and communication, or any other entity of a unique nature essential to achieve excellence in research. Such infrastructures may be "single-sited" or "distributed" (an organised network of resources).

Resource Registration: A functionality that creates an entry in a resource registry and inserts resource object or a reference to a resource object in specified representations and semantics.

Role : A role in a community is a prescribing behaviour that can be performed any number of times concurrently or successively.

Science Gateway: Community portal for interacting with an infrastructure.

Scientific Modelling and Simulation: A functionality that supports the generation of abstract, conceptual, graphical or mathematical models, and to run an instance of the model.

Scientist (synonym: Researcher): An active role, which is a person who makes use of the data and application services to conduct scientific research.

(Scientific) Workflow Enactment: A specialisation of Workflow Enactment, which support of composition and execution a series of computational or data manipulation steps, or a workflow, in a scientific application. Important processes should be recorded for provenance purposes.

Security Service: Oversight service for authentication and authorisation of user requests to the infrastructure.

Semantic Annotation: link from an information object (single datum, data set, data container) to a concept within a conceptual model, enabling the discovery of the meaning of the information object by human and machines.

Semantic Broker: Broker for establishing semantic links between concepts and bridging queries between semantic domains.

SV Community Behaviour: A behaviour enabled by a *Semantic Mediator* that unifies similar data (knowledge) models based on the consensus of collaborative domain experts to achieve better data (knowledge) reuse and semantic interoperability.

Semantic Laboratory: Community proxy for interacting with semantic models.

Semantic Mediator: A passive role, which is a system or middleware facilitating semantic mapping discovery and integration of heterogeneous data.

Sensor: A passive role, which is a converter that measures a physical quantity and converts it into a signal which can be read by an observer or by an (electronic) instrument.

Sensor Network: A passive role, which is a network consists of distributed autonomous sensors to monitor physical or environmental conditions.

Service: Service or process, available for reuse.

Service Consumer: Either an active or a passive role, which is an entity using the services provided.

Service Description: Services and processes, which are available for reuse, be it within an enterprise architecture, within a research infrastructure or within an open network like the Internet, shall be described to help avoid wrong usage. Usually such descriptions include the accessibility of the service, the description of the interfaces, the description of behavior and/or implemented algorithms. Such descriptions are usually done along service description standards (e.g. WSDL, web service description language). Within some service description languages, semantic descriptions of the services and/or interfaces are possible (e.g. SAWSDL, Semantic Annotations for WSDL)

Service Provider: Either an active or a passive role, which is an entity providing the services to be used.

Service Registry: A passive role, which is an information system for registering services.

Setup Mapping Rules: Specify the mapping rules of data and/or concepts.

Specification of Investigation Design: This is the background information needed to understand the overall goal of the measurement or observation. It could be the sampling design of observation stations, the network design, the description of the setup parameters (interval of measurements) and so on... It usually contains important information for the allowed evaluations of data. (E.g. the question whether a sampling design was done randomly or by strategy determines which statistical methods that can be applied or not).

Specification of Measurements or Observations: The description of the scientific measurement model which specifies:

- what is measured;
- how it is measured;
- by whom it is measured; and
- what the temporal design is (single /multiple measurements / interval of measurement etc.)

Specify Investigation Design: specify design of investigation, including sampling design:

- geographical position of measurement or observation (site) -- the selections of observations and measurement sites, e.g., can be statistical or stratified by domain knowledge;
- characteristics of site;
- · preconditions of measurements.

Specify Measurement or Observation: Specify the details of the method of observations/measurements.

Stakeholder (synonyms: Private Investor, Private Consultant): An active role, a person, who makes use of the data and application service for predicting market so as to make business decision on producing related commercial products.

Storage: A passive role, which is memory, components, devices and media that retain digital computer data used for computing for some interval of time.

Storage Administrator: An active role, which is a person who has the responsibilities to the design of data storage, tune queries, perform backup and recovery operations, raid mirrored arrays, making sure drive space is available for the network.

Store Data: Archive or preserve data in persistent manner to ensure continuing accessible and usable.

Subsystem: a set of capabilities that collectively are defined by a set of interfaces with corresponding operations that can be invoked by other subsystems. Subsystems can be executed independently, and developed and managed incrementally.

Technician: An active role, which is a person who develop and deploy the sensor instruments, establishing and testing the sensor network, operating, maintaining, monitoring and repairing the observatory hardware.

Track Provenance: Add information about the actions and the data state changes as data provenances.

Unique Identifier (UID): With reference to a given (possibly implicit) set of objects, a unique identifier (UID) is any identifier which is guaranteed to be unique among all identifiers used for those objects and for a specific purpose.

User Behaviour Tracking: A behaviour enabled by a Data Use Subsystem that to track the Users. User Behaviour Tracking is the analysis of visitor behaviour on a website. The analysis of an individual visitor's behaviour may be used to provide options or content that relates to their implied preferences; either during a visit or in the future visits. Additionally, it can be user to track content use and performance.

User Group Work Supporting: A behaviour enabled by a Data Use Subsystem that to support controlled sharing, collaborative work and publication of results, with persistent and externally citable PIDs.

User Profile Management: A behaviour enabled by a Data Use Subsystem that to support persistent and mobile profiles, where profiles will include preferred interaction settings, preferred computational resource settings, and so on.

User Working Space Management: A behaviour enabled by a Data Use Subsystem that to support work spaces that allow data, document and code continuity between connection sessions and accessible from multiple sites or mobile smart devices.

User Working Relationships Management: A behaviour enabled by a Data Use Subsystem that to support a record of working relationships, (virtual) group memberships and friends.

Virtual Laboratory: Community proxy for interacting with infrastructure subsystems.

Virtual Research Environment (VRE, synonyms:Science Gateway, Collaboratory, Digital Library, Inhabited Information Space, Virtual Laboratory): a web-based working environment tailored to

serve the needs of a research community. A VRE is expected to provide an array of commodities needed to accomplish the research community's goal(s); it is open and flexible with respect to the overall service offering and lifetime; and it promotes fine-grained controlled sharing of both intermediate and final research results by guaranteeing ownership, provenance and attribution.