21st International Congress on Modelling and Simulation, Gold Coast, Australia, 29 Nov to 4 Dec 2015
www.mssanz.org.au/modsim2015

Standard Provenance Reporting and Scientific Software Management in Virtual Laboratories

C. Wise, N. J. Car, R. Fraser and G. Squire

CSIRO, GPO Box 664, Canberra ACT Australia 2601

Email: nicholas.car@csiro.au

Abstract: The Virtual Hazards Impact & Risk Laboratory (VHIRL) is a scientific workflow portal that provides researchers with access to a cloud computing environment for natural hazards eResearch tools. It allows researchers to construct experiments with data from a variety of sources and execute cloud computing processes for rapid and remote simulation and analysis. The service currently includes tools for the simulation of three major hazards affecting the Asia-Pacific region: earthquakes, tsunamis and tropical cyclones.

For scientific results, the establishment of provenance is key to reproducibility and trust. Thus the need for any virtual laboratory to provide provenance information for the tasks it manages is obvious, but the appropriate way to report and manage provenance information is not always so straightforward.

Many virtual laboratories and workflow systems provide bespoke provenance management with a focus on internal system use. This has clear benefits for reproducibility within the system, but it limits the interoperability of systems. For VHIRL, a provenance solution was required that was as interoperable with other, external, provenance systems as possible.

A related common issue facing workflow tools and virtual laboratories is the need to manage software code. With this comes well-known issues associated with code sharing: licensing, source code management, version management and dependency resolution. There are a wide selection of commonly used tools to help solve these problems, for example Git and Subversion.

A key goal of VHIRL was to externalise as much information management as was reasonable. VHIRL is a virtual laboratory: it is not designed to be a data store, software repository, or records management system. A solution was required that could hand off the management of provenance records and code to external services, with links between them, other data services and VHIRL jobs where appropriate.

Scientific software can be quite complicated and systems for managing dependencies and source vary from system to system. In order to provide the least friction for authors of software, we designed a system called the Scientific Software Solution Centre (SSSC) to manage solutions to scientific problems and deliver the solution templates, code and dependencies that enable them for use in VHIRL and other Virtual Laboratories and applications.

We discuss our integration of the PROMS provenance toolkit into VHIRL in order to generate standardised provenance reports according to the W3C's PROV ontology. We describe the collection of provenance information through the VHIRL service and the submission of that information to an external provenance system. We also present the SSSC design as an example of how to meet the disparate requirements of managing scientific codes and solutions while dealing with publishing, discovery, licensing and versioning. We show how the SSSC assists in provenance report generation.

Keywords: *Provenance, software deployment, software management, virtual laboratories*

1 INTRODUCTION

1.1 VHIRL

The Virtual Hazards, Impact and Risk Laboratory was begun in 2013[1] as an extension of the ongoing CSIRO work in the Geophysical Virtual Laboratory explained in **?**. This family of projects, including VHIRL, is designed to exploit generic cloud infrastructure to provide services for analysis of primarily, but not exclusively, geospatial data sets. VHIRL is designed for the needs of those modelling hazards, their impact and risks. Examples include the behaviour and impact of tropical cyclones and the inundation and damage caused by a tsunami. These models are often run with incremental parameter changes and with consistent input data sets, however they may take in excess of eight hours to run. VHIRL provides an easy way to set up a number of experiments, run them, and monitor their progress, without using local resources so users are able to continue with other work. Since VHIRL provides a space for scientists to run experiments, it is crucial that it provides all the tools user scientists need to ensure their experiments are appropriately recorded and reproducible.

1.2 Provenance

There are different approaches to recording provenance information. These depend on the kinds of data and processes being recorded, as in **?**, and the way in which the provenance record needs to be used, see for example **?**. Many workflow tools and virtual laboratories choose to tightly couple the provenance information to the system, in order to use it to easily replicate experiments internally as **?**. In many cases the input data and results are maintained in external data stores. If the workflow tool or virtual laboratory is not the only service of importance to the understanding or future use of the data, then, clearly, provenance information should be stored in a standardised manner. Some virtual laboratories have already realised the importance of storing provenance information in a standardised way. Both WebLab[2] and Callimachus [3], conform to the World Wide Web Consortium's Recommendation, PROV-O, the PROV Ontology[4]. This provides a formalisation of a data model that can be used for understanding provenance information beyond the virtual laboratory.

1.3 Scientific Software

There have been many attempts to create catalogs of software for use in other systems, such as the Information Visualisation Software Repository **?** and SeisCode[5]. Most of these are designed to provide libraries of code for development or running locally.

In VHIRL there are multi-part collections of software to be run on the cloud, which may have complicated dependencies. As much as possible, VHIRL tries to not manage this 'dependency hell', but instead enables scientists to provide working examples, as virtual machine images. These can then be overlaid with management scripts, as required, to be used in a VHIRL job.

It is also important that any system provides an easy-to-use interface for the scientists, in order to reduce friction in system adoption. Also, where possible, we should link to systems they are already using, such as source code management and public websites. For these reasons, a new web-based system was built called the Scientific Software Solution Centre (SSSC) that is both easily accessible allows code and resources from existing systems to be used in VHIRL.

[1] For a presentation on VHIRL, as distinct from other Virtual Labs, see https://publications.csiro.au/rpr/pub?pid=csiro:EP151745
[2] http://dl.acm.org/citation.cfm?id=2457367
[3] See the designer's blog post Provenance and Traceability in RDF with Callimachus
[4] http://www.w3.org/TR/prov-o/
[5] http://adsabs.harvard.edu/abs/2012AGUFM.S51C2427T

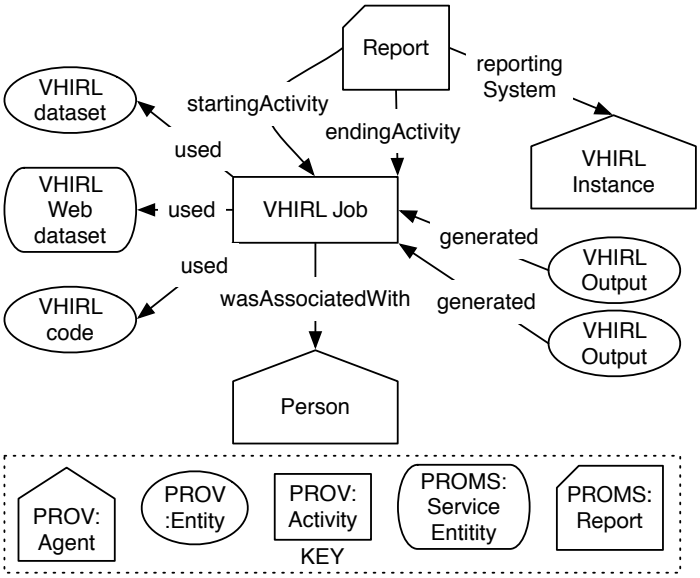

Figure 1: Mapping VHIRL job elements to PROV-O and PROMS-O. Properties are from PROV-O

2 PROMS

The PROMS (PROvenance Managment System) Server exists to validate, store, manage and query (via RESTful API and a SPARQL endpoint) provenance data. Users do not need to be familiar with the underlying technologies for use (**?**). There are PROMS *toolkits* in various programming languages for reporting provenance information. These do not require in-depth knowledge of RDF, PROV-O, or Semantic Web technologies for use; instead they present Object-Oriented interfaces which are easily understood by most programmers. They are used to collect the required metadata and other information required to generate and transmit a valid report.

The PROMS ontology is a very lightweight ontology that extends the PROV-O with some extra classes, such as `prov:Entities` which are the result of Web Service calls. The toolkits produce PROMS ontology-compliant Reports verified using validation code known as *RuleSets*, see (**?**).

2.1 Mappings

For VHIRL to create PROV-O compatible PROMS records, a mapping was made between concepts in VHIRL and PROV-O & PROMS-O. This involved a careful investigation of what information VHIRL already had, what was reasonable to collect, and what we could not reliably know.

Figure 1 illustrates a high-level view of the mapping, using the symbols found in the PROV-O standard. Note that the `proms:ServiceEntity` is a subclass of `prov:Entity` and `proms:Report` is a subclass of `prov:Activity`.

Agent. There are two Agents involved in any VHIRL job: the first is the VHIRL instance, identified by its URL. Multiple instances of VHIRL may be active, with versions customised for specific clients or provided for testing so they need to be distinguishable.

The second Agent is the VHIRL user who created the job. All users authenticate with VHIRL via Google's OpenID Connect protocol[6] and as part of that process, VHIRL receives the user's Google+ Profile URL. This URL is used to identify the user. It should exist for the lifetime of the user's Google account.

[6]See the Google Identity Platform: https://developers. google.com/identity/protocols/OpenIDConnect

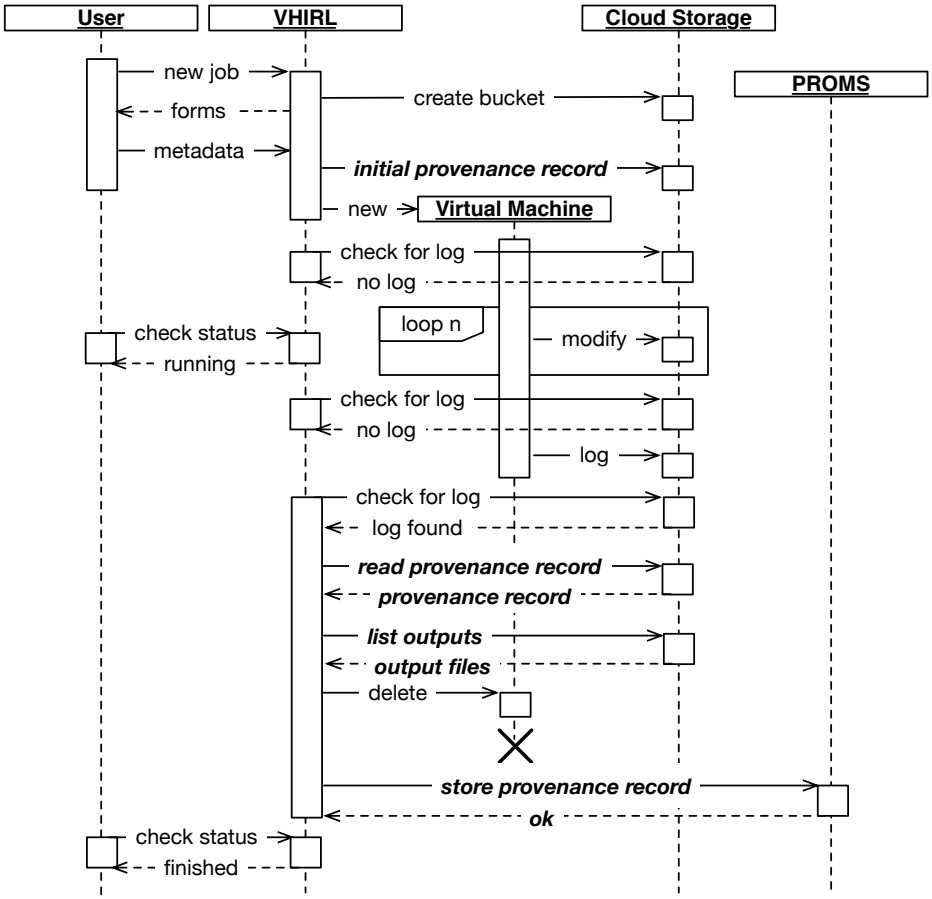

Figure 2: VHIRL Process Diagram, highlighting how provenance is recorded.

The Google+ Profile URLs are not linked-data compliant, however they have a few important features. Firstly, they can be used with the Google+ Platform to get machine-readable information about the user. Secondly, the Google+ Profile url is guaranteed to exist for all VHIRL users. Today, it is unlikely many users have linked-data compliant URLs they could supply instead—they may have profile URLs with various social media services or with their place of work—but these are no more likely to be helpful than the Google+ Profile URL, which we can get automatically, and with minimum friction.

Entity. The primary source of Entities in VHIRL are input and output files. VHIRL makes no distinctions between kinds of output files: any file that appears in the cloud storage after a job is started is considered an 'output', though they may be intermediate results. This is because VHIRL distances itself from knowing the details of the process in order to remain a more generic service.

There are three primary types of input files to a VHIRL job. The first is a file uploaded by the user. The second is a result of a web call, such as to a geographic information system or a data repository. The third is the scientific code that runs as part of the job, the *Solution* from the SSSC.

For the input Entities we often have very little reliable metadata. For a directly uploaded file, we may only have the file name. We can ask the user for additional information, such as the owner's name or creation date, but the user may not know. Solutions which come from the SSSC will have more metadata associated with them, such as an author.

Activity. There is only one `prov:Activity` in a VHIRL job: that which represents the VHIRL job itself. A VHIRL job can supply start and end times, a name, a description, for the Activity and

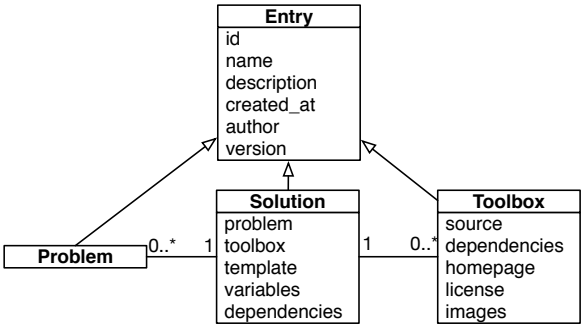

Figure 3: Class diagram for the Scientific Software Solution Centre.

the Google+ URL for the user who started it. Since the job is a core part of VHIRL, this information is much easier to gather and much more reliable than for any other component.

PROMS extension: ServiceEntity. `proms:ServiceEntity` is a sub-class of `prov:Entity` designed to represent a data source accessed via a web service call as opposed to downloading a static data source. It is important to record when the source was accessed, as well as what parameters were used in the call. In the context of VHIRL, Service Entities are used to represent mapping data sources. Using the VHIRL interface, the user selects a data subset required and this subset is extracted by passing a query to the web service.

2.2 Provenance Service

To create the provenance record, the VHIRL job must be inspected at several points, the process of which is shown in Figure 2. It shows that the collection of the provenance record is almost entirely automatic, a very desirable situation. The initial provenance record contains some metadata about the job, such as its name and creator, and the inputs files given to VHIRL. This is stored in the Cloud Storage. Once the VHIRL job is complete, the initial provenance record is read, it can then be completed by inspecting the Cloud Storage for files created during the job. The last stage is to report the record to PROMS Server. Here, VHIRL uses the Java PROMS Toolkit to wrap the information in a `proms:Report` which it then posts to the registered PROMS.

3 Scientific Software Solution Centre

The SSSC contains a selection of *Problems*, describing questions to be answered. Each is associated with none, one or more *Solutions*. A Solution defines a workflow resolving a Problem and consists of a *Toolbox*, which is code wrapped in a Python script and a description of the required inputs. A Toolbox includes a reference to a source code repository plus metadata and a list of virtual machine system images, provided by the toolbox builder, known to contain the required environment. Toolboxes can also have extra dependencies, links to websites for more information and licences. These relationships are illustrated in Figure 3.

Problems, Solutions and Toolboxes are created by SSSC scientist users but managed by the SSSC itself and the consistent formatting means the parts can be accessed by a variety of automated systems. For this there are two interfaces: the first, for users, is via a web browser where they can view, edit and create all entry types. The second is a RESTful (**?**) JSON (**?**) API, which allows external services, such as VHIRL, to view and query the content. This is described in Figure 4.

When a VHIRL user creates a job they must nominate a Solution from those available in the SSSC. VHIRL is then able to request the appropriate image from the SSSC and instantiate it in the cloud. It then applies the other Toolbox dependencies to create a virtual machine which can execute the job.

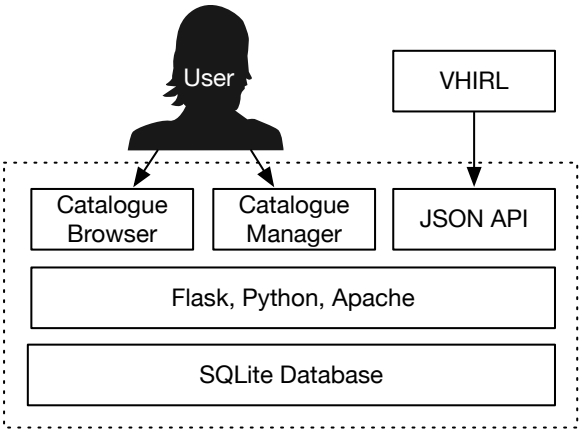

Figure 4: Architecture of the SSSC, illustrating how it is used by VHIRL and directly by users.

4 Discussion & Future Work

This process of a workflow exporting reports compliant with PROV-O is likely to be a common task now that the PROV standard exists. Inevitably, the effort required to export to PROV will vary depending on how the system is configured and how granular you desire the PROV records to be. For comparison, our estimates of the time and effort taken for this work are 2 months full-time for a novice to both the workflow system and PROV, given documentation for the workflow system, and a suitable PROMS toolkit. This would be less if the person was familiar with the workflow system, since understanding the system is the most time-consuming part of the exercise. However, if the workflow system was highly complex with many steps to document and many operations—and all this was to be captured—it could take a long time to find the appropriate points to collect the information required and to test that it was done correctly for various different workflows.

In order to use a PROMS toolkit, one only requires a brief understanding of PROV-O, as the toolkit functions somewhat like a form, specifying required and optional parameters and their types. Producing a PROMS toolkit that generates valid RDF is outside the scope of the work done here, and requires a much deeper understanding of PROV-O.

VHIRL contains metadata not currently captured in the provenance recording. Some is important only to VHIRL, such as the name of the cloud storage bucket, but some may be important for reproducibility, such as the cloud compute provider or operating system used. An extended VHIRL-PROMS mapping could include this information. A very useful extension to this work would be to enable VHIRL to read a provenance record from PROMS Server and generate another job from it. This is crucial to close the provenance loop and to introduce job reproducibility, independent of a VHIRL deployment. It would also be valuable to allow a VHIRL instance to check if any input data sets had provenance records and include that link in the provenance report to PROMS.

When selecting input datasets to a VHIRL job, it is currently possible to upload a file directly. During development this has been an important pragmatic choice as datasets are often not as well managed or published as one would ideally like. It would be better to direct all users wanting to upload a file to use a general-purpose data repository, or one specific to the disaster management space, and then retrieve a link. This relieves VHIRL of data management responsibilities and makes the provenance reports more stable as Entity persistence is managed by a purpose-built repository.

There is an opportunity to extend the capabilities of the SSSC to be used by more virtual labs and perhaps also as a more general-purpose repository for software which users can run on the cloud. This might include extending the SSSC's ability to manage dependencies and its dealings with alternative types of cloud infrastructure beyond virtual machines.

5 CONCLUSION

In this paper we have described the processes and systems we used to generate standardised (according to PROV-O) provenance reports from the Virtual Hazards Impact & Risk Laboratory (VHIRL). We detailed the mapping we made between VHIRL data and metadata and elements of the PROV ontology and a derivative ontology of it, PROMS. We explained our use of the PROMS toolkits and PROMS Server systems to generate and store provenance information. We also described the Scientific Software Solutions Centre (SSSC), a key system used by VHIRL to manage software and virtual machines used to run it. Finally, we discussed possible future work regarding both generation and use of provenance information and possible extensions to the SSSC to make it more general purpose.