Git2PROV: Exposing Version Control System Content as W3C PROV

Tom De Nies[1], Sara Magliacane[2], Ruben Verborgh[1], Sam Coppens[1], Paul Groth[2], Erik Mannens[1], and Rik Van de Walle[1]

[1] Ghent University - iMinds - Multimedia Lab
`{tom.denies,ruben.verborgh,sam.coppens,`
`erik.mannens, rik.vandewalle}@ugent.be`
[2] VU University Amsterdam
`{s.magliacane,p.t.groth}@vu.nl`

Abstract. Data provenance is defined as information about entities, activities and people producing or modifying a piece of data. On the Web, the interchange of standardized provenance of (linked) data is an essential step towards establishing trust [2]. One mechanism to track (part of) the provenance of data, is through the use of version control systems (VCS), such as Git. These systems are widely used to facilitate collaboration primarily for both code and data. Here, we describe a system to expose the provenance stored in VCS in a new standard Web-native format: W3C PROV [4]. This enables the easy publication of VCS provenance on the Web and subsequent integration with other systems that make use of PROV. The system is exposed as a RESTful Web service, which allows integration into user-friendly tools, such as browser plugins.

1 Introduction

Version control systems (VCS) have a long history in computing. The first such system was the Source Code Control System, developed in 1972 [10]. Nowadays, VCS are widely popular and becoming more so with the advent of cloud-based services, such as Github[3] and Bitbucket[4], that both simplify the management of the VCS and expose their information through Web interfaces. For example, Github has over 3 million users and maintains over 6 million repositories.[5]

Versioning of data is an aspect of *provenance*: information about entities, activities and people producing or modifying a piece of data. Provenance is critical in contexts ranging from scientific reproducibility to journalism. Furthermore, a number of sub-disciplines of computer science, including databases, distributed systems and the Web have been addressing issues related to provenance [8]. Provenance is particularly important to the Semantic Web community because

[3] `http://www.github.com`
[4] `http://www.bitbucket.com`
[5] See http://thenextweb.com/insider/2013/04/11/code-sharing-site-github-turns-five-and-hits-3-5-million-users-6-million-repositories/

of the need to ascertain the trust of data originating from multiple interlinked sources [5]. Because of the importance of provenance to the Web, the W3C produced the PROV recommendations for the interchange of provenance on the Web [4]. PROV was recently released and already has over 60 implementations and is in use by several Linked Data datasets.

Thus, the aim of the system described in this paper is to enable the provenance within VCS to be exposed in a *Web-native and interoperable format*. This provenance can then be consumed by other PROV enabled tools. Indeed, given that the software used to create many Linked Data sets is available through public version control systems[6], we believe that the tool can be used to enrich the provenance of many of these datasets.

The rest of this paper is organized as follows. We briefly discuss related work and present a mapping from the Git version control system to PROV. This is followed by a description of the implementation and demonstration of our system. In particular, our demo illustrates how the resulting information can be consumed by other PROV enabled systems. The demonstration (live and video) is available at the following URI: `http://git2prov.org`.

2 Related work

This paper is part of a greater effort to create more interoperability among different platforms that track provenance information, by mapping them to a standard interlingua such as [4]. Two relevant examples are the Dublin Core Mapping to PROV [1], and the mapping of the revision history of Wikipedia to OPM (an ancestor of PROV) [9].

Currently, the most commonly used VCS is Git, an example of a distributed version control system. Due to brevity we will omit a detailed explanation of Git, and refer to [6] for an overview.

3 A Mapping from VCS to PROV

Our mapping, shown in Fig. 1, was created by identifying whether the data could represent information from one or more broad classes of provenance information. The three classes we used are identified below. For each class, we describe how provenance can be expressed using concepts from the PROV Data Model [7].

- Dependency - a dependency between two objects expressed as the relationship between two prov:Entity objects using prov:wasDerivedFrom and prov:specializationOf , e.g. a file f_c was derived from another previous file f_{c-1}, both are a specialization of a certain file f;
- Activities - a process expressed as a prov:Activity that connects two prov:Entity objects, expressed through prov:used and prov:wasGeneratedBy relations, e.g. a commit c uses a file f_{c-1} and generates a file f_c;

[6] e.g. `https://github.com/dbpedia` or `https://github.com/jimmccusker/twc-healthdata`

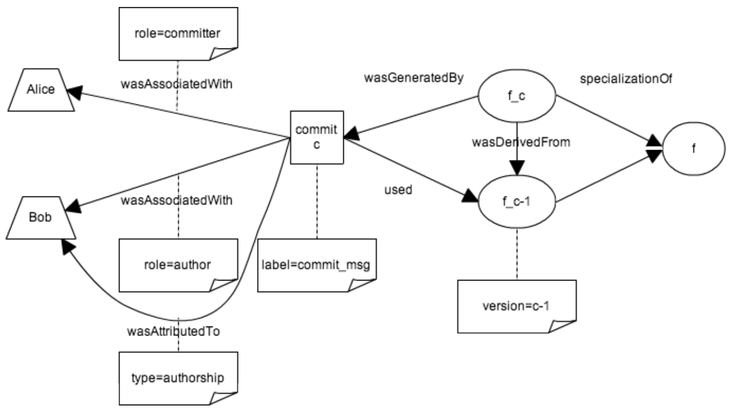

Fig. 1. Mapping of Git operations to PROV concepts. Note that the Activity *Start* and *End* concepts of PROV are not depicted, and correspond to, respectively, the author time and the commit time of each commit.

– Attribution - attribution information expressed as the prov:Agent that created the Entity using the prov:wasAttributedTo and prov:wasAssociatedWith relations, modeling the two potentially distinct roles of a author and a committer.

These classes reflect the three use-case perspectives on provenance identified by the W3C Provenance Primer [3]: object-oriented, process-oriented and agent-oriented.

4 Implementation

Because we want our conversion tool to be as flexible as possible, we chose to build a RESTful git2prov Web Service for this purpose, using the Node.js[7] framework. The service is available at the following URI: `http://git2prov.org/git2prov?gituri=<your_git_uri>`.

The only required input for this service is a URL **giturl** that refers to a git repository. In this first proof-of-concept implementation, only openly accessible repositories are supported. However, adding support for secure repositories is part of our future work. In addition to **giturl**, the service accepts a number of optional parameters, with the default value in bold:

serialization *(possible values: [**PROV-N**, PROV-O, PROV-JSON, PROV-XML])* This parameter is used to specify the desired PROV serialization.

shortHashes *(possible values: [**false**, true])* This parameter forces the service to use the short commit hash in the exported provenance, to increase readability for human users

[7] http://nodejs.org

ignore *(possible values: a provenance relation)* This parameter is used to filter the specified relation from the converted provenance.

Note that each provenance document generated by the git2prov service includes a link to the complete document (without any restricting parameters).

Upon receiving a request, the service *clones* the git repository to a temporary location, and performs a *git log* command on it. The output of this log is then *mapped* to PROV as described in Sect. 3, and written to the HTTP response in the requested serialization. To demonstrate our service, we wrote a small web application, available at `http://git2prov.org`. A video is also available at `http://vimeo.com/70980809`.

5 Conclusions and Future Work

We believe that systems such as Git2PROV have the potential to become an important enabler of the widespread interchange of standardized provenance. With our proof-of-concept implementation, we have shown that it is certainly feasible to build a lightweight RESTful Web service to convert versioning systems into PROV. In future work, we aim to improve our work by including more semantic annotations in combination with the provenance to allow further reasoning over it, with the prospect of deriving trust assessments.

Acknowledgements These research activities were funded by Ghent University, iMinds, the IWT Flanders, the FWO-Flanders, and the European Union, as well as the Data2Semantics project in the Dutch national program COMMIT.

References

[1] Daniel Garijo, K.E.: Dublin core to prov mapping - w3c working group note
[2] De Nies, T., Coppens, S., Verborgh, R., Vander Sande, M., Mannens, E., Van de Walle, R., Michaelides, D., Moreau, L.: Easy Access to Provenance: an Essential Step Towards Trust on the Web. In: METHOD 2013, Kyoto, Japan (2013)
[3] Gil, Y., Miles, S., et al.: PROV Model Primer. W3C Working Group Note (2013)
[4] Groth, P., Moreau, L.: PROV-Overview: An Overview of the PROV Family of Documents. W3C Working Group Note (2013)
[5] Groth, P., Gil, Y.: Editorial - using provenance in the semantic web. Web Semantics: Science, Services and Agents on the World Wide Web 9(2) (2011), `http://www.websemanticsjournal.org/index.php/ps/article/view/196`
[6] Loeliger, J.: Version Control with Git: Powerful Tools and Techniques for Collaborative Software Development. O'Reilly Media, Inc. (2009)
[7] Moreau, L., Missier, P., (Eds.) et al: PROV-DM: The PROV Data Model. W3C Recommendation (2013)
[8] Moreau, L.: The foundations for provenance on the web. Foundations and Trends in Web Science 2(2–3), 99–241 (2010)
[9] Orlandi, F., Passant, A.: Modelling provenance of DBpedia resources using wikipedia contributions. Web Semantics pp. 149 – 164 (2011)
[10] Rochkind, M.J.: The source code control system. Software Engineering, IEEE Transactions on (4), 364–370 (1975)