

# Detailed Provenance Capture of Data Processing

Ben De Meester, Anastasia Dimou, Ruben Verborgh, and Erik Mannens

Ghent University – imec – IDLab,  
Department of Electronics and Information Systems, Ghent, Belgium  
`{firstname.lastname}@ugent.be`

**Abstract.** A large part of scientific output entails computational experiments, e.g., processing data to generate new data. However, this generation process is only documented in human-readable form or as a software repository. This inhibits reproducibility and comparability, as current documentation solutions do not provide detailed metadata and rely on the availability of specific software environments. This paper proposes an automatic capturing mechanism for interchangeable and implementation independent metadata and provenance that includes data processing. Using declarative mapping documents to describe the computational experiment, term-level provenance can be automatically captured, for both schema and data transformations, and storing both the used software tools as the input-output pairs of the data processing executions. This approach is applied to mapping documents described using RML and F<sub>NO</sub>, and implemented in the RMLMapper. The captured metadata can be used to more easily share, reproduce, and compare the dataset generation process, across software environments.

**Keywords:** Computational Experiment, Data Processing, F<sub>NO</sub>, Provenance, RML

## 1 Introduction

Reproducibility is an important factor to judge scientific claims [15]. A large part of scientific output entails computational experiments, e.g., data being processed to generate new data. Thus, the reproducibility of these computational experiments is of great importance. The ten newly introduced datasets of ISWC 2016's Resource Track<sup>1</sup> were generated using some sort of data processing, e.g., parsing raw data, interlinking existing datasets, or performing Natural Language Processing (NLP). One of the most widely-known examples is DBpedia, where Wikitext is processed to generate the DBpedia dataset [12].

However, the description of how these – and other – datasets are generated is mostly available as a scientific paper, e.g., [12], or software repository, e.g., <https://github.com/dbpedia/extraction-framework>. This inhibits reproducibility and

<sup>1</sup> <http://iswc2016.semanticweb.org/pages/program/accepted-papers.html>

comparability, as these current documentation solutions do not provide machine-interpretable detailed metadata describing the dataset generation process. This demands manual intervention and specific software and hardware environments to reproduce or compare a generated dataset, if possible at all<sup>2</sup>.

In this paper, we propose an automatic capturing mechanism suitable for reproducibility and comparability of data processing, without relying on the availability of specific software and hardware environments, or implying restrictions on the complexity of the data processing. This detailed metadata and provenance that describes the continuously increasingly complex generation processes of published datasets thus improves reproducibility and accountability.

After providing a background on provenance in Section 2, we show the different relevant provenance types and underlying model using the PROV Ontology (PROV-O) [1] in Section 3. In Section 4, we show how using declarative statements to describe the computational experiment allows us to automatically capture term-level provenance. We apply our model to the Function Ontology (F<sub>h</sub>O) [3] which, on its own turn, is aligned with the RDF Mapping Language (RML) [7]. Our proposed approach is implemented in the RML and F<sub>h</sub>O tool chain, namely, the RMLMapper and FunctionProcessor [4], and used to generate metadata based on the generation of a sample DBpedia dataset. We conclude in Section 5.

## 2 Automatic Metadata for Linked Data Generation and Publishing

In this paper, we propose automatically capturing machine-interpretable and detailed metadata concerning the data processing of a dataset generation to improve reproducibility. In Section 2.1, we introduce metadata formats to enable reproducibility, and in Section 2.2, we provide existing work that automatically captures machine-interpretable metadata, without considering data processing.

### 2.1 Provenance

Provenance can be considered information describing materials and transformations applied to derive the data and the processes that enabled their creation [17]. It has several applications [10], namely to assess *data quality*, trace the *audit trail*, aid in describing *replication recipes*, establish *attribution*, and be *informational*, i.e., provide context. As such, providing provenance of a data processing alongside the generated dataset can improve general reproducibility.

Providing this provenance as Linked Data has advantages, as its distributed nature allows us (i) to publish provenance separate from the actually published dataset, and (ii) to easily interlink different provenance dimensions without tight coupling. To improve interoperability, we use commonly used Linked Data vocabularies to describe the provenance. Provenance vocabularies already exist,

<sup>2</sup> Virtualization tools such as Docker (<https://www.docker.com/>) do abstract certain software environment requirements, however, they still rely on the (public) availability of all needed software tools.

namely, the PROV Ontology (PROV-O) [1], a W3C recommendation that uses the Web Ontology Language (OWL2) [2] to represent and interchange provenance generated in different systems and under different contexts. Describing the generation process using provenance modeled in PROV-O thus allows us to generate machine-interpretable and interoperable metadata.

## 2.2 Automatic Capture of Provenance

A provenance capture mechanism falls into three main classes: workflow-, process-, and operating system-based (OS) [9]. Workflow-based mechanisms are attached to or integrated in a workflow system, process-based mechanisms require each involved service or process to document itself, and OS-based mechanisms rely solely on the availability of specific functionalities at the OS level, without modifications to existing scripts or programs. Considering a data generation process as a single step within a workflow, and aiming to provide an implementation independent solution – thus an OS independent solution – the provenance of a dataset generation process is best captured using a process-based mechanism. As such, it is complementary to workflow capturing mechanisms such as implemented in the Pegasus Workflow Management System [5], and OS capturing mechanisms such as implemented in PANDA [8].

Related work [6] automatically captures metadata and provenance information decoupled from the implementation by relying on declarative descriptions, both for (i) the mapping rules that specify how to generate the Linked Data in RDF, and (ii) the raw data access interfaces. Different detail levels have been identified to capture metadata and provenance [6], namely, on the dataset, named graph, partitioned dataset, triple, and term level. Furthermore, multiple ways of adding provenance information to the declarative descriptions using PROV-O have been identified [6], namely using explicit graphs, implicit graphs, singleton properties [14], or reification. Implicit graphs and reification have the advantage that they do not influence the generated RDF data, whilst explicit graphs might conflict with generated named graphs, and singleton properties require changing the schema level transformations [6].

However, aforementioned work does not include data processing, i.e., it only takes raw extracted data values into account. Meanwhile, most generated datasets are related to specific data processing, and its provenance is an essential part in the reproducibility of a generated dataset.

## 3 Metadata and Provenance Capture for Data Transformations

We first state the different dimensions to take into account when capturing the metadata and provenance for data processing in Section 3.1, after which we propose our model in Section 3.2.

### 3.1 Metadata and Provenance Dimensions

*Schema vs Data Transformations* Dataset generation depends on both *schema* and *data* transformations [16]. Schema transformations involve (re-)modeling the original data, describing how objects are related, and deciding which vocabularies and ontologies to use [11]. Data transformations are needed to support any changes in the structure, representation or content of data [16]. However, instead of coupling these transformations, aligning them allows them to be executed separately as well as combined [3]. Namely, aligning instead of coupling the provenance of these transformations allows us to reproduce data transformations without needing to reproduce the schema transformations and vice versa.

*Interaction vs Actor* The execution of a generation process involves multiple actors, namely, the processor executing the generation process, and the different processes that perform the data transformations. The relation between these actors is a client-service relation, i.e., the generation process (client) calls the different data transformations (service). Two kinds of provenance are generated by these actors [17]: (i) *interaction provenance*, which describes the input and output parameters of each execution, generated and confirmed by both client and service actor, and (ii) *actor provenance*, which is metadata about the actor's own state during an execution (e.g. implementation details or hardware configuration) and is not verifiable by the other actors. Both kinds of provenance are complementary, i.e., interaction provenance can be used to compare results without relying on the implementation, whilst actor provenance can be used to reproduce performance measurements.

### 3.2 Metadata and Provenance Model

Whereas capturing provenance can be applied on different levels in the data generation process [6], it is most relevant on *term level*, as only then it can be unambiguously defined which data transformation process contributed to which value. For instance, DBpedia data involves parsing, e.g., date values from infoboxes in Wikipedia. In particular the provenance of how this date value was parsed is important for the DBpedia generation provenance, and currently not measured nor published [4]. When capturing term level provenance, data transformations are decoupled from the schema transformation, i.e., the captured metadata and provenance is defined on value level, and does not rely on the relationships between resources and used vocabularies or ontologies. In addition to the existing domains to capture metadata and provenance (i.e., mapping rules definition and data sources retrieval), we introduce another domain: the processing domain. This covers the data transformations, complementary to the schema transformations covered by the mapping rules.

To capture both interaction and actor provenance, it is necessary to capture and align implementation specific data (e.g., the software release as actor provenance) with implementation independent data (e.g., the input and output values as interaction provenance). However, on top of input and output values,

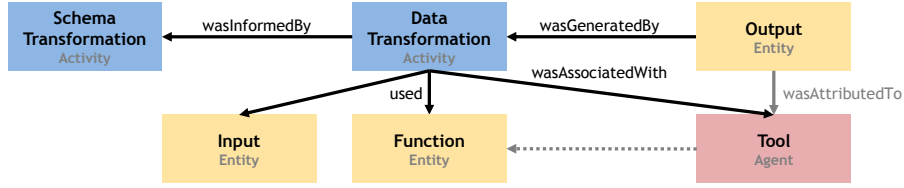


Fig. 1: PROV-O model of data transformations for dataset generation. Relations that can be derived from existing data are depicted gray, relations not within the PROV-O model are dotted.

we argue additional implementation independent data is needed to allow comparability between data generation processes, namely, the type of data processing executed. This allows comparing interaction provenance across tools. When two different tools implement the same type of data processing function, the input and output values of the first data generation process with the first tool can be used to compare with the second tool. For the remainder of the paper, we will make the distinction between *function* (i.e., the implementation independent description) and *tool* (i.e., the implementation specific description).

Our model is visualized in Figure 1 and mapped to PROV-O [1]. There is a distinction between schema and data transformation, we provide actor and interaction provenance, and both function and tool are included. Schema and data transformations are a `prov:Activity`, where the latter is informed by (`prov:wasInformedBy`) the schema transformation. Input, function, and output are a `prov:Entity`, and the tool is a `prov:Agent`. The data transformation uses (`prov:used`) the input<sup>3</sup> and function, and the output is generated by (`prov:wasGeneratedBy`) the data transformation. The data transformation is associated with (`prov:wasAssociatedWith`) the tool, thus we can derive that the output is attributed to (`prov:wasAttributedTo`) the tool. The relation between the function and the tool can also be derived, but this is not part of the PROV-O model.

## 4 Application

Capturing metadata and provenance within the dataset generation process – specifically when including data transformations – requires term-level capturing mechanisms. Instead of providing a tool-specific solution, i.e., changing a specific system, our approach considers capturing metadata and provenance based on machine interpretable descriptions of the dataset generation process. This way, the approach is independent of the actual implementation.

As exemplary case, we consider the RDF Mapping Language (RML) [7] to provide the machine interpretable mapping rules for the schema transformations, and the Function Ontology (F<sub>n</sub>O) [3] to describe the data transformations. An alignment between RML and F<sub>n</sub>O is presented in previous work [4]. RML is considered because it is the only language that allows uniformly defining the mapping

<sup>3</sup> The model puts no restriction on amount of inputs, but it is not visualized for clarity.

```

1  grel:toTitleCase a fno:Function, prov:Entity ;
2  fno:name "title case" ;
3  fno:expects ( [ fno:predicate grel:stringInput ] ) ;
4  fno:output ( [ fno:predicate grel:stringOutput ] ) .
5
6  :exe a fno:Execution, prov:Activity ;
7  prov:wasInformedBy :RDFdataset_Generation .
8  implementation :grelJavaImpl ;
9  fno:executes grel:toTitleCase ;
10 grel:stringInput :input ;
11 grel:stringOutput :output .
12
13 :input a prov:Entity ; rdf:value "ben de meester" .
14 :output a prov:Entity ; rdf:value "Ben De Meester" .

```

**Listing 1:** Function descriptions and Executions using FnO

rules over heterogeneous data sources, thus covers the most dataset generation use cases. FnO is considered as it allows implementation independent description of functions, their input parameters, and returning results.

Existing work that allows capturing metadata and provenance for dataset generation [6] is extended to include data transformations. On top of relying on the declarative descriptions of (i) the mapping rules and (ii) the raw data access interfaces, we include the declarative descriptions of (iii) the data transformations. For all detail levels except the term level, it suffices to include which data transformation functions have been used, using their FnO description.

For the term detail level, we extend existing work to include the model as presented at Section 3.2, relying on the mapping rules described in RML to trigger the execution of a function described in FnO. As example, we present a simple mapping process that maps person names, and requires a title case data transformation<sup>4</sup>. In detail: a data source (`:Source a prov:Entity`) is mapped using a mapping process (`:RDFdataset_Generation a prov:Activity`). This mapping process executes the schema transformations. When generating the triple `:Ben foaf:name "Ben De Meester"`, a data transformation needs to be executed on the object, thus, a mapping rule triggers the execution of a function. The description of the `grel:toTitleCase` function is given in Listing 1, lines 1–4. When executing this function, the actual implementation (`:grelJavaImpl a prov:Actor`) needs to be retrieved. The execution of that function with specific input data, together with the PROV types for clarification, is given in Listing 1, lines 7–14 using the FnO statements [3]. For capturing the provenance of literal values, we need an intermediate resource and `rdf:value` relation to attach additional metadata. This is not needed in the generated dataset.

Based on this FnO execution description, additional PROV-O statements such as `:exe prov:wasInformedBy :RDFdataset_Generation` as well as `:output prov:wasGeneratedBy :exe` can be derived. On <https://fno.io/prov/>, an extended de-

<sup>4</sup> More advanced mapping processes, with more complicated data processing, e.g., NLP, or complex algorithms to generate data values or resources are similar, as the provenance model makes no assumptions on the execution complexity.

scription and example is given which, due to page constraints, could not be incorporated in the paper. As such, we can capture all needed metadata and provenance using RML and F<sub>NO</sub>, with an additional cost of about ten triples for every literal or resource generated. This increases the total amount of generated triples with an order of magnitude. However, the captured metadata and provenance can be published separately and does not affect the generated dataset.

Our approach was implemented in the RMLMapper, the reference implementation for both RML and its alignment with F<sub>NO</sub> [4]. We then captured all metadata and provenance for a sample of DBpedia, together with additional description available at <https://fno.io/prov/dbpedia/>. Both F<sub>NO</sub> as PROV-O statements are available, and exemplary queries show how this approach can improve reproducibility and comparability of dataset generation processes. As DBpedia's new generation process includes RML and F<sub>NO</sub> [13], we can provide detailed provenance alongside the DBpedia data, e.g., all releases of the used tools to generate the sample DBpedia dataset can be requested (i.e., all data transformation tools, including the release of the RMLProcessor). Moreover, all executions' input and output concerning the DBpedia parsing functions used to parse the Wikipedia input data into normalized RDF literals can be requested. This allows the functional evaluation of a new parsing function by comparing its output values with those of the existing parsing functions. The inclusion of `prov:startedAtTime` and `prov:endedAtTime` statements allows performance evaluation, given that hardware context is also included and can be compared. Moreover, decoupling schema and data transformations allows us to reproduce and compare data transformations without needing to execute the schema transformations and vice versa.

## 5 Conclusions

Dataset generation consumes a large part of scientific output, including complex and specific data processing. The publication of this generation process however is not reproducible, as current documentation options do not provide machine-intepretable detailed metadata that do not rely on specific software environments. In this work, we present automatic capturing of metadata and provenance of data processing on term level, whilst separating schema and data transformations and capturing both actor and interaction provenance.

As we tested our approach on the DBpedia generation, it is evident that our solution allows reproducibility, as details about the used tools for every input-output pair are available, and data transformations can be executed decoupled from the schema transformations. Comparability is also improved, as the connection with implementation independent description of the used function, together with the input-output pair, can be used to compare different tools.

## References

1. Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., Zhao, J.: PROV-O: The PROV Ontology. Recommendation, World Wide Web Consortium (W3C) (Apr 2013), <https://www.w3.org/TR/prov-o/>

2. Bock, C., Fokoue, A., Haase, P., Hoekstra, R., Horrocks, I., Ruttenberg, A., Sattler, U., Smith, M.: OWL 2 Web Ontology Language – Structural Specification and Functional-Style Syntax (Second Edition). Recommendation, World Wide Web Consortium (W3C) (Dec 2012), <http://www.w3.org/TR/owl2-syntax/>
3. De Meester, B., Dimou, A., Verborgh, R., Mannens, E., Van de Walle, R.: An Ontology to Semantically Declare and Describe Functions. In: The Semantic Web: Proceedings of the 13th ESWC Satellite Events. LNCS, vol. 9989, pp. 46–49. Springer, Cham, Heraklion, Crete, Greece (Oct 2016)
4. De Meester, B., Maroy, W., Dimou, A., Verborgh, R., Mannens, E.: Declarative data transformations for Linked Data generation: the case of DBpedia. In: The Semantic Web: Proceedings of the 14th International Conference, ESWC. LNCS, vol. 10250, pp. 33–48. Springer, Cham, Portorož, Slovenia (May 2017)
5. Deelman, E., Vahi, K., Juve, G., Rynge, M., Callaghan, S., Maechling, P.J., Mayani, R., Chen, W., Ferreira da Silva, R., Livny, M., Wenger, K.: Pegasus, a workflow management system for science automation. *Future Generation Computer Systems* 46(C), 17–35 (May 2015)
6. Dimou, A., De Nies, T., Verborgh, R., Mannens, E., Van de Walle, R.: Automated metadata generation for Linked Data generation and publishing workflows. In: Proceedings of the 9th Workshop on Linked Data on the Web. vol. 1593. CEUR, Montreal, Canada (Apr 2016)
7. Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van de Walle, R.: RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In: Proceedings of the 7th Workshop on Linked Data on the Web. vol. 1184. CEUR, Seoul, Korea (Apr 2014)
8. Dolan-Gavitt, B.F., Hodosh, J., Hulin, P., Leek, T., Whelan, R.: Repeatable reverse engineering for the greater good with PANDA (2014)
9. Freire, J., Koop, D., Santos, E., Silva, C.T.: Provenance for computational tasks: A survey. *Computing in Science & Engineering* 10(3), 20–30 (May 2008)
10. Goble, C.: Position statement: Musings on provenance, workflow and (semantic web) annotations for bioinformatics. In: Workshop on Data Derivation and Provenance. vol. 3. Chicago, USA (Oct 2002)
11. Hyland, B., Atemezing, G., Villazón-Terrazas, B.: Best Practices for Publishing Linked Data. Working Group Note, World Wide Web Consortium (W3C) (Jan 2014), <https://www.w3.org/TR/ld-bp/>
12. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia. *Sem Web* (2015)
13. Maroy, W., Dimou, A., Kontokostas, D., De Meester, B., Verborgh, R., Lehmann, J., Mannens, E., Hellmann, S.: Sustainable linked data generation: The case of DBpedia. In: Proceedings of the 16th International Semantic Web Conference: In-Use Track. Springer, Vienna, Austria (Oct 2017)
14. Nguyen, V., Bodenreider, O., Sheth, A.: Don’t like RDF reification?: Making s-statements about statements using singleton property. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 759–770. ACM, New York, NY, USA (2014)
15. Peng, R.D.: Reproducible research in computational science. *Science* 334(6060), 1226–1227 (Dec 2011)
16. Rahm, E., Do, H.H.: Data cleaning: Problems and current approaches. *IEEE Data Engineering Bulletin* 23(4), 3–13 (2000)
17. Simmhan, Y.L., Plale, B., Gannon, D.: A survey of data provenance in e-science. *SIGMOD Rec.* 34(3), 31–36 (Sep 2005)