4-1-2013

An Ontological Approach to Capture Data Provenance Across Multiple Platforms

Leonardo Salayandia
University of Texas at El Paso, leonardo@utep.edu

Comments:
Technical Report: UTEP-CS-13-24

An Ontological Approach to Capture Data Provenance Across Multiple Platforms

Leonardo Salayandia
University of Texas at El Paso

Abstract

The process of collecting and transforming data can extend across different platforms, both physical and digital. Capturing provenance that reflects the actions involved in such a process in a consistent manner can be difficult and involve the use of multiple tools. An approach based on formal ontologies and software engineering practices is presented to capture data provenance. The approach starts by creating ontologies about data collection and transformation processes. These ontologies, referred to as Workflow-Driven Ontologies, establish a consistent view of the process that is independent of the platform used to carry out the process. Next, software modules are generated, targeting specific types of platforms on which data processes are implemented, so that data provenance can be captured as the process is being carried out. This paper presents the software architecture of the approach and discusses the generation of software modules, leveraging the structure and terminology of Workflow Driven Ontologies to capture data provenance. The result of this approach is the creation and population of knowledge bases that capture the processes used to collect and transform data, as well as provenance about how individual datasets were produced.

1. Introduction

End-to-end data processes often extend across multiple platforms, starting from field observations and measurements that may be recorded in non-digital formats, going through digital ingestion phases, using computing resources to further transform the data, and eventually going through human analysis phases to reason about the data and interpret it. While various data processing systems, such as those based on workflow or database models offer mechanisms to capture data provenance within their domains of operation, the challenge of comprehensively capturing data provenance across multiple process platforms is not addressed. This paper discusses an approach, based on formal ontologies and software engineering practices to capture data provenance across multiple data processing platforms.

Previous work has established the Workflow-Driven Ontologies (WDO) Framework to facilitate the construction of formal ontologies by end-users to model their data processes [1]. Additionally, formal ontologies created with the WDO framework have been used to capture data provenance in two target platforms: 1) a manual platform where data processing activities and data provenance recording are performed manually [2], and 2) a scripting platform where the data process consists of a pipe-and-filter pattern [3].

The *first contribution of this work* is an architectural design that can be used to capture data provenance for a data processing environment that extends across multiple platforms. The *second contribution of this work* is a mapping from formal ontologies created with the WDO Framework to software modules that can be used to enhance data processing systems to capture data provenance in a knowledge base. This second contribution is an extension of previous work to create software modules in light of the proposed software architecture, as well as to target additional types of data processing platforms: Object Oriented languages, workflow management systems, and black box tools.

Previous work has defined a set of criteria to evaluate user support in defining data processes and using provenance traces [4], which is expected to be applicable to the evaluation of this work. Other criteria related to system performance, manual implementation effort, and source code invasiveness are applicable as well [6].

2. Approach

The approach consists of creating a conceptual representation of the end-to-end data collection and transformation process, which will serve as guidance for the capture of data provenance of interest. While other work emphasizes a distinction between coarse- and fine-grained provenance [5], the emphasis in establishing a conceptual data process from the start is to focus on an appropriate level of detail for a given use case. Once a conceptual data process has been established, the next phase of the approach is to generate software modules based on the components of the conceptual data process and that are customized to target the specific platforms used to carry out the actual data process. Finally, the generated software modules are used to intercept data and metadata about the state of data processing at specific points of interest. Previous work has

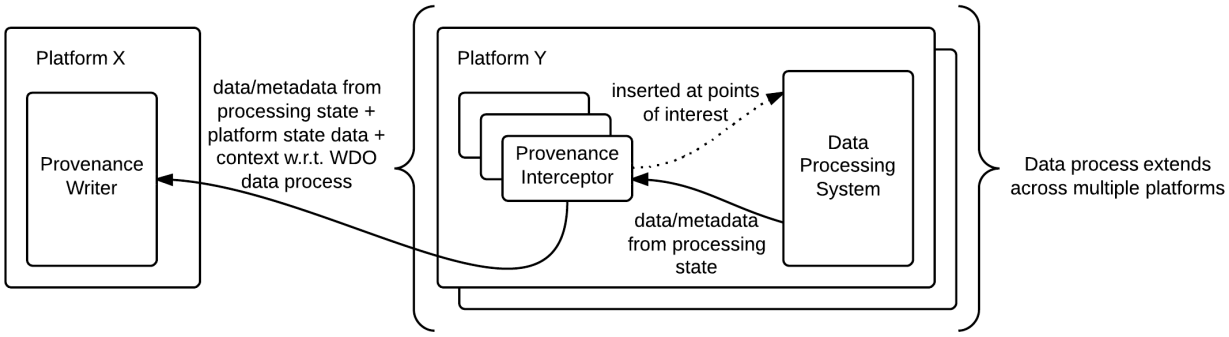

Figure 1. Software architecture to capture data provenance across multiple data process platforms.

achieved this last step for processes carried out in a scripting platform [3]. Other work has described the notion of point of interest with respect to parts of a data processing system where provenance information can be intercepted [6].

The Workflow-Driven Ontologies (WDO) framework is used to model data collection and transformation processes by focusing solely on data flow [1]. Data flow dependencies are captured in terms of input and output ontological class restrictions. All other restrictions based on control flow or execution-related conditions are omitted. The intention is to facilitate the creation of data process models for end users. End users refer to the people responsible for designing and carrying out data processes, who typically consider data a first class citizen and technical details secondary to their endeavors. Another advantage of focusing solely on data flow is that the resulting ontologies, i.e., WDOs, are useful to model data processes regardless of the execution platform used; WDOs are just as applicable to manual processes that consist of manual activities carried out in the physical world, as they are to automated processes carried out in high performance computing machines. Furthermore, the metamodel of WDOs is aligned to the PML data provenance ontology [7]. Hence, WDOs can also be leveraged to construct knowledge bases about data provenance. Future work on WDOs will investigate the alignment of WDOs to other provenance models such as PROV-O [8].

3. Software Architecture

The software architecture for this approach includes three types of components: *data processing system*, *provenance interceptor*, and *provenance writer*. These components are illustrated in Figure 1.

The *Data Processing System* is responsible to carry out parts or whole data collection and transformation processes. Depending on the types of activities involved in the process, this component can be a software application executed in a computing platform or a set of manual activities carried out in the physical world. Figure 1 illustrates *Platform Y* as the platform hosting a *Data Processing System*. However, notice that there can be multiple platforms, each of which could host data processing systems that execute parts of a comprehensive data process. It is assumed there exists an entity that orchestrates the multiple parts of the process, establishing an order and interaction protocol. For example, a person can be such an entity; the person manipulates multiple data processing systems, transferring data among them as necessary to achieve intended outcomes. Initially, the comprehensive data process may be implicit in the person's mind. The person explicitly documents the process as a WDO to capture provenance consistently from end-to-end.

The *Provenance Interceptor* is responsible to intercept data and metadata at points of interest within the *Data Processing System* in order to build the data provenance record. While the *Provenance Interceptor* necessarily has to operate under the constraints of the hosting platform of the *Data Processing System*, the *Provenance Interceptor* does not intervene with the normal operation of the *Data Processing System*. The data and metadata that is necessary to build the provenance record can vary depending on the type of use case being supported and the provenance language being used. Comprehensively, the provenance record contains information related to: 1) *What*: refers to the data itself and which can be passed by reference or by value; 2) *Where*: can be determined by the platform on which the data processing system is being executed, e.g., where data is being processed, or by the data itself, e.g., the latitude and longitude columns of the dataset being processed; 3) *When*: can be determined by the platform on which the data is being processed, e.g., a timestamp generated by the executing platform, or by the data itself, e.g., the timestamp column of the dataset being processed; 4) *How*: refers to the metadata related to the

operations being carried out by the system, e.g., reference to the extrapolation routine implementation; and 5) *Who*: can be determined by the platform on which the data processing system is being executed, e.g., user login information, or by metadata included in the dataset being processed.

Provenance Interceptors are generated from the WDO, representative of a comprehensive, end-to-end data process. Given the focus on data flow, the WDO includes *Method* concepts and their input and output relations to *Data* concepts. *Method* concepts are representations of activities in a data process, which can be implemented on various platforms. A *Provenance Interceptor* is created for each *Method* concept included in the WDO and targeted to the specific platform of choice. For example a WDO may contain the *Extrapolation* method, which requires a dataset as input and is expected to produce an extrapolated dataset. Such a method could be implemented in a Java program as the *kriging* method:

```
public class Dataset {…}
public class ExtrapolatedDataset {…}
public class DataProcessing {
        …
  ExtrapolatedDataset kriging (Dataset x) { … }
        …
}
```

A person that is familiar with the technical implementation of the data process is required to identify points of interest in the source code or protocol, and to integrate the generated *Provenance Interceptor* to intercept the data and metadata that is representative of the data process state related to the targeted process activity. For the example above, the *Provenance Interceptor* would be customized to capture the resulting dataset after the *kriging* method is terminated (i.e., What), which would be passed by value because the state of the resulting variable would not be persisted after the program terminates. Additionally, the *Provenance Interceptor* would capture metadata about the hosting computer (i.e., Where), a system timestamp generated right after the *kriging* method returns (i.e., When), a reference to the *kriging* function (i.e., How), and user login information (i.e., Who).

The *Provenance Writer* is responsible to build the provenance record based solely on the information provided by *Provenance Interceptors*. Timing at which *Provenance Interceptors* log their corresponding activities is critical to build the provenance record. However, it is assumed that race conditions or other timing concerns are handled by the logic of *Data Processing Systems*, as well as by orchestrating entities responsible to coordinate parts of the data process across multiple platforms.

4. Data Process Platforms

In this work, a data process platform refers to the type of hosting system used to carry out a process to collect and transform data. The data process platform establishes the types of actions that can be carried out for a process. The data process designer determines the platform best fitted for the requirements of the data process at hand.

As previously mentioned, *Provenance Interceptors* need to run under the same data process platform used by the *Data Processing System*. As a result, this approach to capture data provenance across platforms requires generators of *Provenance Interceptors* to target multiple platforms. Several platforms are discussed next with respect to the feasibility of generating and customizing *Provenance Interceptors*.

4.1. Manual Platform

Manual platforms support data process activities that are driven by humans. For example, collecting data by physically taking measurements for a variable of interest. This kind of platform is a special case where a person plays the role of a *Provenance Interceptor*, manually entering data and metadata of interest to build the provenance record. DerivA is a Java application for people to manually create provenance records encoded in the Proof Markup Language (PML) and based on conceptual data processes created with the WDO framework [2].

4.2. Scripting Language Platform

Scripting language platforms, typically in UNIX-like operating systems, support a wide range of control flow structures, have access to OS environment variables and settings, and can make calls to OS commands and executable programs. Data process systems implemented in this type of platform typically follow a pipe-and-filter pattern, where data is processed by some program and its output is piped as input to another program for further processing. Data interceptors have been generated for these types of systems using the WDO framework [3], typically requiring intermediate data processing state to be written to files in order to be referenced in the provenance record.

4.3 Object Oriented Language Platform

Object Oriented (OO) language platforms, such as Java and C++, use encapsulation and abstraction software development techniques to support the creation of complex *Data Processing Systems*. *Provenance Interceptors* can be generated as abstract classes, which can be extended by the classes of the *Data Processing System*. In the case where the OO design of the *Data Processing*

System closely matches the structure of the WDO conceptual process, determining the points of interest of where abstract classes can wrap data processing classes is trivial. For the opposite situation, however, other solutions may be needed. For example, Schäler et al. have investigated the use of Aspects to capture provenance in OO systems as a crosscutting concern [6]. A *Provenance Interceptor* generator for this type of platform is being investigated.

4.4. Workflow System Platform

Workflow Systems for scientific data processing provide platforms that support executable data processing modules that can be reused to build *workflows*. Additionally, workflow systems typically support composition mechanisms by which workflows can be composed of other workflows, providing multiple levels of abstraction. WDOs could be used to build an abstract workflow in the target workflow system to wrap an executable workflow. By leveraging remote communication capabilities of the platform, e.g., Web Service calls, communication with a remote *Provenance Writer* to capture provenance would be possible. A *Provenance Interceptor* generator for this type of system is being investigated, initially targeting the Kepler Workflow System [9].

4.5. Black-box Tool Platform

Black-box tool platforms refer to proprietary tools that cannot be modified, e.g., MATLAB. The goal for these types of platforms is to keep track of the inputs and outputs towards building the provenance record. For these platforms, an approach like that of a manual platform can be used, where the user of the tool is responsible to logging in his/her input and output data. Another alternative is to use an approach like that of scripting language platforms, where inputs and output data locations are pre-established and scripts are scheduled to automatically create provenance records.

5. Summary

An approach has been described to capture data provenance across multiple platforms. The approach uses an ontological representation of end-to-end data processes to establish a level of detail at which to capture data provenance, as well as to generate software modules that can be customized to capture provenance as the data processing system is being executed.

Preliminary work has provided evidence of the feasibility of the approach. The approach is described here as a software architecture and support for additional data process platforms is being investigated.

Despite the extension of end-to-end data processes across multiple platforms, a consistent data provenance record can be achieved by establishing a comprehensive data process and by using a common provenance language throughout. Current prototypes focus on PML and PROV-O is being investigated.

Previous work has defined a set of criteria to evaluate user support in defining data processes and using provenance traces [4]. It is expected that these criteria, are applicable towards the evaluation of this work. Other criteria related to system performance, manual implementation effort, and source code invasiveness are also applicable [6].

By using formal ontologies as the foundation of this approach, the intention is to support the creation and population of knowledge bases that capture the processes used to collect and transform data, as well as provenance about how individual datasets were produced.

References

[1] Leonardo Salayandia. *Ontologies for Scientific Data Transformation*. PhD thesis, University of Texas at El Paso, December 2012.

[2] Antonio Garza. http://trust.utep.edu/derivA/. Web. January 2013.

[3] Paulo Pinheiro da Silva, Leonardo Salayandia, Nicholas Del Rio, and Ann Q. Gates. *On the use of abstract workflows to capture scientific process provenance*. In Proceedings of the 2nd Workshop on the Theory and Practice of Provenance (TaPP'10), San Jose, CA, February 2010.

[4] Leonardo Salayandia, Ann Q. Gates, and Paulo Pinheiro. *An approach to evaluate scientist support in abstract workflows and provenance traces*. In Discovery Informatics Symposium: The Role of AI Research in Innovating Scientific Processes, AAAI Fall Symposium Series, Arlington, VA, November 2012.

[5] Yael Amsterdamer, Susan B. Davidson, Daniel Deutch, Tova Milo, Julia Stoyanovich, Val Tannen. *Putting Lipstick on Pig: Enabling Database-style Workflow Provenance*. CoRR abs/1201.0231 (2012)

[6] Martin Schäler, Sandro Schulze, and Gunter Saake, *Toward Provenance Capturing as Cross-Cutting Concern*, TaPP Workshop 2012.

[7] Deborah L. McGuinness, Li Ding, Paulo Pinheiro da Silva, and Cynthia Chang. *PML 2: A modular explanation interlingua*. In Proceedings of the 2007 Workshop on Explanation-aware Computing, ExaCt-2007, 2007.

[8] PROV-O: The PROV Ontology http://www.w3.org /TR/prov-o/. Web. January 2013.

[9] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. *Scientific workflow management and the kepler system*. In Concurr. Comput. : Pract. Exper, page 2006, 2005.