DATASET VERSIONING THROUGH CHANGELOG ANNOTATION

By

Benno Lee

Prepared for:

Peter Fox, Thesis Advisor

Jim Hendler, Advisor

Deborah MacGuiness, Member

Beth Plale, Member

Rensselaer Polytechnic Institute
Troy, New York

November 2016
(For Graduation December 2017)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

ABSTRACT

This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text.

This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text.

This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text.

This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text. This is a sentence used to take up space and look like text.

CHAPTER 1
INTRODUCTION

Software development followed a stiff production cycle prior to the early 2000s. However, as technology developed, software development required a system to adapt and change to evolving conditions leading to the Agile Manifesto. Likewise, data collected by researchers grew at an astounding rate with new technology. NASA's Atmospheric Science Data Center reported a growth from hosting around five million files to twenty million files between 2001 and 2004. Many datasets have required reprocessings of their data, either to improve data quality or to correct for errors [CALIPSO, ARM].

Dataset versioning tracks and documents the changes which occur in datasets. Version numbers are commonly seen in relation to new releases of software such as MATLAB or R, but current version naming schemes borrow from software contexts. While it may have been possible to manually manage the changes to data when datasets were relatively small, research centers must now supervise on the order of tens of millions of files with changes being made at rates of thousands of processing jobs per day.

Versioning information is widespread across devices and software in the modern world. From the numbering of the latest smart phone to the patch number of the newest release of MATLAB, scientists must deal with a range of labels and formats when performing their research. Science data, likewise, also has a tendency to change. Datasets are subjected to data audits and error corrections regularly to maintain a level of data quality.

Agencies and research groups have collected new data at an incredible rate. The amount of data housed by NASA quadrupled from 2001 to 2004 [4] and high energy physics labs can generate on the order of 4000 new datasets every day [5].

1.1 Provenance

In a number of papers, authors describe models or systems that track changes in the workflow and how modifications to the flow would then generate new versions of datasets. The information that details the activities and agents involved in generating a data entity is known as provenance. Barkstrom identifies times when new versions should be generated by locating changes in provenance (specifically calibrations, scripts, and input data) at different levels of data processing in NASA remote sensing workflows. Software revision management tools such as Git and SVN also keep track of provenance information when logging new commits to a project. Since it plays a significant role in triggering new version generations, provenance is often conflated with versioning by scientists. However, it does not sufficiently characterize a versioning event.

Provenance is the data used to describe the origin of an object (Merrium Webster). In the field of semantic technologies, the W3C recommendation, PROV provides a data model to encode provenance information so that the lineage of data products can be traced. However, PROV expresses relationships between versions with the wasRevisionOf or alternateOf property. The properties do not allow for more elaboration as to what changes were made to transform version one into its alternate, nor should it. Explicitly itemizing the changes is unnecessary in order to communicate the provenance relationship between the two versions. That there exists an association is sufficient. This gap illustrates why versioning information is necessary because itemization allows data consumers to determine the significance of changing to version two.

1.2 Types of Change

One of the challenges facing data versioning is that changes within a version often mean different things to different people. From a data producers standpoint, changing the units within a dataset does not have significant impact to the science they are trying to communicate, but it may have catastrophic impacts on a data consumer who is expecting the dataset to be formatted in specific units.

Figure 1.1: This is the Caption for Figure 1 make it long to illustrate how it looks when wrapped around to the next line

Table 1.1: This is the Caption for Table 1[1]

Here's	an	example
of	a	table
floated	with	the
`table`	environment	command.

1.3 Changelogs

1.4 RDFa

1.5 Why Semantic Technologies?

This is a sentence to take up space [1]. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

Please refer to Figure 1.1.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

1.6 This is a Section Heading

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

1.6.1 This is a Subsection Heading

This is a sentence to take up space and look like text. This is a sentence to take up space [2]. This is a sentence to take up space and look like text.

1.6.1.1 This is a Subsubsection Heading

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. Text before the footnote.[1] Text after the footnote. This is a sentence to take up space and look like text.

[1] Here's the text of the footnote.

CHAPTER 2
PREVIOUS WORK

PROV is a W3C recommendation that deliniates a method to express data provenance with semantic technologies. Using the model of relating activities, agents, and entities, data managers can express the origins of their datasets. However, when an entity is revised, the PROV data model can only express the relationship as a revision or that the new dataset was derived from the original. This leaves

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

This is shown in table 2.1.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

Figure 2.1: This is the Caption for the First Figure in Chapter 2. It is a long, long caption; we do not want to put the whole thing in the List of Figures. A Shorter Caption can go in the square brackets.

Table 2.1: This is the Caption for Table 2

Here's	another	example
of	a	table
floated	with	the
`table`	environment	command.

2.1 This is a Section Heading

This is a sentence to take up space and look like text. This is a sentence to take up space [3]. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text.

2.1.1 This is a Subsection Heading

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. Text before a footnote.[2] Text after the footnote.

This is a sentence to take up space and look like text. This is a sentence to take up space and look like text. Text before another footnote.[3] Text after the footnote. This is a sentence to take up space and look like text.

[2] Here's the text of the footnote.
[3] Here's the text of the footnote.

CHAPTER 3
CONCEPTUAL MODEL

The conceptual model used within this thesis is built around the expression of three core versioning operations: addition, invalidation, and modification. These three activities can be represented by interacting with three types of concepts: versions, attributes, and changes. Versions represent the data entities being compared. These could be two different editions of a book or versions of software. It is important to understand that a version is an abstraction as it can be represented by multiple physical files. In the sections that follow, operations will only consider the interaction between two versions and will be explained later in the chapter. Versions then contain attributes representing a quantity being modified. Specifically for tabular data, attributes would correspond to an identifier that refers to particular rows or columns within the data. Attributes of the two versions are then connected by a change. This link functions as a very general concept which can be subclassed into more informative types such as unit changes, improving the expressiveness of the model beyond PROV's revisionOf concept.

3.1 ADDITION

When a change adds a new attribute to a version, it only needs to refer to version two and its corresponding attribute. The reasoning should be fairly obvious as the attribute never existed in version one, and therefore, there is nothing to refer to and no need to form a relationship between the change and version one. However, by linking the addition change to version one, we address a difficulty with comparing provenance graphs. When two data objects have identical structures, it is difficult to determine what time the objects were added to the dataset and which version they belong to. As a result, determining the compatability of the two objects becomes difficult. The change contributions to the dataset evolution appears naturally using this construction. The resulting model can be seen in Figure 3.1. Some relationships are specifically left out, such as that between Change A and Version 2, to not confuse

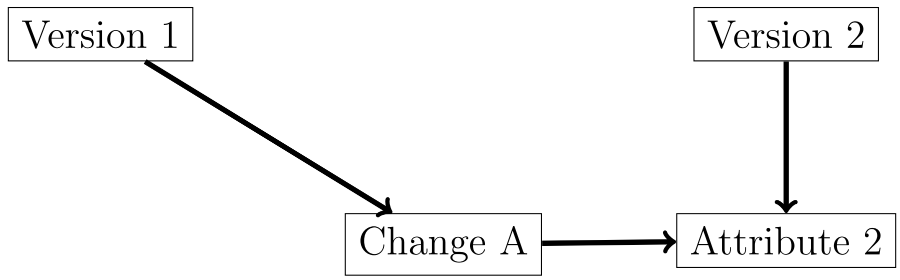

Figure 3.1: Model of the relationships between Versions 1 and 2 when adding an Attribute 2 to Version 2 as a result of Change A

identification of other types of changes. The relationship between Change A and Version 2 can still be implied from Attribute 2.

3.2 INVALIDATION

The Invalidation operation corresponds to the delete concept found in other applications. The choice of invalidation over delete results from the policy that, in versioning, data should never be deleted. In practicality, this may not be particularly feasible due to space limitations and relative validity. In either case, the change invalidates an attribute in version one, resulting in version two. Unlike the Addition operation, Invalidation forms a clear relationship between both versions, which can be seen in Figure 3.2. Notice again that since Attribute 1 no longer exists in Version 2, there is no corresponding Attribute 2 to refer to.

From Figure 3.1, we can see the confusion that could result from requiring explicit relationships between versions and changes in both the Addition and Invalidation operations. Linking Change A to Version 2 would create a duplicate connection and provides a mechanism to identify when items specifically enter or leave a version.

3.3 MODIFICATION

The final operation is Modification, and it maps a change from one attribute from version one to its corresponding attribute in version two. The particular type of change in this case is purposely left out in order to allow data producers to subclass and customize the resulting graph to properly reflect the versioning that they desire.

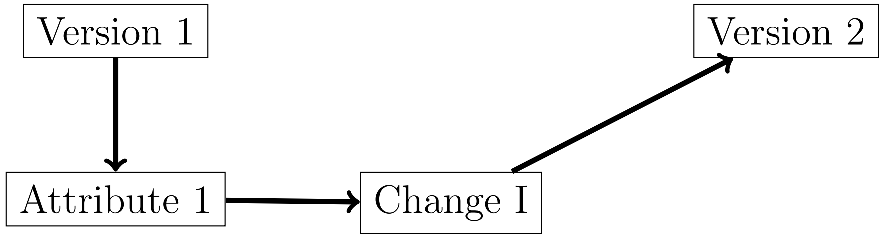

Figure 3.2: Model of the relationships between Versions 1 and 2 when invalidating Attribute 1 from Version 1 as a result of Change I

3.4 MULTIPLE LINKED VERSIONS

Using the construction outlined in the previous three sections, many changes can be compiled together into a graph in a changelog. After all additions, invalidations, and modifications have been compiled into a single graph, a complete mapping from version one to version two may be developed. The orientation of the relationships in the graph allows a flow to be created from attributes in version one to corresponding attributes in version two. Taking version two and performing the same graph construction to a version three results in not only a flow from version two to version three, but also from version one to version three. As a result, the flow can be used to construct a mapping from version one to version three or any future version.

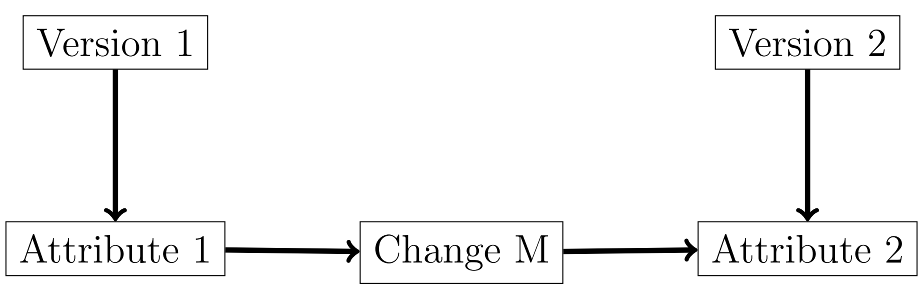

Figure 3.3: Model of the relationships between Versions 1 and 2 when modifying Attribute 1 from Version 1 as a result of Change M, resulting in Attribute 2 from Version 2

CHAPTER 4
VERSIONING SPREADSHEETS

Two datasets were initially studied. The "Noble gas isotopes in hydrocarbon gases, oils and related ground waters" database, initially published on June 11, 2013 and then republished a second version on March 8, 2015. The physical structure of the database changed from eight separate Excel spreadsheets to a single sheet. The model does not explicitly account for a change like this except that the identifiers used to refer to the versions or attributes involved may show that the items from version one originate from different files. The original dataset also had 195 columns. However, these were reduced to 54 columns in the second release. In addition, many new locations were surveyed and added to the second release. These are the most challenging

The Paragenetic Mode for Copper Minerals database produced two versions, one at a workshop on June 8, 2016 and another at a following workshop on August 21, 2016. These both take the form of Excel spreadsheets, which has the benefit of having strict row and column numbering. This allows unique identifiers to be used when referring to individual pieces of data and providing a level of abstraction. The structural changes made to the Copper Dataset resemble those found in the Noble Gas Dataset.

4.1 Provenance Analysis

The first approach to determining the provenance distance for the datasets began with the Noble Gas Dataset. The dataset provides a set of references from which the values were extracted and compiled into the dataset. As a result, a simple provenance mapping was constructed, using PROV, from each reference to its corresponding row in the spreadsheet. After this was done for each version of the dataset, we can generate graphs to compare objects from the two versions like the one found in Figure 4.1. We can tell from the labels that different Activities were used to compile the data entry, but the structure of the graph does not provide

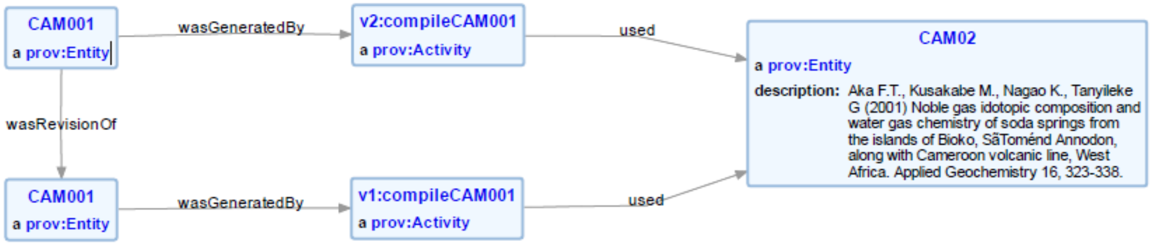

Figure 4.1: Provenance graph for the entry CAM001 entry of the Noble Gas Database. Other than the labels, the structure of each of the data objects is very much the same.

any information as to how extensive this change was for this version. Here we can see how the wasRevisionOf relationship would break down in determining the magnitude of change between the two versions. The relationship must, therefore, be expanded in order to provide the desired data necessary to make an evaluation.

4.2 Versioning Comparison

The initial challenge for both of these datasets is producing an appropriate mapping between their previous and latest versions. Through the second compilation process, many columns were removed from the dataset or moved around, and as a result, the identifiers associated with the Attribute in version two did not necessarily correspond to their identifier in version one. The column headings in the Noble Gas Dataset is stored in the spreadsheet as data and makes it difficult for a general system to automate mapping columns between the two versions since any number of rows may contain metadata information. Additionally, while the second release of the Noble Gas Dataset did have a document detailing the organization of the columns contained within the dataset, it did not have any information to map entries from the old dataset to the new dataset. This limits the ability to map and update changes to human actors. Immediately apparent is that any columns that have a mapping from version one to version two means that these columns (Attributes from the model) will only undergo Modification operations since there exists an associated Attribute in both the previous and current version. The conclusion then follows that all remaining Attributes (including both columns and rows) belonging to version one were Invalidated and those belonging to version two were Added. Using this conclusion, we can pre-calculate the Added and Invalidated Attributes and separate any report into changes grouped by operator.

Once a mapping exists between the two versions, a comparison was performed to determine whether an Attribute of the data object changed. In this case, a simple equality operator was used to determine if anything was modified. In practice, more advanced or complicated methods can be used to determine equality. Since, all Additions and Invalidations have already been predetermined, we can output each Modification as we see them. As is common in versioning, the changes were outputted to a changelog document formatted using HTML. The idea here is that by providing the changelog in HTML, the changes can be made available online and therefore accessible by data consumers. Another benefit of providing a changelog in HTML is that the document can be additionally enriched by RDFa. The changelog document becomes no longer restricted to human consumption, but allows autonomous agents to more intelligently interact with dynamic data systems. The conceptual model detailed in Chapter 3 becomes encoded into the changelog and the graph resulting from the log can be extracted automatically.

The resulting graph provides a structure upon which a flow may be calculated. This becomes an alternative method to determine the provenance distance between two datasets with a much higher fidelity. As mentioned previously, the Change concept is meant to be sub-classed to provide more freedom to represent the particular change bridging the two versions. For example, the He Count entry for the Noble Gas Database changed its units from parts per million to cc STP of given gas specie per cc STP of the total gas. This would be better qualified as a unit change and would be associated with a certain weight in contribution to the total change to the dataset.

CHAPTER 5
DATABASE VERSIONING

CHAPTER 6
ONTOLOGY VERSIONING

CHAPTER 7
CONCLUSION

REFERENCES

[1] This is the first item in the Bibliography. Let's make it very long so it takes more than one line. Let's make it very long so it takes more than one line. Let's make it very long so it takes more than one line. Let's make it very long so it takes more than one line.

[2] The second item in the Bibliography.

[3] Another item in the Bibliography.

[4]

[5]