Towards Automatic Capturing of Manual Data Processing Provenance

Andreas Wombacher and Mohammad Rezwanul Huq

University of Twente,
7500 AE Enschede, The Netherlands
(a.wombacher, m.r.huq)@utwente.nl

Abstract. Often data processing is not implemented by a workflow system or an integration application but is performed manually by humans along the lines of a more or less specified procedure. Collecting provenance information during manual data processing can not be automated. Further, manual collection of provenance information is error prone and time consuming. Therefore, we propose to infer provenance information based on the file read and write access of users. The derived provenance information is complete, but has a low precision. Therefore, we propose further to introducing organizational guidelines in order to improve the precision of the inferred provenance information.

1 Introduction

Information in organizations is processed and aggregated by humans where partial information is either exchanged via email or shared on a workgroup application. Example manual data processing workflows are yearly reports involving the contribution of each department, financial statements of public companies which consist of multiple companies, or documents of the European Union (EU) requiring input of each member state. In all these cases, the applications used by the users are not provenance aware. Further, the quality of the resulting document depends upon the integration of the newest revisions of all inputs. However, since there are so many people involved and the exchange of information is unstructured e.g. via email, it is very hard to make an objective quality statement on the final report.

Provenance information could be used to establish whether all latest revisions actually have been considered for the final report or to issue pro-actively warnings if a report does not include a newer contribution. The characteristic of the addressed problem is that

- legacy systems are not provenance aware,
- multiple users and potentially multiple organizations are involved,
- application interfaces are file based, and
- the legacy systems are not integrated, i.e., files are exchanged between users.

In such a scenario, manual provenance acquisition is infeasible since it is too time consuming, and therefore too costly. Therefore, the aim is to explore automatic provenance capturing in manual data processing.

By analyzing the manual data processing the following observations can be made: First, information contributing to a document is often copied from another document. Thus, the source document must have been read/opened before the information can be copied and saved in the target document. Therefore, each document which has been read before a save operation is performed may have contributed to the target document. Provenance information derived based on this principle is very noisy since it contains many contributing documents, which are actually not contributing. This is because people often work on several tasks in parallel during a day, like e.g. reading emails, writing on a report, looking up information for your boss.

Second, users of ICT systems organize their information often in hierarchical structures like e.g. directories. Thus, for a certain task a user has to perform specific directories are relevant. Knowledge of this structure can be used to reduce the noise , i.e., improve the accuracy, of the provenance information, by eliminating open documents which are not in the right parts of the structure.

Third, several revisions of documents are issued over time, where the revision number may or may not be explicated in the file name. However, it is essential to distinguish the various revisions of a document to provide good provenance information.

Fourth, documents are exchanged between users in various ways. Documents can be sent via email either as an attachment or as a piece of text in the email. Another possibility is using a memory stick or a CD to personally deliver the document. Thus, to acquire provenance information on the exchange of documents it is necessary to restrict the means of exchanging documents.

In this paper, an approach to automatically capture provenance information for manual data processing is proposed. The basic idea is that all documents, which have been read by a user, may have contributed to a document saved later on. Since this derived provenance results is rather imprecise it is proposed to facilitate knowledge of the data processing workflow and the organization of the documents by the user to increase the precision. In particular, it is proposed to introduce organizational directives, i.e., guidelines for the user on how to organize information relevant to the data processing workflow. The more strict these guidelines are the higher the precision that can be achieved. However, strict guidelines lower the degree of freedom of a user to organize 'her' data and therefore may result in non complying users effecting the data quality of captured data. The proposed approach is based on a WebDAV infrastructure which supports versioning of files and documents read and write access to files. The proposed approach has been implemented and evaluated on the paper writing of this paper.

In the following the use cases are introduced (Sect 2) before related work is discussed (Sect 3). The approach is presented on a conceptual level (Sect 4) while a more technical view on the derivation of provenance information is provided in Sect 5. The paper ends with an evaluation of the proposed approach (Sect 6) and conclusions (Sect 7).

2 Use Cases

The use case is a data processing workflow of writing this technical paper with multiple authors in Latex. After initializing the project, two authors are writing together on the same paper. The used files are stored on a network drive and Subversion (a version control system) is used. A BPMN notation of the data processing workflow is depicted in Fig 1[1].

In the workflow the following data processing tasks can be executed in arbitrary order with an arbitrary number of repetitions:

- creation, update and conversion tasks on figures and graphics files
- creation and update tasks on bibtex and latex files
- pdflatex task, for creating a pdf file of the paper
- bibtex task, for creating the bibliography file (bbl file) related to the paper

There are two people involved in the paper writing process, which makes it interesting to determine whether a specific revision of a generated pdf file contains all the latest file revisions. Especially whether all figures have been properly converted before executing the pdflatex task. Further, it can be inferred whether in a specific pdf revision of the paper all indices and the bibliographic information is up to date, since this requires the following task sequence: pdflatex - bibtex - pdflatex -pdflatex.

3 Related Work

Automatic collection of provenance information is often applied in e-science workflow systems, like e.g. Kepler [1] or Taverna [2]. Most systems even rely on exchanging data via files. In previous work [3] we investigated inference of provenance information for stream processing workflow system using a temporal model. However, the workflow system is executing a workflow and all involved tasks are executed automatically which is a major difference to our requirements.

Provenance information is also collected in closed systems like e.g. a data warehouse [4] or a relational database [5]. The level of granularity in these approaches varies between fine-grained and coarse-grained data provenance [6]. The fact that the provenance acquisition is limited to the system makes it infeasible for our scenario.

Automatic collection of provenance information focusing on the exchange of information is addressed in various ways. In [7] provenance information is captured by monitoring a service bus where the invoked services must not be provenance aware. Although this is quite close to the scenario at hand, in this case all processing has to be implemented as a service, which is not the case.

[1] Created with `http://oryx-project.org/`

The approach in [8] records file manipulation operations including system variables and their changes. However, this approach is limited to a single computer since it is woven into the kernel of a linux system which differs significantly from our scenario. This is maybe the closest match to our approach. In [9] the authors automatically collect provenance information based on events recorded by browsers. However, the approach is limited to a single application.

Systems for storing provenance information are e.g. Tupelo2 [10] or Karma2 [11]. These systems provide an into store provenance information and provide means to query the data. The provenance information derived by the proposed approach could be stored in such a system. Further, the acquired provenance information could be made accessible in different provenance models like e.g. the Open Provenance Model (OPM) [12] or the value centric model (TVC) [13]. However these are just alternative representations of the derived provenance data, while the focus of this paper is on acquiring the provenance data rather then how to represent them.

4 Approach

The aim is to infer provenance information from a manual data processing workflow execution without the user providing any information and the legacy applications not being provenance aware. Since files are used to exchange information between different tasks of the workflow, the approach is based on documenting data manipulations on files. Combining file manipulation information with data dependencies of the workflow allows to infer which revision of which file may have been contributed to a revision of a file written by a particular user.

The intuition is that data processing is based on zero or several input files producing one or several output files. In particular, all files which have been opened before the point in time a file is saved potentially contributed to the creation of the saved file. These derived provenance relations have a very low precision, i.e., are too broad especially when considering the amount of files opened in the coarse of a day at a desktop computer.

Therefore, we propose to facilitate knowledge of the data processing workflow to derive organizational directives, i.e., rules for the user performing the data processing activities to be able to associate files with specific tasks in the workflow. These rules specify how to handle files without technically enforcing them. The organizational directives are the basis to derive rules for inferring provenance information. Organizational directives are a mechanism of assigning a responsibility to the employee without technically enforcing the execution of the directive. Organizational directives are widely implemented in organizations, like e.g. you are not allowed to install software on your company laptop, you are not allowed to download copyright protected material, you are obliged to make backups or to encrypt your hard disc.

The approach is depicted in Fig 4. Based on the data processing workflow (lower left corner of Fig 4) the organizational directives are derived, of which inference rules for provenance relations can be derived. Further, continuously

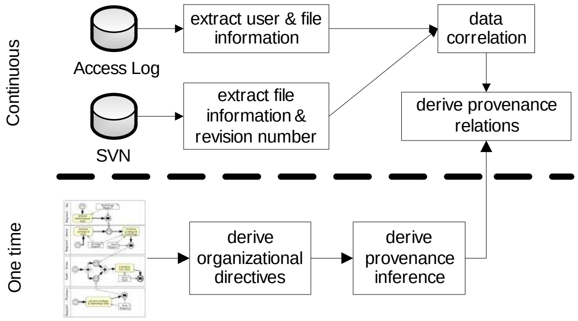
Continuous
Access Log
SVN
extract user & file information
extract file information & revision number
data correlation
derive provenance relations
One time
derive organizational directives
derive provenance inference

Fig. 1. Prototype data processing

automatically acquired file information from the SVN log and the access log are correlated. These correlated data are the basis to apply the derived provenance inference rules to derive provenance relations. In particular, a relation exists, if a file with revision v_s has been written after a file with revision v_r has been read and $v_r < v_s$ assuming a global versioning scheme. In the following the basic derivation of provenance relations (continuous part) and the creation of organizational directives (one time part) of the approach are discussed in more detail after some generic directives are introduced.

4.1 Infrastructure to Capture User Behavior

The proposed approach is based on a Web based infrastructure for file storage and versioning, in particular, a WebDAV infrastructure. WebDAV is an HTTP based protocol for managing content on a web server, i.e., accessing, writing and moving files. From a usability point of view, WebDAV has the advantage that there exist clients for many operating systems. Many clients seamlessly integrate into the operating system, i.e., the WebDAV server appears like a mounted network drive and therefore is intuitively usable also for basic ICT users.

Further, since the WebDAV protocol is based on HTTP it easily supports distributed and cross-organizational scenarios, like the motivating examples in Sect 2. Furthermore, WebDAV can also be used for information dissemination. From a provenance capturing point of view, WebDAV has the advantage that it supports logging of file read and save operations via the underlying web server access log and it supports versioning via an underlying Subversion system configured as auto-commit. Thus, the versioning is a 'version on every write', which introduces overhead providing optimization options in future work.

A schematic overview of the infrastructure is depicted in Fig 5. The center is the WebDAV server combined with a Subversion (SVN) for versioning. Access rights are managed by a User Management system (in our case an LDAP server[2])

[2] `http://www.openldap.org/`

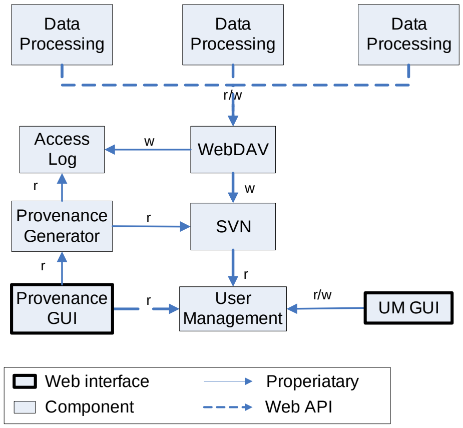

Fig. 2. Data Usage relations

with a User Management Graphical User Interface (UM GUI). WebDAV combined with SVN server is a module[3] of the Apache HTTP Server [4]. The HTTP server provides an access log containing the read (GET), write (PUT), move (MOVE), and delete (DELETE) operations applied to files on the server. Further, the SVN maintains internal logs on write (Add, Modify), delete (Delete) and move (combined Add and Delete) operations. Both sources are required to capture the user behavior and therefore have to be correlated (see Sect 5.2). The correlated information is used by the provenance generator to continuously derive provenance information. The provenance generator uses generic and organizational directives to infer provenance relations between various files and revisions of files. A provenance GUI[5] allows to query and visualize the provenance information. It is based on AJAX requests on the stored provenance information and the InfoVis toolkit [6]. Users performing the data processing activities use files from the WebDAV server and manipulate them with their preferred data processing application.

4.2 Generic Organizational Directives

The proposed approach is based on observing the handling of files. Thus, some generic directives have to be provided to ensure that the handling of files can be observed at the first place. Similar to directives in organizations that all important information has to be saved on a network drive because local disks are not backuped, a data security directive is introduced requiring the user to save data on a mounted network drive, i.e., a WebDAV server.

[3] http://httpd.apache.org/docs/2.0/mod/mod_dav.html
[4] http://httpd.apache.org/
[5] accessible at http://www.sensordatalab.org/offline_provenance_web/
[6] http://thejit.org/

Directive 1 (Data Security) *Users must save all files related to a data processing workflow on the network drive. Thus, it is not allowed to store files related to the data processing on a local disc.*

This directive is necessary, since local file systems can not be monitored with regard to file handling that easily from outside the computer. Thus, it is required to prevent users to store files locally which are related to the data processing workflow execution.

A standard directive in organizations is that login and password information must not be shared between different users. It must always be possible to identify a responsible user for any observed action in the infrastructure. In case of users being sick or on vacation, a delegation mechanism must be in place. The same applies for performing activities in the data processing workflow resulting in the following directive.

Directive 2 (Delegation) *In case of vacation or illness the execution of activities must be delegated.*

In many organizations you have a clean desk policy, which means that at the end of the day all business relevant documents must be removed from the desk. Translating this into the digital world it means that the desktop computer has to be switched off at the end of the day, which is also in the context of green IT getting more attention. This results in the following directive:

Directive 3 (Clean Desk Policy) *The user must shut down his/her computer at the end of the day, i.e., there are no open files or applications active anymore.*

Provenance relations are based on the observed reading and saving of files. Since it can not be observed by WebDAV when a file is closed again, the clean desk policy directive enforces that at the end of a day all files are closed. Thus, this is a synchronization point for deriving provenance relations by excluding files opened at previous days.

4.3 Provenance Relations

The generic directives are the basis to infer provenance relations based on the WebDAV commands. A provenance relation is a relation between a read or save operation of a file A and a save operation of a file B, where the read or save operation of file A is performed before the save operation of file B. The order of the operations can be determined by the timestamps at which the operations are observed and on the associated revision numbers.

Provenance relations are transitive, i.e., the provenance graph can be reduced to the necessary provenance relations. A provenance relation between file A and C can be ignored iff there is a provenance relation between file A and B and a provenance relation between B and C.

Due to directive 3 only read operations which have been performed at the same day as the write operation are considered. Further, according to directive

2 each file access is associated with a specific user, thus, provenance relations require that the files are read and saved by the same user.

With regard to the motivating scenario in Sect 2.1 the observed file access depicted in Fig 2 results in provenance relations enumerated in Tab 1. As an example, the modification of the file F1 by Ida results in a provenance relation (ID 11) of reading file F1 in revision 1 to save a file with the same name resulting in revision 2 of the file.

ID	**read file**	**read revision**	**saved file**	**saved revision**	**user**
1	F1	1	F2	1	Jenny
2	F1	1	F3	1	Jenny
3	F2	1	F3	1	Jenny
4	F1	1	F2	2	Jenny
5	F2	1	F2	2	Jenny
6	F3	1	F2	2	Jenny
7	F1	1	F3	2	Jenny
8	F2	1	F3	2	Jenny
9	F2	2	F3	2	Jenny
10	F3	1	F3	2	Jenny
11	F1	1	F1	2	Ida
12	F3	2	F5	1	Anna
13	F4	1	F5	1	Anna
14	F'	1	F5	1	Anna

Table 1. Provenance relations of the file example in Fig 2

These provenance relations explicate some artifacts, which have to be resolved by user specific organizational directives.

- The file F' is not related to the data processing workflow at all and therefore has to be excluded (provenance relation 14).
- There is a provenance relation (ID 6) between file F3 revision 1 and file F2 revision 2, which is inconsistent with the data processing workflow assuming that file F2 is the output of the 'retrieve ecological data' task and F3 is the output of the 'combine ecology & hydrology' task.
- The provenance relation (ID 8) between file F2 revision 1 and file F3 revision 2 is implicitly included by transitivity from the provenance relation (ID 5) between file F2 revision 1 and file F2 revision 2 and the provenance relation (ID 9) between file F2 revision 2 and file F3 revision 2.

In the following user specific organizational directives are derived from the data processing workflow to infer that provenance relations 14 and 6 are excluded.

4.4 User specific Organizational Directives

The goal of the organizational directives is to increase the precision of data provenance by excluding observed provenance relations which are actually irrelevant. The directives are derived from the data processing workflows a specific user is involved in.

The workflow is based on swimlanes, where each lane is associated with a user or role[7]. Thus, to determine a set of directives, it is necessary to understand which activities in which data processing workflows a user is responsible for and whether their execution overlaps. For example the appearance of file F' in Tab 1 in provenance relation 14 indicates that Anna is involved in at least one task of another workflow.

Since provenance relations are based on sets of read files with a single point in time when files are certainly closed again (see Directive 3), additional mechanisms are required to determine the precise inference of data provenance. One way is to apply some directives on the hierarchical structure of data on the network drive. Such a structure enables the exchange of information between different organizations and to control access rights. In particular, one possible approach is to establish a basic directory structure where each directory is associated to a particular task in a data processing workflow, as it is known e.g. from group ware solutions like BSCW or groove. An alternative is to use filters based on regular expressions exploiting file naming conventions or specific file extensions being unique for the output of an task.

However, the derived directives might be not specific enough or too complicated to implement resulting in imprecise provenance information. The challenge is to find a good balance between usability of organizational directives and the targeted data provenance precision.

With regard to the motivating scenario in Sect 2.1 and the inference in provenance relations enumerated in Tab 1 the following directives should be instantiated:

Directive 4 (EU water directive: State - region1 directive) *All files exchanged between region1 and state are stored in directory region1.*

Directive 5 (EU water directive: State - region2 directive) *All files exchanged between region2 and state are stored in directory region2.*

Directive 6 (EU water directive: 'combine region data' directive) *The directory state is exclusively used for files related to the 'combine region data' task of the 'EU water directive' data processing workflow.*

The above three directives indicate that only provenance relations are relevant, where the saved or read file is in directory *region1* or *region2* or *state*[8] and the saved file is in directory *state*. As a consequence we assume that files F3 and F4 are in directory region1 and region2 respectively. We further assume that file F' is in a different directory since it is not related to an exchange with any region nor is it related to the 'combine region data' task. Thus, due to these directives, the provenance relation 14 can be excluded.

To discard the provenance relation 6 the following directive is required

[7] Roles are at run-time associated with a user again.

[8] This directory is included due to transitivity property of provenance relations.

Directive 7 (EU water directive: region1 directive) *All filenames saved as a result of the 'combine ecology & hydrology' task start with 'combined_'. No other files are allowed to be named in a similar way.*

This directive allows to uniquely associate file F2 and F3 to activities 'retrieve ecological data' and 'combine ecology & hydrology' performed by Jenny. Since task 'retrieve ecological data' is a pre-condition for task 'combine ecology & hydrology' files of the latter one can not contribute to the first one. Thus, it can be derived that file F3 is not contributing to file F2. As a consequence it can be inferred that provenance relation 6 can be excluded.

Based on these directives and the eliminated provenance relations 6 and 14 the precision is now optimal, i.e., has a value of 1.

5 Provenance Relation Derivation

The previous section discussed the more conceptual view on the approach while in this section the focus is on a more technical view on how the provenance information is derived.

The provenance information forms a provenance graph, which consists of vertices and edges. A vertex is either an access log entry or a SVN log entry. The edges represent the provenance relations. In this paper three classes of provenance relations are distinguished: provenance relation correlating SVN and access log entries, SVN step relations, and relations derived from directives. In an access log contains read (GET), write (PUT), move (MOVE), and delete (DELETE) operations(WebDAV commands) applied to files. Further, the SVN internal logs contains write (Add, Modify), delete (Delete) and move (combined Add and Delete) entries.

All relation classes are discussed in the following subsections after a discussion on observations of file handling using WebDAV clients.

5.1 File Handling Observations

The manipulation of files via a WebDAV client and documenting them in SVN and access logs is not as straight forward as initially expected. In particular, the following observations can be made:

The first observation is that adding and updating files is realized in different combinations of WebDAV commands by different applications.

The second observation is that in the access log the WebDAV MOVE command documents only the source filename of the move, but not the destination filename. However, by using WebDAV with autocommit each change on a file results in a new revision number, thus, only a MOVE command results in a delete and an add SVN log entry with the same revision number. This can be facilitated to correlate SVN and access log entries.

The third observation is that a WebDAV DELETE command removes a file from the SVN repository. After the command is executed the file is not in the repository anymore, thus, the revision number of the removal of the file requires an extra query to the SVN log.

The fourth observation is that reading a file is only documented in the access log. Which revision of a file has been read has to be inferred from the state of the SVN and the size of the file read.

The final observation is that the correlation between SVN and access log entries can not be based on time, since the timestamps recorded by the SVN and the access log are points in time when the event has been recorded in the corresponding system. Thus, the timestamp of the access log entry is always before the entry in the SVN log since the HTTP request is first processed by the HTTP server which forwards the request to the WebDAV and therefore to the SVN. However, in case of two fast subsequent operations, it is possible that the access log has two entries before any entry is recorded in the SVN log. Thus, it is sometimes hard to infer the correlation of access to SVN log entries.

5.2 Data Correlation and SVN Step Relations

To determine the correlation between SVN and access log entries the corresponding sequences of events have to be synchronized. I.e., the sequence of WebDAV commands PUT, DELETE, MOVE and GET documented in the access log must be correlated to Add, Modify, and Delete commands in the SVN log. Be aware that GET WebDAV commands do not have a correspondence in the SVN log since read access does not change revisions of the SVN. The correlation is based on the fact that access log entries always precede corresponding SVN log entries. Besides the sequence of events, the time difference between subsequent events, the user name, and the file size are used to infer the correlation between access and SVN log entries.

A simplified version of the correlation algorithm is depicted in Alg 1. The token variable indicates whether the SVN entry can be correlated to an access log entry. A correlation is possible if the SVN entry is either a Delete or an Add entry (line 3). Further, a correlation is possible if the SVN entry is a Modify entry and the previous entry has been older than 2 seconds (line 4). The two second bound is based on the observation that an incremental upload of a file to the SVN resulted in a new revision approximately every second in our test system.

If an entry can not be correlated to an access log entry (line 7), than it is inferred that an incremental upload is occurring. Incremental updates are documented as a provenance relation between SVN log entries called SVN step relation (line 9). Otherwise, Add and Modify entries are correlated with PUT entries (line 11) and Delete entries are correlated either with DELETE (line 12) or with MOVE entries (line 13).

Since the file access is not documented in the SVN log entries, a second loop is executed on the access log entries (line 14). In particular, all GET entries are selected (line 15). For each GET entry the corresponding SVN entry is inferred (line 16) and the entries are correlated (line 17). Fig 6 depicts a subset of the example depicted in Fig 6. You can see Add SVN entries are correlated with

```
token=true;
forall the SVN entries do
    if Delete or Add then token=true;
    if Modify and size>0 and time difference to previous event<2sec then
        token=true;
    if not token then
        ignore entry for correlation;
        document SVN step relation with previous event;
    else
        if Add or Modify then correlate to PUT;
        if Delete then
            correlate to DELETE;
            if not possible then correlate to MOVE;
forall the access log entries do
    if GET then
        find SVN entry preceding the GET with same filename and size;
        correlate entries;
```

Algorithm 1: Simplified Data Fusion

PUT entries at the access log. Subsequent Modify entries are not correlated to access log entries, but are connected to preceding entries via the SVN step relation. Please be aware that the Modify SVN entry of revision 5 is correlated to a PUT entry since the time to the preceding entries is more than 2 seconds.

An interesting case is the *GET(F3,Anna)* entry which is properly correlated with revision 6 of the SVN although the update in the SVN occurs after the read access has been issued by Anna. The right association is made since according to the sequence of the access log there has first to be an update on the SVN log sequence before the read access can be performed. The distinction between revision 5 and 6 is made based on the file size retrieved by the read access. The derived non-transitive relations are depicted in Fig 6.

5.3 Directive Provenance Relations

The last type of relations are the ones derived from organizational directives. This type of relation is inferred between access log entries only. As outlined in Sect 4 all files read at the same day as another file is saved are potentially contributing to the saved file. The relation is labeled by the associated task in the data processing workflow. The organizational directives can be translated into constraints on these potential provenance relations mainly by constraining the files locations and filenames of sources and targets of a provenance relation.

For example Directives 4-6 as discussed in Sect 4.4 determine that only files in directory region1, region2 and state are relevant for the 'combine region data' task. Thus, files of other directories do not contribute to any provenance relation associated to this task.

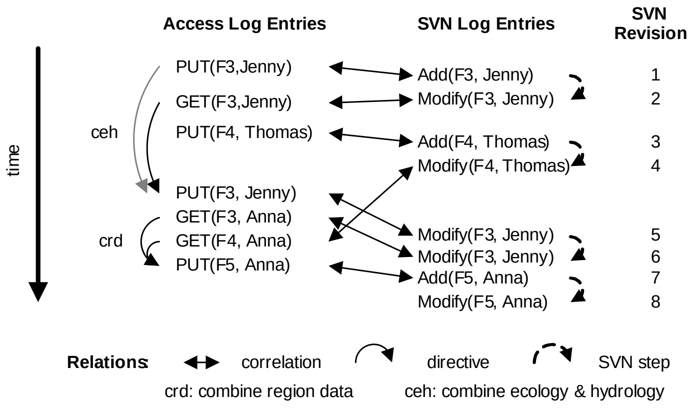

Fig. 3. Relations in a subset of the file example in Fig. 2

Directive 7 further constraints the derivation of provenance relations by stating that all files resulting from task 'combine ecology & hydrology' have a filename which starts with 'combined_'. Since the 'combine ecology & hydrology' task provides the input for the 'combine region data' task, all files from directory region1 contributing to files in state directory start with 'combined_'.

As a result two provenance relations labeled 'crd' remain as depicted in Fig 6. These two relations correspond to relations 12 and 13 in Tab 1.

Another provenance relation associated to task 'combine ecology & hydrology' is depicted in Fig 6 corresponds to relation 10 in Tab 1. It relates the first occurrence of the PUT(F3, Jenny), the GET(F3, Jenny) and the second occurrence of the PUT(F3, Jenny) command. These relations are inferred from directive directive 6 and 7. Since it corresponds to the right filename and is in the right directory. The directive is not clear whether the handling of file F3 is incremental or absolute, i.e., whether the content of previous versions of the file is re-used like e.g. by writing a letter, or whether the previous content is not relevant like e.g. with a file conversion from eps to pdf. In case of the latter the directive is imprecise and has to be reformulated. The grey representation of the provenance relation indicates that this is a transitive relation which will be reduced next.

5.4 Transitive Reduction

The aim is to get a minimal provenance graph, therefore a standard transition reduction algorithm is applied like e.g. [14]. As a consequence it is possible to remove the relation between the first occurrence of PUT(F3, Jenny) and the second occurrence due to the following transitivity:

PUT(F3, Jenny) $\overset{correlated}{\leftrightarrow}$ Add(F3, Jenny) revision 1 $\overset{SVNstep}{\rightarrow}$ Modify(F3, Jenny) revision 2 $\overset{correlated}{\leftrightarrow}$ PUT(F3, Jenny).

The resulting minimal graph based in the provenance relations is used by an active component to detect the user interface to answer provenance queries as well as actively detecting provenance inconsistencies.

6 Evaluation

To evaluate the proposed approach the use case described in Sect 2.2 has been applied on the writing of this paper. The idea was to experience personally the effects of network delays caused by the WebDAV and the 'inconveniences' of organizational directives.

The scenario described in Sect 2.2 is a data processing workflow of writing a scientific paper using latex, where figures are usually created in Microsoft Visio, which are then converted to Enhanced Windows Metafile (emf) files, and further into Encapsulated PostScript (eps) files. The pdflatex command uses a library to convert the eps figure files into pdf figure files, which can then be used in the resulting pdf file.

6.1 Organization Directives

Based on the data processing workflow the following organizational directives can be defined:

Directive 8 (Scientific paper writing) – *All bibtex files are located in a bib subdirectory of the project root directory.*

- *All pictures and figures are located in a pics subdirectory of the project root directory.*
- *All latex files, project files and auxiliary files are located in the project root directory.*
- *Figures may require a conversion of file formats. The source figure filename is a prefix of the target figure filename.*
- *An update of a bibtex file only depends on the old bibtex file, i.e. the filename of source and target are equivalent.*
- *The execution of a pdflatex command reads data from all project directories and writes a pdf file in the project root directory.*
- *The execution of a bibtex command reads from the project root directory and the bib directory and writes a bbl file in the project root directory.*

Please be aware that the generic organizational directives (directives 1-3 in Sect 4.2) are also applied although they are not discussed in detail again.

These organizational directives are not hard on us since we use the same way of structuring a paper writing project for years already. Our guess is that for many data workflows this is similar, since people tend to organize their data rather on content and topics than on time.

6.2 Overhead

The paper is based on two latex files, which uses three style files contained in the WebDAV. There are six bibtex files used and four image files with their corresponding emf,eps and pdf files. Further there are two BPMN files directly available as pdf files. The processing uses 4 auxiliary files and writes a single pdf file.

The overhead perceived during this experiment is clearly experienced due to the network delay of using the WebDAV compared to a local disc. The factor is about 1000 for my ADSL line at home. In particular, for a pdflatex command 840kB are read and 377kB are written, thus, in total 1.2MB are transferred. Further, for a bibtex command 250kB are read and 3kB are written, thus, in total 253MB are transferred. So far we have approximately 120 executions of the pdflatex command during the complete paper writing process. As a conclusion, since a build is not performed that often, the overhead is effecting our working experience only marginally.

6.3 Quality

Quality assessment is problematic since there is no ground truth. As a consequence the only possibility is to manually assess the quality of the inferred provenance information by inspection. The manual inspection does not show any missing provenance relations. Sometimes additional provenance relations are reported, which are e.g. artifacts of file transfers (temporary files). From our inspection we have not found cases with missing information. The precision depends on the organizational directives, which can be adjusted to the required level of precision as discussed before. Please be aware that the current alternative is no provenance information at all.

7 Conclusion

In this paper, a provenance capturing approach for manual data processing workflows involving provenance un-aware legacy systems is proposed. Further, the derivation of different classes of provenance relations is discussed. It has been argued that the recall of the proposed approach is very high while the precision depends on organizational directives, i.e., constraints on handling files as a basis for deriving provenance relations.

Future work should address the currently used 'version on every write' approach to minimize the versions used. A further topic is instead of deriving organizational directives to observe directives and assess the quality of the data handling as applied by the users for deriving provenance relations.

References

1. Ludascher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E., Tao, J., Zhao, Y.: Scientific workflow management and the Kepler system. Concurrency and Computation: Practice and Experience **18** (2006) 1039–1065

2. Oinn, T., Addis, M., Ferris, J., Marvin, D., Greenwood, M., Carver, T., Pocock, M., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics **20** (2004) 3045–3054
3. Huq, M.R., Wombacher, A., Apers, P.M.G.: Facilitating fine grained data provenance using temporal data model. In: Proc 7. Intl Workshop on Data Management for Sensor Networks, DMSN, ACM (2010) 8–13
4. Cui, Y., Widom, J.: Lineage tracing for general data warehouse transformations. VLDB Journal **12** (2003) 41–58
5. Szomszor, M., Moreau, L.: Recording and reasoning over data provenance in web and grid services. In: On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. (2003) 603–620
6. Simmhan, Y.L., Plale, B., Gannon, D.: A survey of data provenance in e-science. SIGMOD Rec. **34** (2005) 31–36
7. Allen, M.D., Chapman, A., Blaustein, B.T., Seligman, L.: Capturing provenance in the wild. In: IPAW. Volume 6378 of LNCS., Springer (2010) 98–101
8. Seltzer, M., Muniswamy-Reddy, K.K., Holland, D.A., Braun, U., Ledlie, J.: Provenance-aware storage systems. In: Proceedings of the USENIX Annual Technical Conference (USENIX'06). (2006)
9. Margo, D.W., Seltzer, M.I.: The case for browser provenance. In Cheney, J., ed.: Workshop on the Theory and Practice of Provenance, USENIX (2009)
10. Futrelle, J.: (Tupelo server) http://tupeloproject.ncsa.uiuc.edu/.
11. Simmhan, Y.L., Plale, B., Gannon, D.: Karma2: Provenance management for data driven workflows. Intl J of Web Services Research **5** (2008) 1–23
12. Moreau, L., Freire, J., Futrelle, J., McGrath, R., Myers, J., Paulson, P.: The open provenance model: An overview. Provenance and Annotation of Data and Processes (2008) 323–326
13. Misra, A., Blount, M., Kementsietsidis, A., Sow, D., Wang, M.: Advances and Challenges for Scalable Provenance in Stream Processing Systems. Provenance and Annotation of Data and Processes (2008) 253–265
14. Aho, A.V., Garey, M.R., Ullman, J.D.: The transitive reduction of a directed graph. (SIAM J. Comput.)