Preserving Workflow Reproducibility: The RePlay-DH Client as a Tool for Process Documentation

Markus Gärtner[1*] Uli Hahn[2] Sibylle Hermann[3*]**
[1]Institute for Natural Language Processing [2]Communication and Information Centre
[3]University Library [*]University of Stuttgart [**]Ulm University
markus.gaertner@ims.uni-stuttgart.de, uli.hahn@uni-ulm.de
sibylle.hermann@ub.uni-stuttgart.de

Abstract

In this paper we present a software tool for elicitation and management of process metadata. It follows our previously published design idea of an assistant for researchers that aims at minimizing the additional effort required for producing a sustainable workflow documentation. With the ever-growing number of linguistic resources available, it also becomes increasingly important to provide proper documentation to make them comparable and to allow meaningful evaluations for specific use cases. The often prevailing practice of post hoc documentation of resource generation or research processes bears the risk of information loss. Not only does detailed documentation of a process aid in achieving reproducibility, it also increases usefulness of the documented work for others as a cornerstone of good scientific practice. Time pressure together with the lack of simple documentation methods leads to workflow documentation in practice being an arduous and often neglected task. Our tool ensures a clean documentation for common workflows in natural language processing and digital humanities. Additionally, it can easily be integrated into existing institutional infrastructures.

Keywords: reproducibility, metadata, documentation

1. Introduction

Several decades ago Claerbout and Karrenbach (1992) formulated the idea that in a research environment of ever increasing digitization the published articles are only advertisement for the actual scholarship. The scholarship itself then also includes all the scripts, tools, data and additional information that was used to create the results. This view is even more pronounced today, considering how already minor adjustments or changes in intermediary or preprocessing steps in typical natural language processing (NLP) tasks can have significant impact on the outcome of down-stream tasks, as has been exemplified by Elming et al. (2013). While it is not an ultimate guarantee for reproducibility, thorough documentation of workflows certainly aids in being able to reproduce results, evaluate the suitability of resources for a certain task or at least increases ite)usability for others and even for oneself.

In stark contrast to this, day-to-day research often exhibits a serious neglect of documentation efforts when it comes to minor details in a project's workflow. Reasons for this include for example the competitive and time pressure that largely dominates today's research practice. On the road to a publishable end-result it is easy to omit workflow documentation, especially if it requires a substantial amount of extra effort and there is no apparent gratification for it. If at all, such documentation is typically created retrospectively at a later time, where the risk that some details of the workflow might have already been lost is high.

The matter is further complicated by the diverse nature of workflows in the field of NLP, computational linguistics (CL) or digital humanities (DH). They draw from a vast pool of available resources and tools to intermix strictly automatized steps with purely manual work or any form of hybrids between those two. Resulting workflows can also be linear, branched or highly iterative with only small differences between recurring steps. They can further involve multiple persons working in an collaborative effort on the same data and joining results. Looking at this complexity one cannot emphasize enough the importance of providing detailed workflow and provenance information for published resources and results.

We previously proposed our design of a software tool (Gärtner et al., 2018) and an associated workflow metadata scheme to fill this need for a way of comfortably collecting process documentations. Our approach addresses documentation already during an active workflow top of the version control system Git[1]. It models workflows in a very generic way and is suitable to describe automatic steps as well as manual work. In order to increase sustainability our design uses several standards and established practices and offers simple interfaces for publishing and/or archiving individual stages within a workflow.

In this paper we focus on the applicability of our tool to typical tasks in NLP and CL. We show how common tasks fit into our metadata scheme and how the tool can be integrated into existing institutional infrastructures.

2. Related Work

For the task of documenting research processes we essentially distinguish two conceptually different types of systems, namely the ones used for workflow management and those for workflow tracking.

Used for setting up (and often even executing) workflows as collections of interdependent steps, *workflow management systems* (WMSs) contribute to the overall documentation effort prior to or during an active workflow. In contrast, a system for *workflow tracking* represents a more reactive approach and provides the documentation during or after a workflow. Looking at available WMS, the list of (commercial) systems for general-purpose or enterprise use is extensive. Their usability for specialized research

[1]https://git-scm.com/

workflows however is usually rather limited. In the context of certain research fields customized WMS instances have emerged, that allow researchers to build and execute workflows from catalogs of predefined analysis or processing steps. Popular web-based examples for this include *GenePattern*[2] (Reich et al., 2006) for genome research or in the field of NLP the *Language Application Grid* (Ide et al., 2016). The latter is an application of the *Galaxy* (Afgan et al., 2016) platform for biomedical analyses that has been tailored to the field of natural language processing. For local execution of NLP pipelines there have been approaches using the *Apache UIMA™project*, such as the *DKPro Core* (Eckart de Castilho and Gurevych, 2014) framework. With such systems the workflow documentation is very much covered by the actual description of the workflow setup if the system allows to export this kind of information.

When looking at the topic of workflow tracking, the core actions addressed often boil down to either monitor the data flow to/from processes or the physical changes to a designated set of resources. Tools like *YesWorkflow* (McPhillips et al., 2015) are examples of the first type and offer the ability to annotate data flows and operations on the code level. The implicit workflow can then be visualized based on the annotated information. The second type of workflow tracking systems are version control systems like Git[3] or Apache Subversion (SVN)[4], which are common in software development. They are intended for documenting complex collaborative development workflows exist in various flavors, such as centralized (SVN) versus decentralized (Git).

While the elaborate solutions listed above do provide a wide range of approaches to workflow documentation, they often are not applicable to a given research workflow (e.g. NLP/CL/DH): Their general focus on executable workflows makes them incompatible with workflows that also contain manual steps, such as annotation or curation tasks. Solutions such as version control provide the means of tracking very fine-grained changes, but leave out dedicated mechanism to formally describe what actions were conducted to cause those changes. Another important aspect of documentation systems regarding their usability is the level of technical expertise they require. This is especially true for technically less-skilled users, where a complex system is likely to act more as deterrent than encouragement. As a result it is not uncommon for researchers to document their workflows manually by means of a local Word or Excel file.

Besides approaches to documenting the actions in a workflow, it is also important to consider existing infrastructures that deal with metadata for the objects used in the workflow. As such initiatives like the *LRE Map* (Calzolari et al., 2010) or *CLARIN* (Hinrichs and Krauwer, 2014) and many others already provide wide coverage of metadata repositories for communities in computational linguistics and digital humanities. By no means a replacement of proper process documentation, linking to content of those infrastructures does already provide a valuable foundation to build on.

Also related to the topic of workflow documentation is the notion of resource provenance. Here the *PROV Family of Documents*[5] from the W3C Working Group provides models and exchange formats for describing provenance in a very expressive way. While not used as its native model for workflows, the RePlay-DH Client supports the PROV concept for the description of datasets that are exported.

Field	#	Description
Title	1	Short label of a step
Description	1	Human readable explanation of the actions performed
Person	0..n	Human subjects involved such as annotators or curators
Tool	0..1	Processing software used to generate the output
Input	0..n	All resources used to produce the output, including external things like annotation guidelines
Output	0..n	Resources generated or modified as result of the performed step
Properties	0..n	Custom metadata entries to store additional information in an organized (machine readable) way

Table 1: Overview of the top-level fields used in our process metadata scheme, their multiplicity and brief description.

3. Target Workflows

Workflows in RePlay-DH are assumed to be representable as directed acyclic graphs of individual steps and their dependencies. This is in line with observations regarding the most common workflow structures in Deelman et al. (2009). Our main targets are *data centered* workflows, hence the explicit `input` and `output` field in the metadata scheme described in the next section. However, it is noteworthy that the scheme is flexible enough to also model steps consisting of only title and description, e.g. ones that represent a cognitive progress in a workflow.

Under the data centered assumption, a single workflow step models the generation or modification of one or more resources by one or more actors[6], optionally with the aid of a software tool. The role of *actor* is implicitly assumed to be filled in by the user, but for situations with additional persons involved such a step can carry information to identify and describe those persons.

Naturally, the amount of individual actions that together form a complete workflow step is highly dependent on the actual workflow and we do not impose any direct constraints on the overall granularity of what can be recorded as a workflow step. We do however limit each step to contain at most one instance of processing software (see the `tool` field in Section 4). This is done to preserve proper information about individual input and output resources in the event of pipeline architectures where several tools are applied by the user consecutively.

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

[3] `https://git-scm.com/`

[4] `https://subversion.apache.org`

[5] `https://www.w3.org/TR/prov-overview/`

[6] We only count persons as actors in a workflow step, who perform manual work on the data. Merely executing an automatic processing tool or script does not suffice.

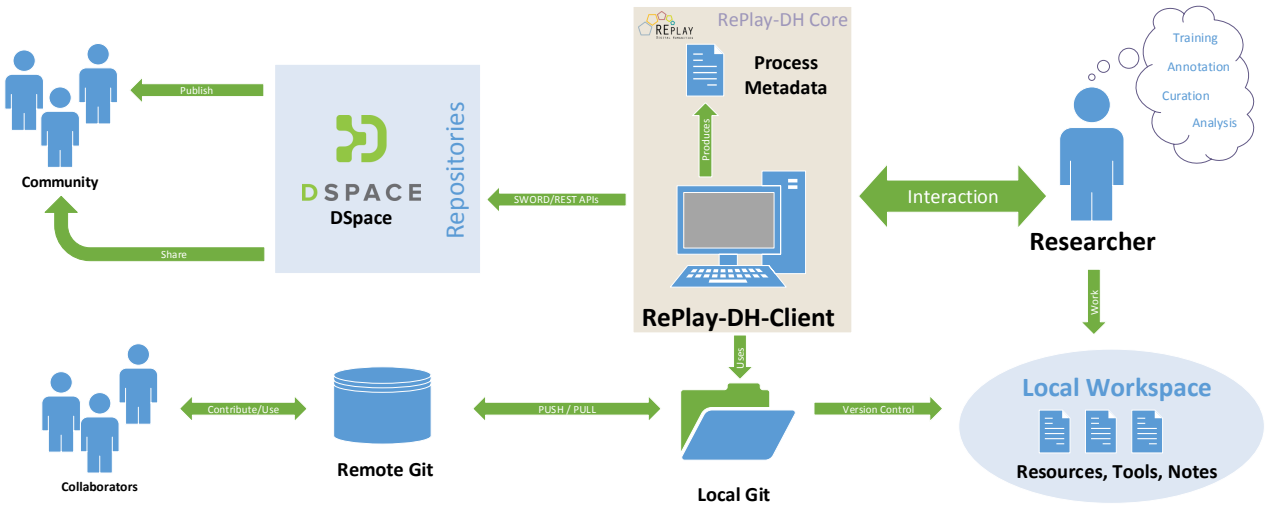


Figure 1: Overview of the RePlay-DH Client architecture and its interfaces to existing infrastructure components.

A recurring issue we noticed when documenting workflows involving very different processing tools, was the need to sometimes split resources into multiple fragments, to meet input requirements of certain tools. While not a problem for processing per se, it does however pose a challenge for workflow documentation. To avoid this, the RePlay-DH Client adds another organizational layer and lets the user group multiple physical resources (such as files) when creating process documentation, so that the amount of information within the metadata for a workflow step remains readily comprehensible.

4. Metadata

For the purpose of documenting workflows we distinguish between *process metadata* and *object metadata*. So far infrastructure initiatives in the NLP community have largely been focused on solutions for object related metadata, i.e. the detailed documentation of (finished) resources or tools in public repositories such as the CLARIN Virtual Language Observatory (VLO)[7] with its Component MetaData Infrastructure (CMDI)[8]. While some of those metadata schemes also provide the means to record information regarding the provenance or creation process of a resource, documenting that sort of workflow in detail is generally outside their scope.

Our approach is to model an entire workflow graph, including experimental steps or paths that led to dead-ends. It therefore features the individual workflow steps as central units and not only the one successful workflow path that resulted in the final output data. This is in contrast to classic approaches for documenting resource provenance such as PROV, which are focused on modeling provenance chains. We therefore designed a compact model and associated metadata scheme for recording process metadata in data centric workflows with the expressed goal of being able to build on the richness of existing object metadata infrastructures.

Table 1 gives a brief overview of the top-level fields for individual workflow steps in our metadata scheme. For a detailed explanation we refer to the original description in Gärtner et al. (2018). Of special importance in the context of this paper is the flexible mechanism used for identifying resources in our scheme and linking them to existing object metadata records. For this, every resource entry (`Person`, `Tool`, `Input`, `Output`) contains a collection of *typed identifiers*. An identifier consists of a *type* definition (similar to a namespace declaration) and the actual textual *id* itself, for instance `VLO-HANDLE` as type for entries in the VLO and the actual handle URL as id for a CMDI metadata record of the TIGER Corpus[9].

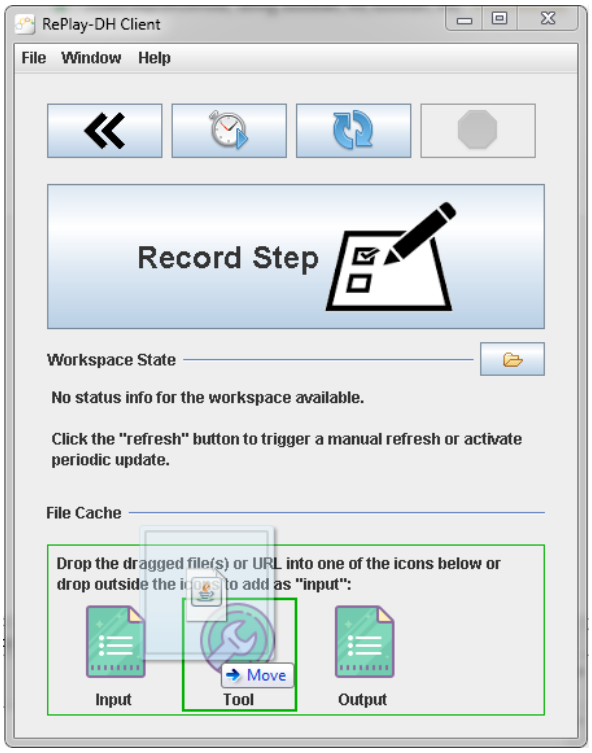
With this scheme our process metadata can link directly to object metadata entries in established repositories and reduce both redundancy and the overall effort required to describe resources in detail. For situations where no external repositories are available or needed, our client provides an integrated object metadata repository. This local repository can be used to store metadata records based on the Dublin Core scheme (Powell et al., 2005).

The RePlay-DH Client assists the user in the creation of metadata and uses JSON[10] for recorded workflow steps to serialize metadata into a representation that is stored alongside the data inside the underlying Git repository. We chose JSON for its simplicity, human readability (visible in Figure 3) and interoperability, which makes it easy to process exported instances of the process metadata with other tools or to extend the scheme for future needs.

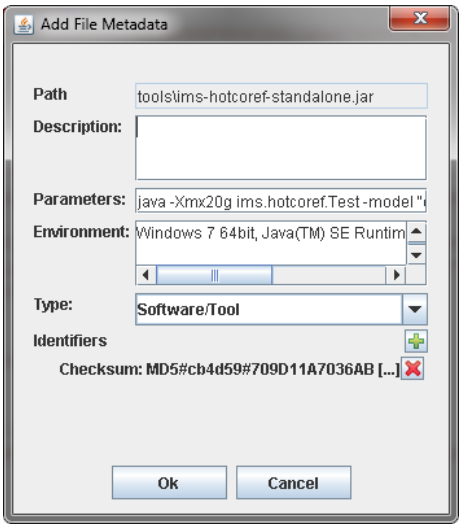
5. Architecture and Integration

The central product of our project is the RePlay-DH Client, the architecture and interfaces of which are depicted in Figure 1. Its main role is to bundle as many aspects of

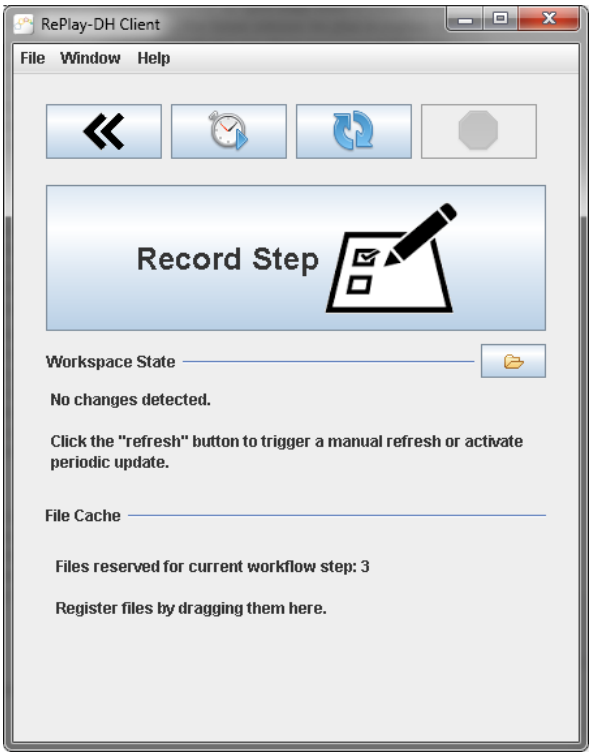
[7] `https://vlo.clarin.eu`
[8] `http://www.clarin.eu/cmdi`
[9] `http://hdl.handle.net/11022/1007-0000-0000-8E2D-F`
[10] `http://www.json.org/`



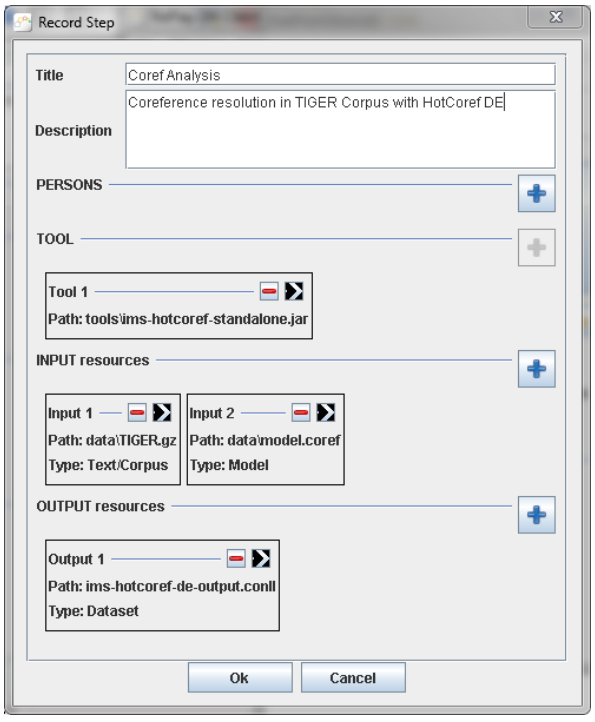
(a) Registering a resource as part of an ongoing workflow step via dragging a file onto the RePlay-DH Client interface.



(b) Recording metadata for the new resource.



(c) The RePlay-DH Client having three files registered as part of the current workflow step.



(d) Dialog for finally recording the active workflow step.

Figure 2: Phases in the process of registering resources (a) and their metadata (b) during a workflow step, marking them for later (c) and then finally recording the step itself (d).

```
{"title":"Coref Analysis",
  "description":"Coreference resolution on TIGER Corpus with HotCoref DE",
  "timestamp":"2017-12-27 17:13:12",
  "input":[
    {"systemId":"cbf13c01-d735-40e7-865c-477f4090e8c1",
    "type":"Text/Corpus",
    "identifiers":[
      {"type":"path","id":"data\\TIGER.gz"},
      {"type":"vlo-handle","id":"http://hdl.handle.net/11022/1007-0000-0000-8E2D-F"},
      {"type":"checksum","id":"MD5#7bca4b#452E036994C0F8E74D971D73861ADB2251"}]
    },
    {"systemId":"f73ad47c-8b4c-4afa-948a-dc5927a00261",,
    "type":"Model"
    "identifiers":[
      {"type":"path","id":"data\\model.coref"},
      {"type":"checksum","id":"MD5#1309#745412406E25E8B85EEFED6C58D1A8A574"}]
    }
  ],
  "tool":{
    "systemId":"53b0194f-d926-4cc1-b468-04feac22e404",
    "type":"Software/Tool",
    "environment":"Windows 7 64bit, Java(TM) SE Runtime Environment (build 1.8.0_161-b12)"
    "parameters":"java -Xmx20g -cp . ims.hotcoref.Test -model \"data/model.coref\" -in \"data/TIGER.gz\" -out \"
        ims-hotcoref-de-output.conll\" -cores \"4\" -lemmaBased -beam 20",
    "identifiers":[
      {"type":"checksum","id":"MD5#cb4d59#709D11A7036AB3778A3856E380564D1A6B"},
      {"type":"vlo-handle","id":"http://hdl.handle.net/11022/1007-0000-0000-8E69-B"},
      {"type":"path","id":"tools\\ims-hotcoref-standalone.jar"}]
  },
  "output":[
    {"systemId":"ff805df8-4e0e-43cf-8c25-d62cdfceeb45",
    "type":"Dataset",
    "identifiers":[
      {"type":"path","id":"ims-hotcoref-de-output.conll"},
      {"type":"checksum","id":"MD5#145e#EDC0555E82BDD9EF93E9A49CB0B52642ED"}]
    }
  ]
}
```


Figure 3: Example instance of process metadata for a single processing step in JSON. It describes an automatic analysis involving a coreference resolver as `tool`, a target corpus and trained model files as `input` and the result file as `output`. To conserve space, some values have been shortened and the default JSON layout has been condensed.

the workflow documentation process as possible in order to provide the researcher with a single optimized interface for that process. As mentioned in the introductory section, an omnipresent factor in today's research is time pressure. Therefore, a documentation tool's cost-benefit ratio (with 'cost' representing the effort required to use it) is most likely to be the decisive factor for its suitability. With this in mind our client is designed to be non-invasive and to minimize the documentation overhead, so that researchers can focus on their actual workflows. In its most basic configuration it functions without any external dependencies other than the libraries it ships with, making it very lightweight and flexible.

Git as foundation. Internally the client uses the popular and well-maintained JGit[11] library to put local workspace folders under version control and thereby monitor them. As a result no additional installation of any Git-related local software is required.

Each recording of a workflow step is wrapped into a Git commit and the associated process metadata for that step is stored in serialized form (JSON) as commit message. This provides a tight coupling of metadata and the observed physical changes of each workflow step within the Git commit graph.

As an added benefit of using a local Git repository comes also the possibility of connecting it to arbitrary remote repositories such as an institutional GitLab[12] instance. Use cases for this kind of interfacing are for example collaborative work or simply having an additional layer of backup available.

Incremental metadata construction. Delayed documentation, i.e. process metadata that gets created at a (significantly) later point in time than the actual actions it describes, runs the risk of having seemingly minor, but potentially important, steps omitted. To counter this, our client allows the user to build the final metadata for an active workflow step incrementally while working, "on the fly". Files (or URL strings) can be added via drag and drop functionality and immediately enriched with metadata. This process is depicted in the Figures 2a and 2b. The screenshots show the user dragging a tool in the form of a JAR file onto the client interface and then filling out the metadata form for the `tool` section of a workflow step (such as the parameters used to execute the tool on a command-line interface).

With this functionality the RePlay-DH Client enables users to document usage of a resource directly when actually using it. Entries cached this way (as seen in the lower part of Figure 2c) are then automatically added to the documentation of a workflow step when the user decides to record

[11] `https://eclipse.org/jgit/`

[12] `https://gitlab.com`

it. The final dialog in Figure 2d shows the three previously registered resources and the automatically detected output. Especially for automatic steps with a runtime of more than a few moments this can lead to a more efficient use of time for users and minimize the time overhead required for the documentation itself, besides the time that is used on the workflow.

Resource and metadata repositories. Our client is designed to directly interface with repository systems to make intermediate stages or the results of entire workflows available to others. Depending on the research context, different domains (Treloar et al., 2007) and restrictions apply. Innately the client works in private domains without external interfacing. In addition to that, the client also offers the possibility of collaborating in the shared (but not publicly open) domain. The possibility to share data within defined communities, is an important aspect when working with sensitive data.
For publishing partial or final results with a permanent identifier (DOI[13]) in the public domain the popular repository software DSpace[14] (Smith, 2002) is supported as one possible use case for the repository-client interaction.

Simplification and extensibility. The client implementation effectively shields the user from the complexity of underlying systems such as Git. This way we lower the barrier of entry to workflow documentation significantly and make it a convenient and accessible task for a wide target audience. The focus is on tracking changes of resources in a workflow and to assist the researcher in the documentation process by filling as many parts of the metadata as possible automatically (e.g. reuse information previously entered by the user for the same resource in another workflow step). For sharing and publishing workflow data we directly support a system frequently used for institutional repositories, as described above. Integration into existing institutional infrastructures is made possible by a plugin framework, allowing the RePlay-DH Client to be tailored to individual needs by adding custom implementations for interfaces.

Interoperability. Besides assisting in the elicitation of process metadata, the RePlay-DH Client also allows to export it. This export functionality is available for various formats and levels of granularity, such as an entire workflow graph, a certain workflow path or individual steps. While not directly derived from the PROV model or its extensions such as P-PLAN[15], our metadata scheme shares many aspects with those and transformation is a simple task. When exporting, the user can therefore choose between the native representation (serialized to JSON) or OWL-based variants. Due to the underlying plugin engine additional export formats can easily be integrated on demand.

6. Applicability

In this section we show in what ways the client is suitable for various types of NLP or DH tasks based on three examples and also how it can face recurring challenges when dealing with linguistic data or common processing steps.

[13] `https://www.doi.org/`
[14] `http://www.dspace.org`
[15] `http://www.opmw.org/model/p-plan/`

Manual annotation. Manually annotated corpora are one of the pillars of research in NLP. To evaluate a corpus for a given task researchers require precise knowledge about annotation guidelines, curation steps, automatic preprocessing or cleaning of the primary data. The ability to link to arbitrary resources as `input` for a step makes our metadata scheme well-suited for the first two aspects. Since the majority of annotation formats are represented textually, Git is a natural fit for tracking fine-grained changes made to them and ensuring their documentation.

Automatic processing. Quickly performed iterative and automatic processing steps are a hotbed for incomplete documentation when for example seemingly minor adjustments are left out. Our metadata scheme limits granularity of processing steps to have at most one `tool` instance. This way we ensure that no intermediary information gets lost. While this might appear demanding, the client supports mechanisms like drag & drop of files for usability and reduces the time required to document a single step.

No output. In the previous two data centered task types there is always a resource being modified or generated. To also document cognitive progress in a workflow, we allow the recording of steps that have no apparent effect on the resources in a workspace. Being essentially comparable to memos, they offer a great way to document insights gathered from a set of (`input`) resources.

Data size. Depending on the task, corpus or model resources can grow very large, making duplication caused by Git prohibitive. Our client allows users to exclude files from version control (either manually or based on a customizable size threshold), but to still cover them in the documentation. Combined with proper links to object metadata and documentation of previous steps, this can at least provide a level of reproducibility sufficient for many use cases.

Duplicates. With the lack of actually applied standards, it is quite common that within a single workflow resources are converted into multiple different formats for processing. While in principle still the same resource, those physically distinct instances could cause confusion in documentations. Our flexible approach to resource identification (see Section 4) allows proper unification of several physical instances of the same resource.

7. Pilot Project

For the design of our process metadata scheme described in Section 4 an associated annotation project was highly influential: In the manual annotation efforts for the gold standard in the GRAIN[16] release of the SFB732 Silver Standard Collection (Eckart and Gärtner, 2016) a setup very similar to RePlay-DH was used. Annotations were joined in a Git repository and the annotators had to enter formalized descriptions of their performed steps as part of their commits. These descriptions took the form of a preliminary version of the RePlay-DH process metadata and were provided in a simplified JSON format. Due to the lack of a dedicated support tool for eliciting the metadata (such as the client software presented in this paper) at the time, annotators were

[16] Also appearing as an article in this volume.

given textual templates to fill in relevant information in order to minimize overhead.
Subsequent automatic extraction of the metadata from the Git repository showed promising results in terms of usability for visualizing compositional information of the corpus. In addition, the pipelines creating automatic annotations for GRAIN were also designed in such a way that they create the same kind of process metadata as part of their analysis output. It is planned to release a curated version of the process metadata alongside the actual corpus data.

8. Availability

The tool itself is implemented in Java and runs on all major operating systems. Besides an installed Java Runtime Environment of version 8 or higher, no additional software is required for its use. Executable binaries, documentation and further information regarding the client software are available online[17]. Additional metadata for documentation is published in the framework of CLARIN [18].

9. Outlook

In this paper we presented the implementation of our previously proposed software tool for supporting process documentation by using version control as foundation. We contextualized the tool in the landscape of existing systems that deal with various aspects of workflow management or tracking applicable to tasks in NLP and DH. We also showed how our approach of separating different aspects of workflow documentation – namely the distinction between metadata describing *objects* used in a workflow and the *actions* performed – allows it to easily integrate into the diverse landscape of existing infrastructures and to better exploit the richness of available metadata repositories.
Discovery of available metadata records for resources in a workflow remains an open issue when it comes to external repositories. For future releases we plan to explore the feasibility of interfacing our client even more tightly with such systems to partly automate this discovery process. The possibility of defining metadata templates for entire steps which are frequently used in certain types of workflows is also something we intend to evaluate.

10. Acknowledgements

This work was funded by the Ministry for Science, Research and the Arts in Baden-Württemberg (MWK) via project RePlay-DH and the German Research Foundation (DFG) via the SFB 732, project INF.

11. Bibliographical References

Afgan, E., Baker, D., van den Beek, M., Blankenberg, D., Bouvier, D., Čech, M., Chilton, J., Clements, D., Coraor, N., Eberhard, C., Grüning, B., Guerler, A., Hillman-Jackson, J., Von Kuster, G., Rasche, E., Soranzo, N., Turaga, N., Taylor, J., Nekrutenko, A., and Goecks, J. (2016). The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic Acids Res.*, 44(W1):W3–W10.

Calzolari, N., Soria, C., Gratta, R. D., Goggi, S., Quochi, V., Russo, I., Choukri, K., Mariani, J., and Piperidis, S. (2010). The lrec map of language resources and technologies. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).

Claerbout, J. and Karrenbach, M. (1992). Electronic documents give reproducible research a new meaning. In *Proc. 62nd Ann. Int. Meeting of the Soc. of Exploration Geophysics*, pages 601–604.

Deelman, E., Gannon, D., Shields, M., and Taylor, I. (2009). Workflows and e-science: An overview of workflow system features and capabilities. *Future Gener. Comput. Syst.*, 25(5):528–540, May.

Eckart, K. and Gärtner, M. (2016). Creating Silver Standard Annotations for a Corpus of Non-Standard Data. In Stefanie Dipper, et al., editors, *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016)*, volume 16 of *BLA: Bochumer Linguistische Arbeitsberichte*, pages 90–96, Bochum, Germany.

Eckart de Castilho, R. and Gurevych, I. (2014). A broad-coverage collection of portable nlp components for building shareable analysis pipelines. In *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT*, pages 1–11, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

Elming, J., Johannsen, A., Klerke, S., Lapponi, E., Martinez Alonso, H., and Søgaard, A. (2013). Down-stream effects of tree-to-dependency conversions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 617–626, Atlanta, Georgia, June. Association for Computational Linguistics.

Gärtner, M., Hahn, U., and Hermann, S. (2018). Supporting sustainable process documentation. In Georg Rehm et al., editors, *Language Technologies for the Challenges of the Digital Age*, pages 284–291, Cham. Springer International Publishing.

Hinrichs, E. and Krauwer, S. (2014). The CLARIN Research Infrastructure: Resources and Tools for e-Humanities Scholars. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 1525–1531, May.

Ide, N., Pustejovsky, J., Cieri, C., Nyberg, E., DiPersio, D., Shi, C., Suderman, K., Verhagen, M., Wang, D., and Wright, J., (2016). *The Language Application Grid*, pages 51–70. Springer International Publishing, Cham.

McPhillips, T. M., Song, T., Kolisnik, T., Aulenbach, S., Belhajjame, K., Bocinsky, K., Cao, Y., Chirigati, F., Dey, S. C., Freire, J., Huntzinger, D. N., Jones, C., Koop, D., Missier, P., Schildhauer, M., Schwalm, C. R., Wei, Y., Cheney, J., Bieda, M., and Ludäscher, B. (2015).

[17] http://hdl.handle.net/11022/1007-0000-0007-C62F-6

[18] http://hdl.handle.net/11022/1007-0000-0007-C630-3

YesWorkflow: A user-oriented, language-independent tool for recovering workflow information from scripts. *CoRR*, abs/1502.02403.

Powell, A., Nilsson, M., Naeve, A., and Johnston, P. (2005). Dublin core metadata initiative - abstract model. White Paper.

Reich, M., Liefeld, T., Gould, J., Lerner, J., Tamayo, P., and Mesirov, J. P. (2006). GenePattern 2.0. *Nature Genetics*, 38(5):500–501, May.

Smith, M. (2002). Dspace: An institutional repository from the mit libraries and hewlett packard laboratories. In Maristella Agosti et al., editors, *Research and Advanced Technology for Digital Libraries*, volume 2458 of *Lecture Notes in Computer Science*, pages 543–549. Springer Berlin Heidelberg.

Treloar, A., Groenewegen, D., and Harboe-Ree, C. (2007). The data curation continuum: Managing data objects in institutional repositories. *D-Lib Magazine*, 13(9/10), September/October.