Supporting Sustainable Process Documentation

Motivation
- Documentation of complex research processes is often lacking.
- If done at all, it usually is performed after the process.
- Sustainable process documentation requires lots of additional effort.
- Existing version control solutions or workflow management systems are typically not suitable for processes in the fields of CL and DH.

Goals
- Assist in creating documentation already during an active research workflow.
- Provide a simple metadata schema for workflow documentation.
- Minimize effort required from researchers for clean process documentation.
- Idea: Build on Git as foundation for workflow tracking, but hide the complexity by channeling all the documentation work through a single graphical application.

Process Metadata
Workflows are modeled as directed acyclic graphs of interdependent steps. We collect and store metadata for individual workflow steps, following a simple schema:

- **Title:** User-defined short label for the workflow step.
- **Description:** More detailed human readable description of the workflow step as free text.
- **Input (0..n):** Resources used to perform the action (e.g., corpora, model files, annotation guidelines).
- **Output (0..n):** Resources generated or modified by the workflow step (annotation files, notes, ...).
- **Tool (0..1):** The executable resource or web-service used for processing (including configuration parameters).
- **Person (0..n):** Human subjects involved in the workflow step (e.g., annotators, curators, experiment participants).
- **Custom properties (0..n):** Arbitrary classic textual key-value metadata entries to provide additional machine readable information.
- **Serialization format for our process metadata is JSON, making it easy to process for others.**

Local Git
Each local workspace is put under version control, directly providing several benefits:
- Once recorded in a workflow, no data or information gets lost (effectively a local backup).
- Process metadata collected during the workflow is stored together with the physical data in every Git commit.
- By means of branching users can comfortably try alternatives in their workflow without clogging the workspace with additional files.

Remote Git
Local workspaces in the RePlay-DH client can be linked to a remote Git repository such as an institute or university GitLab instance:
- Distributed storage provides an additional layer of backup for important research data.
- Remote Git can be used as archiving solution.
- Multiple users can collaborate on the same project and data through a shared remote repository.

Design Principles

Independence:
No external infrastructure or additional third-party software required for the basic client. Workflow documentation and local object metadata management in a simple schema following Dublin Core [1] available.

Extensibility:
Plugin-architecture to incorporate the client into existing institutional infrastructure such as repositories for metadata or publishing.

External Repositories
Planned interfacing of the client with repositories for different domains:

- **Public Domain:** Repository software DSpace [2] for publishing data with a persistent identifier (DOI).
- **Shared Domain:** With better rights management RePlay-DH-Client