CWLProv

Retrospective provenance capture and its challenges

https://www.commonwl.org/
common-workflow-language/cwltool

Farah Zaib Khan · Stian Soiland-Reyes · Richard O. Sinnott · Andrew Lonie · Michael R. Crusoe

The University of Melbourne, Australia
The University of Manchester, UK
Common Workflow Language Project

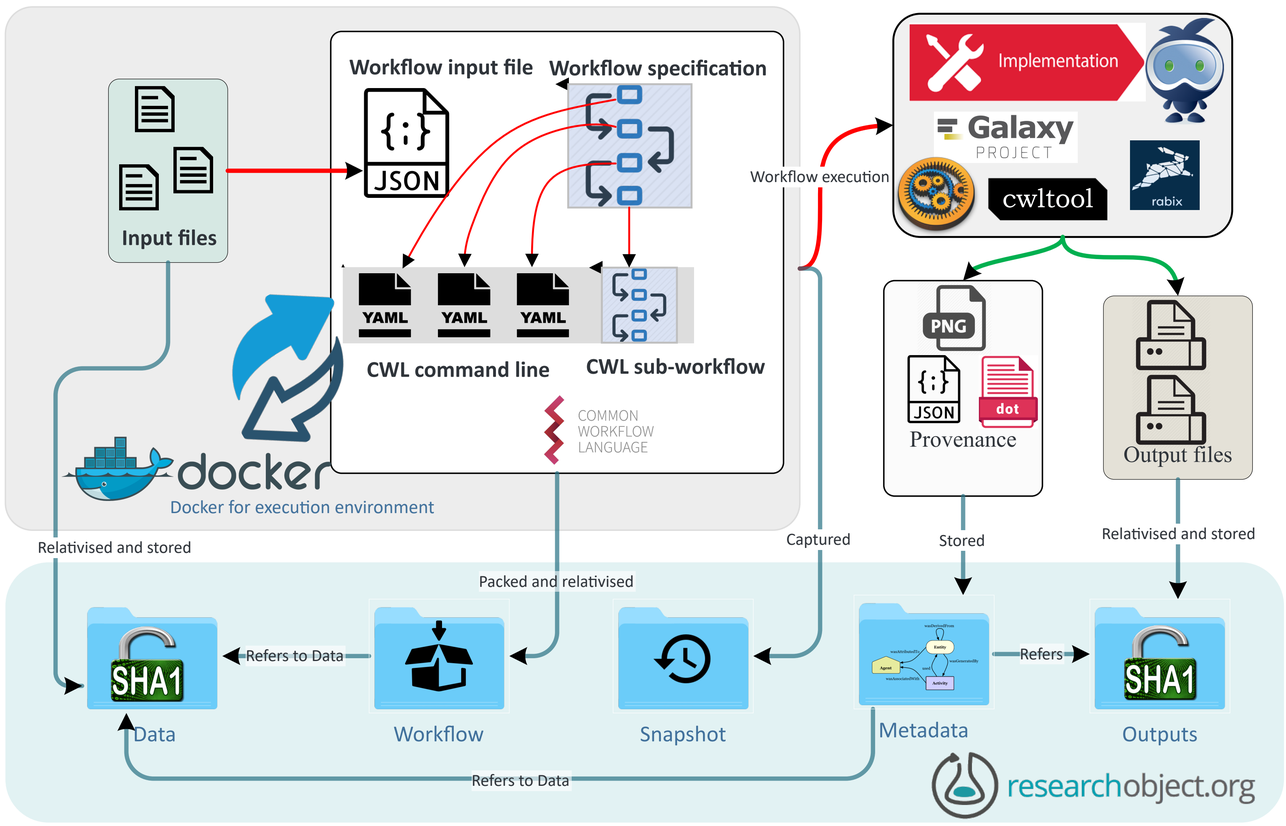
Capturing workflow provenance

CWLProv adds provenance capture to **cwltool,** the reference implementation of CWL.

Snapshots of the workflow and input files are captured in the **Research Object** (RO) with relativizing paths. The generated **input JSON file** allows re-execution.

Data generated by the workflow run, including intermediates, are SHA1-hashed and added to the RO with **content-addressable** identifiers.

The retrospective **provenance trace** of the run refers to data using their hash, linking to the workflow definition.

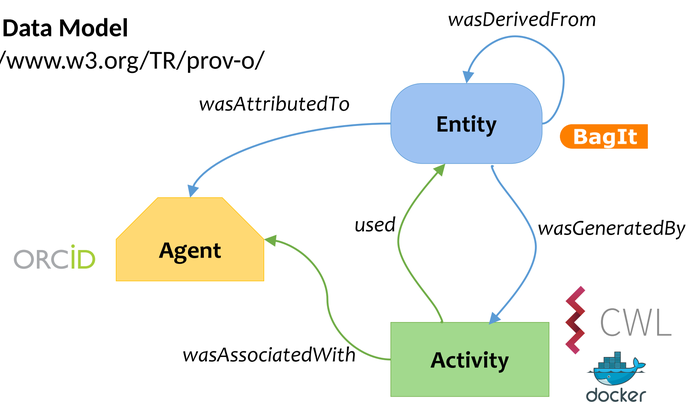


Workflow Provenance Profile

L	PROV type	Subtype	Relations	
1	**Plan**	wfdesc:Workflow	wfdesc:hasSubProcess	wfdesc:Process
1		wfdesc:Process		
0	**Activity**	wfprov:WorkflowRun	wasAssociatedWith	wfprov:WorkflowEngine
1			hadPlan	wfdesc:Workflow
0			wasStartedBy	wfprov:WorkflowEngine
0			atTime	*(ISO8601 timestamp)*
2			wasStartedBy	wfprov:WorkflowRun
0			wasEndedBy	wfprov:WorkflowEngine
0			atTime	*(ISO8601 timestamp)*
1		wfprov:ProcessRun	wasStartedBy	wfprov:WorkflowRun
1			atTime	*(ISO8601 timestamp)*
1			used	wfprov:Artifact
1			role	wfdesc:InputParameter
1			wasAssociatedWith	wfprov:WorkflowRun
1			hadPlan	wfdesc:Process
1			wasEndedBy	wfprov:WorkflowRun
1			atTime	*(ISO8601 timestamp)*
2		SoftwareAgent	wasAssociatedWith	wfprov:ProcessRun
2			cwlprov:image	*(docker image)*
0	**SoftwareAgent**	wfprov:WorkFlowEngine	wasStartedBy	Person *(ORCID)*
0			label	*(cwltool –version)*
1	**Entity**	wfprov:Artifact	wasGeneratedBy	wfprov:ProcessRun
1			role	wfdesc:OutputParameter

*CWLProv profile of **W3C PROV**, extended with Research Object Model **wfdesc** (prospective provenance) and **wfprov** (retrospective provenance). Indentation indicates n-ary relationships (hadPlan/atTime/role/image).*

PROV Data Model
http://www.w3.org/TR/prov-o/



Best Practices for Publishing Workflow Analyses

R1: Workflows should be treated as **first class data objects** (Corcho 2012).

R2: Workflow specification alone is insufficient to ensure reusability of scientific experiments (Zhao 2012). Complete provenance **capture of workflow enactment** should be published along with the workflow specification (Garijo 2017). This can also help to avoid *workflow decay*.

R3: A **structured description** of the experimental steps carried out in a workflow using a "system-neutral" language can ensure well-documented and well described workflows for enhanced understandability of methods (Belhajjame 2015).

R4: Availability of the **underlying software** needed by each step of a given workflow is critical. (Kanwal 2017). More recently **container** technologies such as *Docker*, *OpenVZ* or *LXC* can be exploited to package the environment and configuration together.

R5: While publishing digital scholarly objects, **open licensing** should be adapted as a practice to allow *sharing* and *reproducing* of published analyses (Stodden 2016).

R6: The description of the underlying software is not enough for reproducing an analysis; instead **workflow specifications** and **configuration** should also be published (Garijo 2017).

R7: Intermediate data products should be captured (if feasible) to facilitate debugging, error handling and thorough examination of the published workflows and associated results (Sandve 2013).

Provenance levels

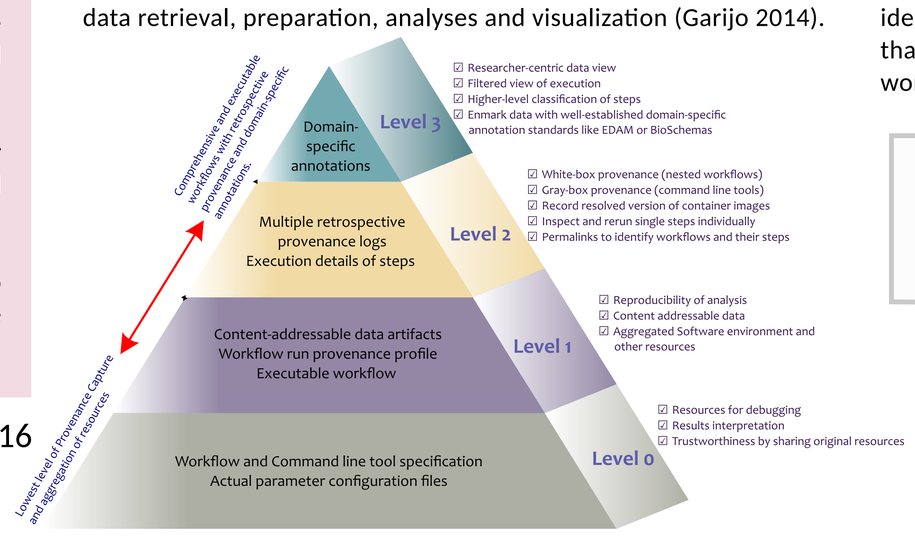
The granularity level of the workflow execution in the RO can be classified into four levels to realize a hierarchical and modular solution towards interoperable domain-specific provenance.

Level 0 adds **snapshots** of the files used to start a workflow enactment and optionally the final outputs. This can be used for result interpretation (e.g. workflow parameters) and facilitates *re-executing* or *repurposing* the workflow as a whole. In the simplest case this logs the workflow engine command line arguments; in CWLProv we create a re-executable *cwl-runner* **job file** with relative links to inputs and workflow definition files.

Level 1 adds *retrospective provenance* with *content-addressable* data to enable reproducibility of the main analysis, step by step. In CWLProv this takes the form of a **W3C PROV** file. Activities link to local identifiers of steps within the workflow *plan*, providing *prospective provenance* (Zhao 2006).

Level 2 (work in progress) logs *within steps*, e.g. the exact Docker image ids fetched, as well as adding retrospective provenance of **nested workflows.** Although technically this can be added to the same PROV file as level 1, we found it cleaner with separate PROV files per step execution so each step is re-runnable independently.

Level 3 (planned) adds *augmented* and *filtered* provenance, **domain-specific** and focused only on data/steps relevant to the researcher. In CWLProv we will build on existing work like *CWL Viewer*, *LabelFlow* (Alper 2018) and *BioCompute Objects* (BCO) to extract CWL annotations using domain-specific ontologies (**EDAM**, **BioSchemas**) apply them to the produced data, and classify steps into *motifs* like data retrieval, preparation, analyses and visualization (Garijo 2014).



Common Workflow Language

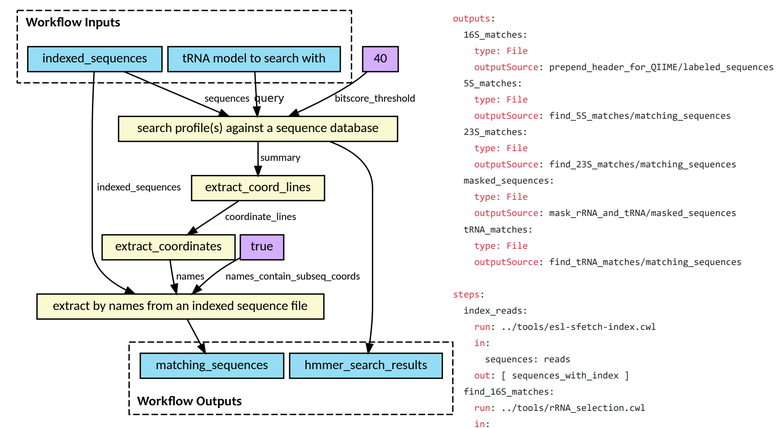
Common Workflow Language (CWL) aims to provide extensible, open source standards supporting interoperable, portable and reproducible workflow-based research (Amstutz 2016).

CWL is a community-driven effort, being adapted by leading workflow design and execution platforms including UCSC' *Toil*, Curoverse's *Arvados*, Seven Bridges' *Rabix*, Broad Institute's *Cromwell*, IBM's *CWLExec*, *Galaxy* and *Apache Taverna* (incubating).

CWL provides declarative constructs for workflow and command line tool definition and make minimal assumptions about base software dependencies, configuration settings, software versions, parameter settings or the execution environment more generally.

To a large extent CWL achieves workflow **portability** by executing bioinformatics tools distributed as **Docker** containers or **BioConda** packages, ensuring the correct tool version and its dependencies are installed. Defining tool descriptions in CWL encourages reuse of steps across workflows and researchers.

The CWL community commonly develops workflows as open source in GitHub repositories, which can be visualized in the **CWL Viewer**. CWL definitions can be repurposed or reused across multiple labs, as owing to the **interoperability** aspect of CWL they do not need to agree on workflow engine or compute architecture.



tRNA selection, EMG pipeline (Mitchell 2018)
Rob Finn, Michael Crusoe
EBI-Metagenomics/ebi-metagenomics-cwl

CWLProv Research Objects

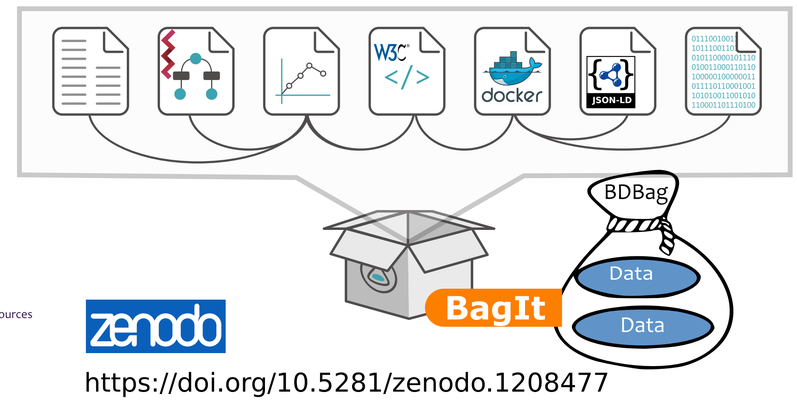
One challenge when capturing or distributing details of a workflow enactment is handling *multiple files* (workflow, scripts, datasets), *non-digital* resources (e.g. people, equipment, samples), execution *dependencies* (Bechhofer 2013) as **Docker** or **BioConda** packages.

These resources often have separate *identifiers*, *versions*, *attributions* and can be produced by humans or automated processes. Large data (e.g. metagenomic reads) can be non-trivial to transfer correctly or efficiently, and may reside in multiple repositories (Madduri 2018).

Research Objects encapsulate the digital artefacts associated with a given computational analysis, richly described with structured metadata in a **JSON-LD manifest**. Aggregated resources may include *input* and *output* data for experiment results; computational *methods* such as command line tools and workflow specifications; *attribution* details; retrospective and prospective *provenance*; and machine-readable *annotations* regarding the included artefacts and the relation between them (Belhajjame 2015).

The RO approach, developed in full for **workflow preservation** by capturing provenance of executing bioinformatics (Hettne 2014) and virtual astronomy workflows (Ruiz 2014), has expanded into other domains to cover investigations and datasets in systems biology (Wolstencroft 2013), earth sciences (Garcia-Silva 2017), health informatics (Pavis 2015) and precision medicine (Alterovitz 2018).

CWLProv updates the original *Workflow Research Object* profile with the **BagIt** serialization (Kunze 2018), a Library of Congress supported digital archive mechanism that emphasizes file authenticity and completeness. RO and BagIt, combined with the **MinId** distributed identification resolution mechanism, forms **BDBag** (Big Data Bag), that can capture and transfer datasets from large-scale genomics workflow runs (Chard 2016), used in *NIH Data Commons*.



zenodo
https://doi.org/10.5281/zenodo.1208477

https://f1000research.com/posters/7-916

We gratefully acknowledge **The University of Melbourne** for providing funding support as MIRS and MIFRS scholarship.

This work has been done as part of the **BioExcel CoE** (www.bioexcel.eu), a project funded by the European Union contract H2020-EINFRA-2015-1-675728.