SciDAP

Airflow and CWL-powered bioinformatics platform

Nicholas Luckey
Michael Kotliar

Human readable workflow representation

Workflow provenance support

Modular workflow structure

Apache Airflow

Interoperability and portability independent of vendor

task-driven workflow management system developed by Airbnb

Rich ecosystem of available workflows widely supported by scientific community

Common Workflow Language

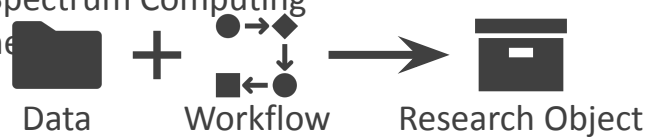
open standard for describing workflows and makes them p

```
cwlVersion: v1.2
class: Workflow
inputs:
  input_file:
    type: File
outputs:
  output_file:
    type: File
    outputSource: second_step/output_file
steps:
  alternative_step:
    run: another.cwl
    in:
      input_file: input_file
    out:
      - output_file
  second_step:
    run: second.cwl
    in:
      input_file: first_step/output_file
    out:
      - output_file
```

Arvados Project
Curii
Seven Bridges Genomics
Galaxy Project
Apache Taverna
Institut Pasteur
Wellcome Trust Sanger Institute
University of California Santa Cruz
Harvard T.H. Chan School of Public Health
Cincinnati Children's Hospital Medical Center
Broad Institute
University of Melbourne Center for Cancer Research
Netherlands eScience Center
Agave Platform
CyVerse
Institute for Systems Biology
ELIXIR Europe
BioExcel CoE
BD2K
EMBL Australia Bioinformatics Resource
IBM Spectrum Computing
DNAne
CERN

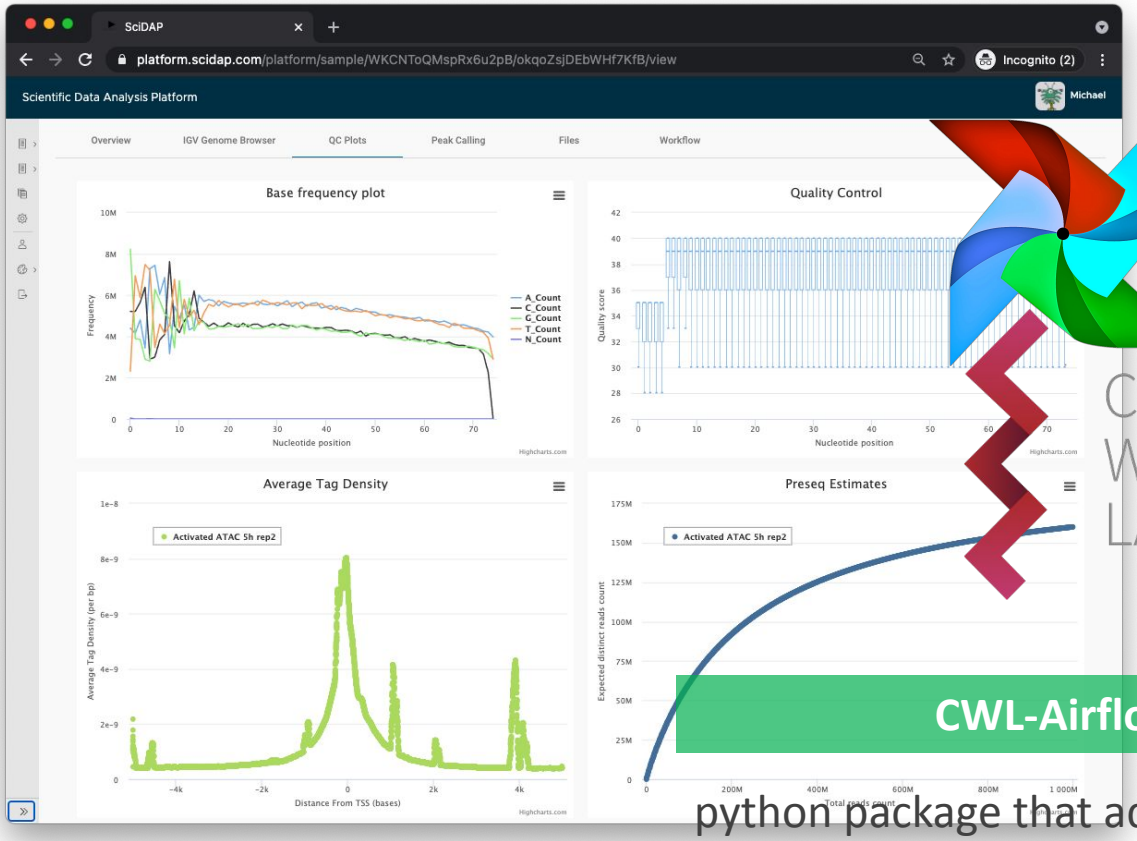
CWL-Airflow

python package that adds support for CWL to the Apache Airflow



Findability
Accessibility
Interoperability
Reuse

SciDAP



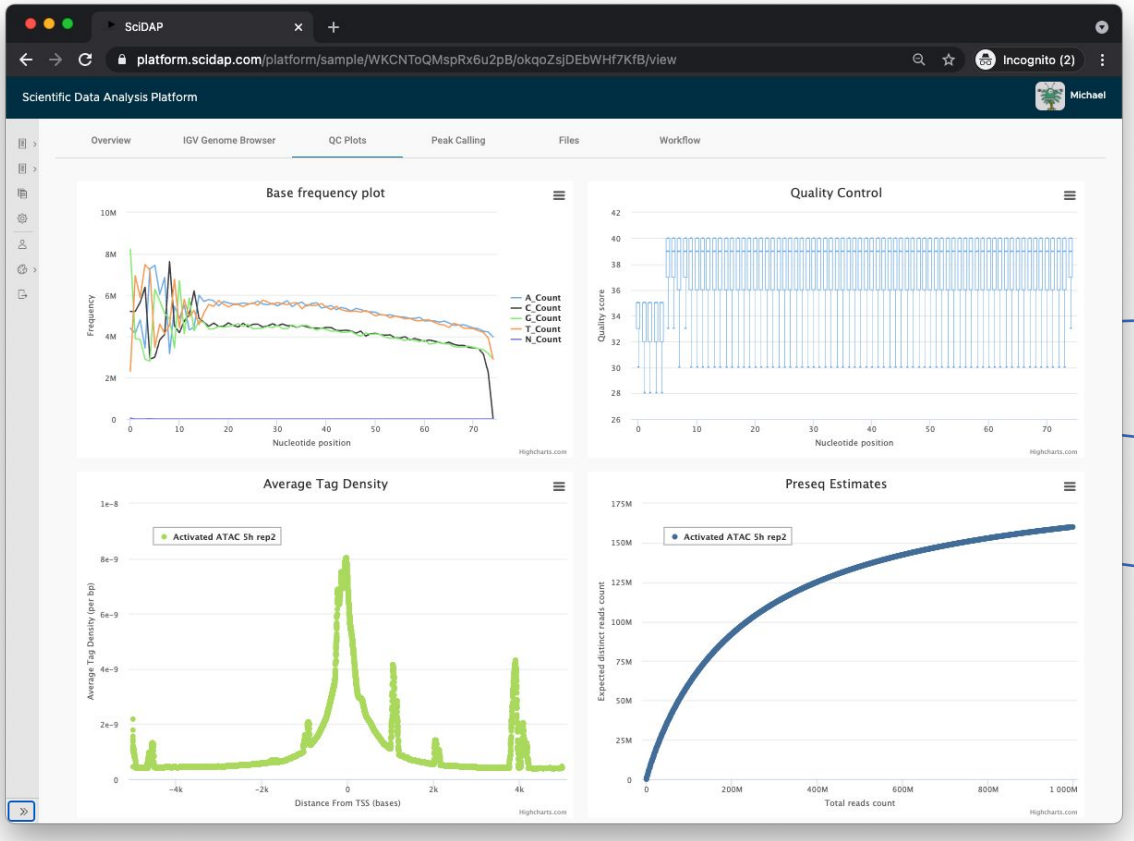
user-friendly scientific data analysis platform

COMMON
WORKFLOW
LANGUAGE

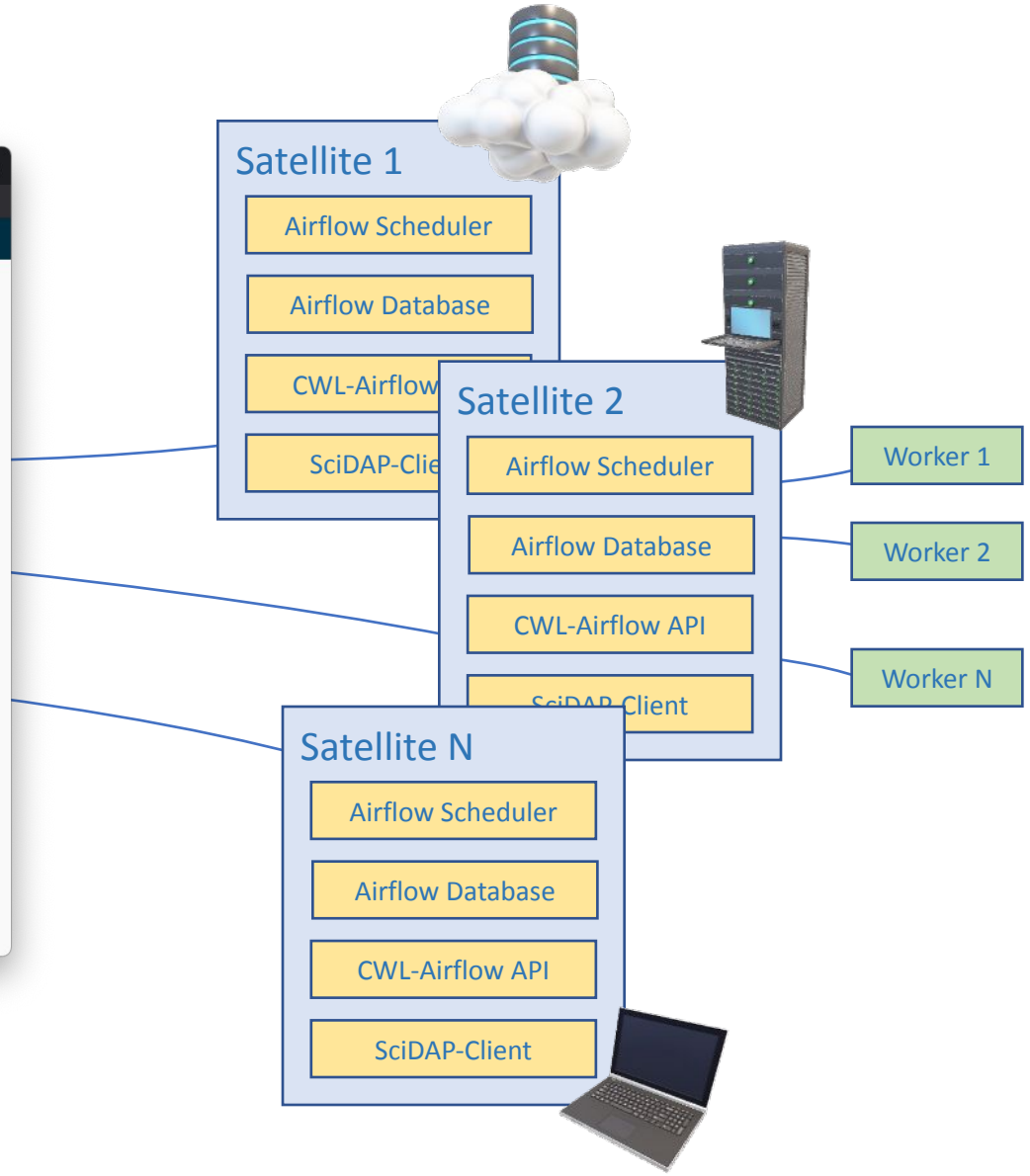
CWL-Airflow

python package that adds support for CWL to the Apache Airflow

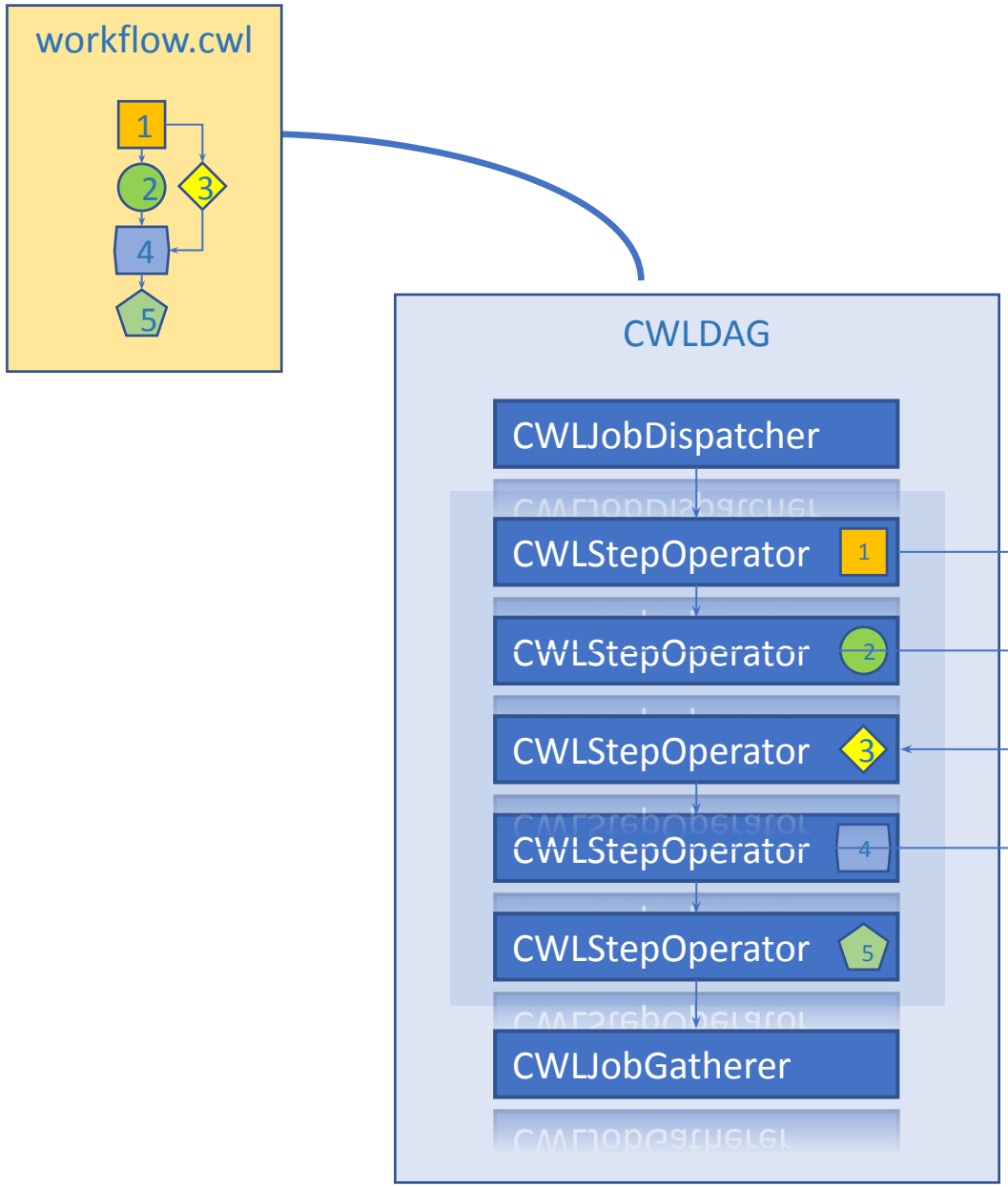
SciDAP



user-friendly scientific data analysis platform



Challenge 1 Converting CWL workflow to Airflow DAG



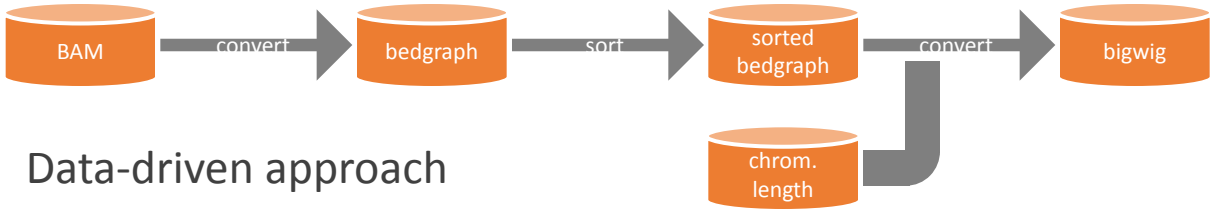
my_dag.py

```
#!/usr/bin/env python3
from cwl_airflow.extensions.cwldag import CWLDAG
dag = CWLDAG(
        workflow='workflow.cwl',
        dag_id='bam-bedgraph-bigwig'
)
```

my_dag_with_embedded_workflow.py

```
#!/usr/bin/env python3
from cwl_airflow.extensions.cwldag import CWLDAG
dag = CWLDAG(
        workflow='H4sIAAn3/F8/8 ... TikAAA==',
        dag_id='bam-bedgraph-bigwig'
)
```

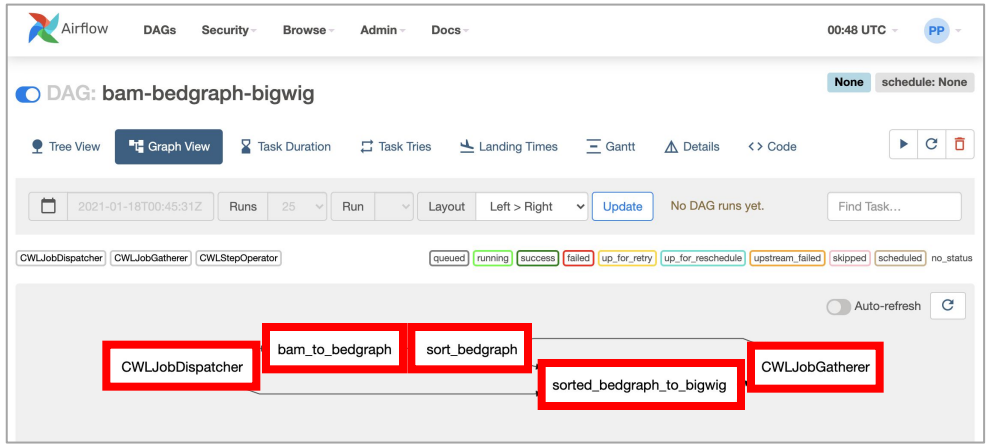
Challenge 2 CWL is a data-driven workflow standard

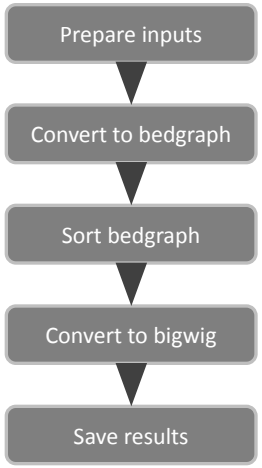


Data-driven approach

Combine tasks based on the CWL workflow step inputs and outputs

Define a mechanism for transfering data between tasks





Traditional for Airflow DAG-based task-driven approach

Task starts its execution if all its predecessors have successfully completed

Task is not supposed to exchange data with other task

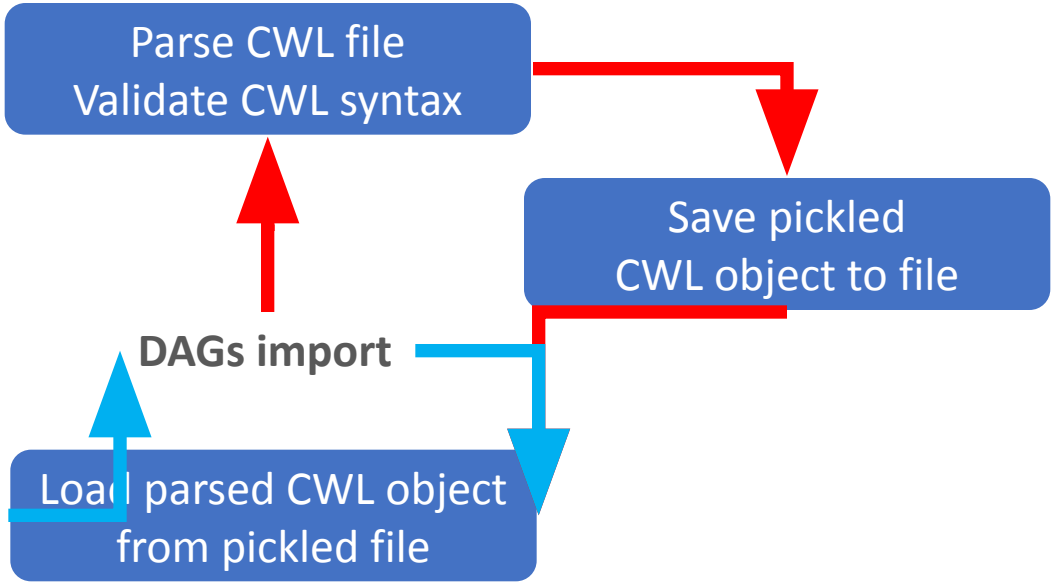
Challenge 3 CWL slows down Airflow DAG import

1. Parse CWL file
2. **Validate CWL syntax**
3. Create DAG

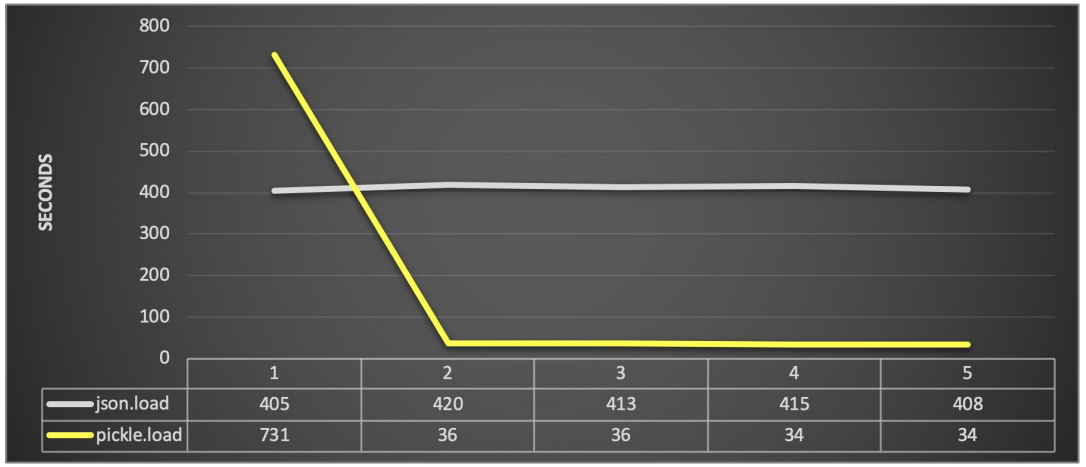
my_dag.py

```
#!/usr/bin/env python3
from cwl_airflow.extensions.cwldag import CWLDAG
dag = CWLDAG(
        workflow='workflow.cwl',
        dag_id='bam-bedgraph-bigwig'
)
```

How long before timing out a python file import?
dagbag_import_timeout = 30.0



515 CWLDAGs



	1	2	3	4	5
json.load	405	420	413	415	408
pickle.load	731	36	36	34	34

CWL in Airflow: pros and cons for scientific community

Advantages:

Cross-vendor portability

Shallow learning curve for newcomers

Designed with FAIRness in mind

Rich ecosystem of available and tested workflows

Workflow visualization tools
for easy pipeline building and viewing

Broad list of participating organizations

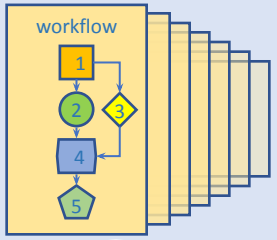
Disadvantages:

Limited functionality implied by using CWL specification

Extra level of complexity, thus more places to look for errors

CWL-Airflow and its role in scientific research

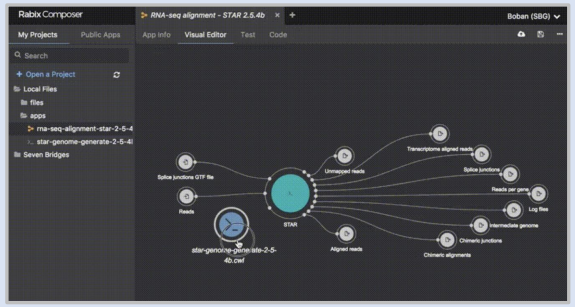
CWL workflows



Innovations

Science
nature
(GIGA)n SCIENCE

Visual workflow editors



Workflow management systems

SevenBridges

TOIL
Massively Parallel Workflows
(and fire-breathing dragon slugs)

SCIDAP

Arvados

reana

Future CWL-Airflow improvements

- Complete CWL specification support
  - Conditional step execution
  - Scattered step execution
  - Abstract operations support
- Support other workflow specifications
  - WDL
  - Nextflow
  - Snakemake
- Complete Cloud support

Airflow Ecosystem