
# Workflow Provenance registration with COMPSs

Raül Sirvent – Barcelona Supercomputing Center

Innovative HPC workflows for industry, Munich
October 25, 2023

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955558. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway. MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR (PCI2021-121957)

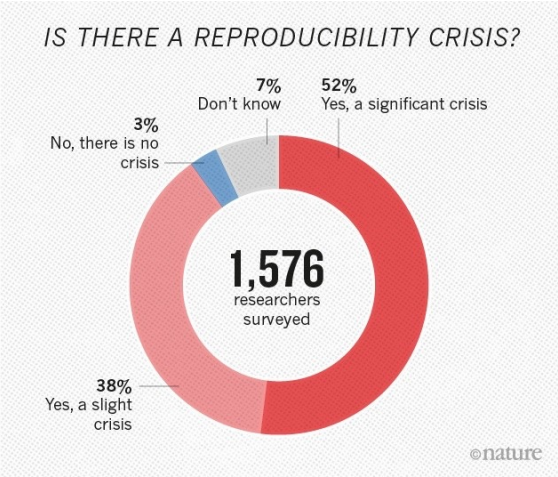
# Outline


- Motivation and Background
- Design of the Workflow Provenance recording
- Using Workflow Provenance with COMPSs
- Inspecting registered metadata
- Live demo with WorkflowHub

# MOTIVATION AND BACKGROUND

# Motivation

- Large number of **Scientific Workflows** experiments
  - Keep track of results - **Governance**
- **Reproducibility** crisis in scientific papers
  - Conferences now request artifacts
- **Provenance recording** can help with both problems




©M. Baker, Nature, 2016

- **Provenance:** The chronology of the origin, development, ownership, location, and changes to a system or system component and associated data
  - Need to record metadata
  - Our focus: Workflow Provenance (data + software)

# Motivation

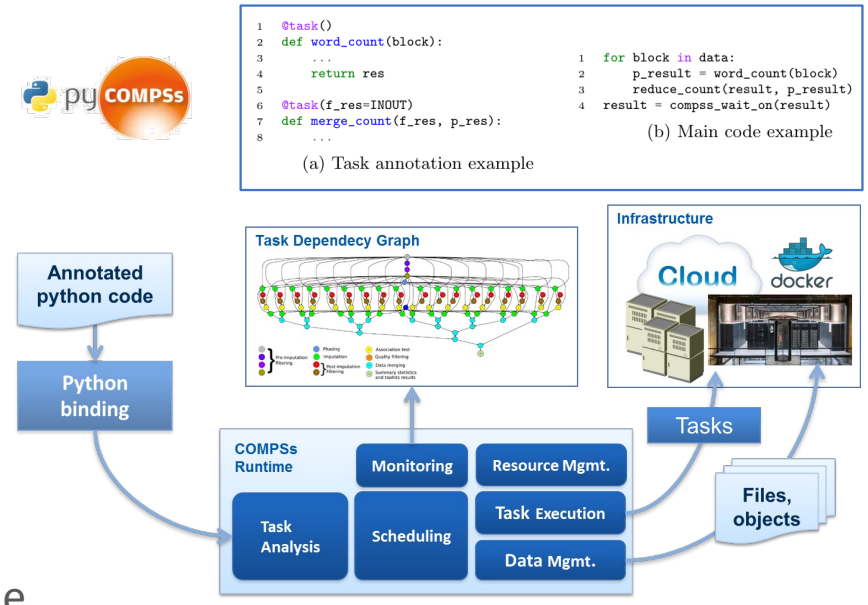
eFlows4HPC

- Provenance is **MORE** than just Reproducibility
  - **Governance** (availability, usability, consistency, …)
  - **Replicability** (exchange inputs)
  - **Traceability** (validation/verification, visualisation)
  - **Knowledge extraction** (queries, mining)

- Our claim: desired features for **Workflow Provenance** registration
  - **Automatic:** lower user burden
  - **Efficient:** no overheads
  - **Scalable:** large workflows (tasks and data assets used)

# Background: COMPSs

- **Sequential** programming, **parallel** execution
- **General purpose** programming language + **annotations/hints** (identify tasks and directionality of data)
- Builds a **task graph** at runtime (potential concurrency)
- Tasks can be **sequential**, **parallel** (threaded or MPI)
- Offers to applications a **shared memory illusion** in a distributed system (Big Data apps support)
- Support for **persistent storage**
- **Agnostic** of computing platform: enabled by the runtime for **clusters**, **clouds** and **container** managed clusters




- **Advanced features:** heterogeneous infrastructures, task constraints, streamed data, task faults, task exceptions, checkpointing, elasticity

# Background: Research Object Crate


- Package research data + metadata
- Evolution from:
  - **Research Object**: describe digital and real-world resources
  - **DataCrate**: aggregate data with metadata
- Lightweight format
  - Both **machines** and **humans** can read it
- JSON Linked Data (JSON-LD)
  - Vocabulary: Schema.org
  - Structure:
    - **Root Data Entity**
    - **Data Entities** (files, directories)
    - **Contextual Entities** (non-digital elements)
- Strong ecosystem, we use:
  - ro-crate-py library
  - WorkflowHub

# Background: RO-Crate Profiles

- RO-Crate is very **generic** (wide scope)
  - Profiles enable **Interoperability**
    - Set of conventions, types and properties (MUST, SHOULD, …)
- **Workflow RO-Crate** profile
  - MUST **ComputationalWorkflow**, **mainEntity** (Root Dataset)
  - SHOULD **WorkflowSketch**
- **Workflow Run RO-Crate** profile collection (MUST **CreateAction**)
  - Process Run Crate (set of tools)
  - Workflow Run Crate (computational workflow)
  - Provenance Run Crate (detailed computational workflow)

# DESIGN OF WORKFLOW PROVENANCE RECORDING

# Design Requirements

- Target HPC workflows (commonly large)
- Provenance representation format
  - Simple but able to represent complex workflows
- **Automatic** provenance registration (no explicit annotations)
- **Efficient** provenance registration (avoid overheads at run time)
- **Scale** to large workflows (thousands of files and tasks)

# COMPSs runtime modifications


- Flags –p or --provenance trigger it after execution
- Can be manually invoked if provenance generation time becomes an issue (i.e., extreme large workflows)

dataprovenance.log

After application finishes…

generate_COMPSs_RO-Crate.py

ro-crate-info.yaml

ro-crate-py 0.8.0

COMPSs_RO-Crate_[uuid]/

- Lightweight approach: record file accesses, generate provenance later

```
3.2
lysozyme_in_water.py
App_Profile.json
file://s01r2b54-ib0/home/bsc19/bsc19057/DP_Test_3_demo/dataset/2hs9.pdb IN
file://s01r2b54-ib0/home/bsc19/bsc19057/DP_Test_3_demo/output/2hs9.gro OUT
file://s01r2b54-ib0/home/bsc19/bsc19057/DP_Test_3_demo/output/2hs9.top OUT
...
```

- It's the *crate*
- ro-crate-metadata.json
- Application source files, command line arguments, workflow image and profile


# generate_COMPSs_RO-Crate.py features


- Detects and records **COMPSs version** used and the **mainEntity**
  - Looks for alternatives, if not found
- Automatically detects overall **inputs** and **outputs** of the workflow
  - Discards intermediate generated results as inputs
- Respects application **source files** sub directory structure
- If data persistence, machine paths translated to crate paths
  - Identifies **common paths** to correctly arrange files
    - E.g. inputs/00/input_file.txt
- If no persistence: **URIs** to files are generated, **size** and **modification** date of files are stored to record the file version

# USING WORKFLOW PROVENANCE WITH COMPSS

# Steps to record and publish Workflow Provenance in COMPSs

- Install ro-crate-py (if needed)
- Provide YAML information file
- Run with -p or --provenance
  - The ***crate*** is generated (a sub-folder COMPSs_RO-Crate_[uuid])
- Publish it at WorkflowHub, using the crate
- Generate a DOI, cite your results in papers


# Install ro-crate-py

- pip install rocrate
- pip install rocrate --user
  - Typically, installs the library in ~/.local/
- pip install -t install_path rocrate
  - Specify target directory

https://github.com/ResearchObject/ro-crate-py

# YAML information to be provided

- Non-automatically gathered info: **ro-crate-info.yaml**
- Sections:
  - COMPSs Workflow Information
  - Authors
  - Submitter
- Data persistence: True or False
- No inputs/outputs are provided, automatically detected by the provenance generation script

```
COMPSs Workflow Information:
  name: COMPSs Matrix Multiplication
  description: Blocks as hypermatrix
  license: Apache-2.0
  sources_dir: [src, ~/java/matmul/xml]
  files: [~/java/matmul/pom.xml, Readme]
  data_persistence: True
Authors:
  - name: Rosa M. Badia
    e-mail: Rosa.M.Badia@bsc.es
    orcid: https://orcid.org/0000-0003-2941-5499
    organisation_name: Barcelona Supercomputing
    Center
    ror: https://ror.org/05sd8tv96
Submitter:
  name: Raül Sirvent
  e-mail: Raul.Sirvent@bsc.es
  orcid: https://orcid.org/0000-0003-0606-2512
  organisation_name: Barcelona Supercomputing
    Center
  ror: https://ror.org/05sd8tv96
```

# Run your COMPSs application

- runcompss -p
- enqueue_compss -p
- pycompss run -p

- Either **-p** or **--provenance**

- Post-process automatically triggered after the end of the application
- Log and time statistics are provided
  - grep PROVENANCE
- If provenance generation fails for any reason:
  - Still possible to invoke it manually (commands provided in the output log)

```
...
PROVENANCE | COMPSs RO-Crate created successfully in subfolder COMPSs_RO-Crate_aaf0cb82-a500-4c28-bbc8-439c37c2e210/
PROVENANCE | RO-CRATE dump TIME: 0.004969120025634766 s
PROVENANCE | RO-CRATE GENERATION TOTAL EXECUTION TIME: 0.014089107513427734 s
PROVENANCE | ENDED DATA PROVENANCE SCRIPT
```

# The Crate (resulting folder)

- application_sources/
- dataset/
- complete_graph.svg
- App_Profile.json
- compss_command_line_arguments.txt
- ro-crate-metadata.json

```
|-- App_Profile.json
|-- application_sources
|   |-- Readme
|   |-- pom.xml
|   |-- src
|   |   `-- main
|   |       `-- java
|   |           `-- matmul
|   |               |-- arrays
|   |               |   |-- …
|   |               |   `-- Matmul.java
|   |               |-- files
|   |               |   |-- Block.class
|   |               |   |-- Block.java
|   |               |   |-- Matmul.class
|   |               |   |-- Matmul.java
|   |               |   |-- MatmulImpl.class
|   |               |   |-- MatmulImpl.java
|   |               |   |-- MatmulItf.class
|   |               |   `-- MatmulItf.java
|   |               `-- objects
|   |                   |-- …
|   |                   `-- Matmul.java
|   `-- xml
|       |-- project.xml
|       `-- resources.xml
|-- complete_graph.svg
|-- compss_command_line_arguments.txt
|-- dataset
|   |-- …
|   `-- C.1.1
|-- ro-crate-info.yaml
`-- ro-crate-metadata.json

10 directories, 41 files
```

# Publish your results with WorkflowHub

- zip –r crate.zip COMPSs_RO-Crate_[uuid]/
- Login to WorfklowHub
- Create -> Workflow
  - Upload/Import Workflow RO-Crate tab -> Local file (crate.zip)
  - Click Register
- Review automatically obtained information
- Select the visibility of your workflow in the Sharing tab (for both general public, and for teams selected)
- Click Register again

# Cite your results with WorkflowHub

- Freeze your workflow version
  - Overview tab -> Citation box -> Freeze version
  - Actions menu -> Freeze version
- Generate DOI
  - **IMPORTANT:** make sure your version is final
  - Citation box -> Generate a DOI
  - Actions menu -> Generate a DOI
  - Select Mint DOI
- The **final generated DOI** for the workflow results can be found in the Citation box

https://doi.org/10.48546/workflowhub.workflow.484.1

# INSPECTING REGISTERED METADATA

# Inspecting registered metadata


```
"@id": "application_sources/matmul_files.py",
"@type": ["File", "SoftwareSourceCode", "ComputationalWorkflow"],
"contentSize": 1948,
"description": "Main file of the COMPSs workflow source files",
"encodingFormat": "text/plain",
"image": {"@id": "complete_graph.svg"},
"name": "matmul_files.py",
"programmingLanguage": {"@id": "#compss"}
```

```
"@id": "#compss",
"@type": "ComputerLanguage",
"alternateName": "COMPSs",
"citation":
    "https://doi.org/10.1007/s10723-
    013-9272-5",
"name": "COMPSs Programming Model",
"url": "http://compss.bsc.es/",
"version": "3.2"
```


```
"@id": "complete_graph.svg",
"@type": ["File", "ImageObject", "WorkflowSketch"],
"about": {"@id": "application_sources/matmul_files.py"},
"contentSize": 6681,
"description": "The graph diagram of the workflow, automatically generated by COMPSs runtime",
"encodingFormat": [[”image/svg+xml",{"@id": "https://www.nationalarchives.gov.uk/PRONOM/fmt/92"}]],
"name": "complete_graph.svg"
```

# Inspecting registered metadata

## Auxiliary Files

```
"@id": "application_sources/matmul_tasks.py",
"@type": ["File", "SoftwareSourceCode"]
"contentSize": 1549,
"description": "Auxiliary File",
"encodingFormat": "text/plain",
"name": "matmul_tasks.py"
```

## Command line arguments

```
"@id": "compss_command_line_arguments.txt",
"@type": "File",
"contentSize": 709,
"description": "COMPSs command line
    execution command (runcompss),
    including flags and parameters passed",
"encodingFormat": "text/plain",
"name": "compss_command_line_arguments.txt"
```

## COMPSs Task Profiling

```
"@id": "App_Profile.json",
"@type": "File",
"contentSize": 247,
"description": "COMPSs application Tasks profile",
"encodingFormat": ["application/json", {"@id": "https://www.nationalarchives.gov.uk/PRONOM/fmt/817"}],
"name": "App_Profile.json"
```

# Inspecting registered metadata

## Persistent Data

```
"@id": "dataset/A.0.0",
"@type": "File",
"contentSize": 16,
"dateModified": "2023-09-07T09:20:20",
"name": "A.0.0",
"sdDatePublished": "2023-09-07T09:20:27+00:00"
```

## Non-Persistent Data

```
"@id": "file://s07r1b33-ib0/home/bsc19/bsc19057/DP_Test_3_demo/dataset/133l.pdb",
"@type": "File",
"contentSize": 116154,
"dateModified": "2022-04-20T13:20:58",
"name": ”133l.pdb",
"sdDatePublished": "2022-10-18T08:03:08+00:00"
```

```
"@id": "file://s02r2b26-ib0/home/bsc19/bsc19057/DP_Test_3_demo/config/energy.selection"
```

Hostname

Location path in hostname

# Inspecting registered metadata

## CreateAction


```
"@id": "#COMPSs_Workflow_Run_Crate_marenostrum4_SLURM_JOB_ID_30132875",
"@type": "CreateAction",
"actionStatus": {"@id": "http://schema.org/CompletedActionStatus"},
"agent": {"@id": "https://orcid.org/0000-0003-0606-2512"},
"description": "Linux s01r2b48 4.4.59-92.20-default #1 SMP Wed May 31 14:05:24 UTC 2017 (8cd473d)
    x86_64 x86_64 x86_64 GNU/Linux SLURM_JOB_NAME=matmul-DP COMPSS_PYTHON_VERSION=3.9.10
    SLURM_JOB_QOS=debug SLURM_MEM_PER_CPU=1880 COMPSS_BINDINGS_DEBUG=1 SLURM_JOB_ID=30132875
    SLURM_JOB_USER=bsc19057 COMPSS_HOME=/apps/COMPSs/3.2/ SLURM_JOB_UID=2952
    SLURM_SUBMIT_DIR=/gpfs/home/bsc19/bsc19057/COMPSs-DP SLURM_JOB_NODELIST=s01r2b48
    SLURM_JOB_GID=2950 SLURM_JOB_CPUS_PER_NODE=48 COMPSS_MPIRUN_TYPE=impi SLURM_SUBMIT_HOST=login3
    SLURM_JOB_PARTITION=main SLURM_JOB_ACCOUNT=bsc19 SLURM_JOB_NUM_NODES=1 COMPSS_MASTER_NODE=s01r2b48
    COMPSS_WORKER_NODES=",
"endTime": "2023-09-07T09:46:26+00:00",
"instrument": {"@id": "application_sources/matmul_files.py"},
"name": "COMPSs matmul_files.py execution at marenostrum4 with JOB_ID 30132875",
```

# Inspecting registered metadata

## CreateAction


```
"object": [{"@id": "dataset/A.0.0"}, {"@id": "dataset/A.0.1"}, {"@id": "dataset/A.1.0"}, {"@id":
    "dataset/A.1.1"}, {"@id": "dataset/B.0.0"}, {"@id": "dataset/B.0.1"}, {"@id": "dataset/B.1.0"},
    {"@id": "dataset/B.1.1"}, {"@id": "dataset/C.0.0"}, {"@id": "dataset/C.0.1"}, {"@id":
    "dataset/C.1.0"}, {"@id": "dataset/C.1.1"}],

"result": [{"@id": "dataset/C.0.0"}, {"@id": "dataset/C.0.1"}, {"@id": "dataset/C.1.0"}, {"@id":
    "dataset/C.1.1"}, {"@id": "./"}],

"subjectOf": ["https://userportal.bsc.es/"]
```

# Conclusions

- **FAIR HPC workflows** combining COMPSs + RO-Crate + WorkflowHub
  - WMS that use RO-Crate (Galaxy, Nextflow, Streamflow, Sapporo, Autosubmit)
- Paper* experiments show
  - We provide **automatic** provenance registration (whenever possible)
  - We are **efficient** (no run time overhead appreciated)
  - We can **scale** and deal with large workflows (shown by use cases)
- Future Work
  - Integration with: WfExS, ROHub (RO-Crate)
  - Automatic reproducibility with the PyCOMPSs CLI
  - Governance and Knowledge extraction

*Raül Sirvent et al. “Automatic, Efficient and Scalable Provenance Registration for FAIR HPC Workflows” In: 2022 IEEE/ACM Workshop on Workflows in Support of Large-Scale Science (WORKS). IEEE, 2022. p. 1-9.

# LIVE DEMO WITH WORKFLOWHUB


https://compss-doc.readthedocs.io/en/latest/Sections/05_Tools/04_Workflow_Provenance.html

**www.eFlows4HPC.eu**

@eFlows4HPC  eFlows4HPC Project

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955558. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway.