



18/11/2024



Optimising workflow lifecycle management: development, HPC-ready containers deployment and reproducibility

Raül Sirvent, Rosa M Badia

SC24 tutorial, Atlanta, 18 November 2024

Tutorial website

https://github.com/bsc-wdc/Tutorial_SC24



Agenda

8:30 – 8:35	Overview of tutorial agenda	Rosa M Badia
8:35 – 9:05	Part 1.1: Hybrid HPC+AI+DA workflow development with PyCOMPSs - Context of the workflows at BSC - Overview of workflow development with PyCOMPSs - Extensions for the integration of HPC with AI and DA - Sample workflows	Rosa M Badia
9:05 – 9:35	Part 1.2: Workflows' reproducibility through provenance - Motivation for workflow provenance - Design of the recording mechanism - Sharing experiments for reproducibility	Raül Sirvent
9:35 – 9:50	Part 1.3: HPC ready container images - Motivation for architecture specific containers - Overview of the Container Image Creation service - Example of HPC ready container generation	Rosa M Badia
9:50 – 10:00	Hands-on preparation (credentials distribution, how to access, etc)	All presenters



Agenda

10:00 - 10:30	Coffee break	
10:30 – 11:00	Part 2.1: Hands-on session: Sample workflows with PyCOMPSs, execution with containers, task-graph generation, tracefile generation (optional)	Rosa M Badia
11:00 – 11:45	Part 2.2: Hands-on session: How to automatically record workflow provenance and use it to share experiments in WorkflowHub	Raül Sirvent
11:45 - 12:00	Tutorial conclusions	All presenters

Outline

- Motivation and baseline technologies
- Design of Workflow Provenance recording
- Using Workflow Provenance with COMPSS

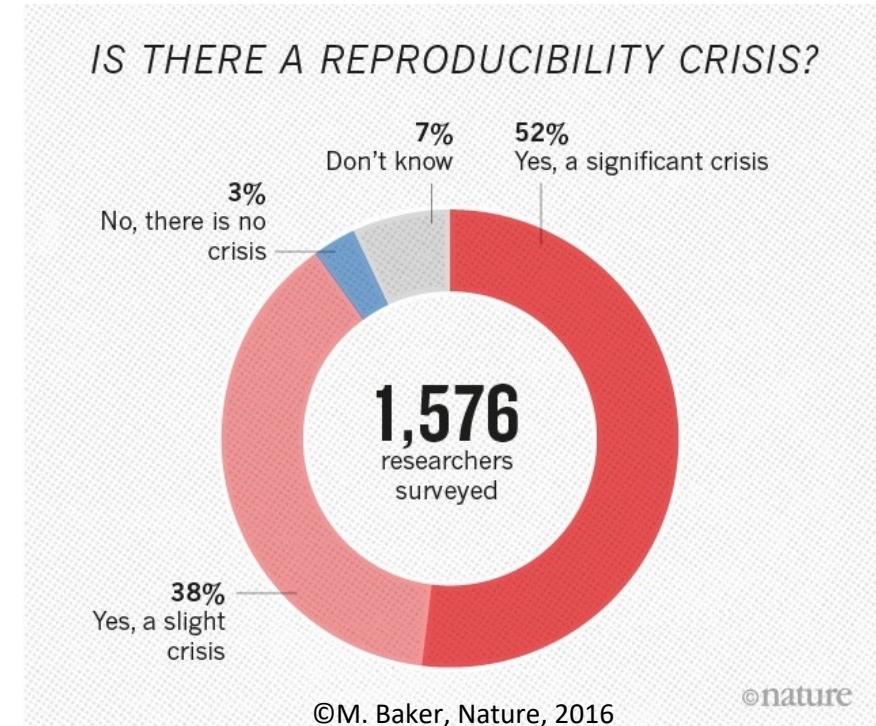
Motivation and baseline technologies



**Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

Motivation

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



Motivation

- Provenance is **MORE** than just Reproducibility
 - **Governance** (availability, usability, consistency, ...) (e.g. **FAIR Workflows**)
 - **Replicability** (exchange inputs)
 - **Knowledge extraction** (queries, mining)
 - **Traceability** (validation/verification, visualisation) (e.g. **AI Explainability**)
- Our claim: desired features for **Workflow Provenance** registration
 - **Automatic**: lower user burden
 - **Efficient**: no overheads
 - **Scalable**: large workflows (both tasks and data assets used)

Baseline: COMPSs



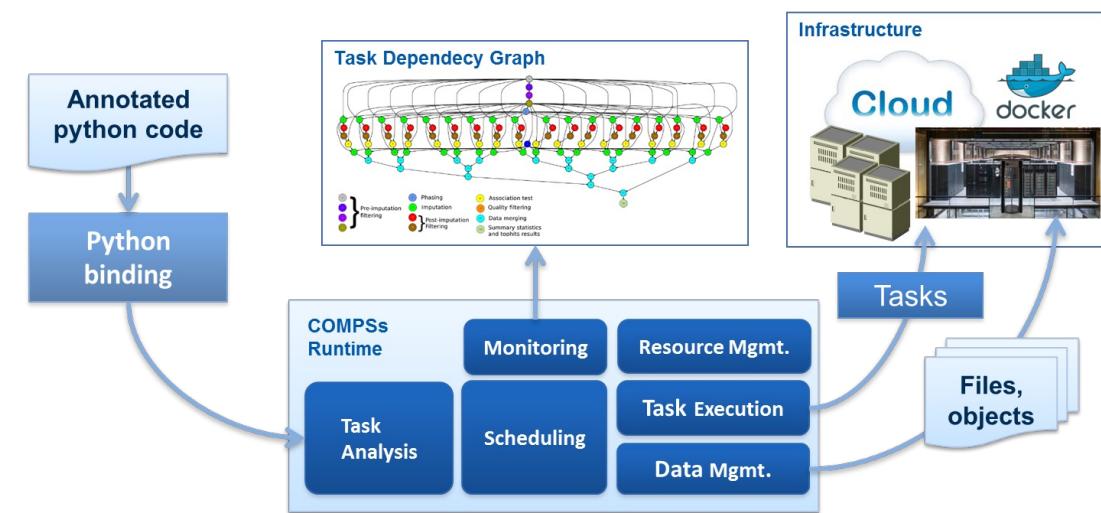
- Sequential programming, parallel execution
- General purpose programming language + annotations/hints (identify tasks and directionality of data)
- Builds a **task graph** at runtime (potential concurrency)
- Tasks can be **sequential, parallel** (threaded or MPI)
- Offers to applications a **shared 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

```
1 @task()
2 def word_count(block):
3     ...
4     return res
5
6 @task(f_res=INOUT)
7 def merge_count(f_res, p_res):
8     ...
```

(a) Task annotation example

```
1 for block in data:
2     p_result = word_count(block)
3     reduce_count(result, p_result)
4 result = compss_wait_on(result)
```

(b) Main code example



Baseline: 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, more than 30 tools/systems:
 - Describo, ro-crate-html, ro-crate-zendodo, ...
- We use:
 - ro-crate-py library (generation / consumption), rocrate-validator, WorkflowHub (interoperability)



Baseline: 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)



Simone Leo et al. “Recording provenance of workflow runs with RO-Crate” PLoS ONE 19(9): e0309210 (Sept 2024)

WMS/tools using WRROC for Provenance Recording



- runcrate
- rocrate-validator



Design of Workflow Provenance recording



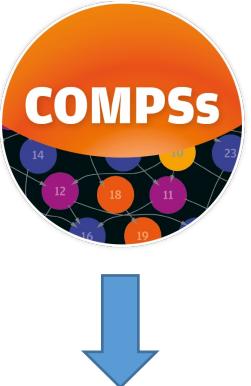
**Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

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)

After application finishes...

dataprovenance.log

- Lightweight approach: record file accesses, generate provenance later

generate_COMPSS_RO-Crate.py ro-crate-info.yaml

ro-crate-py 0.11.0

COMPSS_RO-Crate_[uuid]/

3.3.rc2407

```
lysozyme_in_water.py
App_Profile.json
2024-08-12T13:25:07.717499Z
file:///s01r2b54-ib0/home/msc19/msc19057/DP_Test_3_demo/dataset/2hs9.pdb IN
file:///s01r2b54-ib0/home/msc19/msc19057/DP_Test_3_demo/output/2hs9.gro OUT
file:///s01r2b54-ib0/home/msc19/msc19057/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

Generating Workflow Provenance with COMPSs



**Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

Steps to record Workflow Provenance in COMPSs

- Install ro-crate-py (if needed)
- Provide YAML information file
- Run with -p or –provenance=my_file.yaml
 - The *crate* is generated (a sub-folder COMPSs_RO-Crate_[uuid]/)

YAML information to be provided

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

COMPSs Workflow Information:

```
name: COMPSs Matrix Multiplication
description: Blocks as hypermatrix
license: Apache-2.0
sources: [src/, ~/java/matmul/xml/, ~/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
```

Agent:

```
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
```

Automatic search of ORCID and ROR

- For machines with internet connectivity only
- Gets the rest of missing items, if available
- Convenient for applications with large number of authors

COMPSS Workflow Information:

name: COMPSS Matrix Multiplication
description: Blocks as hypermatrix
license: Apache-2.0
sources: [src/, ~/java/matmul/xml/,
~/java/matmul/pom.xml, Readme]
data_persistence: True

Authors:

- name: Rosa M. Badia
- name: Nicolò Giacomini
- name: Fernando Vázquez Novoa
organisation_name: Barcelona Supercomputing Center
- name: Cristian Cătălin Tatu
- orcid: <https://orcid.org/0000-0001-5081-7244>
organisation_name: Barcelona Supercomputing Center
- orcid: <https://orcid.org/0000-0001-6401-6229>
ror: <https://ror.org/05r78ng12>
- name: Francesc Lordan
ror: <https://ror.org/05sd8tv96>

Agent:

orcid: <https://orcid.org/0000-0003-0606-2512>

Using Workflow Provenance with COMPSS



**Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

What you can do with the generated provenance

- Publish your experiment in WorkflowHub
- PyCOMPSs CLI (v3.3.4): `pycompss inspect`
- COMPSs Reproducibility Service

Steps to record and publish Workflow Provenance in COMPSS

- Publish it at WorkflowHub, **uploading the crate or using GitHub**
- Generate a DOI, cite your results in papers
- **Large datasets** can be shared in external repositories (e.g. Zenodo, FigShare), and included as a reference



Sharing Use Cases with WorkflowHub



- Asset sharing focus:
 - Share your **workflow** (+some example usage/test datasets attached)
 - Share your **dataset results** (+ the workflow that generated them, for reproducibility)
- Sharing audience:
 - **Privately** upload your code or data in a team, to share with your peers
 - Upload the most interesting code versions / output datasets
 - Once something is worth sharing **publicly** (code or data), change the permission of the specific result, generate DOI, cite it, etc...

SC Conference Reproducibility Initiative

- Artifacts Available ✓

- Artifacts used in the research (including data and code) are permanently archived in a **public repository** that assigns a **global identifier** and **guarantees persistence**, and are made available via **standard open licenses** that maximize artifact availability



- Artifacts Evaluated-Functional ✓

- Documentation: Are the artifacts sufficiently **documented** to enable them **to be exercised** by readers of the paper?
- Completeness: Do the submitted artifacts **include all of the key components** described in the paper?
- Exercisability: Do the submitted artifacts **include the scripts and data needed to run the experiments** described in the paper, and can the software be successfully executed?



- Results Replicated

- Reproduce Behavior: determine the equivalent or approximate behavior on available hardware
- Reproduce the Central Results and Claims of the Paper



pycompss inspect [crate_folder/ | crate.zip]

- Visualise metadata content through the console in a friendly way

```
COMPSS_R0-Crate_310c93f5-ff00-45b2-aaf7-de202bbf1349/
├── Date Published
│   └── Friday, 20 of September of 2024 - 09:14 UTC
├── Name
│   └── COMPSS Matrix Multiplication, out-of-core using files
├── Authors
│   └── Raül Sirvent (Barcelona Supercomputing Center) (Raul.Sirvent@bsc.es)
├── License
│   └── CC-BY-NC-ND-4.0
├── Software Dependencies
│   ├── NumPy
│   └── GROMACS
├── COMPSS Runtime version
│   └── 3.3.rc2408
├── R0-Crate Profiles compliance
│   ├── Process Run Crate (0.5)
│   ├── Workflow Run Crate (0.5)
│   ├── Provenance Run Crate (0.5)
│   └── Workflow R0-Crate (1.0)
├── Description
│   └── Hypermatrix size 2x2 blocks, block size 2x2 elements
>CreateAction (execution details)
├── Agent
│   └── Rosa M. Badia (Universitat Politècnica de Catalunya) (Rosa.M.Badia@upc.edu)
├── Application's main file
│   └── matmul_files.py
├── Hostname
│   └── MacBook-Pro-Raul-2018.local
├── Description (machine details)
│   └── Darwin MacBook-Pro-Raul-2018.local 23.6.0 Darwin Kernel Version 23.6.0: Mon Jul 29 21:13:00 PDT 2024; root:xnu-10063.141.2~1/RELEASE_X86_64 x86_64
├── Environment
│   └── COMPSS_HOME = /Users/rsirvent/opt/COMPSS/
└── Resource Usage
    ├── #localhost.matmul_tasks.multiply.maxTime = 1015
    ├── #localhost.matmul_tasks.multiply.executions = 8
    ├── #localhost.matmul_tasks.multiply.avgTime = 553
    ├── #localhost.matmul_tasks.multiply.minTime = 94
    └── #overall.matmul_tasks.multiply.maxTime = 1015
...

```

```
└── Start Time
    └── Friday, 20 of September of 2024 - 09:14:41 UTC
└── End Time
    └── Friday, 20 of September of 2024 - 09:14:48 UTC
└── TOTAL EXECUTION TIME
    └── 0:00:07 s
└── INPUTS
    └── dataset/A.0.0 (16 bytes)
...
└── dataset/C.1.1 (20 bytes)
    └── https://zenodo.org/records/10782431/files/lysozyme_datasets.zip (267,235,780 bytes)
└── OUTPUTS
    ├── dataset/C.0.0 (20 bytes)
    ├── dataset/C.0.1 (20 bytes)
    ├── dataset/C.1.0 (20 bytes)
    └── dataset/C.1.1 (20 bytes)
...
└── https://zenodo.org/records/10783183/files/results_2003_0521_boumardes_BS.tar.gz (711,341,852 bytes)
    └── ./
```

- Helps to inspect crate zipped files, without having to unzip them

Reproducibility Service

- Reproduce a previously run COMPSS experiment

```
compss_reproducibility_service [ro_crate_folder/ | url ]
```

- Example

```
compss_reproducibility_service https://workflowhub.eu/workflows/1072/ro_crate?version=2
```

- Two main use cases:
 - Data persistence:
 - Experiment can be re-executed on any machine
 - Non data persistence:
 - Re-execute on a machine that has access to the data assets paths



Reproducibility Service

- Metadata is used to verify input files
 - Ensure re-execution will be done with EXACTLY the same data assets (size, mod_date)

S.No.	Filename	File Path	File Accessible	File Size Verified
1	application_sources/src/wc_reduce.py	/home/archit/Desktop/COMPSS-Reproducibility-Service/reproducibility_service_20240820_012206/Workflow/application_sources/src/wc_reduce.py	✓	✓
2	file0.txt	/home/archit/Desktop/COMPSS-Reproducibility-Service/reproducibility_service_20240820_012206/Workflow/dataset/dataset_4f_16mb/file0.txt	✓	✓
3	file1.txt	/home/archit/Desktop/COMPSS-Reproducibility-Service/reproducibility_service_20240820_012206/Workflow/dataset/dataset_4f_16mb/file1.txt	✓	✓
4	file2.txt	/home/archit/Desktop/COMPSS-Reproducibility-Service/reproducibility_service_20240820_012206/Workflow/dataset/dataset_4f_16mb/file2.txt	✓	✓
5	file3.txt	/home/archit/Desktop/COMPSS-Reproducibility-Service/reproducibility_service_20240820_012206/Workflow/dataset/dataset_4f_16mb/file3.txt	✓	✓

✓: SUCCESS | ✘: FAILURE | -: NOT IN METADATA

Metadata format



**Barcelona
Supercomputing
Center**

Centro Nacional de Supercomputación

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"}
"softwareRequirements": [{"@id": "https://numpy.org/"}, {"@id":
  "https://www.gromacs.org/"}]
```



```
"@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": "3.3.rc2408"
```

```
"@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"
"sha256": "44018036dc20569b90104113928291e997f2f5cecb2e01af19b5dc2e0043eb6f"
```

Inspecting registered metadata

Software Dependencies

```
{  
    "@id": "https://numpy.org/",  
    "@type": "SoftwareApplication",  
    "name": "NumPy",  
    "url": "https://numpy.org/",  
    "version": "1.24.1"  
},  
{  
    "@id": "https://www.gromacs.org/",  
    "@type": "SoftwareApplication",  
    "name": "GROMACS",  
    "url": "https://www.gromacs.org/",  
    "version": ":-) GROMACS - gmx, 2024.2-Homebrew (-:\nExecutable:  
/usr/local/bin/..../Cellar/gromacs/2024.2/bin/gmx\nData prefix:  
/usr/local/bin/..../Cellar/gromacs/2024.2\nWorking dir:  
/Users/rsirvent/.COMPSs/lysozyme_in_water_full_no_mpi.py_30\nCommand line:\n  gmx --  
version\n  nGROMACS version: 2024.2-Homebrew\n  Precision: mixed\n  Memory model:  
  64 bit\n  MPI library: thread_mpi\n  OpenMP support: enabled (GMX_OPENMP_MAX_THREADS =  
  128)\n",  
},
```

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_submission_command_line.txt",  
"@type": "File",  
"contentSize": 709,  
"description": "COMPSs submission command line  
  (runcompss / enqueue_compss), including flags  
  and parameters passed to the application",  
"encodingFormat": "text/plain",  
"name": "compss_submission_command_line.txt"  
"sha256":  
  "7093e1aa31723f8748148e8413fc7831fc3df2bcabe26  
  0775f9e3ab86f117e48"
```

COMPSs Task Profiling (deprecated)

```
"@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"  
"sha256": "83f7239637dd90008d48a5b99f6d36bf478ec9db9a912126af856a0c8747670"
```

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/1331.pdb",
"@type": "File",
"contentSize": 116154,
"dateModified": "2022-04-20T13:20:58",
"name": "1331.pdb",
"sdDatePublished": "2022-10-18T08:03:08+00:00"
```

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



Inspecting registered metadata

CreateAction

```
"@id": "#COMPSs_Workflow_Run_Crate_marenostrum5_SLURM_JOB_ID_5143097",
"@type": "CreateAction",
"actionStatus": {"@id": "http://schema.org/CompletedActionStatus"},
"name": "COMPSs matmul_files.py execution at marenostrum5 with JOB_ID 5143097",
"description": "Linux gs14r1b58 5.14.0-284.30.1.el9_2.x86_64 #1 SMP PREEMPT_DYNAMIC Fri Aug 25
 09:13:12 EDT 2023 x86_64 x86_64 x86_64 GNU/Linux"
"instrument": {"@id": "application_sources/matmul_files.py"},
"agent": {"@id": "https://orcid.org/0000-0003-0606-2512"},
"environment": [{"@id": "#slurm_job_user"}, ... {"@id": "#compss_worker_nodes"}],
"resourceUsage": [{"@id": "#localhost.matmul_tasks.multiply.maxTime"}, ... {"@id":
  "#overall.matmul_files.py.executionTime"}],
"startTime": "2024-09-02T14:22:40+00:00",
"endTime": "2024-09-02T14:22:47+00:00",
"subjectOf": ["https://userportal.bsc.es/"]
```

Inspecting registered metadata

CreateAction

```
{  
    "@id": "#slurm_job_user",  
    "@type": "PropertyValue",  
    "name": "SLURM_JOB_USER",  
    "value": "bsc019057"  
},  
{  
    "@id": "#slurm_job_uid",  
    "@type": "PropertyValue",  
    "name": "SLURM_JOB_UID",  
    "value": "2952"  
},  
{  
    "@id": "#slurm_submit_dir",  
    "@type": "PropertyValue",  
    "name": "SLURM_SUBMIT_DIR",  
    "value": "/gpfs/home/bsc/bsc019057/COMPSS-DP"  
},  
...  
}
```

```
{  
    "@id": "#localhost.matmul_tasks.multiply.maxTime",  
    "@type": "PropertyValue",  
    "name": "maxTime",  
    "propertyID":  
        "https://w3id.org/ro/terms/compss#maxTime",  
    "unitCode": "https://qudt.org/vocab/unit/MilliSEC",  
    "value": "369"  
},  
{  
    "@id": "#localhost.matmul_tasks.multiply.executions",  
    "@type": "PropertyValue",  
    "name": "executions",  
    "propertyID":  
        "https://w3id.org/ro/terms/compss#executions",  
    "value": "8"  
},  
...  
}
```

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": "./"}],
```

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
 - Detect and Install software dependencies
 - Governance and Knowledge extraction (from execution metrics)
 - XAI: eXplainable Artificial Intelligence

*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.

BoF: Reproducibility and Provenance for HPC and Distributed Environments



Rocío Carratalá-Sáez



Michela Taufer



Tanzima Islam



Raül Sirvent

Tuesday, 19 November 2024
5:15pm - 6:45pm EST
Room B201

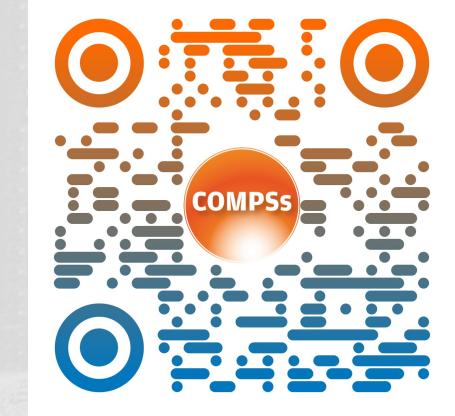


SC24
Atlanta, GA | hpc creates.



**Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

Thanks!



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

Raul.Sirvent@bsc.es