

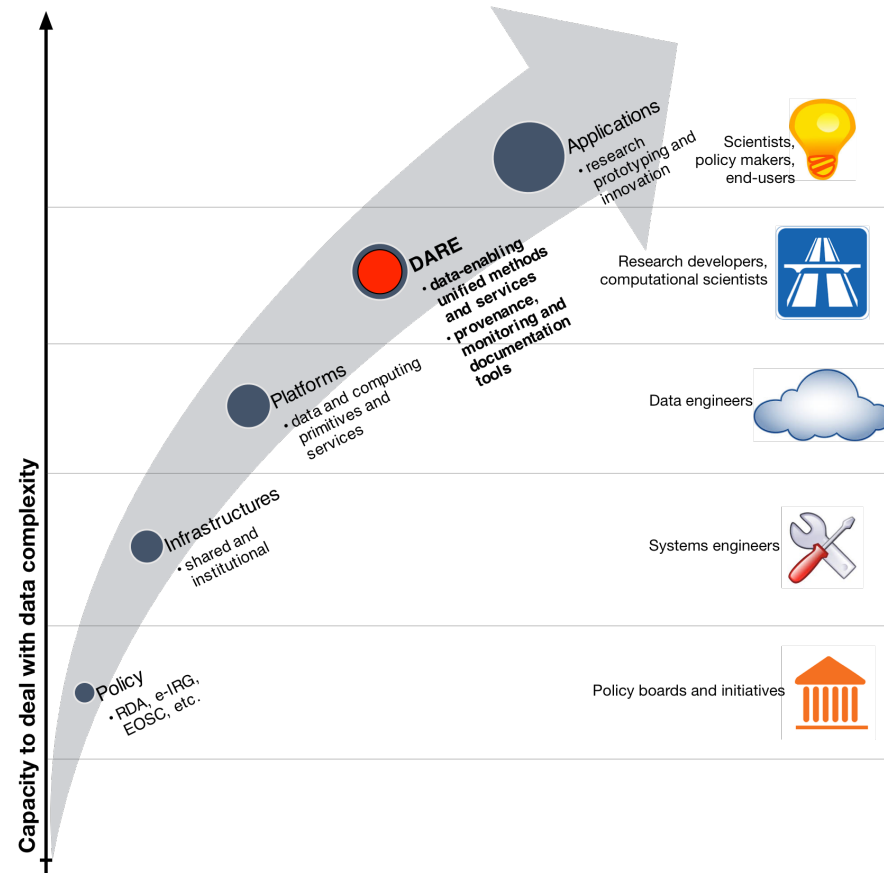
# DARE

## Integrating solutions for Data-Intensive and Reproducible Science

Alessandro Spinuso (KNMI) & The DARE Team

# DARE

## Delivering Agile Research Excellence on European e-Infrastructures



**Working environment** for professionals wrestling with challenges involving complexity of methods and data

*Domain Expert - Computational Scientists - Research Developers*

- **Mapping from abstract methods to concrete applications** executed by different enactments seamlessly
- **Validation and Traceability of runs and products: diagnose, monitor, repeat**
- **Organisation of campaigns** reusing data and methods from **multiple runs**

**Accelerate productivity of expert teams.**

# Requirements Collected from EPOS/IS-ENES



# Requirements Collected from EPOS/IS-ENES

- **Control over** heterogeneous applications (HPC simulation, data processing).
- **Develop new atomic functions adopting community libraries.**
- **Access to distributed data sources** to gather input data products.



# Requirements Collected from EPOS/IS-ENES

- **Control over** heterogeneous applications (HPC simulation, data processing).
- **Develop new atomic functions adopting community libraries.**
- **Access to distributed data sources** to gather input data products.



- **Abstract** workflow pipelines from implementation.
- **Process large data volumes**, near(er) to the data storage.
- **Exploit Research infrastructures and e-infrastructures services (ESGF, EOSC).**



# Requirements Collected from EPOS/IS-ENES

- **Control over** heterogeneous applications (HPC simulation, data processing).
- **Develop new atomic functions adopting community libraries.**
- **Access to distributed data sources** to gather input data products.



- **Find, Repeat, Reuse and Trace** execution's outcomes.
- **Integration within existing community portals, web-services.**
- **Automatic allocation of resources for large computations.**

- **Abstract** workflow pipelines from implementation.
- **Process large data volumes**, near(er) to the data storage.
- **Exploit Research infrastructures and e-infrastructures services (ESGF, EOSC).**

is-enes  
INFRASTRUCTURE FOR THE EUROPEAN NETWORK  
FOR EARTH SYSTEM MODELLING



# DARE Unified Actions and Remote APIs

## User Facing Tools

Integration in  
Development  
environments



Invoked on  
demand  
climate4impact  
VERCE



# DARE Unified Actions and Remote APIs

## User Facing Tools

Integration in  
Development  
environments



Invoked on  
demand  
climate4impact  
VERCE



## Unified Actions

Manage Working  
Sessions

Validate, Repeat,  
Reuse Results

Configure  
Run

Execute  
Applications  
and Workflows

Compose  
Register  
Workflows






# DARE Unified Actions and Remote APIs

## User Facing Tools

Integration in Development environments



Invoked on demand

climate4impact  
VERCE



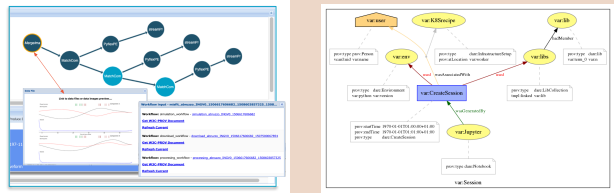
Open Geospatial Consortium, Inc.

## Unified Actions

- Manage Working Sessions
- Validate, Repeat, Reuse Results
- Configure Run
- Execute Applications and Workflows
- Compose Register Workflows

## Services (Provenance, Execution and Catalogues)

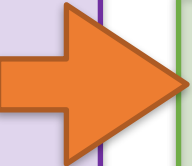
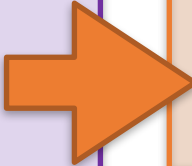
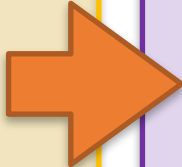
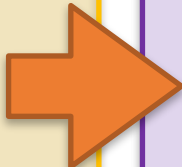

### Provenance Services



### Computational Contexts allocation and customisation Services



### Workflow Mapping and Execution



# DARE Unified Actions and Remote APIs

## User Facing Tools

Integration in  
Development  
environments



Invoked on  
demand  
climate4impact  
VERCE



## Unified Actions

Manage Working  
Sessions

Validate, Repeat,  
Reuse Results

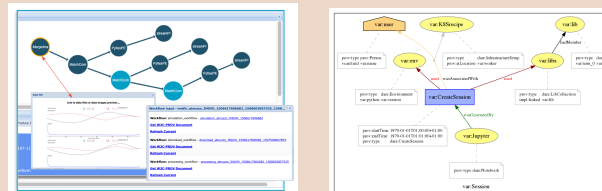
Configure  
Run

Execute  
Applications  
and Workflows

Compose  
Register  
Workflows

## Services (Provenance, Execution and Catalogues)

### Provenance Services



### Computational Contexts allocation and customisation Services



### Workflow Mapping and Execution

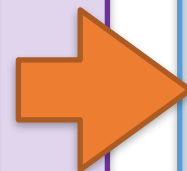
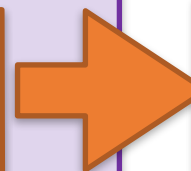
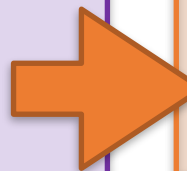
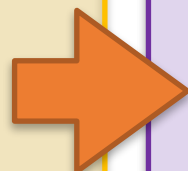
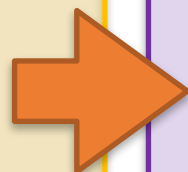


### DARE Catalogues for Data and Methods



User  
interaction  
and lineage

Optimisation

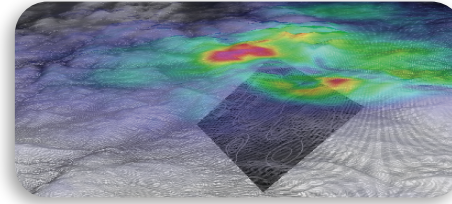


# Develop, Describe and Run Workflows

## Test Case: Rapid Assessment Workflow Decomposition

Reusable Tasks running at different scale.  
May require human monitoring and intervention

Rapid Ground Motion Assessment (RA)



Choose/upload seismic wavespeed & mesh

run waveform simulation

Get pre-processed synt and data

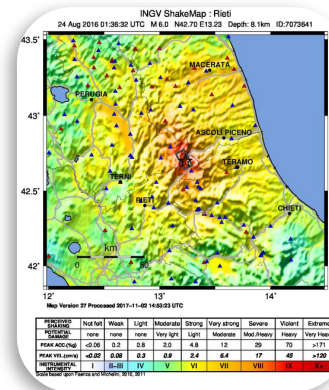
Get ground motion parameters

Compare/integrate synthetic and observed ground motion data

Store data, metadata, provenance

Choose/upload seismic source (point or fault)

Gather observed data



# Develop, Describe and Run Workflows

## Test Case: Rapid Assessment Workflow Decomposition

Reusable Tasks running at different scale.  
 May require human monitoring and intervention

Rapid Ground Motion Assessment (RA)

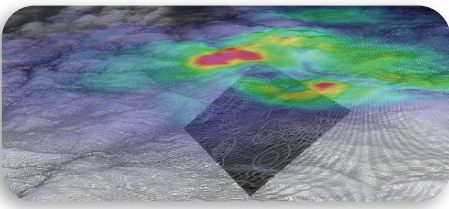
Choose/upload seismic wavespeed & mesh

Choose/upload seismic source (point or fault)

MPI Simulation  
 run waveform simulation

Gather observed data

Get pre-processed synt and data



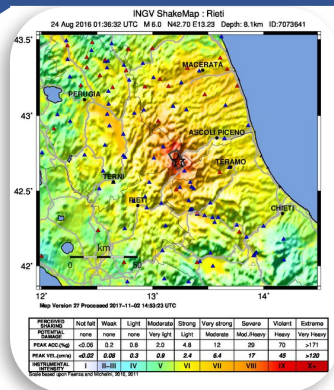
Get ground motion parameters

Compare/integrate synthetic and observed ground motion data

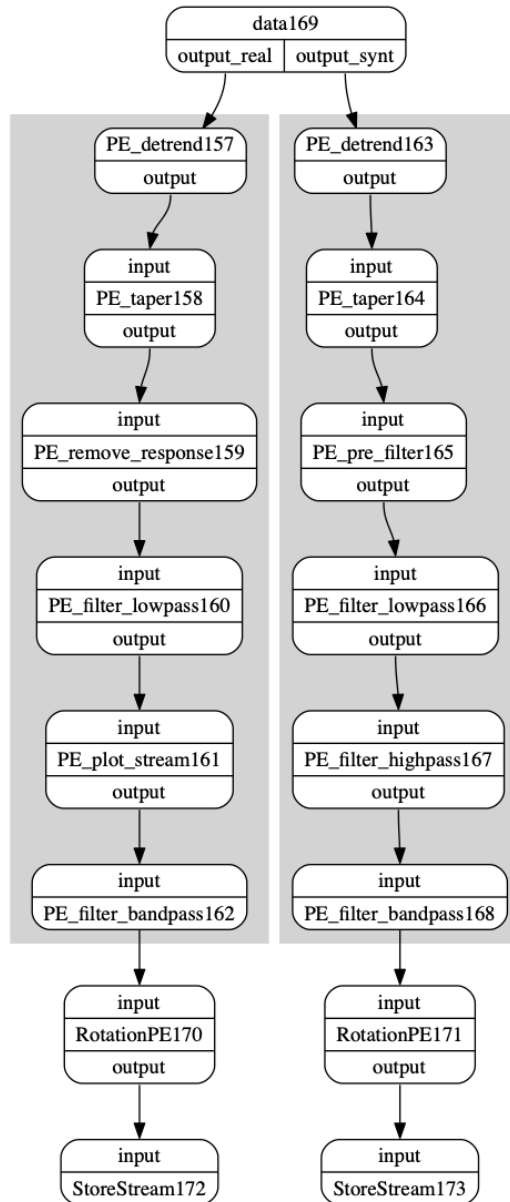
Store data, metadata, provenance

Data Analysis  
 Waveform Preprocessing

Misfit Analysis



## Waveform Preprocessing



## pipeline JSON Description (eg. from file)

## Manual Extensions

## Workflow encoded in Python

```
def buildWorkflow():
    real_preprocess = create_processing_chain(proc['data_processing'])
    synt_preprocess = create_processing_chain(proc['synthetics_processing'])
    print(real_preprocess)
    graph = WorkflowGraph()
    read = ReadDataPE()
    read.name = 'data'
    read.output_units = proc['output_units']
    rotate_real = RotationPE('data')
    rotate_synt = RotationPE('synth')
    store_real = StoreStream('data')
    store_synt = StoreStream('synth')
    graph.connect(read, 'output_real', real_preprocess, 'input')
    graph.connect(read, 'output_synt', synt_preprocess, 'input')
    if proc['rotate_to_ZRT']:
        graph.connect(real_preprocess, 'output', rotate_real, 'input')
        graph.connect(synt_preprocess, 'output', rotate_synt, 'input')
        graph.connect(rotate_real, 'output', store_real, 'input')
        graph.connect(rotate_synt, 'output', store_synt, 'input')
    else:
        graph.connect(real_preprocess, 'output', store_real, 'input')
        graph.connect(synt_preprocess, 'output', store_synt, 'input')

    return graph
```

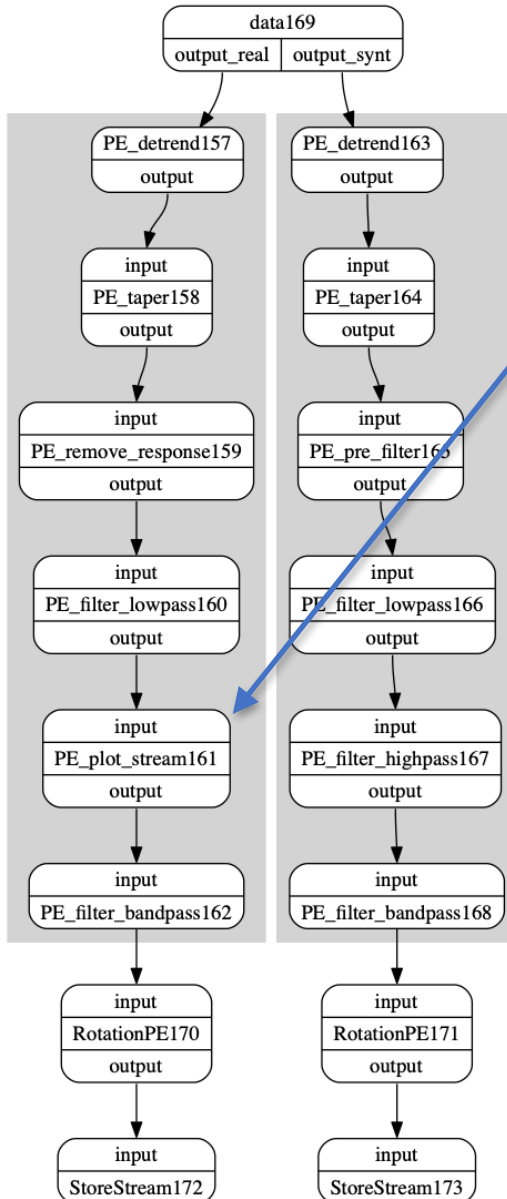
```
graph=buildWorkflow()
```

```
from dispel4py.visualisation import display
display(graph)
```



# Provenance Configuration for lineage usability

Waveform  
Preprocessing



pipeline  
JSON  
Description  
(eg. from file)

Manual  
Extensions

## Functions encoded in Python with User Defined Metadata

```
def plot_stream(stream, output_dir, source, tag):  
    stats = stream[0].stats  
    filename = source + "-%s.%s.%s.%s.png" % (  
        stats['network'], stats['station'], stats['channel'], tag)  
    path = os.environ['STAGED_DATA'] + '/' + output_dir
```

```
    if not os.path.exists(path):  
        try:  
            os.makedirs(path)  
        except:  
            pass
```

```
    dest = os.path.join(path, filename)  
    stream.plot(outfile=dest)  
    #return stream
```

```
    prov = {'location': "file://" + socket.gethostname() + "/" + dest,  
           'format': 'image/png',  
           'metadata': {'myterm': tag}}
```

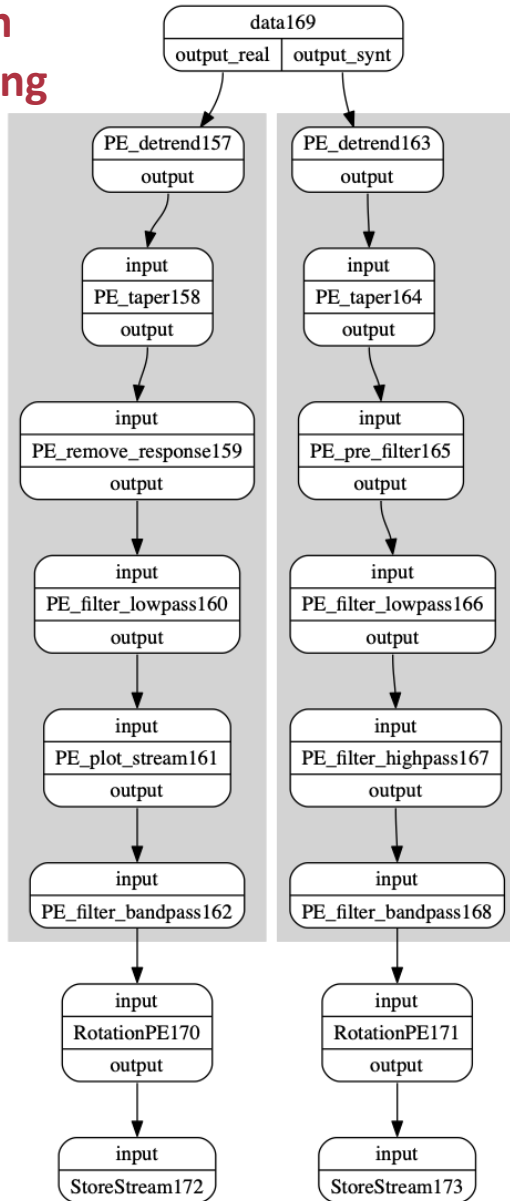
```
    return {'_d4p_prov': prov, '_d4p_data': stream}
```



User Defined Metadata

# Provenance Configuration for lineage usability

## Waveform Preprocessing

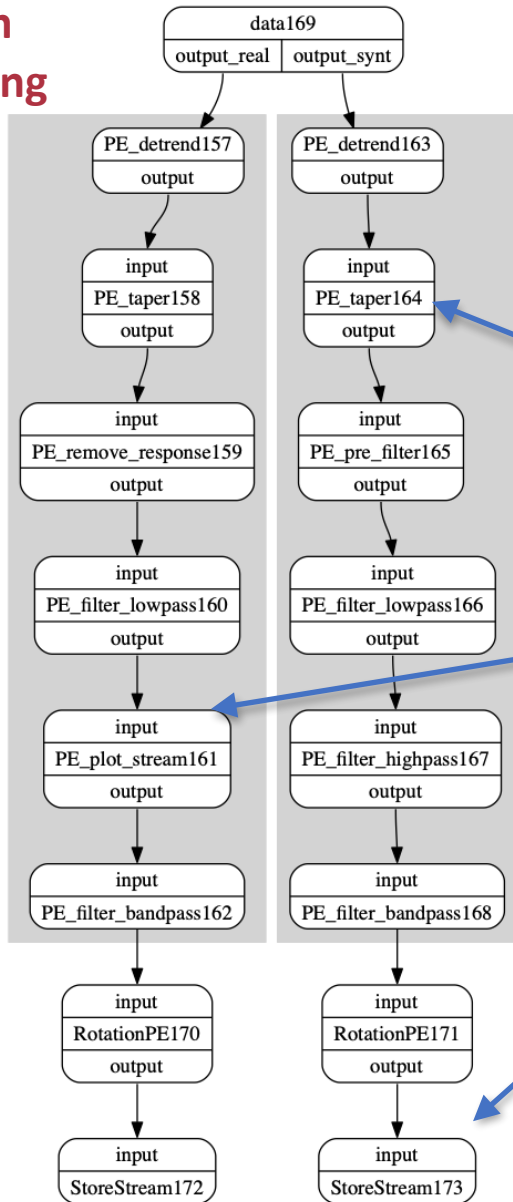


## Configuration Profile in JSON with Provenance Types

```
{  
  'provone:User': "aspinuso",  
  's-prov:description' : "provdemo demokritos",  
  's-prov:workflowName' : "demo_epos",  
  's-prov:workflowType' : "seis:preprocess",  
  's-prov:workflowId' : "workflow process",  
  's-prov:save-mode' : 'service',  
  # defines the Provenance Types and Provenance Clusters for the Workflow Components  
  's-prov:componentsType' :  
    { 'PE_taper' : { 's-prov:type':(SeismoPE, ),  
                    's-prov:prov-cluster': 'seis:Processor' },  
      'PE_plot_stream' : { 's-prov:prov-cluster': 'seis:Visualisation',  
                           's-prov:type':(SeismoPE, ) },  
      'StoreStream' : { 's-prov:prov-cluster': 'seis:DataHandler',  
                        's-prov:type':(SeismoPE, ) }  
    }  
}
```

# Provenance Configuration for lineage usability

## Waveform Preprocessing



## Configuration Profile in JSON with Provenance Types

```
{  
  'provone:User': "aspinuso",  
  's-prov:description' : "provdemo demokritos",  
  's-prov:workflowName' : "demo_epos",  
  's-prov:workflowType' : "seis:preprocess",  
  's-prov:workflowId' : "workflow process",  
  's-prov:save-mode' : 'service',  
  # defines the Provenance Types and Provenance Clusters for the Workflow Components  
  's-prov:componentsType' :  
    { 'PE_taper' : { 's-prov:type' : (SeismoPE, ),  
                   's-prov:prov-cluster' : 'seis:Processor' },  
      'PE_plot_stream' : { 's-prov:prov-cluster' : 'seis:Visualisation',  
                          's-prov:type' : (SeismoPE, ) },  
      'StoreStream' : { 's-prov:prov-cluster' : 'seis:DataHandler',  
                       's-prov:type' : (SeismoPE, ) }  
    }  
}
```

starttime: 2013-02-16T21:16:09.240000Z

delta: 0.01

calib: 1

sampling\_rate: 100

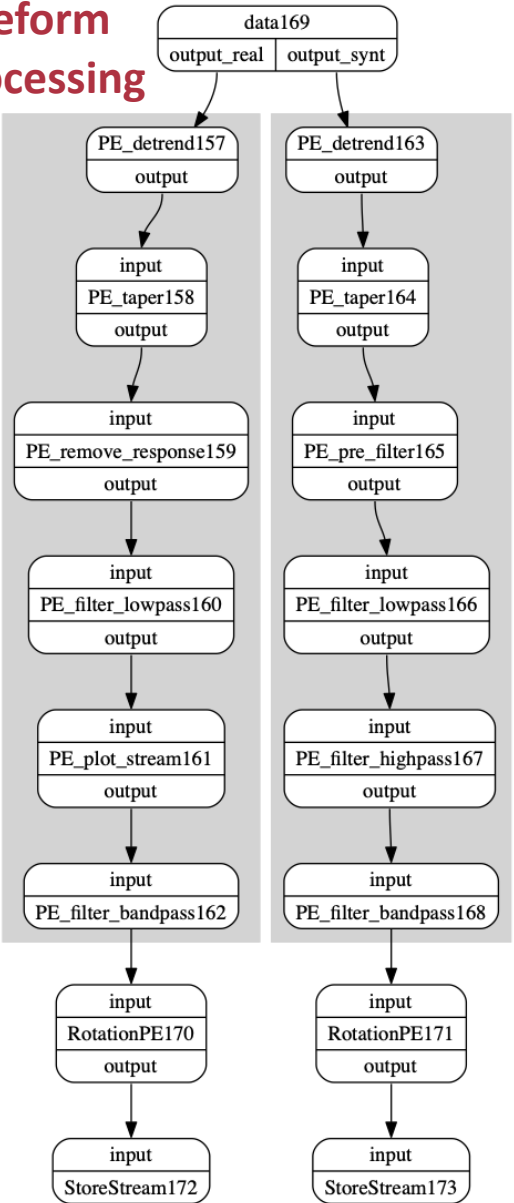
**ProvenanceType for outputs' Metadata Contextualisation and Lineage Patterns**



# Monitor, search and analyse results through lineage



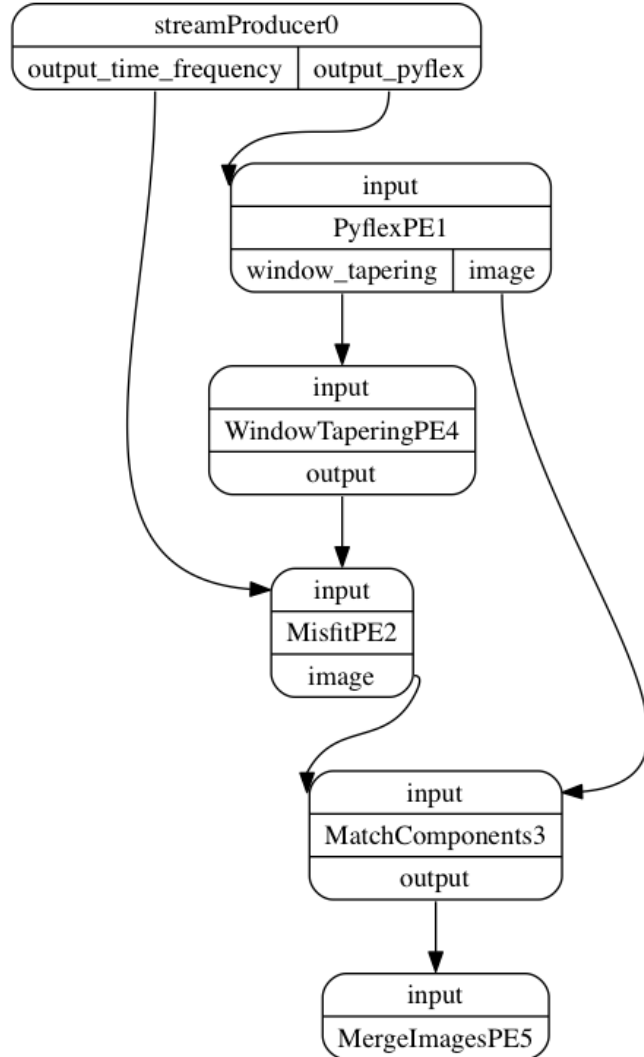
## Waveform Preprocessing



The screenshot shows a 'Data Dependency Graph' interface. At the top, it says 'Data Dependency Graph' and 'Double Click on the border data-nodes to expand. Right Click on each data-node to access its info'. Below this is a legend for node types: trace-bw, trace-fw, stateful, cross-run, file, Incomplete. The graph shows a sequence of nodes: PE\_taper (highlighted with a red dashed box and labeled 'SeismoType provenance capturing'), PE\_pre\_f, PE\_filt\_e, and PE\_plot\_ (highlighted with a red dashed box and labeled 'Inline provenance capturing'). A 'Data Detail' window is open over the PE\_plot\_ node, showing 'Output Files : Open', 'Output Metadata: station: ARRO, myterm : observed, network: IV' (labeled 'User Defined Metadata'), and 'ProvenanceType Metadata: starttime: 2013-02-16T21:16:09.240000Z, delta: 0.01, calib: 1, sampling\_rate: 100'. At the bottom, there is a 'Data products' section with buttons for 'Search', 'Filter Current', and 'Produce Download Script', and 'Output Files :', 'Output Metadata:' sections.

# Monitor, search and analyse results through lineage

## Misfit Analysis



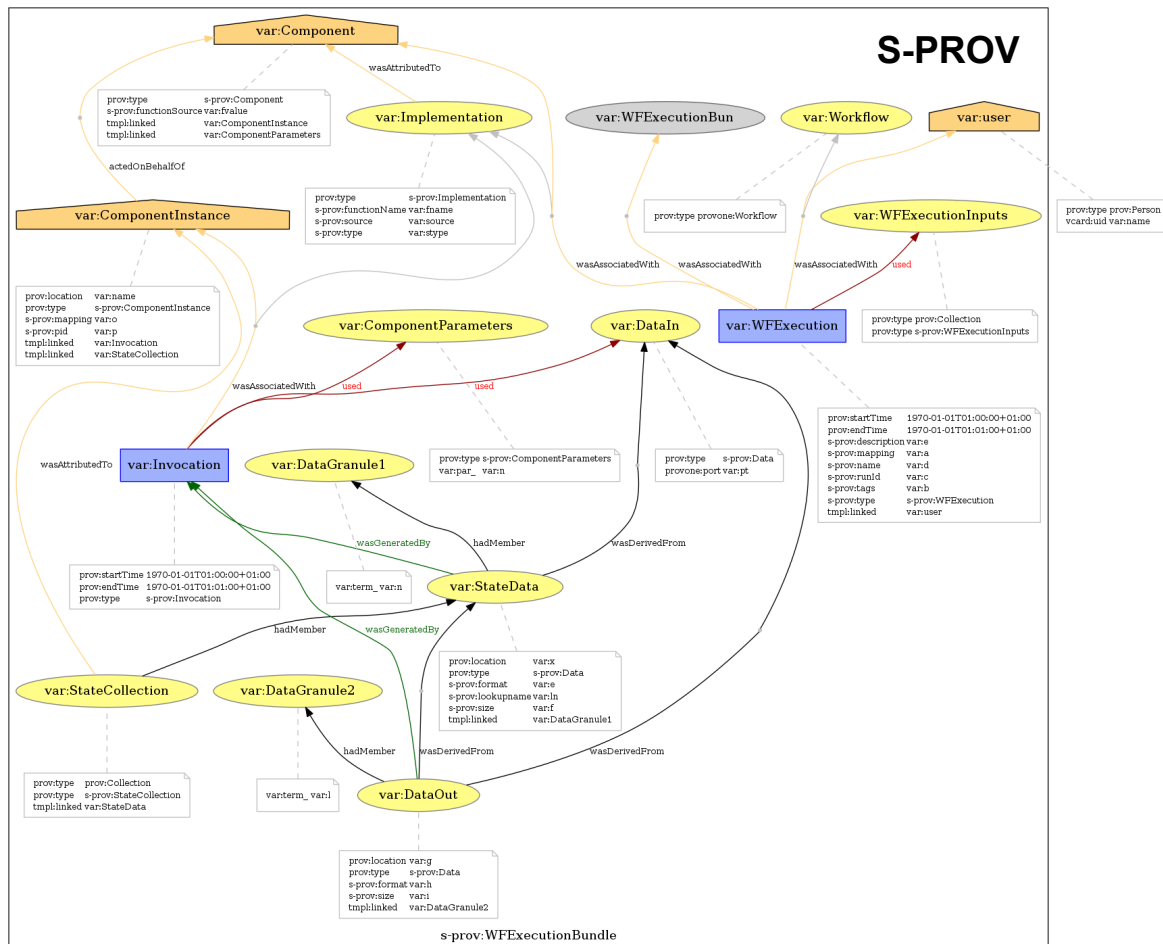
**Workflow input - misfit\_abruzzo\_INGV0\_150661760682\_1508603857325\_1508...**

- Workflow:** simulation\_workflow - [simulation\\_abruzzo\\_INGV0\\_150661760682](#)  
[Get W3C-PROV Document](#)  
[Refresh Current](#)
- Workflow:** download\_workflow - [download\\_abruzzo\\_INGV0\\_150661760682\\_1507500067854](#)  
[Get W3C-PROV Document](#)  
[Refresh Current](#)
- Workflow:** processing\_workflow - [processing\\_abruzzo\\_INGV0\\_150661760682\\_1507500067854](#)  
[Get W3C-PROV Document](#)  
[Refresh Current](#)

Interconnected workflows executions

# Large Scale Lineage Representation, Management, Exploitation

Provenance Model, Services and Tools combining system, domain and user-driven information about the computation



## Processes Runtime Monitor

**Processes Runtime Monitor**

Results: `aspluhsu - Runtime Instances monitor - CORR_LARGE_orfeus-ws-92819-7405d3c-9624-11e7-a...`

ID	Last Event	Data count	Worker	Message	Changed
1	MaxClipe-Ins...	2017-09-10 1...	6		
2	CorrCoef-Ins...	2017-09-10 1...	9900		
3	CorrCoef-Ins...	2017-09-10 1...	10100		
4	Source-Instan...	2017-09-10 1...	2		
5	Source-Instan...	2017-09-10 1...	2		
6	Source-Instan...	2017-09-10 1...	2		
7	Source-Instan...	2017-09-10 1...	2		
8	Source-Instan...	2017-09-10 1...	2		
9	Source-Instan...	2017-09-10 1...	2		
10	Source-Instan...	2017-09-10 1...	2		
11	Source-Instan...	2017-09-10 1...	2		
12	Source-Instan...	2017-09-10 1...	2		
13	Source-Instan...	2017-09-10 1...	2		
14	Source-Instan...	2017-09-10 1...	2		
15	Source-Instan...	2017-09-10 1...	2		
16	Source-Instan...	2017-09-10 1...	2		
17	Source-Instan...	2017-09-10 1...	2		
18	Source-Instan...	2017-09-10 1...	2		
19	Source-Instan...	2017-09-10 1...	2		
20	Source-Instan...	2017-09-10 1...	2		
21	Source-Instan...	2017-09-10 1...	2		
22	Source-Instan...	2017-09-10 1...	2		
23	Source-Instan...	2017-09-10 1...	2		
24	Source-Instan...	2017-09-10 1...	2		
25	Source-Instan...	2017-09-10 1...	2		
26	Source-Instan...	2017-09-10 1...	2		

**Data Dependency Graph**

Interactive Navigation

State Derivation

Data Resource

Data File

Data Products and Metadata

Output Metadata:

- Iteration: 2
- order: 11
- cliq: [6, 7, 0, 3, 4, 10, 16, 11, 15, 12, 13]

**Monitoring**  
System and user messages, changes, data counter, worker

**Discovery and Filter**  
Search current run for data or filter the current resultset on ancestors' properties

**Preview and Staging**  
Access to concrete data resources for preview or download  
Produce download script for the whole resultset. Allow large transfers with delegated authorisation.

## S-ProvFlow



A. Spinuso: Active Provenance for Data-Intensive research: <https://www.era.lib.ed.ac.uk/handle/1842/33181>

S-ProvFlow: <https://github.com/KNMI/s-provenance>

# S-ProvFlow Lineage API

## lineage

GET /data

POST /data/filterOnAncestor

GET /data/{data\_id}

GET /data/{data\_id}/derivedData

GET /data/{data\_id}/wasDerivedFrom

GET /instances/{instid}

GET /invocations/{invocid}

GET /terms

## export

GET /data/{data\_id}/export

GET /workflowexecutions/{run\_id}/export

## summaries

GET /summaries/collaborative

GET /summaries/workflowexecution

## discovery

GET /workflowexecutions

DELETE /workflowexecutions/{runid}

GET /workflowexecutions/{runid}

## acquisition

POST /workflowexecutions/insert

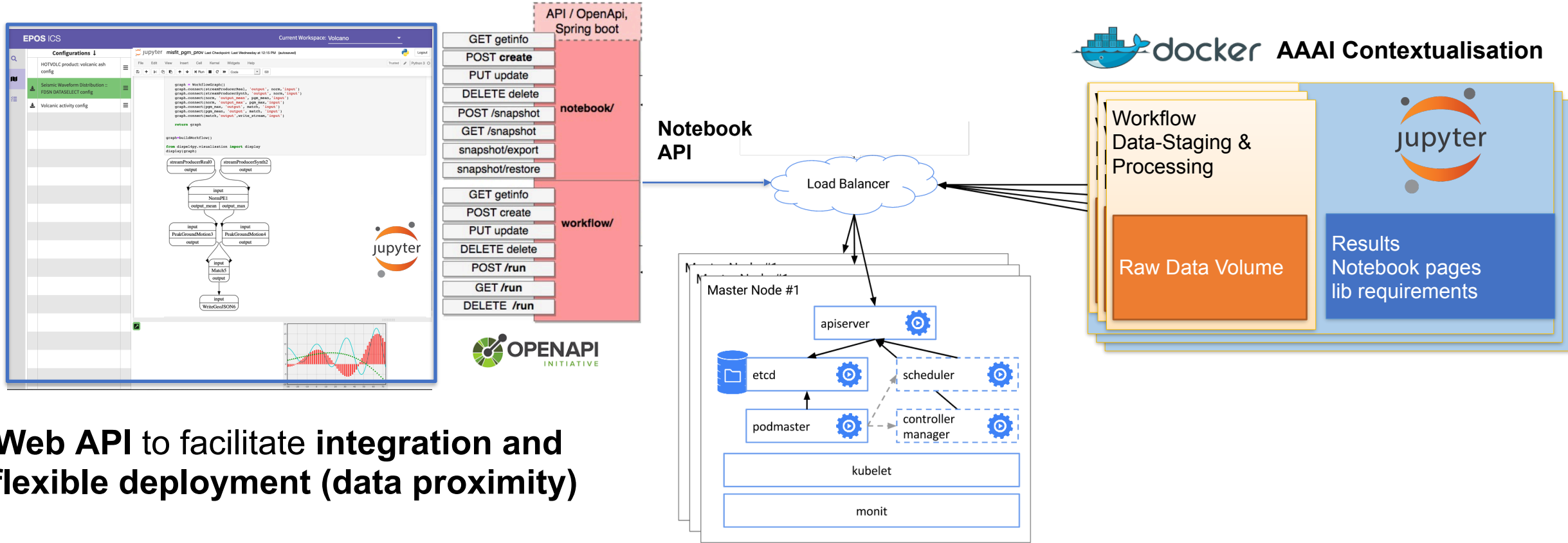
POST /workflowexecutions/{runid}/delete

POST /workflowexecutions/{runid}/edit

## monitor

GET /workflowexecutions/{runid}/showactivity

# Working Sessions as Notebooks Instances

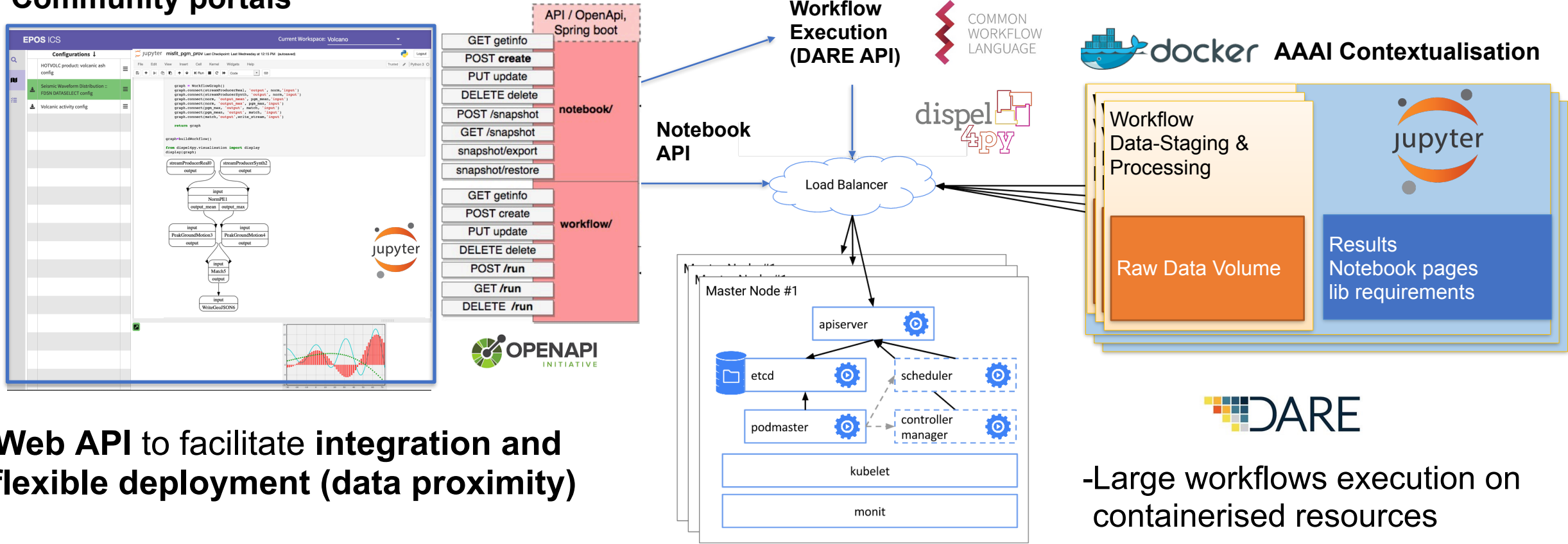


- **Web API to facilitate integration and flexible deployment (data proximity)**
- **Staging and Pre-processing (CWL)**
  - Data staging history
  - Automated resources preparation
  - Local Execution
  - **Traceable Updates of Working Sessions**

  
**kubernetes**  
 AWS, BRGM, GRNET

# Working Sessions as Notebooks Instances

## Community portals

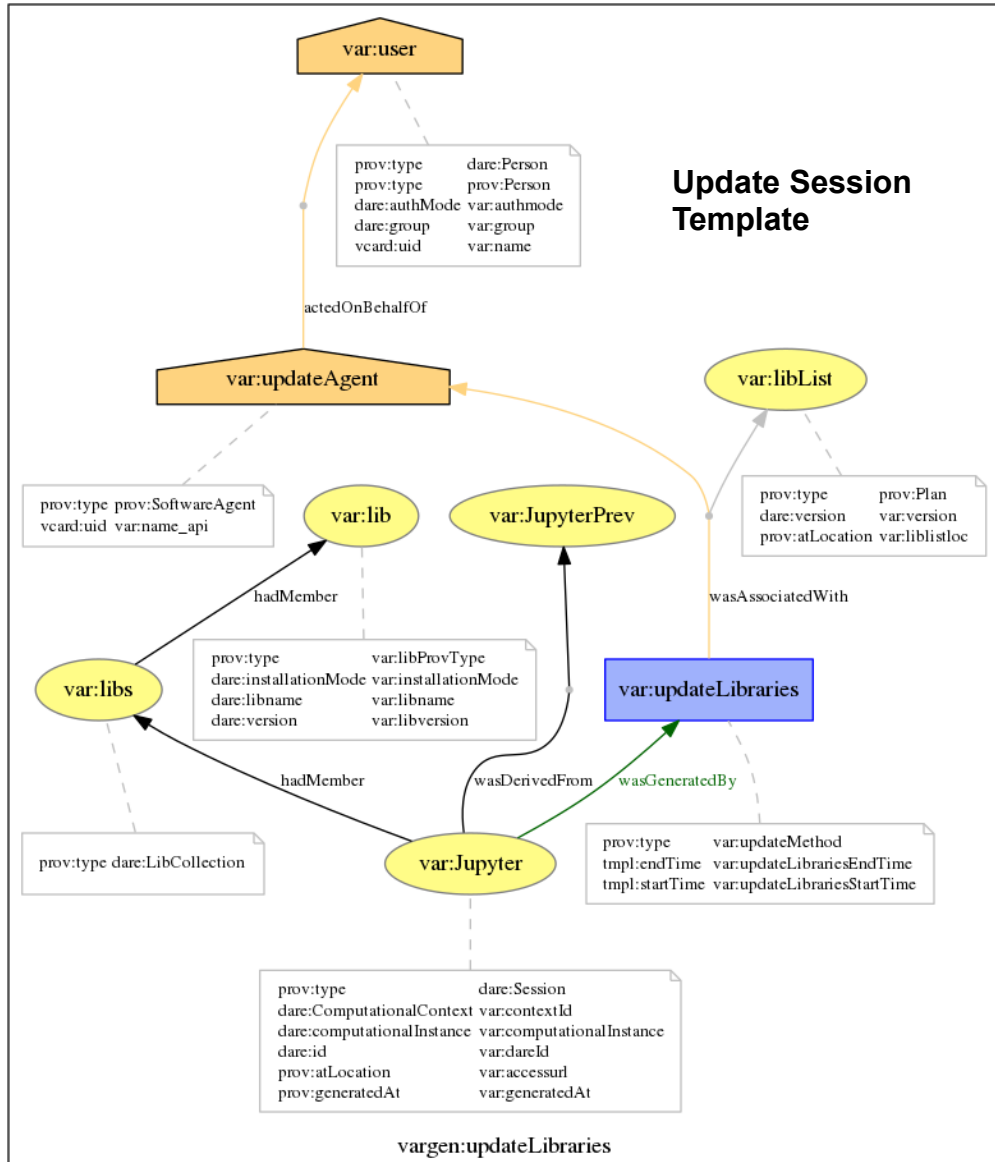


- **Web API to facilitate integration and flexible deployment (data proximity)**
- **Staging and Pre-processing (CWL)**
  - Data staging history
  - Automated resources preparation
  - Local Execution
  - **Traceable Updates of Working Sessions**

- Large workflows execution on containerised resources
- Combine different workflow languages (**CWL, dispel4py**)
- **Capture and analyse lineage**

**kubernetes**  
AWS, BRGM, GRNET

# Trace Notebook Creation and Updates



## PROV Templates

Model provenance of the creation and interactive update of a **Working Sessions**

- Distinction between user's choices and system's concerns
- Reproduce technical setups (.. also onto computational nodes)

- **Templates foster discussions** on provenance relationships involving heterogeneous agents and resources.
- **Modelling of usable and re-usable** provenance scenarios (tailoring vs generalisation)
- **Remove the burden to hardcode provenance editing** (expansion tools/services)
- **Bindings in applications costs less** than building graphs (weight-shift to the service)

Luc Moreau et al. A Templating System to Generate Provenance

<https://eprints.soton.ac.uk/405025/1/provtemplate.pdf>

ProvenanceTemplate Catalogue

<https://github.com/EnvriPlus-PROV/ProvTemplateCatalog>

<https://envriplus-provenance.test.fedcloud.eu/>



**Thanks!**