



# eFlows4HPC

## Demo Session: Deployment and Execution of a Workflow with HPCWaaS

Jorge Ejarque (BSC)

HPC workflows for climate models

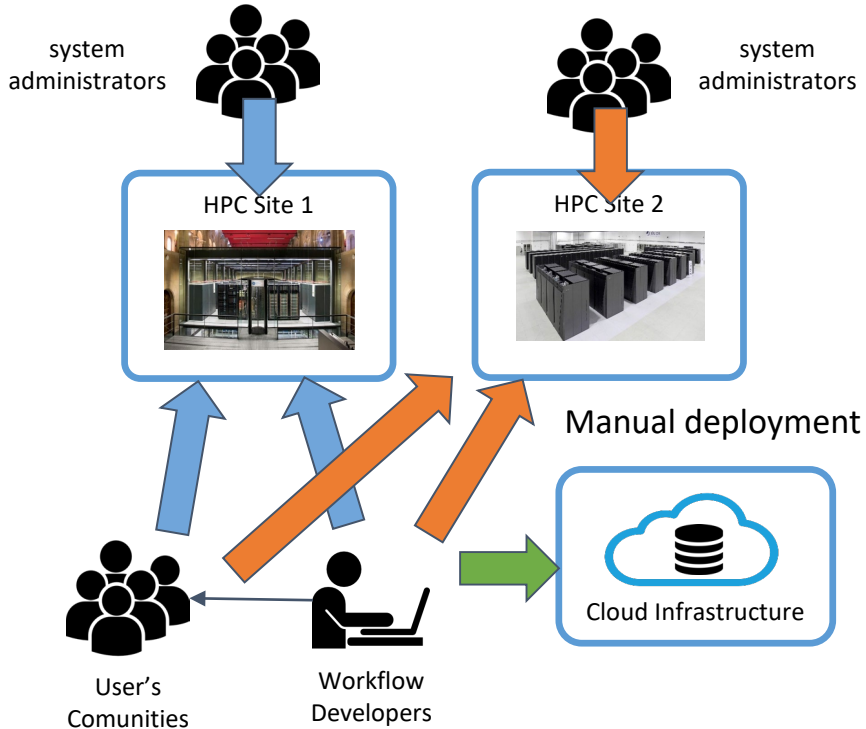
Espoo, October 17, 2023



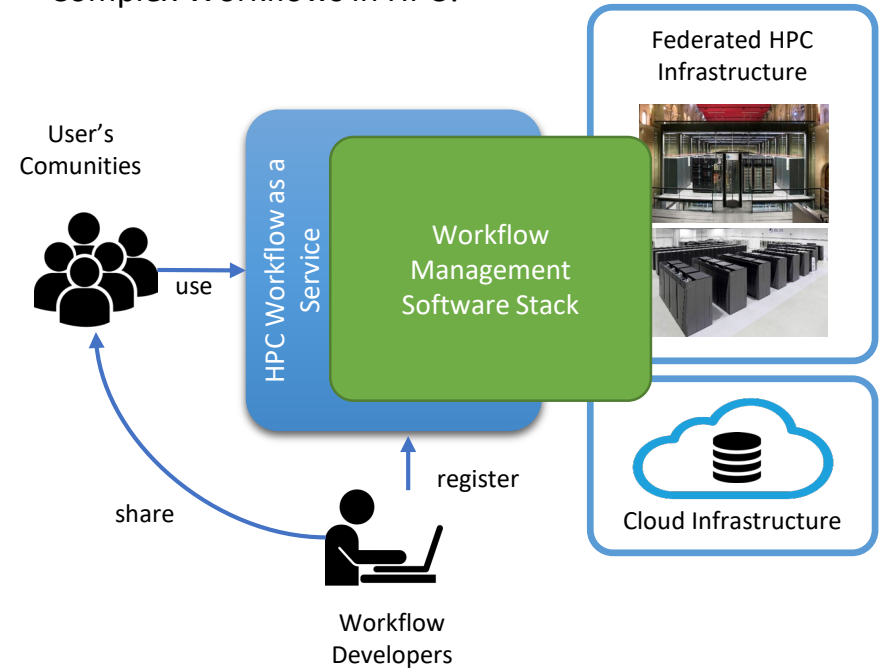
This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955558. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway. MCIN/AEI/10.13039/501100011033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

# Deployment in HPC Environments

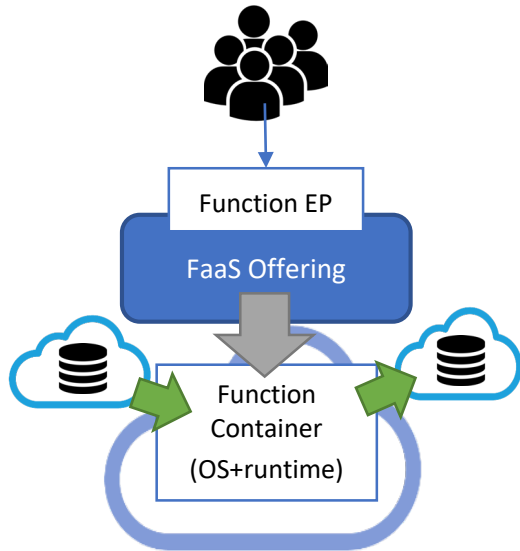
Current approach



Can we apply something like FaaS for Complex Workflows in HPC?



# FaaS vs. HPCWaaS

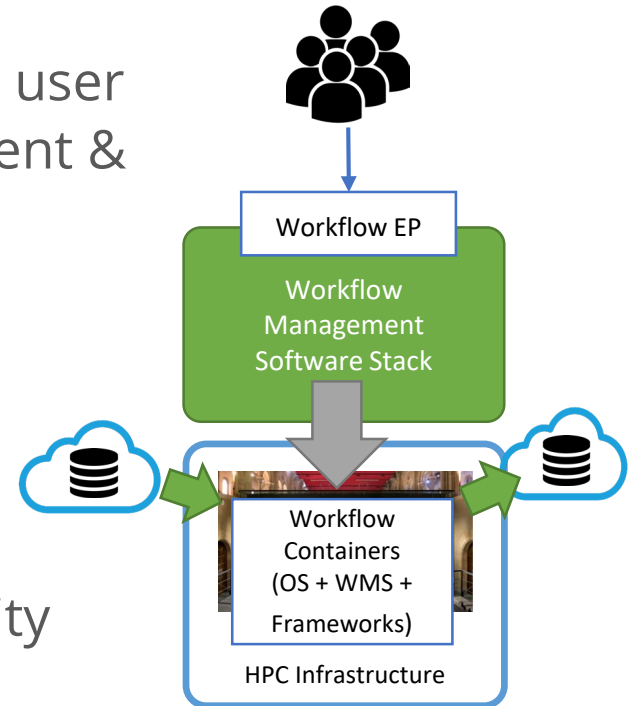


## Similarities

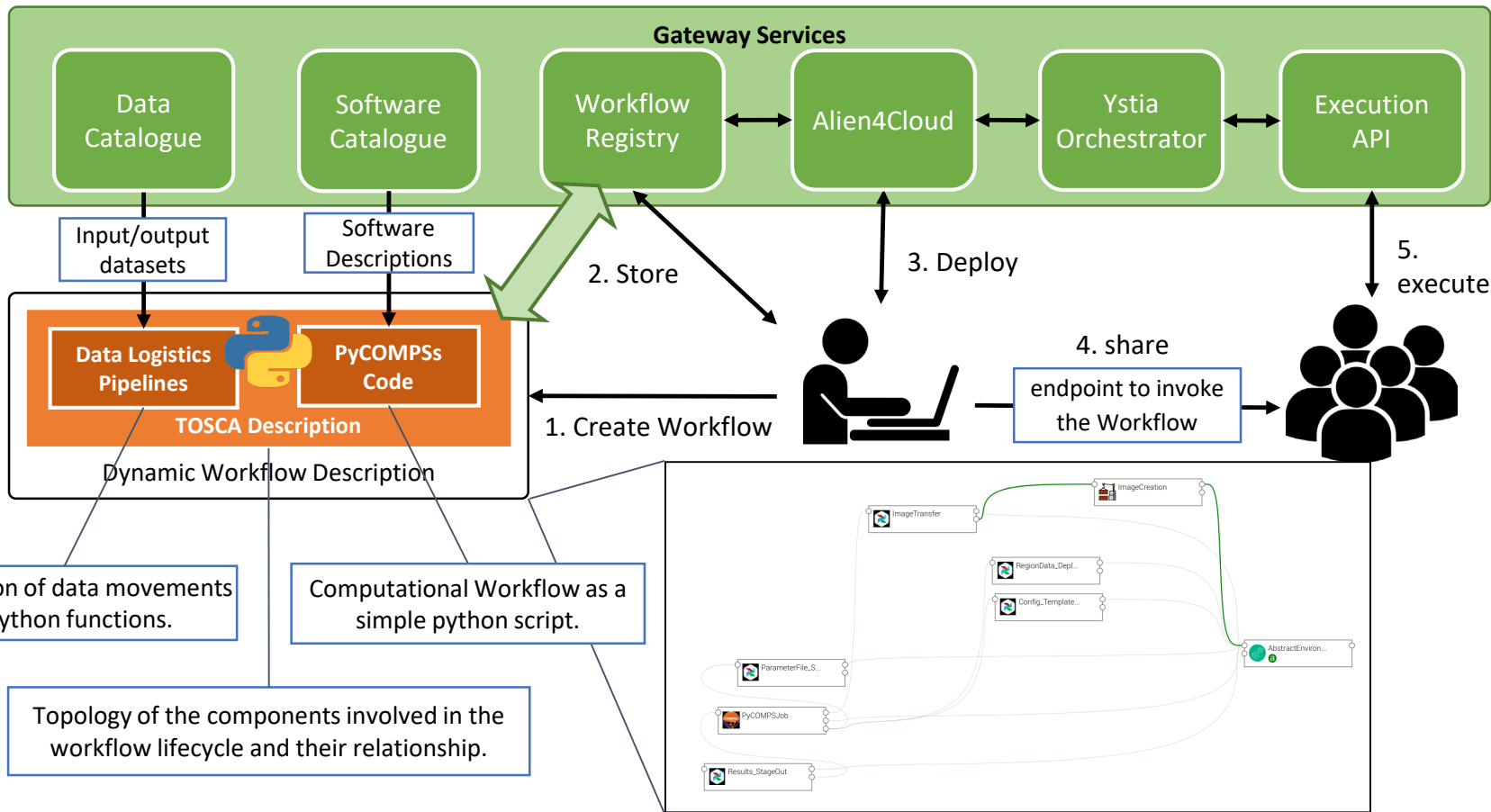
- Easy to use for final user
- Automate deployment & execution
- Data integration
- Containers

## Differences

- Restrictions
- Deployment and Execution Complexity
- Performance

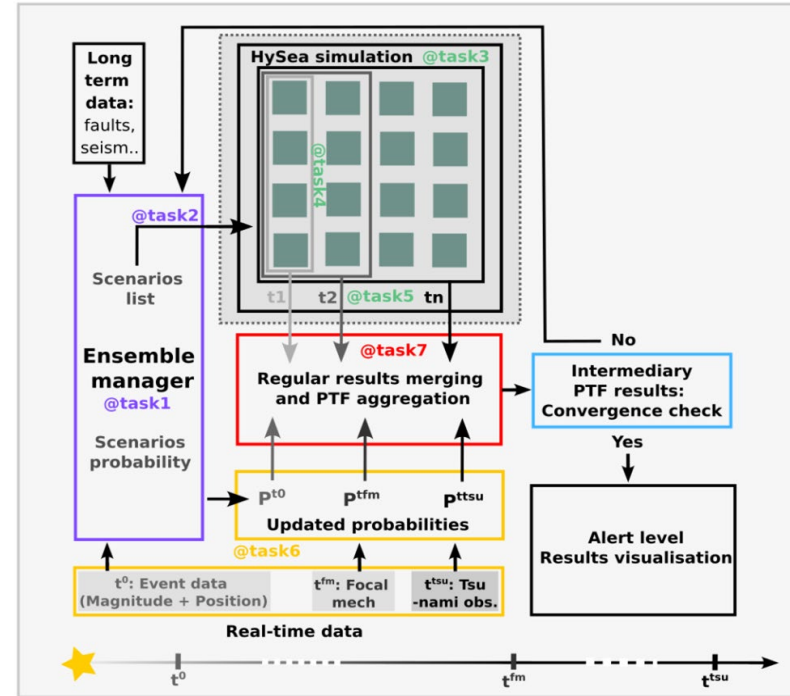


# Development Overview



# PTF Workflow

- Given an event in a region
- Generate a set of scenarios to simulate
- Every N simulation, computes the hazard curves in different parts of the coast
- Demo: Deploy and Execute the PTF Workflow in the CTE-Power9 machine at BSC



- Data Management
  - Required data and Results stored in the B2DROP and must be moved from/to HPC
  - Data Logistics Service and Data Catalogue
- Software Deployment
  - Workflows Code and required software in the HPC with Containers
  - Container Image Creation:
    - ✓ Build a container tailored for the target HPC machine
- Deployment and Execution Automation
  - TOSCA topology in the workflow registry
  - HPCWaaS:
    - ✓ Key management
    - ✓ Orchestration the Image creation, Data pipeline and PyCOMPS executions

```
@binary(binary=config_bin)
@task(config_file=FILE_OUT)
def build_config(config_template, config_file, data_dir, files_step2, par_file, kag, tsu, event_id):
    pass
```

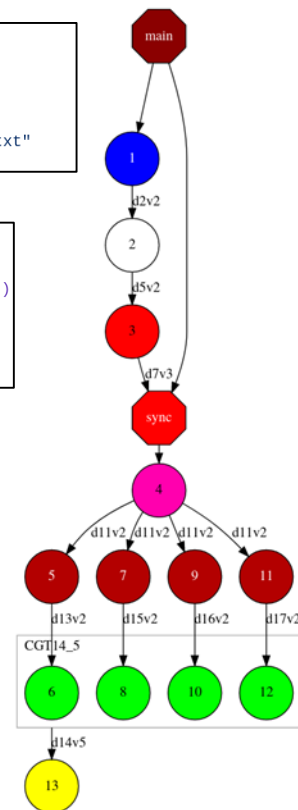
```
@task(config_file=FILE_IN, returns=1)
def step1_func(args, config_file, seistype, sim_files_step1):
    args.cfg=config_file
    run_step1_init(args, sim_files_step1)
    return sim_files_step1 + "/Step1_scenario_list_"+seistype+".txt"
```

```
@binary(binary=simulBS_bin, working_dir="{{wdir}}")
@task(sim_files_step2=FILE_OUT)
def build_structure(seistype, grid, hours, group, sim_files_step2, load_balancing, pois_ts_file,
    pass
```

```
@constraint(processors=[{'processorType':'CPU', 'computingUnits':'1'},
    {'processorType':'GPU', 'computingUnits':'1'}])
@mpi(binary=tsunamiHySEA_bin, args="{{file_in}}", runner="mpirun", processes=gpus_per_exec, processes_per_node=gpus_per_node, working_dir="{{wdir}}")
@task(file_in=FILE_IN, returns=1)
def mpi_func(file_in, wdir):
    pass
```

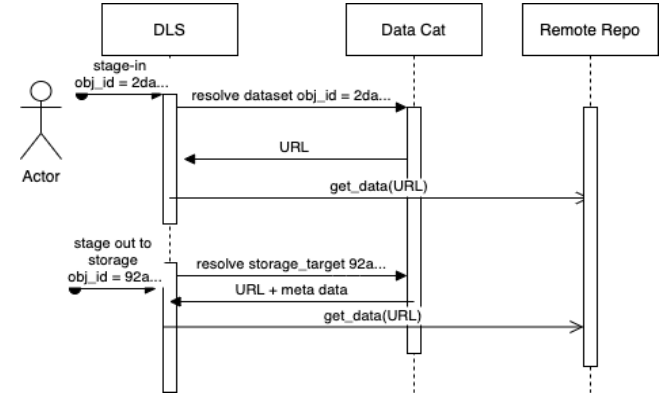
```
@task(ptf_files=COMMUTATIVE, config_file=FILE_IN)
def append_and_evaluate(ptf_files, ptf_file, args, config_file, sim_files_step1, out_step2_path, out_update_path, out_final, depth_file, log_file, sim_pois_ts, num_sims, kag, tsu,
    args.cfg = config_file
    ptf_files.append(ptf_file)
    if (num_sims != 0) and (len(ptf_files) % num_sims == 0):
        step2_create_ptf_input(ptf_files, out_step2_path, depth_file, log_file)
        if kag>0:
            run_step_kagan(args, sim_files_step1, out_update_path)
            sim_files_input=out_update_path
        elif tsu>0:
            run_step_mare(args, sim_files_step1, out_update_path, sim_pois_ts, ptf_files)
            sim_files_input=out_update_path
        else:
            sim_files_input=sim_files_step1
            run_step3_init(args, sim_files_input, out_final, sim_pois_ts, ptf_files)
```

```
@binary(binary="tar", args="zcvf {{outfile}} {{folder}}")
@task(outfile=FILE_OUT)
def compress(folder, outfile, ptf_files):
    pass
```



# Data pipelines

- Implemented in Data Logistics Service
- Reusable for multiple data/workflows
- Configured from Data Catalogue



## eFlows4HPC Data Catalog

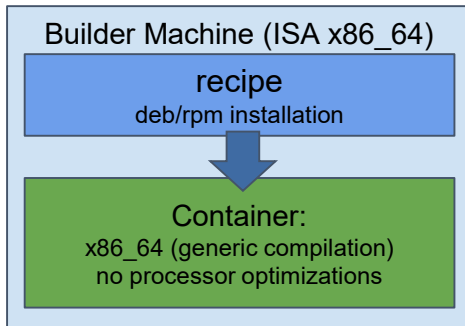
Property	Value
Name	PTF Workflow events and regions Data
OID	37d2f94b-3698-4a4a-937b-645a9c4fe879
URL	https://jorge@b2drop.bsc.es/remote.php/webdav/
<b>Other Metadata</b>	
path	eFlows4HPC/WPs/WP1/Testing_data/PTF/Regions/

All 18 Active 5 Paused 13 Filter DAGs by tag

🔍	DAG	Owner	Runs	Schedule	Last Run
🔵	plainhttp2ssh http ssh wp4	airflow	34 / 5	None	2023-09-12, 08:10:32
🔵	transfer_image example	airflow	20 / 3	None	2023-09-12, 06:15:54
🔵	upload_example example	airflow	2 / 4	None	2023-06-30, 14:51:32
🔵	webdav_stagein UCIS4EQ wps	airflow	40 / 5	None	2023-09-12, 06:15:34
🔵	webdav_stageout	airflow	10 / 0	None	2023-09-12, 10:37:15



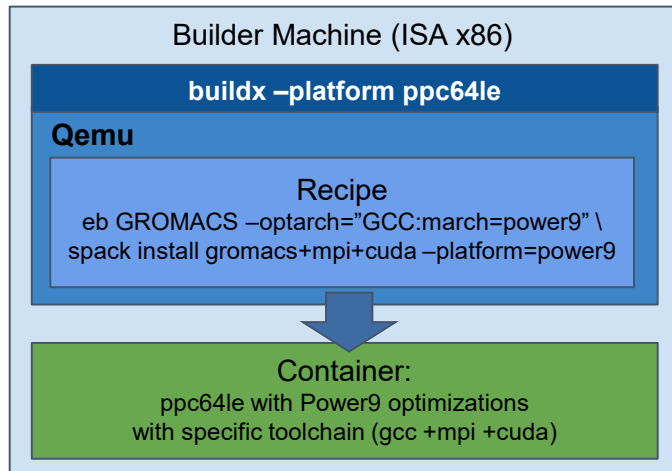
Standard container image creation



- **Simplicity for deployment**
  - Just pull or download the image
- **Trade-Off performance/portability**
  - Architecture Optimizations
- **Accessing Hardware from Containers**
  - MPI Fabric /GPUs
- **Host-Container Version Compatibility**

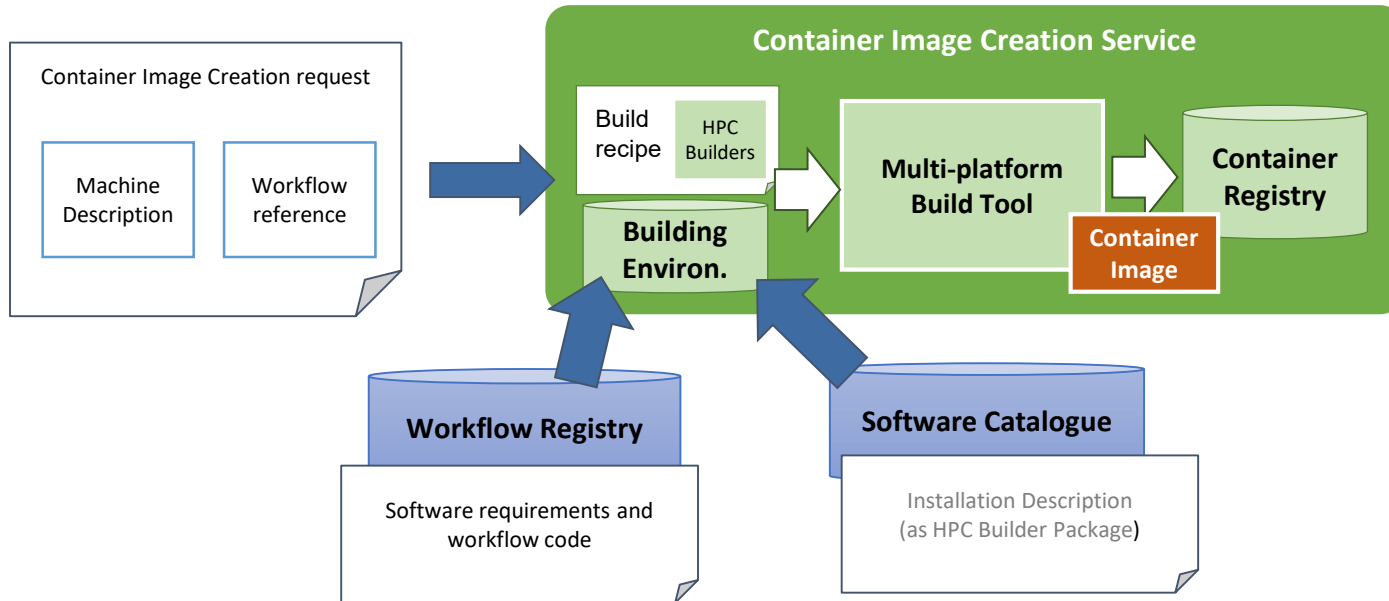
# HPC Ready Containers

eFlows4HPC approach



- **Methodology to allow the creation containers for specific HPC system**
  - Leverage HPC and Multi-platform container builders
- **It is tight to do by hand but let's automate!**

# Container Image Creation Service



# Container Image Creation Service



- Web Interface

The screenshot shows the 'New Container Image Build Request' form in the eFlows4HPC web interface. The form is divided into two main sections: 'Machine Description' and 'Workflow Reference'. The 'Machine Description' section includes a dropdown for 'System Platform', a text input for 'Processor Architecture', another dropdown for 'Container Engine', and optional text inputs for 'MPI version' and 'GPU runtime'. The 'Workflow Reference' section includes text inputs for 'Workflow Name' and 'Sub-workflow Name'. A blue 'Build' button is located at the bottom of the form. The navigation menu on the left includes Home, Builds, Images, and Account. The top header shows the eFlows4HPC logo, 'Dashboard', and 'Logout'.

- REST Interface and CLI

POST /build/

```
{
  "machine": {
    "platform": "linux/amd64",
    "architecture": "rome",
    "container_engine": "singularity",
    "workflow": "minimal_workflow",
    "step_id": "wordcount",
    "force": false
  }
}
```

HTTP/1.1 200 OK  
Content-Type: application/json

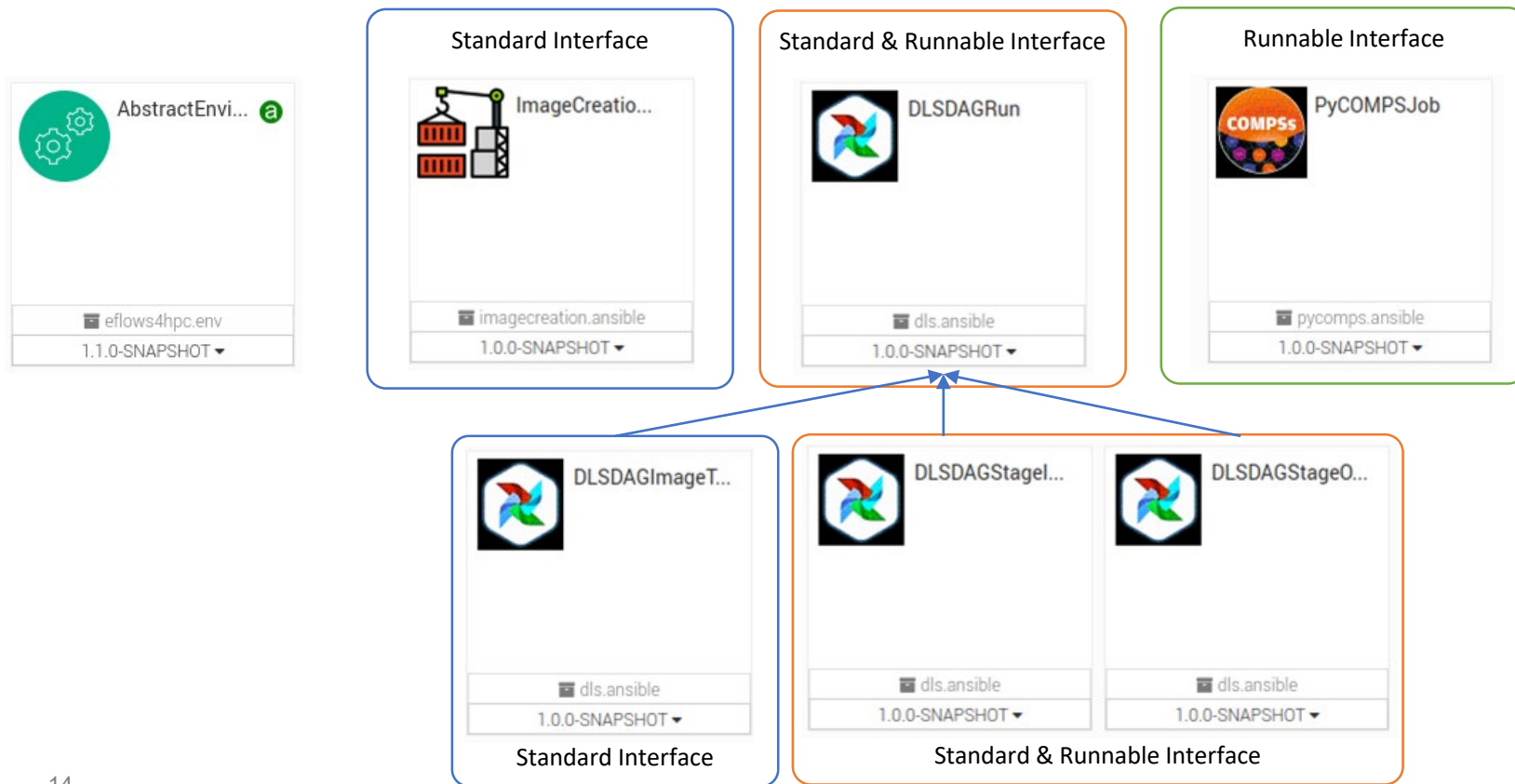
```
{
  "id": "<creation_id>"
}
```

```
localhost:~/image_creation> ./cic_cli <user> <token> https://<image_creation_url> build <request.json>
Response:
{"id": "f1f4699b-9048-4ecc-aff3-1c689b855adc"}
```

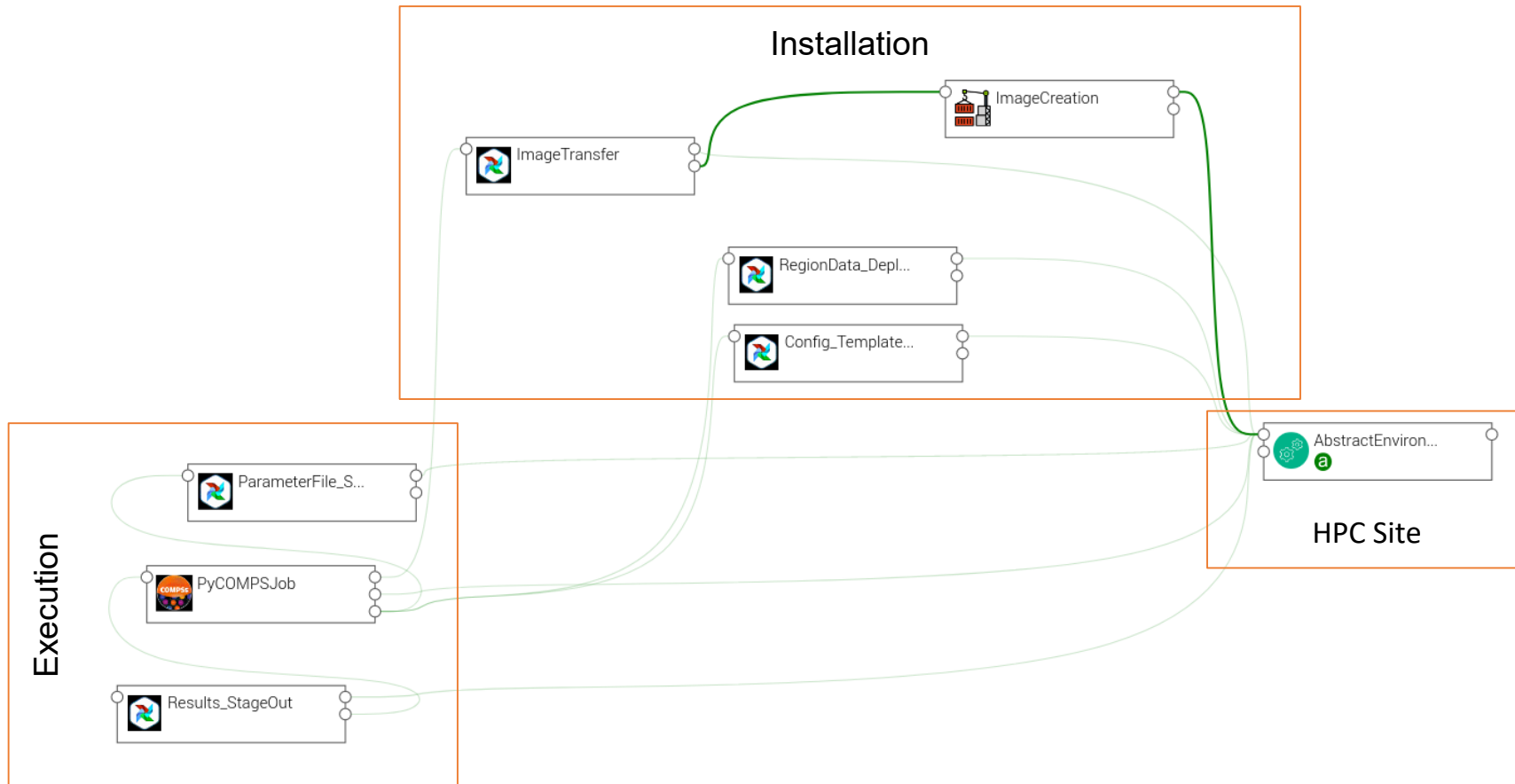
# TOSCA Model

- **Describe the orchestration of the application lifecycle management**
- **Topology of components with dependencies**
  - Application Component:
    - Describe what to do in every lifecycle step
      - ✓ Standard toasca steps (start, stop, delete,...)
      - ✓ Extended runnable (submit, run, cancel,...) Integrate jobs in Tosca.
    - The required input data and properties
  - Dependencies:
    - Describe the data exchanged between components.
- **Workflows**
  - Topology generate the standard TOSCA workflows to deploy/undeploy the application
  - Custom workflows

# eFlows4HPC TOSCA Components



# TOSCA Modelization



# Workflow Deployment (done once per HPC site)



- Set deployment input parameters (user, credential, select HPC location)

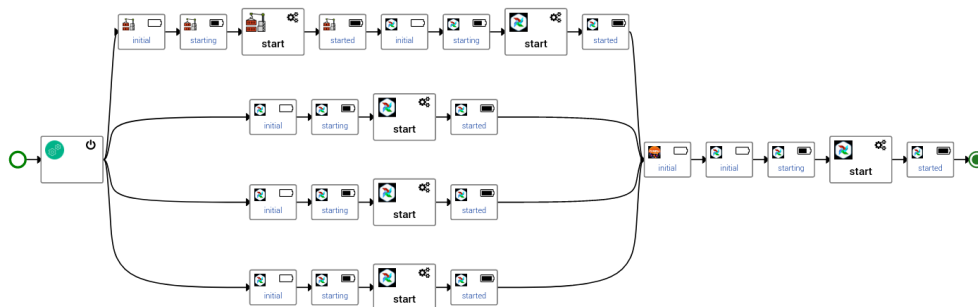
The screenshot shows the 'Inputs' tab of the deployment interface. It features a navigation bar with 'Applications' and 'Catalog' tabs, and a sub-bar with 'pillar\_1', 'Environment', and 'Inputs'. Below this, there are tabs for 'Home', 'Prepare next deployment 0.1.0-SNAPSHOT', and 'Manage current deployment'. A status indicator shows 'Undeployed'. A row of buttons includes 'Version', 'Topology', 'Inputs' (highlighted in blue), 'Locations', 'Matching', and 'Review & deploy'. The main section is titled 'Input properties' and contains a table with the following data:

Property	Value	Actions
debug	<input checked="" type="checkbox"/>	⌵ 0
user_id	bsc19611	⌵ 0
vault_id	eba73c03-470e-430a-bd0e-67f...	⌵ 0
container_image_transfer_directory	/gplfs/projects/bsc44/images	⌵ 0
mid	71e863ac-ae6-4680-a57c-de3...	⌵ 0
register_result_in_datacat	<input type="checkbox"/>	⌵ 0

Below the table, there is a section for 'Preconfigured input properties' which currently shows 'No data available.'

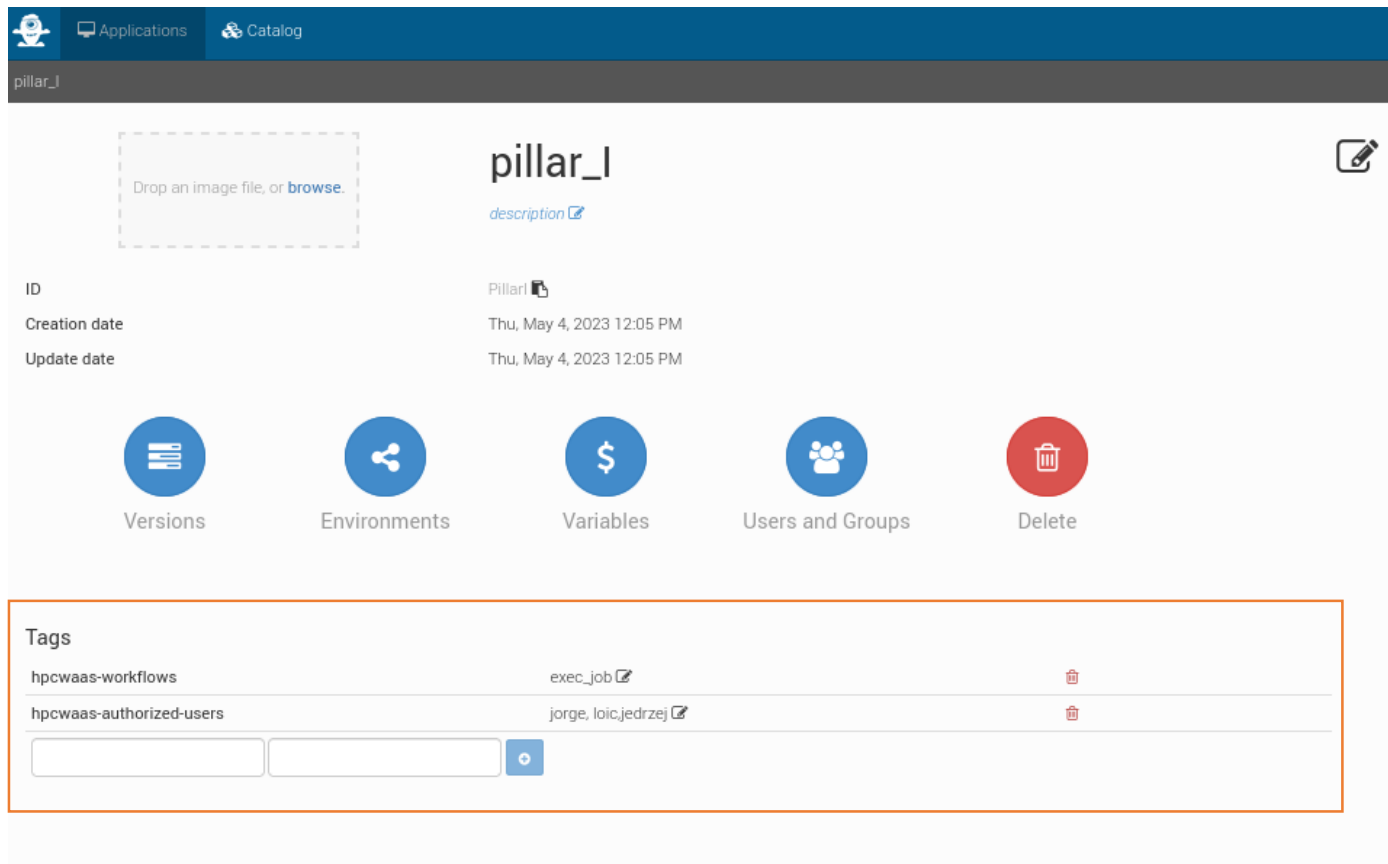
The screenshot shows the 'Matching' tab of the deployment interface. It features a navigation bar with 'Applications' and 'Catalog' tabs, and a sub-bar with 'pillar\_1', 'Environment', and 'Matching'. Below this, there are tabs for 'Home', 'Prepare next deployment 0.1.0-SNAPSHOT', and 'Manage current deployment'. A status indicator shows 'Undeployed'. A row of buttons includes 'Version', 'Topology', 'Inputs', 'Locations', 'Matching' (highlighted in blue), and 'Review & deploy'. The main section is titled 'Policies matching' and 'Nodes matching'. Under 'Nodes matching', there is a dropdown menu for 'AbstractEnvironment'. Below this, there is a table with the following data:

Name	Type
<input type="radio"/> bsc_nord3.1.0.0	<input checked="" type="checkbox"/> eflows4hpc.env.nodes.AbstractEnvironment
<input checked="" type="radio"/> bsc_amd.1.0.0	<input checked="" type="checkbox"/> eflows4hpc.env.nodes.AbstractEnvironment





# Publish workflow and authorize users



The screenshot displays the 'pillar\_1' workflow page in the eFlows4HPC interface. The top navigation bar includes 'Applications' and 'Catalog'. The main content area shows the workflow name 'pillar\_1' with a description link. Below this, a metadata table lists the ID, creation date, and update date. A row of five circular icons provides management actions: Versions, Environments, Variables, Users and Groups, and Delete. A 'Tags' section at the bottom, highlighted with an orange border, contains a table of tags and their associated users, along with input fields for adding new tags.

ID	Creation date	Update date
pillar_1	Thu, May 4, 2023 12:05 PM	Thu, May 4, 2023 12:05 PM

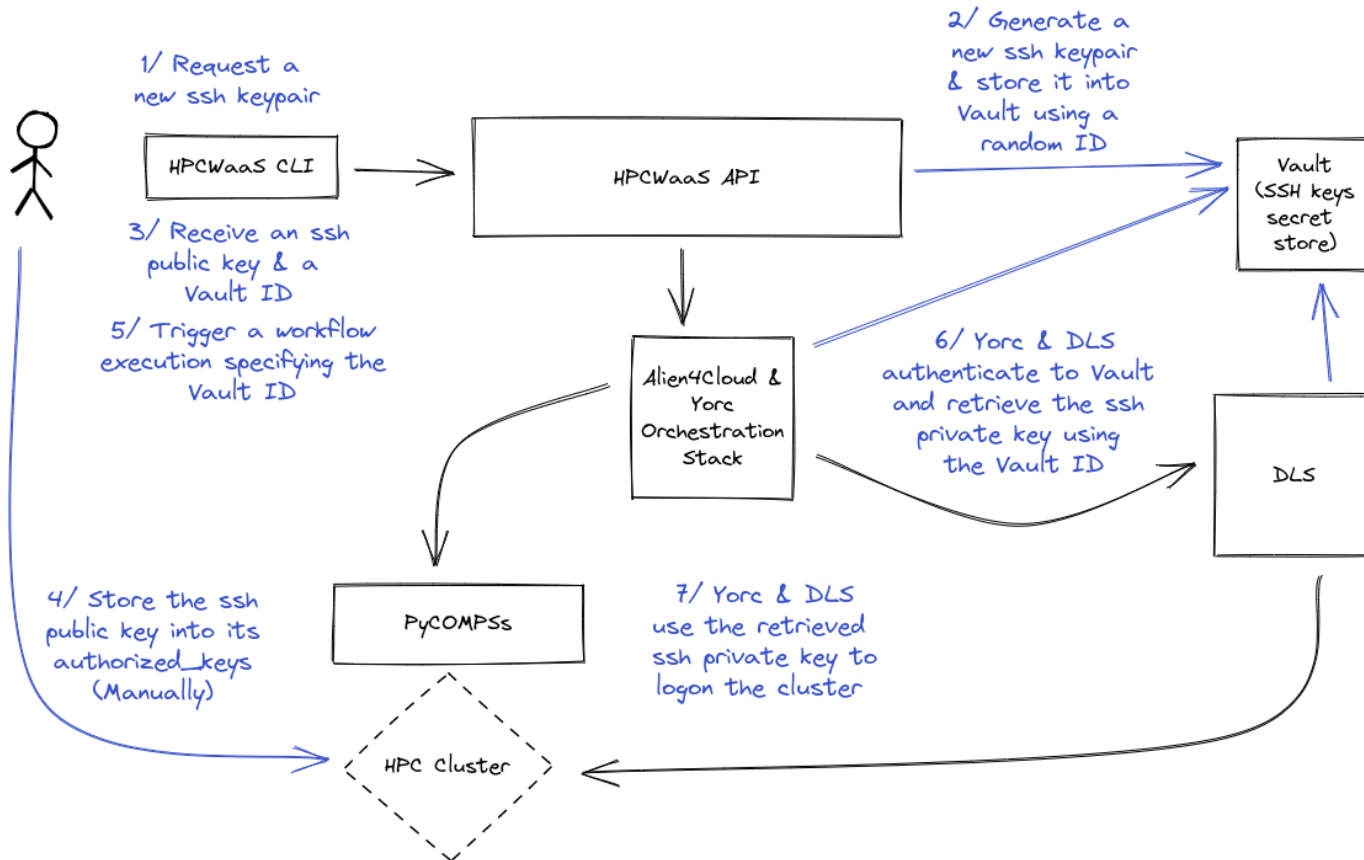
Icon	Action
	Versions
	Environments
	Variables
	Users and Groups
	Delete

Tags		
hpcwaas-workflows	exec_job	
hpcwaas-authorized-users	jorge, loic,jedrzej	

Input fields:

# Workflow Execution End user



# Thank you



## eFlows4HPC

Enabling dynamic and Intelligent workflows  
in the future EuroHPC ecosystem

[www.eFlows4HPC.eu](http://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.