



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

Raül Sirvent, Rosa M Badia

18/11/2024

SC24 tutorial, Atlanta, 18 November 2024

Tutorial website

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



Agenda

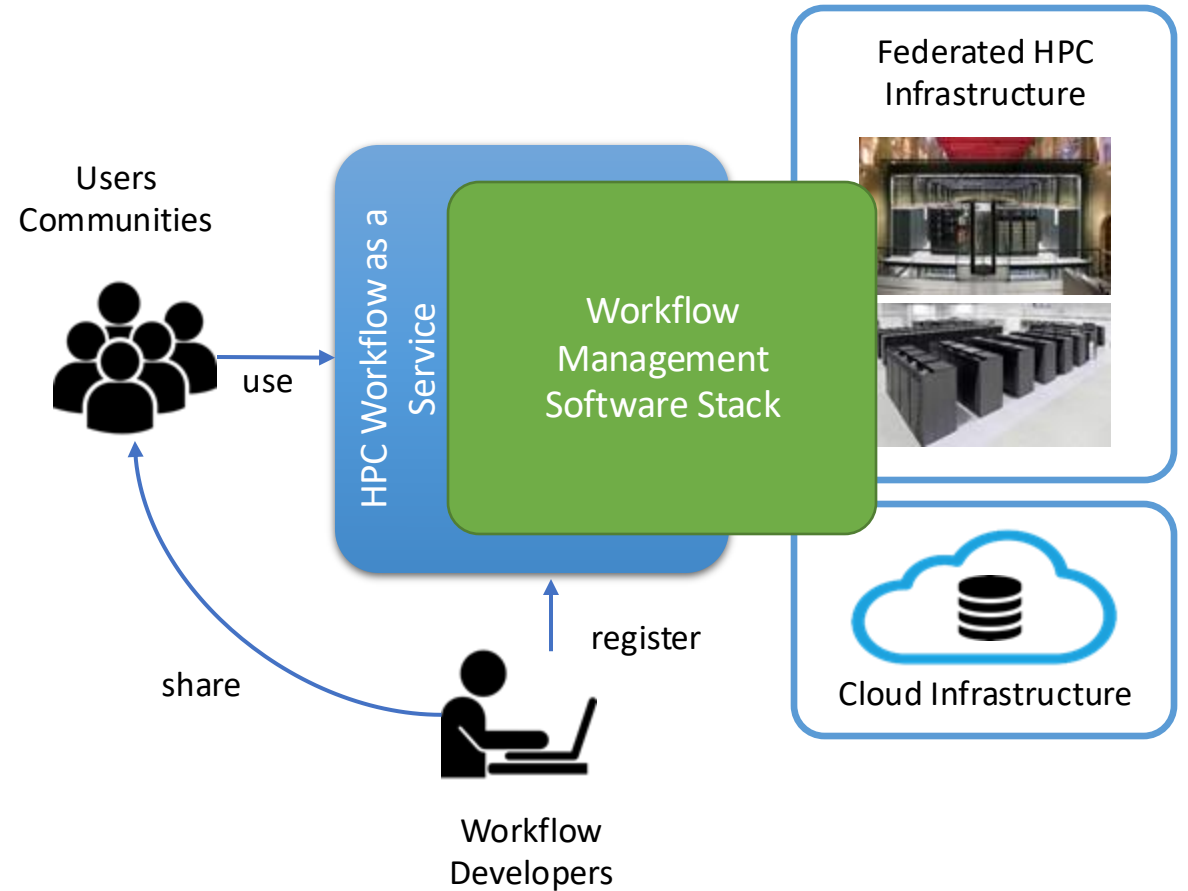
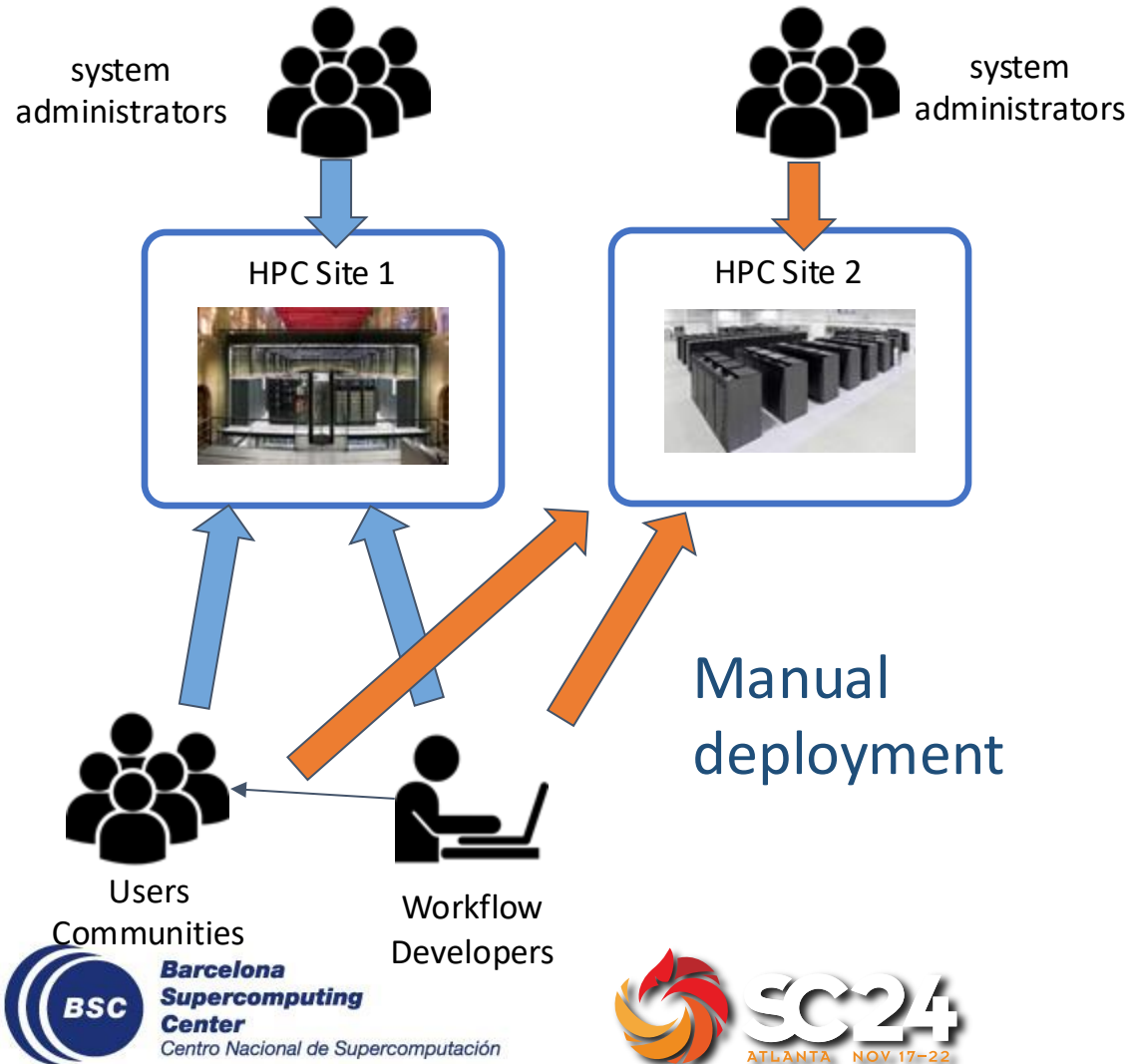
8:30 – 8:35	Overview of tutorial agenda	Rosa M Badia
8:35 – 9:05	Part 1.1: Hybrid HPC+AI+DA workflow development with PyCOMPSs <ul style="list-style-type: none">- Context of the workflows at BSC- Overview of workflow development with PyCOMPSs- Extensions for the integration of HPC with AI and DA- Sample workflows	Rosa M Badia
9:05 – 9:35	Part 1.2: Workflows' reproducibility through provenance <ul style="list-style-type: none">- 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 <ul style="list-style-type: none">- 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

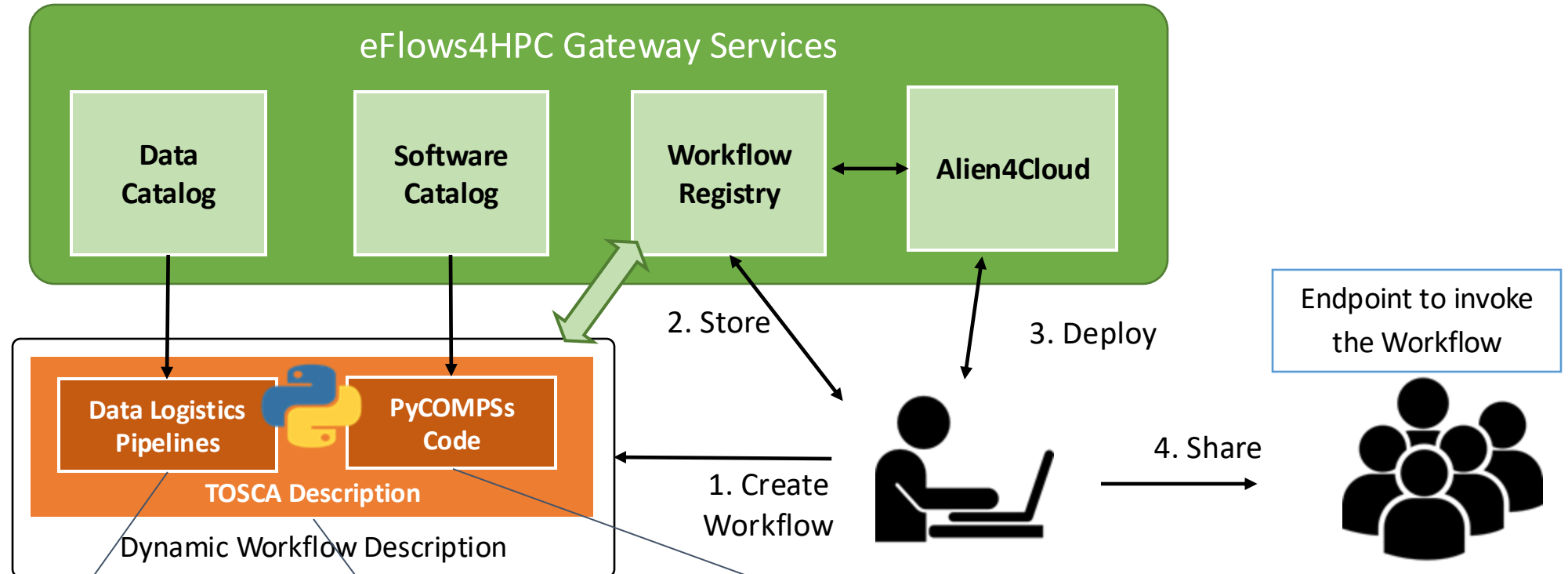
Deployment in HPC Environments

Current approach



Can we design something like FaaS for Complex Workflows for HPC?

HPCWaaS: Workflow lifecycle overview

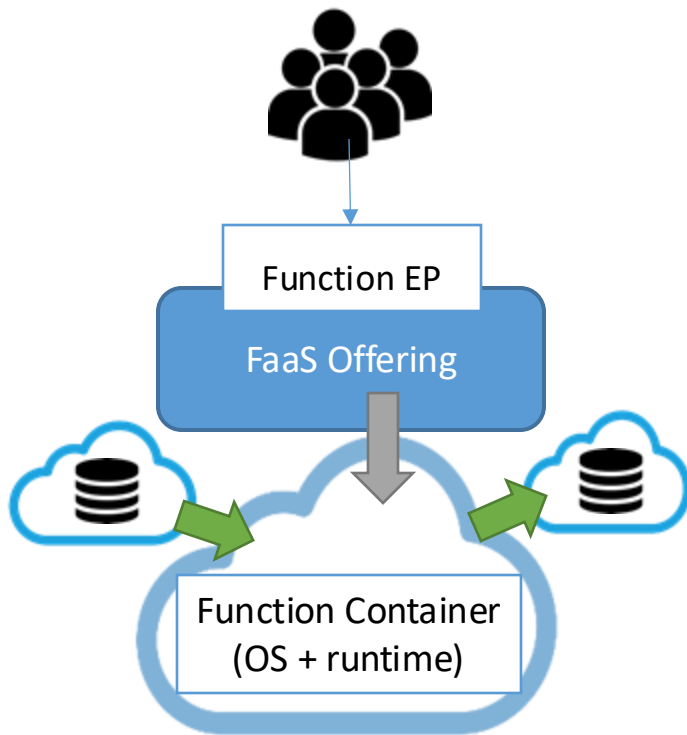


Description of data movements as Python functions.
Input/output datasets described at Data Catalog

Computational Workflow as a simple Python script.
Invocation of software described in the Software Catalog

Topology of the components involved in the workflow
lifecycle and their relationship.

FaaS vs HPCWaaS

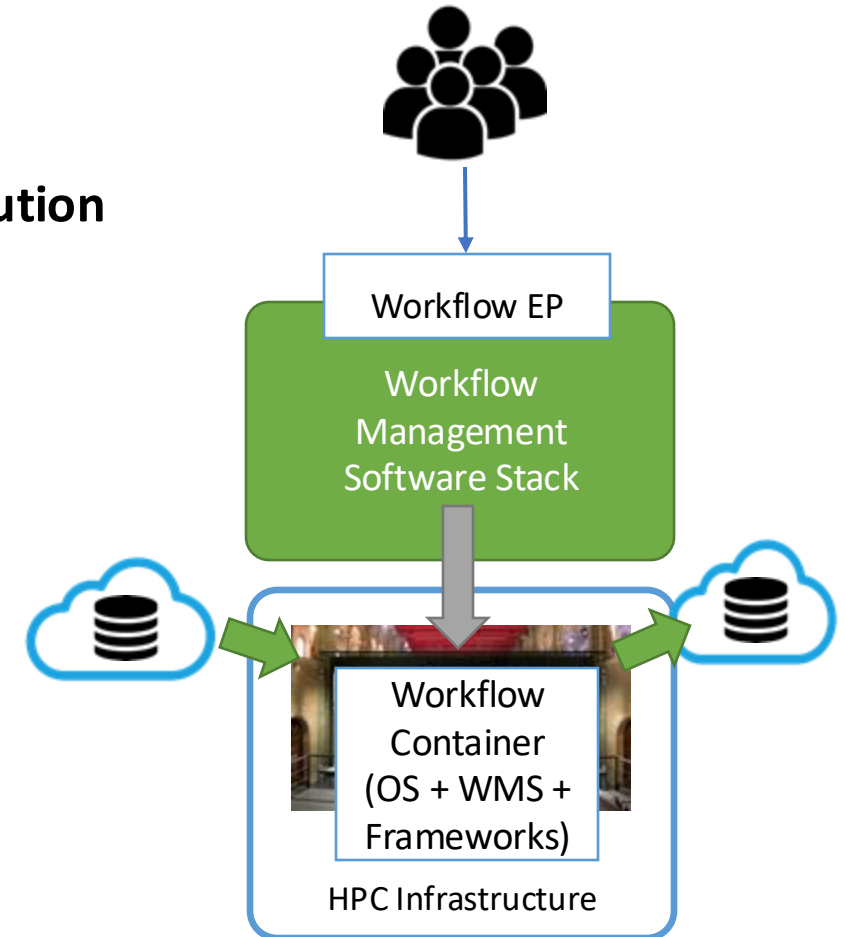


Similarities

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

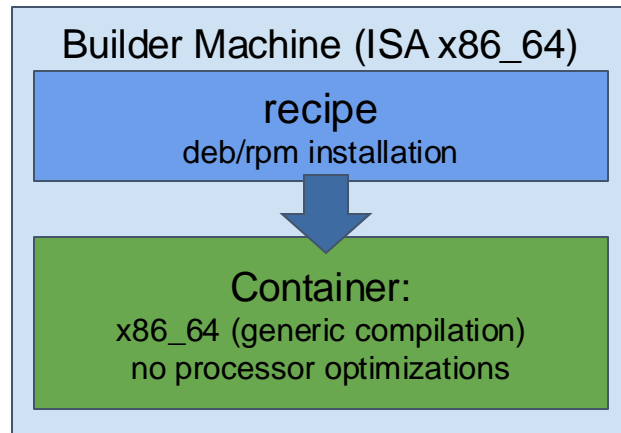
Differences

- HPC policies & requirements
- Deployment and Execution Complexity
- **Performance needs**



Containers and HPC

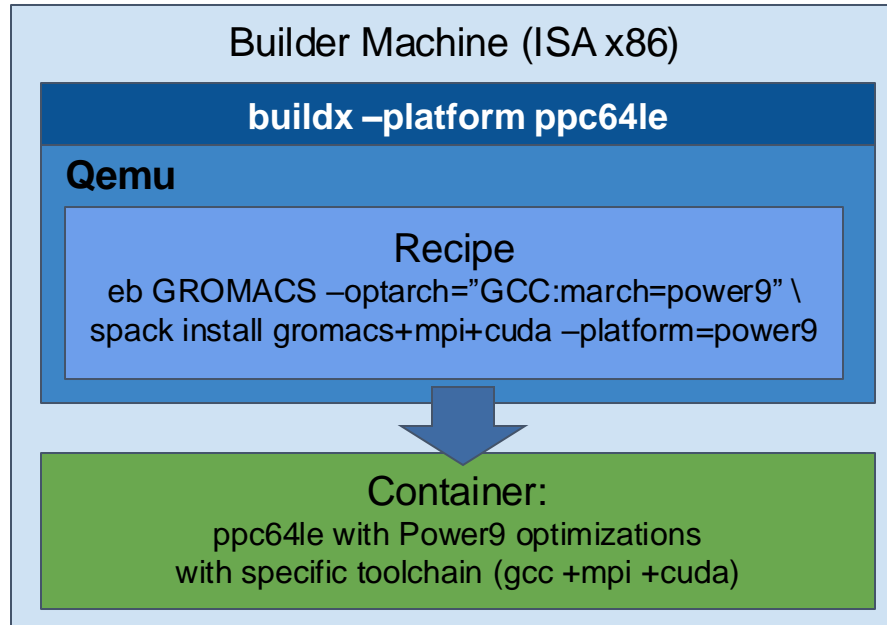
Standard container image creation



- 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!

HPC Ready Containers

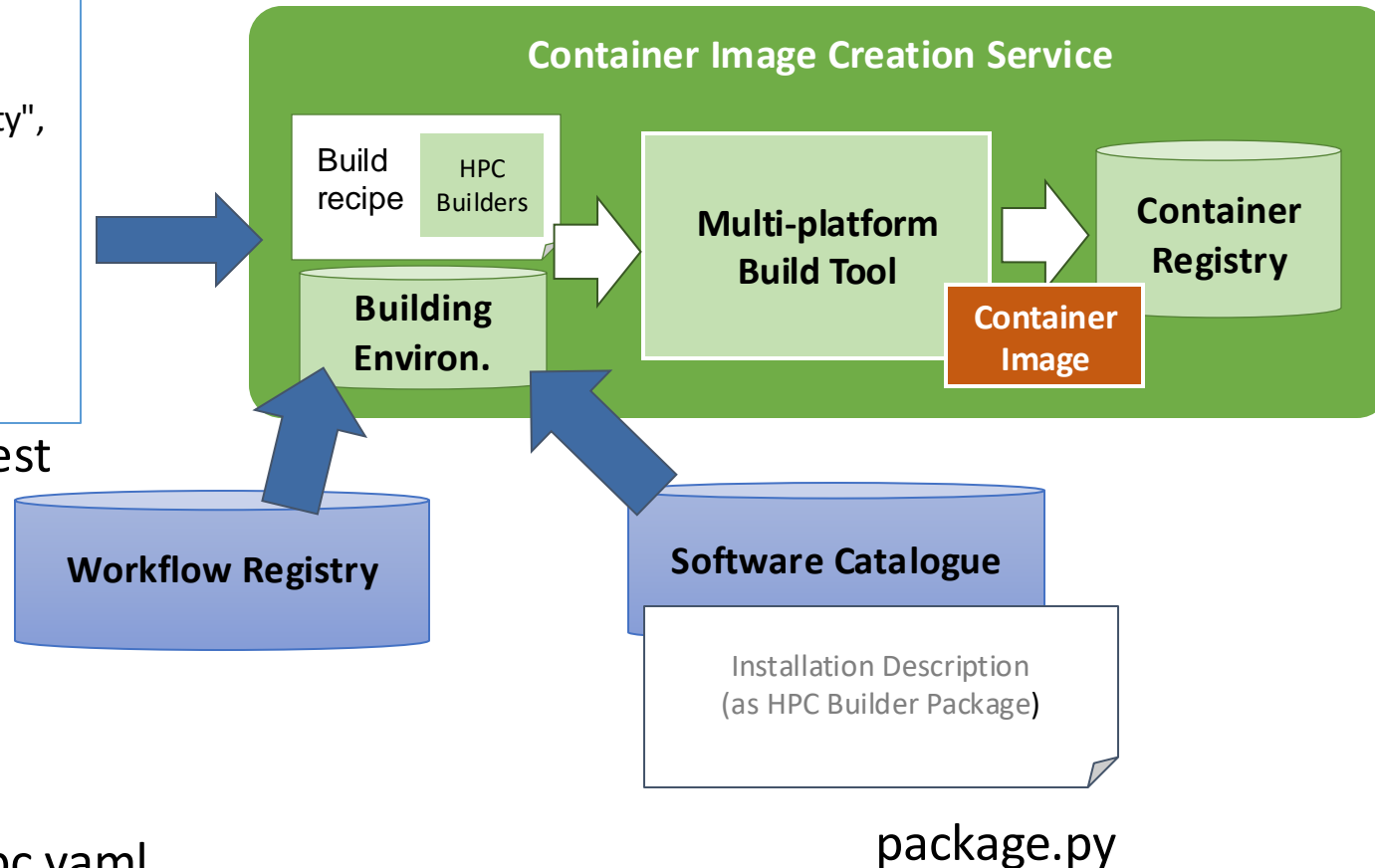
Workflow step + target system

```
{
  "machine": {
    "platform": "linux/amd64",
    "architecture": "skylake",
    "container_engine": "singularity",
    "mpi": "openmpi@4.1.1"
  },
  "workflow": "tutorial",
  "step_id": "HPC_AI_training",
  "workflow_yaml": "eflows4hpc.yaml",
}
```

Service request

```
1 apt:
2   - graphviz
3   - libbz2-dev
4 spack:
5   specs:
6     - compss@3.3.2
7     - py-dislib@master
8     - alya@master
9 pip:
10  - pyyaml
11  - pydoe
12  - pandas
13  - pillow
```

eflows4hpc.yaml



CAELESTIS workflow

<https://github.com/eflows4hpc>

- Example of HPC ready container generation
- Sample request (json):

Yaml file describing modules involved in the workflow

Target: MareNostrum4 architecture

```
{
  "machine": {
    "platform": "linux/amd64",
    "architecture": "skylake",
    "container_engine": "singularity",
    "mpi": "openmpi@4.1.1"
  },
  "workflow": "tutorial",
  "step_id": "HPC_AI_training",
  "workflow_yaml": "eflows4hpc.yaml",
  "force": "True",
  "push": "False"
}
```

Location in
workflow-registry

[workflow-registry](#) / [tutorial](#) / [HPC_AI_training](#) / [eflows4hpc.yaml](#)

Jorge Ejarque add acm_summer_school workflow

Code Blame 16 lines (16 loc) · 293 Bytes

```
1 apt:
2     - graphviz
3     - libbz2-dev
4 spack:
5     specs:
6         - compss@3.3.2
7         - py-dislib@master
8         - alya@master
9 pip:
10    - pyyaml
11    - pydoe
12    - pandas
13    - pillow
14    - recreate
```

CAELESTIS workflow

<https://github.com/eflows4hpc>

[workflow-registry](#) / [tutorial](#) / [HPC_AI_training](#) / [eflows4hpc.yaml](#) 

Specific spack packages

 Jorge Ejarque add acm_summer_school workflow

Code Blame 16 lines (16 loc) · 293 Bytes

```
1 apt:
2     - graphviz
3     - libbz2-dev
4 spack:
5     specs:
6         - compss@3.3.2
7         - py-distlib@master
8         - alya@master
9 pip:
10    - pyyaml
11    - pydoe
12    - pandas
13    - pillow
14    - rocrate
15    - pickle5
16    - contextvars
```

[software-catalog](#) / [packages](#) / [compss](#) / [package.py](#) 

 Jorge Ejarque update versions

Code Blame 72 lines (62 loc) · 3.16 KB

```
1 # Copyright 2013-2021 Lawrence Livermore National Security, LLC and other
2 # Spack Project Developers. See the top-level COPYRIGHT file for details.
3 #
4 # SPDX-License-Identifier: (Apache-2.0 OR MIT)
5
6 # -----
7 # If you submit this package back to Spack as a pull request,
8 # please first remove this boilerplate and all FIXME comments.
9 #
10 # This is a template package file for Spack. We've put "FIXME"
11 # next to all the things you'll want to change. Once you've handled
12 # them, you can save this file and test your package like this:
13 #
14 #     spack install compss
15 #
16 # You can edit this file again by typing:
17 #
18 #     spack edit compss
19 #
20 # See the Spack documentation for more information on packaging.
21 # -----
22
23 from spack import *
24
25
26 class Compss(Package):
27     """COMP Superscalar programming model and runtime."""
28
29     # Add a proper url for your package's homepage here.
30     homepage = "https://compss.bsc.es"
31     url      = "https://compss.bsc.es/repo/sc/stable/COMPSS_2.10.tar.gz"
32
33     # FIXME: Add the source code for this package.
34
35     def install(self):
36         # FIXME: Add the installation steps for this package.
```

CAELESTIS workflow

<https://github.com/eflows4hpc>

[workflow-registry](#) / [tutorial](#) / [HPC_AI_training](#) / [eflows4hpc.yaml](#) 

Specific spack packages

 Jorge Ejarque add acm_summer_school workflow

Code Blame 16 lines (16 loc) · 293 Bytes

```
1 apt:
2     - graphviz
3     - libbz2-dev
4 spack:
5     specs:
6         - compss@3.3.2
7         - py-dislib@master
8         - alya@master
9 pip:
10    - pyyaml
11    - pydoe
12    - pandas
13    - pillow
14    - rocrate
15    - pickle5
16    - contextvars
```

[software-catalog](#) / [packages](#) / [py-dislib](#) / [package.py](#) 

 Jorge Ejarque update versions

Code Blame 59 lines (51 loc) · 2.5 KB

```
1  # Copyright 2013-2021 Lawrence Livermore National Security, LLC and other
2  # Spack Project Developers. See the top-level COPYRIGHT file for details.
3  #
4  # SPDX-License-Identifier: (Apache-2.0 OR MIT)
5
6  # -----
7  # If you submit this package back to Spack as a pull request,
8  # please first remove this boilerplate and all FIXME comments.
9  #
10 # This is a template package file for Spack. We've put "FIXME"
11 # next to all the things you'll want to change. Once you've handled
12 # them, you can save this file and test your package like this:
13 #
14 #     spack install py-dislib
15 #
16 # You can edit this file again by typing:
17 #
18 #     spack edit py-dislib
19 #
20 # See the Spack documentation for more information on packaging.
21 # -----
22
23 from spack import *
24
25
26 class PyDislib(PythonPackage):
27     """FIXME: Put a proper description of your package here."""
```

CAELESTIS workflow

<https://github.com/eflows4hpc>

[workflow-registry](#) / [tutorial](#) / [HPC_AI_training](#) / [eflows4hpc.yaml](#) 

Specific spack packages

 Jorge Ejarque add acm_summer_school workflow

Code Blame 16 lines (16 loc) · 293 Bytes

```
1 apt:
2     - graphviz
3     - libbz2-dev
4 spack:
5     specs:
6         - compss@3.3.2
7         - py-dislib@master
8         - alya@master
9 pip:
10    - pyyaml
11    - pydoe
12    - pandas
13    - pillow
14    - rccrate
15
16
```

[software-catalog](#) / [packages](#) / [alya](#) / [package.py](#) 

 FernandoVN98 Added alya package

Code Blame 59 lines (49 loc) · 2.03 KB

Raw   

```
1  # Copyright 2013-2022 Lawrence Livermore National Security, LLC and other
2  # Spack Project Developers. See the top-level COPYRIGHT file for details.
3  #
4  # SPDX-License-Identifier: (Apache-2.0 OR MIT)
5
6  # -----
7  # If you submit this package back to Spack as a pull request,
8  # please first remove this boilerplate and all FIXME comments.
9  #
10 # This is a template package file for Spack. We've put "FIXME"
11 # next to all the things you'll want to change. Once you've handled
12 # them, you can save this file and test your package like this:
13 #
14 #     spack install alya
15 #
16 # You can edit this file again by typing:
17 #
```

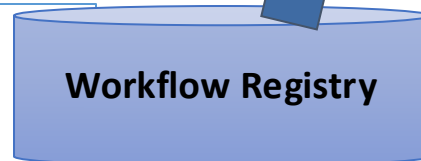
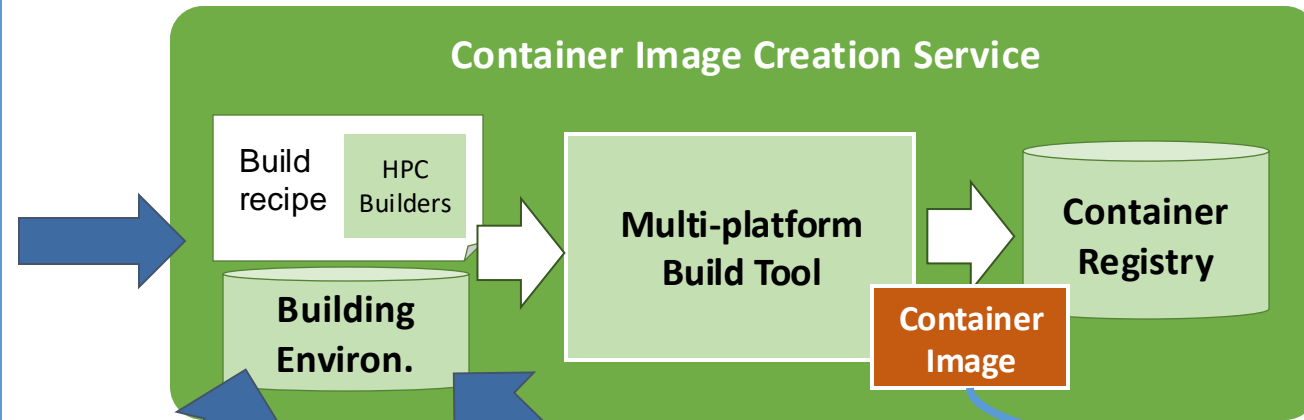
Some taken from default Spack repository

```
22
23 from spack import *
24 import os
25
26 class Alya(CMakePackage):
```

HPC Ready Containers

Service request

```
{
  "machine": {
    "platform": "linux/amd64",
    "architecture": "skylake",
    "container_engine": "singularity",
    "mpi": "openmpi@4.1.1"
  },
  "workflow": "tutorial",
  "step_id": "HPC_AI_training",
  "workflow_yaml": "eflows4hpc.yaml",
}
```



eflows4hpc.yaml



Installation Description
(as HPC Builder Package)

package.py

sc24_workflow_tutorial.sif

```
1 apt:
2     - graphviz
3     - libbz2-dev
4 spack:
5     specs:
6     - compss@3.3.2
7     - py-dislib@master
8     - alya@master
9 pip:
10    - pyyaml
11    - pydoe
12    - pandas
13    - pillow
```


Further Information

- Project page: <http://www.bsc.es/compss>
 - Documentation
 - Virtual Appliance for testing & sample applications
 - Tutorials



- Source Code

<https://github.com/bsc-wdc/compss>



- Docker Image

<https://hub.docker.com/r/compss/compss>

- Applications



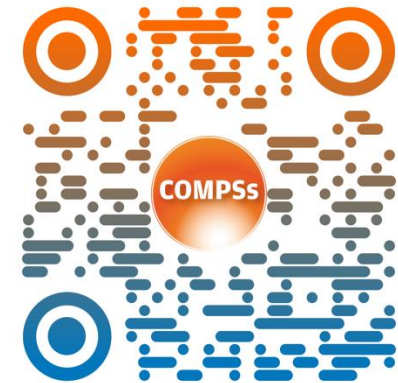
<https://github.com/bsc-wdc/apps>

<https://github.com/bsc-wdc/dislib>



- Dislib

<https://dislib.readthedocs.io/en/latest/>



ACKs



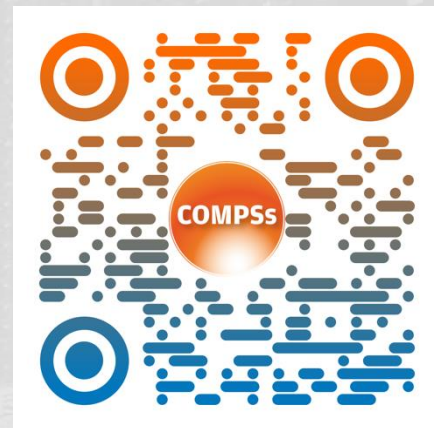
HP2C-DT





**Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

Thanks!



rosa.m.badia@bsc.es